EVALITA
Evaluation
# of NLP
and Speech Tools
for Italian

Proceedings of the Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian Final Workshop (EVALITA 2020)

December 17th, 2020

Editors:

Valerio Basile
Danilo Croce
Maria Di Maro
Lucia C. Passaro

aA
ccademia
university
press

aA

sede legale: c/o Bernardo Magnini, Via delle Cave 61, 38122 Trento
codice fiscale 96101430229
email: info@ai-lc.it

Accademia University Press
via Carlo Alberto 55
I-10123 Torino
info@aAccademia.it

isbn PDF 9791280136275
www.aAccademia.it/EVALITA_2020

Accademia University Press è un marchio registrato di proprietà
di LEXIS Compagnia Editoriale in Torino srl

Table of Contents

Preface.. 1

EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian
Valerio Basile, Danilo Croce, Maria Di Maro, Lucia C. Passaro.. 5

Keynote Talk
Flattening the Curve of the COVID-19 Infodemic: These Evaluation Campaigns Can Help!
Preslav Nakov.. 15

Track "Affect, Hate, and Stance"

AMI: Automatic Misogyny Identification

AMI @ EVALITA2020: Automatic Misogyny Identification
Elisabetta Fersini, Debora Nozza and Paolo Rosso.. 21

UniBO @ AMI: A Multi-Class Approach to Misogyny and Aggressiveness Identification on Twitter Posts Using AlBERTo
Arianna Muti and Alberto Barrón-Cedeño.. 29

fabsam @ AMI: A Convolutional Neural Network Approach
Samuel Fabrizi.. 35

Jigsaw @ AMI and HaSpeeDe2: Fine-Tuning a Pre-Trained Comment-Domain BERT Model
Alyssa Lees, Jeffrey Sorensen and Ian Kivlichan.. 40

PoliTeam @ AMI: Improving Sentence Embedding Similarity with Misogyny Lexicons for Automatic Misogyny Identification in Italian Tweets
Giuseppe Attanasio and Eliana Pastor.. 48

MDD @ AMI: Vanilla Classifiers for Misogyny Identification
Samer El Abassi and Sergiu Nisioi 55

No Place For Hate Speech @ AMI: Convolutional Neural Network and Word Embedding for the Identification of Misogyny in Italian
Adriano dos S. R. da Silva and Norton T. Roman.. 60

ATE_ABSITA: Aspect Term Extraction and Aspect-Based Sentiment Analysis

ATE_ABSITA @ EVALITA2020: Overview of the Aspect Term Extraction and Aspect-based Sentiment Analysis Task
Lorenzo de Mattei, Graziella De Martino, Andrea Iovine, Alessio Miaschi, Marco Polignano and Giulia Rambelli.. 67

SentNA @ ATE_ABSITA: Sentiment Analysis of Customer Reviews Using Boosted Trees with Lexical and Lexicon-based Features
Francesco Mele, Antonio Sorgente and Giuseppe Vettigli 75

ghostwriter19 @ ATE_ABSITA: Zero-Shot and ONNX to Speed up BERT on Sentiment Analysis Tasks at EVALITA 2020
Mauro Bennici 80

App2Check @ ATE_ABSITA 2020: Aspect Term Extraction and Aspect-based Sentiment Analysis
Emanuele Di Rosa and Alberto Durante.. 85

HaSpeeDe: Hate Speech Detection

HaSpeeDe 2 @ EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task
Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti and Irene Russo 93

YNU_OXZ @ HaSpeeDe 2 and AMI : XLM-RoBERTa with Ordered Neurons LSTM for Classification Task at EVALITA 2020
Xiaozhi Ou and Hongling Li 102

DH-FBK @ HaSpeeDe2: Italian Hate Speech Detection via Self-Training and Oversampling
Elisa Leonardelli, Stefano Menini and Sara Tonelli 110

By1510 @ HaSpeeDe 2: Identification of Hate Speech for Italian Language in Social Media Data
Tao Deng, Yang Bai and Hongbing Dai 116

Venses @ HaSpeeDe2 & SardiStance: Multilevel Deep Linguistically Based Supervised Approach to Classification
Rodolfo Delmonte 121

UR NLP @ HaSpeeDe 2 at EVALITA 2020: Towards Robust Hate Speech Detection with Contextual Embeddings
Julia Hoffmann and Udo Kruschwitz 129

Fontana-Unipi @ HaSpeeDe2: Ensemble of transformers for the Hate Speech task at Evalita
Michele Fontana and Giuseppe Attardi 136

TheNorth @ HaSpeeDe 2: BERT-based Language Model Fine-tuning for Italian Hate Speech Detection
Eric Lavergne, Rajkumar Saini, György Kovács and Killian Murphy 142

UO @ HaSpeeDe2: Ensemble Model for Italian Hate Speech Detection
Mariano Jason Rodriguez Cisnero and Reynier Ortega Bueno 148

No Place For Hate Speech @ HaSpeeDe 2: Ensemble to Identify Hate Speech in Italian
Adriano dos S. R. da Silva and Norton T. Roman 154

Svandiela @ HaSpeeDe: Detecting Hate Speech in Italian Twitter Data with BERT
Svea Klaus, Anna-Sophie Bartle and Daniela Rossmann 159

CHILab @ HaSpeeDe 2: Enhancing Hate Speech Detection with Part-of-Speech Tagging
Giuseppe Gambino and Roberto Pirrone 165

Montanti @ HaSpeeDe2 EVALITA 2020: Hate Speech Detection in online contents
Elia Bisconti, Matteo Montagnani 171

SardiStance: Stance Detection

SardiStance @ EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets
Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti and Paolo Rosso 177

UNITOR @ Sardistance2020: Combining Transformer-based Architectures and Transfer Learning for Robust Stance Detection
Simone Giorgioni, Marcello Politi, Samir Salman, Roberto Basili and Danilo Croce 187

ghostwriter19 @ SardiStance: Generating New Tweets to Classify SardiStance EVALITA 2020 Political Tweets
Mauro Bennici 193

QMUL-SDS @ SardiStance: Leveraging Network Interactions to Boost Performance on Stance Detection using Knowledge Graphs
Rabab Alkhalifa and Arkaitz Zubiaga 198

DeepReading @ SardiStance 2020: Combining Textual, Social and Emotional Features
Maria S. Espinosa, Rodrigo Agerri, Alvaro Rodrigo and Roberto Centeno 204

TextWiller @ SardiStance, HaSpeede2: Text or Con-text? A Smart Use of Social Network Data in Predicting Polarization
Federico Ferraccioli, Andrea Sciandra, Mattia Da Pont, Paolo Girardi, Dario Solari and Livio Finos 210

UninaStudents @ SardiStance: Stance Detection in Italian Tweets - Task A
Maurizio Moraca, Gianluca Sabella and Simone Morra 215

SSN NLP @ SardiStance : Stance Detection from Italian Tweets using RNN and Transformers
Kayalvizhi S, Thenmozhi D and Aravindan Chandrabose 220

SSNCSE-NLP @ EVALITA2020: Textual and Contextual Stance Detection from Tweets Using Machine Learning Approach
Bharathi B, Bhuvana J and Nitin Nikamanth Appiah Balaji 224

Track “Creativity and Style”

CHANGE-IT: Style Transfer

CHANGE-IT @ EVALITA 2020: Change Headlines, Adapt News, GEnerate
Lorenzo De Mattei, Michele Cafagna, Felice Dell'Orletta, Malvina Nissim and Albert Gatt 235

TAG-it: Topic, Age and Gender Prediction

TAG-it @ EVALITA2020: Overview of the Topic, Age, and Gender Prediction Task for Italian
Andrea Cimino, Felice Dell'Orletta and Malvina Nissim 243

UO_4to @ TAG-it 2020: Ensemble of Machine Learning Methods
Maria Fernanda Artigas Herold and Daniel Castro Castro 252

UOBIT @ TAG-it: Exploring a Multi-faceted Representation for Profiling Age, Topic and Gender in Italian Texts.
Roberto Labadie Tamayo, Daniel Castro Castro and Reynier Ortega Bueno 256

ItaliaNLP @ TAG-IT: UmBERTo for Author Profiling at TAG-it 2020
Daniela Occhipinti, Andrea Tesei, Maria Iacono, Carlo Aliprandi and Lorenzo De Mattei 263

Track “Semantics and Multimodality”

DANKMEMES: Multimodal Artefacts Recognition

DANKMEMES @ EVALITA 2020: The Memeing of Life: Memes, Multimodality and Politics
Martina Miliani, Giulia Giorgi, Ilir Rama, Guido Anselmi and Gianluca E. Lebani 275

SNK @ DANKMEMES: Leveraging Pretrained Embeddings for Multimodal Meme Detection
Stefano Fiorucci 284

UPB @ DANKMEMES: Italian Memes Analysis - Employing Visual Models and Graph Convolutional Networks for Meme Identification and Hate Speech Detection
George-Alexandru Vlad, George-Eduard Zaharia, Dumitru-Clementin Cercel and Mihai Dascalu 288

ArchiMeDe @ DANKMEMES: A New Model Architecture for Meme Detection
Jinen Setpal and Gabriele Sarti 294

UNITOR @ DANKMEME: Combining Convolutional Models and Transformer-based architectures for accurate MEME management
Claudia Breazzano, Edoardo Rubino, Danilo Croce and Roberto Basili 301

CONcreTEXT: Concreteness in Context

CONcreTEXT @ EVALITA2020: The Concreteness in Context Task
Lorenzo Gregori, Maria Montefinese, Daniele P. Radicioni, Andrea Amelio Ravelli and Rossella Varvara 311

ANDI @ CONcreTEXT: Predicting Concreteness in Context for English and Italian using Distributional Models and Behavioural Norms
Armand Stefan Rotaru 319

CAPISCO @ CONcreTEXT 2020: (Un)supervised Systems to Contextualize Concreteness with Norming Data
Alessandro Bondielli, Gianluca E. Lebani, Lucia C. Passaro and Alessandro Lenci 327

KonKretiKa @ CONcreTEXT: Computing Concreteness Indexes with Sigmoid Transformation and Adjustment for Context
Yulia Badryzlova 334

Ghigliottin-AI: Evaluating Artificial Players for the Language Game “La Ghigliottina”

Ghigliottin-AI@EVALITA2020: Evaluating Artificial Players for the Language Game ``La Ghigliottina"
Pierpaolo Basile, Marco Lovetere, Johanna Monti, Antonio Pascucci, Federico Sangati and Lucia Siciliani 345

“Il Mago della Ghigliottina” @ GhigliottinAI: When Linguistics meets Artificial Intelligence
Federico Sangati, Antonio Pascucci and Johanna Monti 349

GUL.LE.VER @ GhigliottinAI: A Glove based Artificial Player to Solve the Language Game “La Ghigliottina”
Nazareno de Francesco 356

PRELEARN: Prerequisite Relation Learning

PRELEARN @ EVALITA 2020: Overview of the Prerequisite Relation Learning Task for Italian
Chiara Alzetta, Alessio Miaschi, Felice Dell'Orletta, Frosina Koceva and Ilaria Torre.......... 363

B4DS @ PRELEARN: Ensemble Method for Prerequisite Learning
Giovanni Puccetti, Luis Bolanos, Filippo Chiarello and Gualtiero Fantoni 371

UNIGE_SE @ PRELEARN: Utility for Automatic Prerequisite Learning from Italian Wikipedia (short paper)
Alessio Moggio and Andrea Parizzi 376

NLP-CIC @ PRELEARN: Mastering Prerequisites Relations, from Handcrafted Features to Embeddings
Jason Angel and Segun Aroyehun 381

Track “Time and Diachrony”

DaDoEval: Dating Documents

DaDoEval @ EVALITA 2020: Same-Genre and Cross-Genre Dating of Historical Documents
Stefano Menini, Giovanni Moretti, Rachele Sprugnoli and Sara Tonelli.......... 391

matteo-brv @ DaDoEval: An SVM-based Approach for Automatic Document Dating
Matteo Brivio 398

rmassidda @ DaDoEval: Document Dating Using Sentence Embeddings at EVALITA 2020
Riccardo Massidda.......... 403

DIACR-Ita: Diachronic Lexical Semantics

DIACR-Ita @ EVALITA2020: Overview of the EVALITA2020 Diachronic Lexical Semantics (DIACR-Ita) Task
Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti and Rossella Varvara 411

University of Padova @ DIACR-Ita
Benyou Wang, Emanuele Di Buccio and Massimo Melucci.......... 420

UWB @ DIACR-Ita: Lexical Semantic Change Detection with CCA and Orthogonal Transformation
Ondřej Pražák, Pavel Přibáň, Stephen Taylor 426

QMUL-SDS @ DIACR-Ita: Evaluating Unsupervised Diachronic Lexical Semantics Classification in Italian
Rabab Alkhalifa, Adam Tsakalidis, Arkaitz Zubiaga and Maria Liakata 432

CL-IMS @ DIACR-Ita: Volente o Nolente: BERT is still not Outperforming SGNS on Semantic Change Detection
Severin Laicher, Gioia Baldissin, Enrique Castaneda, Dominik Schlechtweg and Sabine Schulte Im Walde 438

OP-IMS @ DIACR-Ita: Back to the Roots: SGNS+OP+CD still Rocks Semantic Change Detection
Jens Kaiser, Dominik Schlechtweg and Sabine Schulte Im Walde 444

UNIMIB @ DIACR-Ita: Aligning Distributional Embeddings with a Compass for Semantic Change Detection in the Italian Language
Federico Belotti, Federico Bianchi and Matteo Palmonari 451

NLP-CIC @ DIACR-Ita: POS and Neighbor Based Distributional Models for Lexical Semantic Change in Diachronic Italian Corpora
Jason Angel 456

Track "New Challenges in Long-standing Tasks"

AcCompl-it: Acceptability & Complexity evaluation

AcCompl-it @ EVALITA2020: Overview of the Acceptability & Complexity Evaluation Task for Italian
Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Simonetta Montemagni, Giulia Venturi and Roberto Zamparelli 465

UmBERTo-MTSA @ AcCompl-It: Improving Complexity and Acceptability Prediction with Multi-task Learning on Self-Supervised Annotations
Gabriele Sarti 473

Venses @ AcCompl-It: Computing Complexity vs Acceptability with a Constituent Trigram Model and Semantics
Rodolfo Delmonte 479

KIPoS: Part-of-speech Tagging on Spoken Language

KIPoS @ EVALITA2020: Overview of the Task on KIParla Part of Speech Tagging
Cristina Bosco, Silvia Ballarè, Massimo Cerruti, Eugenio Goria and Caterina Mauri 489

UniBO @ KIPoS: Fine-tuning the Italian "BERTology" for PoS-tagging Spoken Data
Fabio Tamburini 497

UniBA @ KIPoS: A Hybrid Approach for Part-of-Speech Tagging
Giovanni Luca Izzi and Stefano Ferilli 501

KLUMSy @ KIPoS: Experiments on Part-of-Speech Tagging of Spoken Italian
Thomas Proisl and Gabriella Lapesa 507

Preface to the EVALITA 2020 Proceedings [*]

Welcome to EVALITA 2020! EVALITA is the evaluation campaign of Natural Language Processing and Speech Tools for Italian. EVALITA is an initiative of the Italian Association for Computational Linguistics (AILC, `http://www.ai-lc.it`) and it is endorsed by the Italian Association for Artificial Intelligence (AIxIA, `http://www.aixia.it`) and the Italian Association for Speech Sciences (AISV, `http://www.aisv.it`).

This volume includes the reports of both task organisers and participants to all of the EVALITA 2020 challenges. In the 2020 edition, we coordinated the organization of 14 different tasks belonging to five research areas, being: (i) *Affect, Hate, and Stance*, (ii) *Creativity and Style*, (iii) *New Challenges in Long-standing Tasks*, (iv) *Semantics and Multimodality*, *Time and Diachrony*.

The volume is opened by an overview to the EVALITA 2020 campaign, in which we describe the tasks, provide statistics on the participants and task organizers as well as our supporting sponsors. The abstract of the keynote speech made by Preslav Nakov titled *"Flattening the Curve of the COVID-19 Infodemic: These Evaluation Campaigns Can Help!"* is also included in this collection.

Due to the 2020 COVID-19 pandemic, the traditional workshop was held online, where several members of the Italian NLP Community presented the results of their research. Despite the circumstances, the workshop represented an occasion for all participants from both academic institutions and private companies to disseminate their work and results and to share ideas through online sessions dedicated to each task and a general discussion during the plenary event.

We carried on with the tradition of the "Best system across tasks" award. As in 2018, it represented an incentive for students, IT developers and researchers to push the boundaries of the state of the art by facing tasks in new ways, even if not winning.

We would like to thank our sponsors Amazon Science (`https://www.amazon.science/`), Bnova (`https://www.bnova.it/`), CELI (`https://www.celi.it/`), European Language Resources Association (ELRA, `http://elra.info/en/`) and Google Research (`https://research.google/`) for their support to the virtual event and to the prize for the best system award.

Our gratitude goes also to University of Turin for hosting the online events. In addition, we sincerely thank the Best System across Tasks committee for providing their expertise and experience. Moreover, we acknowledge the AILC Board members for their trust and support, our mentor Nicole Novielli and all the chairs of the 2018 edition, who helped us during all the organization process of EVALITA 2020.

We warmly thank our invited speaker Preslav Nakov for having shared his knowledge and insights with his talk.

Last but not least, we would like to thank all the task organizers and participants, who made this edition special with their enthusiasm and creativity.

December 2020

Valerio Basile
Danilo Croce
Maria Di Maro
Lucia C. Passaro

---

[*] Originally published online by CEUR (CEUR-WS.org, ISSN 1613-0073, Vol-2765, urn:nbn:de:0074-2765-4)

Chairs

Valerio Basile, University of Turin
Danilo Croce, University of Rome “Tor Vergata”
Maria Di Maro, University of Naples “Federico II”
Lucia Passaro, University of Pisa
Advisor: Nicole Novielli, University of Bari “Aldo Moro”

Steering Committee

Chiara Alzetta, Università degli Studi di Genova
Guido Anselmi, University of Milan
Silvia Ballarè, Università degli Studi di Bologna
Pierpaolo Basile, University of Bari Aldo Moro
Cristina Bosco, University of Torino
Dominique Brunato, Istituto di Linguistica Computazionale “Antonio Zampolli” - CNR
Michele Cafagna, University of Pisa
Annalina Caputo, Dublin City University
Tommaso Caselli, University of Groningen, Netherlands
Pierluigi Cassotti, University of Bari Aldo Moro
Massimo Cerruti, University of Torino
Cristiano Chesi, NETS-IUSS
Alessandra Teresa Cignarella, University of Turin / Universitat Politècnica de València
Andrea Cimino, Istituto di Linguistica Computazionale “Antonio Zampolli” - CNR
Gloria Comandini, Università degli Studi di Trento
Graziella De Martino, University of Bari Aldo Moro
Lorenzo de Mattei, University of Pisa
Felice Dell’Orletta, Istituto di Linguistica Computazionale “Antonio Zampolli” - CNR
Elisa Di Nuovo, University of Torino
Elisabetta Fersini, University of Milano-Bicocca
Simona Frenda, University of Torino / Universitat Politècnica de València
Albert Gatt, University of Malta
Giulia Giorgi, University of Milan
Eugenio Goria, University of Torino
Lorenzo Gregori, University of Florence
Andrea Iovine, University of Bari Aldo Moro
Frosina Koceva, Università di Genova
Mirko Lai, University of Torino
Gianluca E. Lebani, Ca’ Foscari University of Venice
Marco Lovetere, Ghigliottiniamo
Caterina Mauri, Università degli Studi di Bologna
Stefano Menini, Fondazione Bruno Kessler
Alessio Miaschi, University of Pisa
Martina Miliani, University for Foreigners of Siena / University of Pisa
Maria Montefinese, University of Padua
Simonetta Montemagni, Istituto di Linguistica Computazionale “Antonio Zampolli” - CNR
Johanna Monti, University of Naples ”L’Orientale”
Giovanni Moretti, Università Cattolica del Sacro Cuore
Malvina Nissim, Faculty of Arts - CLCG University of Groningen
Debora Nozza, Bocconi University
Antonio Pascucci, L’Orientale University of Naples

Viviana Patti, University of Torino
Marco Polignano, University of Bari Aldo Moro
Daniele P. Radicioni, University of Turin
Ilir Rama, University of Milan
Giulia Rambelli, University of Pisa
Andrea Amelio Ravelli, ILC-CNR of Pisa
Paolo Rosso, Universitat Politècnica de València
Irene Russo, Istituto di Linguistica Computazionale “Antonio Zampolli” - CNR
Federico Sangati, Okinawa Institute of Science and Technology Graduate University
Manuela Sanguinetti, University of Torino
Lucia Siciliani, University of Bari Aldo Moro
Rachele Sprugnoli, Università Cattolica del Sacro Cuore
Marco Stranisci, Associazione Acmos
Sara Tonelli, Fondazione Bruno Kessler
Ilaria Torre, University of Genova
Rossella Varvara, University of Florence
Giulia Venturi, Istituto di Linguistica Computazionale “Antonio Zampolli” - CNR
Roberto Zamparelli, Università degli Studi di Trento

Best System Award committee

Giuseppe Attardi, University of Pisa
Francesca Chiusaroli, University of Macerata
Giuseppe Castellucci, Amazon Science
Gloria Gagliardi, “L’Orientale” University of Naples
Nicole Novielli, University of Bari “Aldo Moro”

Website

Manuela Speranza, Fondazione Bruno Kessler

EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian

Valerio Basile
University of Turin
`valerio.basile@unito.it`

Maria Di Maro
University of Naples "Federico II"
`maria.dimaro2@unina.it`

Danilo Croce
University of Rome "Tor Vergata"
`croce@info.uniroma2.it`

Lucia C. Passaro
University of Pisa
`lucia.passaro@fileli.unipi.it`

1 Introduction

The Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA) is the biennial initiative aimed at promoting the development of language and speech technologies for the Italian language. EVALITA is promoted by the Italian Association of Computational Linguistics (AILC)[1] and it is endorsed by the Italian Association for Artificial Intelligence (AIxIA)[2] and the Italian Association for Speech Sciences (AISV)[3].

EVALITA provides a shared framework where different systems and approaches can be scientifically evaluated and compared with each other with respect to a large variety of tasks, suggested and organized by the Italian research community. The proposed tasks represent scientific challenges where methods, resources, and systems can be tested against shared benchmarks representing linguistic open issues or real world applications, possibly in a multilingual and/or multi-modal perspective. The collected data sets provide big opportunities for scientists to explore old and new problems concerning NLP in Italian as well as to develop solutions and to discuss the NLP-related issues within the community. Some tasks are traditionally present in the evaluation campaign, while others are completely new.

This paper introduces the tasks proposed at EVALITA 2020 and provides an overview to the participants and systems whose descriptions and obtained results are reported in these Proceedings[4]. The EVALITA 2020 edition, held online on December 17^{th} due to the COVID-19 pandemic, counts 14 different tasks. In particular, the selected tasks are grouped in five research areas (tracks) according to their objective and characteristics, namely (i) *Affect, Hate, and Stance*, (ii) *Creativity and Style*, (iii) *New Challenges in Long-standing Tasks*, (iv) *Semantics and Multimodality*, (v) *Time and Diachrony*.

This edition was highly participated, with 51 groups whose participants have affiliation in 14 countries. Although EVALITA is generally promoted and targeted to the Italian research community, this edition saw an international participation, also thanks to the fact that several Italian researchers working in different countries contributed to the organization of the tasks or participated in them as authors.

This overview is organized as follows: in Section 2 a brief description of the tasks belonging to the various areas is reported. Section 3 discusses the participation to the workshop referred to several aspects, from the research area, to the affiliation of authors. Section 4 describes the criteria used to assign the best system across tasks award, made by an ad-hoc committee starting from the suggestions of task organizers and reviewers. Finally, section 5 points out on both the obtained results and on the future of the workshop.

---

[1] `http://www.ai-lc.it`

[2] `http://www.aixia.it`

[3] `http://www.aisv.it`

[4] The presentations of these works are publicly available at `https://vimeo.com/showcase/evalita2020` . All videos are also grouped according to different tasks at `https://vimeo.com/user125537954/albums`

2 EVALITA 2020 Tracks and tasks

In the 2020 edition of EVALITA, 14 different tasks were proposed, peer-reviewed and accepted. Data were produced by the task organizers and made available to the participants. For the future availability of this data we are going to release them on GitHub[5], in accordance to the terms and conditions of the respective data sources. Such a repository will also reference alternative repositories managed by the task organizers. The tasks of EVALITA 2020 are grouped according to the following tracks:

Affect, Hate, and Stance

AMI - Automatic Misogyny Identification (Fersini et al., 2020). This shared task is aimed at automatically identifying misogynous content in Twitter for the Italian language. In particular, the AMI challenge is focused on: (1) recognizing misogynous and aggressive messages and (2) discriminating misogynistic contents from the non-misogynistic ones, while guaranteeing the fairness of the model.

ATE_ABSITA - Aspect Term Extraction and Aspect-Based Sentiment Analysis (De Mattei et al., 2020b). A task on Aspect Term Extraction (ATE) and Aspect-Based Sentiment Analysis (ABSA). The task is approached as a cascade of three subtasks: Aspect Term Extraction (ATE), Aspect-based Sentiment Analysis (ABSA) and Sentiment Analysis (SA).

HaSpeeDe - Hate Speech Detection (Sanguinetti et al., 2020). A rerun of the shared task on hate speech detection at the message level on Italian social media texts proposed for the first time in 2018 for the EVALITA evaluation campaign. The main task is a binary hate speech detection task, one in-domain and one out-of-domain. On the same data provided for the main task, the topics of stereotypes in communication and nominal utterances are investigated by of two pilot tasks.

SardiStance - Stance Detection (Cignarella et al., 2020). The goal of the task is to detect the stance of the author towards the target "Sardines movement" in Italian tweets. Two subtasks model (A) Textual Stance Detection and (B) Contextual Stance Detection. Both the subtasks consist on a three-class (in favour, against, neutral) classification problem based on textual information only (A) or on the text provided with additional information about the author and the post of the tweet.

Creativity and Style

CHANGE-IT - Style Transfer (De Mattei et al., 2020a). The first natural language generation task for Italian. Change-IT focuses on style transfer performed on the headlines of two Italian newspapers at opposite ends of the political spectrum. Specifically, the goal is to "translate" the headlines from a style to another.

TAG-it - Topic, Age and Gender Prediction (Cimino et al., 2020). TAG-IT is a profiling task for Italian. It is a follow-up of the GxG task organised in the context of EVALITA 2018. The task is aimed at profiling along with three dimensions (Gender, Age, and Topic). Authors propose several subtasks where participants are asked to predict one or more classes starting from the others.

Semantics and Multimodality

CONcreTEXT - Concreteness in Context (Gregori et al., 2020). The task focuses on automatic assignment of concreteness values to words in context for the Italian and English language. Participants are required to develop systems able to rate the concreteness of a target word in a sentence on a scale from 1 (for fully abstract) to 5 (for maximally concrete).

DANKMEMES - Multimodal Artefacts Recognition (Miliani et al., 2020). The first multimodal task for Italian. The goal of the task is to deal with Italian memes considering textual and visual cues together. Providing a corpus of memes on the 2019 Italian Government Crisis, DANKMEMES features three subtasks: A) Meme Detection, B) Hate Speech Identification, and C) Event Clustering.

---

[5] `https://github.com/evalita2020`

Ghigliottin-AI - Evaluating Artificial Players for the Language Game "La Ghigliottina" (Basile et al., 2020b). The task challenges researchers to develop a system able to defeat human players at the language game "La Ghigliottina", which represents one of the most followed and appreciated quiz games in Italy.

PRELEARN - Prerequisite Relation Learning (Alzetta et al., 2020). The task is devoted to automatically inferring prerequisite relations from educational texts. The task consists in classifying prerequisite relations between pairs of concepts distinguishing between prerequisite pairs and non-prerequisite pairs.

Time and Diachrony

DaDoEval - Dating Documents (Menini et al., 2020). The task focuses on assigning a temporal span to a document, by recognising when a document was issued. A first one coarse-grained classification subtask, participants are asked to provide a document with a class encoding the historical period (5 classes). The second Fine-grained classification task requires to attribute documents with a temporal slice of 5 years.

DIACR-Ita - Diachronic Lexical Semantics (Basile et al., 2020a). The first task on automatic detection of lexical and semantic shift for Italian. The task challenges participants to develop systems that can automatically detect if a given word has changed its meaning over time, given contextual information from corpora.

New Challenges in Long-standing Tasks

AcCompl-it- Acceptability & Complexity evaluation (Brunato et al., 2020). The task is aimed at developing and evaluating methods to classify Italian sentences according to Acceptability and Complexity. Given a set of sentences, two independent subtasks focus on predicting their acceptability and complexity rate.

KIPoS - Part-of-speech Tagging on Spoken Language (Bosco et al., 2020). The first task on Part of Speech tagging of spoken language held at EVALITA. Benefiting from the KIParla corpus, a resource of transcribed spoken Italian, the task provides three evaluation exercises focused on formal versus informal spoken texts.

3 Participation

EVALITA 2020 attracted the interest of a large number of researchers from academia and industry, for a total of 51 teams composed of about 130 individuals participating in one or more of the 14 proposed tasks. After the evaluation period, 58 system descriptions were submitted (reported in these proceedings), i.e., a 70% percentage increase with respect to the previous EVALITA edition (Caselli et al., 2018).

Moreover, task organizers allowed participants to submit more than one system result (called runs), for a total of 240 submitted runs. Table 1 shows the different tracks and tasks along with the number of participating teams and submitted runs. The data reported in the table is based on information provided by the task organizers at the end of the evaluation process. Such data represents an overestimation with respect to the systems described in the proceedings. The trends are similar, but there are differences due to groups participating in more than a task, and groups that have not produced a system report.

Differently from the previous EVALITA editions, the organizers were discouraged from distinguishing the submissions between unconstrained and constrained runs[6]. The rationale for this decision is that the recent spread and extensive use of pre-trained word embedding representations, especially as a strategy to initialize Neural Network architectures, challenges this distinction at its very heart. Participation was quite imbalanced across different tracks and tasks, as reported in Figure 1: each rectangle represents a task whose size reflects the number of participants, while the color indicated the corresponding track.

---

[6] A system is considered *constrained* when using the provided training data only; on the contrary, it is considered *unconstrained* when using additional material to augment the training dataset or to acquire additional resources.

TRACK	TASK	TEAMS	RUNS
Affect, Hate, and Stance	AMI	8	31
	ATE_ABSITA	3	8
	HaSpeeDe	14	27
	SardiStance	12	36
Creativity and Style	CHANGE-IT	0	0
	TAG-it	3	20
New Challenges in Long-standing Tasks	AcCompl-it	2	6
	KIPoS	3	14
Semantics and Multimodality	CONcreTEXT	4	15
	DANKMEMES	5	15
	Ghigliottin-AI	2	2
	PRELEARN	3	14
Time and Diachrony	DaDoEval	2	16
	DIACR-Ita	9	36

Table 1: Number of participating teams and number of runs organized by track and task. The data reported is an overestimation with respect to the systems described in the proceedings (e.g. teams participating in more than a task are counted according to the number of tasks they participated in).

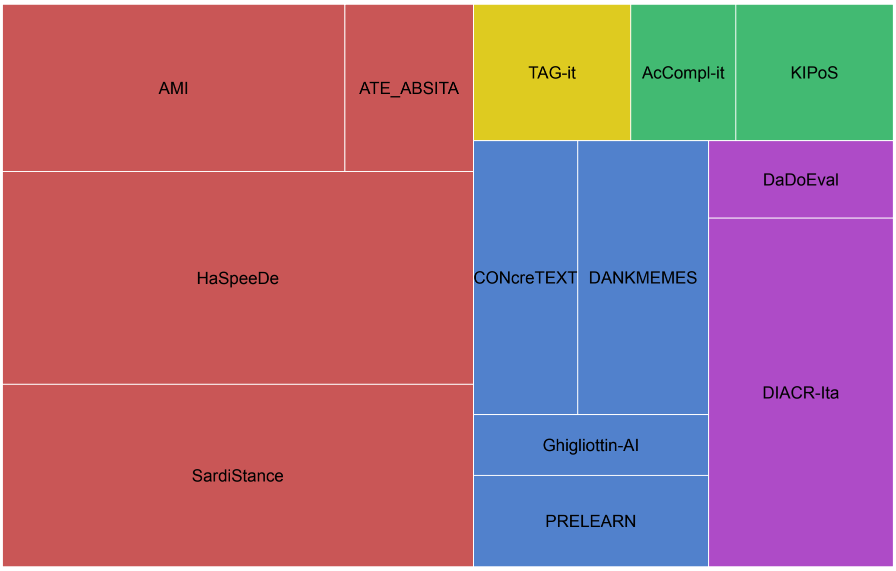


Figure 1: Number of participating teams organized by track (color) and task. The red color is adopted for the track "*Affect, Hate, and Stance*", the yellow color for "*Creativity and Style*", green for "*New Challenges in Long-standing Tasks*", blue for "*Semantics and Multimodality*" and purple for "*Time and Diachrony*".

In line with the previous editions of EVALITA, the track "*Affect, Creativity and Style*" covers about half of the total in terms of participating teams. On the one hand, this demonstrates the well-known interest of the NLP community for Social Media platforms and user-generated content. On the other hand, we report a better balance with respect to the 2018 edition, where about 80% of the teams participated in similar tracks ("*Affect, Creativity and style*" and "*Hate Speech*", which have been merged in this edition). Another significant number of teams participated to the "*Semantics and Multimodality*" and "*Time and Diachrony*" tracks, while the other tracks where less participated. Unfortunately, no team participated to the *CHANGE-IT* task, mainly due to the complexity of the task.

In addition to being widely participated, the over 180 proceedings authors, including both participants and task organizers, have affiliation in 18 countries, with the 64% from Italy and the 36% of participants from Institutions and companies abroad. The group of the 59 task organizers have affiliations in 6 countries (90% from Italy while 10% from Institutions and companies abroad). The gender distribution is highly balanced, with 30 females and males.

4 Award: Best System Across Tasks

In line with the previous edition, we confirmed the award to the best system across-task. The award was introduced with the aim of fostering student participation to the evaluation campaign and to the workshop. EVALITA received sponsorship funding from from Amazon Science, Bnova s.r.l., CELI s.r.l., the European Language Resources Association (ELRA) and Google Research.

A committee of 5 members was asked to choose the best system across tasks. Four of the five members come from academia while the last one is from industry. The composition of the committee is balanced with respect to the level of seniority as well as for their academic background (computer science-oriented vs. humanities-oriented). In order to select a short list of candidates, the task organizers were invited to propose up to two candidate systems participating to their tasks (not necessarily top ranking). The committee was provided with the list of candidate systems and the criteria for eligibility, based on:

- *novelty* with respect to the state of the art;
- *originality*, in terms of identification of new linguistic resources, identification of linguistically motivated features, and implementation of a theoretical framework grounded in linguistics;
- *critical insight*, paving the way to future challenges (deep error analysis, discussion on the limits of the proposed system, discussion of the inherent challenges of the task);
- *technical soundness* and *methodological rigor*.

We collected 10 system nominations from the organizers of 7 tasks from across all tracks. The candidate systems are authored by 20 authors, among whom 12 are students, either at the master's or PhD level. The award recipient(s) will be announced during the final EVALITA workshop, during the plenary session, held online.

5 Final Remarks

A record number of 14 tasks were organized at EVALITA 2020: some of them were revivals of tasks in the past editions (such as *Hate Speech Detection* or *Part-of-Speech Tagging*), while others were completely new (such as the ones involving *Meme Processing* or *Stance Detection*), and were greeted with great enthusiasm by the NLP community.

In this edition, the topics of Affect and Semantics were confirmed as two of the most interesting and thriving ones, both in the number of organized tasks and actual participants. In any case, almost all tasks involved the analysis of written texts. In fact, although the KIPoS task considered transcriptions of spoken Italian utterances, no speech related tasks was proposed.

Anyways, tasks concerning new problems and modalities have been proposed, such as the analysis of memes, and two tasks oriented to the problem of time and diachrony. Moreover, this edition saw an increase in tasks related to creativity and style, despite the fact that one one such tasks, namely CHANGE-it, had no participation, probably due to its complexity and the lack of specific resources for the task in the Italian community. Another task that received a rather low number of submission due to its complexity is GhigliottinAI. Despite being a rather simple word-correlation problem by itself, it required complex modelling of language and semantics to beat the challenge. A very interesting innovation for this task was the evaluation framework, based on APIs, via a Remote Evaluation Server (RES). In general, the most participated tasks have been those by which the linguistic problem could be modelled as a direct classification or regression task.

The competition attracted a record number of participating teams from academia and industry, for a total of 51 teams and more than 180 authors with affiliations in 18 countries. Hopefully, this means that EVALITA is becoming more and more popular also with foreign contributors, and it is becoming an international workshop. First of all, this success confirms the beneficial impact of the organization of the evaluation period based on non-overlapping windows (adopted from EVALITA 2018) in order to help those who want to participate in more than one task. Moreover, we speculate that the technological advancements and ease of use of existing open-source libraries for machine learning and natural language processing improved the accessibility to the tasks, even for master students. In fact, we noticed an increase in the participation of students, that contributed with state-of-the-art solutions to the tasks. We can argue that the spread of frameworks such as PyTorch and Keras, together with pre-trained, off-the-shelf language models, lowered the set-up costs to deal with complex NLP tasks. In general, we noticed that most of the best systems are based on neural approaches. Among them, BERT or similar Transformer-based architectures achieved the best results: more specifically, at least in 11 out of 14 tasks best results (in at least one sub-task) were obtained by neural architectures based on or combined with Transformers.

We are confident that the positive trends observed in this edition, concerning the participation and the proliferation of tasks, has not yet reached a plateau. It would be auspicabile, among other aspects, to see more tasks involving challenging settings such as, for example, multi-modal or multi-lingual analysis involving Italian, in future EVALITA 2020 editions. Several areas represent fertile ground to organize future tasks, such as domain adaptation (which was considered in previous editions of EVALITA), or few-shot learning to support the definition of robust systems in challenging low-resource settings. Finally, we believe in the importance of defining more structured tasks involving real applications to challenge the Italian community, e.g., Question Answering or Dialogue Agents.

Acknowledgments

We would like to thank our sponsors Amazon Science[7], Bnova[8], CELI[9], European Language Resources Association (ELRA)[10] and Google Research[11] for their support to the virtual event and to the best-system across task award.

Moreover, we gratefully acknowledge the members of the AILC board for their trust and support, our EVALITA advisor Nicole Novielli and all the chairs of the 2018 edition, who helped us during the organization process of EVALITA 2020. In addition, we sincerely thank the Best System across Tasks committee for providing their expertise and experience.

Finally, we know that EVALITA 2020 would not have been possible without the tireless effort, enthusiasm, and originality of the task organizers, the colleagues and friends involved in running them, and all the participants who contributed to make the workshop a success.

References

Chiara Alzetta, Alessio Miaschi, Felice Dell'Orletta, Frosina Koceva, and Ilaria Torre. 2020. PRELEARN@EVALITA2020: Overview of the prerequisite relation learning task for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020a. DIACR-Ita@EVALITA2020: Overview of the evalita2020 diachronic lexical semantics (diacr-ita) task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Pierpaolo Basile, Marco Lovetere, Johanna Monti, Antonioa Pascucci, Federico Sangati, and Lucia Siciliani. 2020b. Ghigliottin-AI@EVALITA2020: Evaluating artificial players
for the language game "la ghigliottina". In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Cristina Bosco, Silvia Ballarè, Massimo Cerruti, Eugenio Goria, and Caterina Mauri. 2020. KIPoS@EVALITA2020:overview of the task on kiparla part of speech tagging. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

---

[7] `https://www.amazon.science/`

[8] `https://www.bnova.it`

[9] `https://www.celi.it/`

[10] `http://elra.info/en/`

[11] `https://research.google/`

Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Simonetta Montemagni, Giulia Venturi, and Roberto Zamparelli. 2020. AcCompl-it@EVALITA2020: Overview of the acceptability & complexity evaluation task for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

T. Caselli, N. Novielli, V. Patti, and P. Rosso. 2018. EVALITA 2018: Overview of the 6th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)* , Turin, Italy. CEUR.org.

Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. SardiStance@EVALITA2020: Overview of the task on stance detection in italian tweets. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Andrea Cimino, Felice Dell'Orletta, and Malvina Nissim. 2020. TAG-it@EVALITA2020: Overview of the topic, age, and gender prediction task for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Lorenzo De Mattei, Michele Cafagna, Felice Dell'Orletta, Malvina Nissim, and Albert Gatt. 2020a. CHANGE-IT@EVALITA2020: Change headlines, adapt news, generate. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Lorenzo De Mattei, Graziella De Martino, Andrea Iovine, Alessio Miaschi, Marco Polignano, and Giulia Rambelli. 2020b. ATE_ABSITA@EVALITA2020: Overview of the aspect term extraction and aspect-based sentiment analysis task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020. AMI@EVALITA2020: Automatic misogyny identification. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Lorenzo Gregori, Maria Montefinese, Daniele P. Radicioni, Andrea Amelio Ravelli, and Rossella Varvara. 2020. CONCRETEXT@EVALITA2020: The concreteness in context task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Stefano Menini, Giovanni Moretti, Rachele Sprugnoli, and Sara Tonelli. 2020. DaDoEval@EVALITA2020: Same-genre and cross-genre dating of historical documents. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Martina Miliani, Giulia Giorgi, Ilir Rama, Guido Anselmi, and Gianluca E. Lebani. 2020. DANKMEMES@EVALITA2020: The memeing of life: memes, multimodality and politics. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the evalita 2020 hate speech detection task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

KEYNOTE TALK

Flattening the Curve of the COVID-19 Infodemic: These Evaluation Campaigns Can Help!

Preslav Nakov
Qatar Computing Research Institute, HBKU
`pnakov@hbku.edu.qa`

The World Health Organization acknowledged that "*The 2019-nCoV outbreak and response has been accompanied by a massive 'infodemic' ... that makes it hard for people to find trustworthy sources and reliable guidance when they need it*." While fighting this infodemic is typically thought of in terms of factuality, the problem is much broader as malicious content includes not only "fake news", rumors, and conspiracy theories, but also promotion of fake cures, panic, racism, xenophobia, and mistrust in the authorities, among others. Thus, we argue for the need of a holistic approach combining the perspectives of journalists, fact-checkers, policymakers, social media platforms, and society as a whole, and we present our initial work in this direction.

We further discuss evaluation campaigns at CLEF and SemEval that feature relevant tasks (not necessarily focusing on COVID-19). One relevant evaluation campaign is the CLEF CheckThat! Lab, which has focused on tasks that make human fact-checkers more productive: spotting check-worthy claims (in tweets, political debates, and speeches), determining whether these claims have been previously fact-checked, retrieving relevant pages and passages, and finally, making a prediction about the factuality of the claims. There have been also a number of relevant SemEval tasks related to factuality, e.g., on rumor detection and verification in social media, on fact-checking in community question answering forums, and on stance detection. Other relevant SemEval tasks have looked beyond factuality, focusing on intent, e.g., on offensive language detection in social media, as well as on spotting the use of propaganda techniques (e.g., appeal to emotions, fear, prejudices, logical fallacies, etc.) in the news and in memes (text + image). Of course, relevant tasks can be also found beyond CLEF and SemEval; most notably, this includes FEVER and the Fake News Challenge.

Finally, we demonstrate two systems developed at the Qatar Computing Research Institute, HBKU, to address some of the above challenges: one reflecting the proposed holistic approach, and one on fine-grained propagada detection. The latter system, Prta (https://www.tanbih.org/prta), was featured at ACL-2020 with a Best Demo Award (Honorable Mention).

Short Bio. Dr. Preslav Nakov is a Principal Scientist at the Qatar Computing Research Institute (QCRI), HBKU. His research interests include computational linguistics, disinformation, propaganda and bias detection, fact-checking, machine translation, question answering, sentiment analysis, lexical semantics, and biomedical text processing. He received his PhD degree in Computer Science from the University of California at Berkeley (supported by a Fulbright grant), and he was a Research Fellow in the National University of Singapore, a honorary lecturer in the Sofia University, and research staff at the Bulgarian Academy of Sciences. At QCRI, he leads the Tanbih mega-project, developed in collaboration with MIT, which aims to limit the effect of "fake news", propaganda and media bias by making users aware of what they are reading, thus promoting media literacy and critical thinking. Dr. Nakov is President of ACL SIGLEX, Secretary of ACL SIGSLAV, and a member of the EACL advisory board. He is member of the editorial board of a number of journals including Computational Linguistics, TACL, CS&L, NLE, AI Communications, and Frontiers in AI. He is also on the Editorial Board of the Language Science Press Book Series on Phraseology and Multiword Expressions. He co-authored a Morgan & Claypool book on Semantic Relations between Nominals, two books on computer algorithms, and many research papers in top-tier conferences and journals. Dr. Nakov received a Best Long Paper Award at CIKM-2020, a Best Demo Award (Honorable Mention) at ACL-2020, and the Young Researcher Award at RANLP-2011. He was also the first to receive the Bulgarian President's John Atanasoff award, named after the inventor of the first automatic electronic digital computer. Dr. Nakov's research was featured by over 100 news outlets, including Forbes, Boston Globe, Aljazeera, DefenseOne, Business Insider, MIT Technology Review, Science Daily, Popular Science, Fast Company, The Register, WIRED, and Engadget, among others.

TRACK
“AFFECT, HATE, AND STANCE”

AMI: Automatic Misogyny Identification

AMI @ EVALITA2020: Automatic Misogyny Identification

Elisabetta Fersini[1], **Debora Nozza**[2], **Paolo Rosso**[3]
[1]DISCo, University of Milano-Bicocca
[2] Bocconi University
[3]PRHLT Research Center, Universitat Politècnica de València
elisabetta.fersini@unimib.it
debora.nozza@unibocconi.it
prosso@dsic.upv.es

Abstract

English. Automatic Misogyny Identification (AMI) is a shared task proposed at the Evalita 2020 evaluation campaign. The AMI challenge, based on Italian tweets, is organized into two subtasks: (1) Subtask A about misogyny and aggressiveness identification and (2) Subtask B about the fairness of the model. At the end of the evaluation phase, we received a total of 20 runs for Subtask A and 11 runs for Subtask B, submitted by 8 teams. In this paper, we present an overview of the AMI shared task, the datasets, the evaluation methodology, the results obtained by the participants and a discussion about the methodology adopted by the teams. Finally, we draw some conclusions and discuss future work.

Italiano. *Automatic Misogyny Identification (AMI) é uno shared task proposto nella campagna di valutazione Evalita 2020. La challenge AMI, basata su tweet italiani, si distingue in due subtasks: (1) subtask A che ha come obiettivo l'identificazione di testi misogini e aggressivi (2) subtask B relativo alla fairness del modello. Al termine della fase di valutazione, sono state ricevute un totale di 20 submissions per il subtask A e 11 per il subtask B, inviate da un totale di 8 team. Presentiamo di seguito una sintesi dello shared task AMI, i dataset, la metodologia di valutazione, i risultati ottenuti dai partecipanti e una discussione sulle metodologie adottate dai diversi team. Infine, vengono discusse le conclusioni e delineati gli sviluppi futuri.*

1 Introduction

The expressions of people about thoughts, emotions, and feelings by means of posts in social media have been widely spread. Women have a strong presence in these online environments: 75% of females use social media multiple times per day compared to 64% of males. While new opportunities emerged for women to express themselves, systematic inequality and discrimination take place in the form of offensive content against the female gender. These manifestations of misogyny, usually provided by a man to a woman for dominating or using a sort of power against the female gender, is a relevant social problem that has been addressed in the scientific literature during the last few years. Recent investigations studied how the misogyny phenomenon takes place, for example as unjustified slurring or as stereotyping of the role/body of a woman (i.e., the hashtag #getbacktokitchen), as described in the book by Poland (Poland, 2016). Preliminary research work was conducted in (Hewitt et al., 2016) as the first attempt of manual classification of misogynous tweets, while automatic misogyny identification in social media has been firstly investigated in (Anzovino et al., 2018). Since 2018, several initiatives have been dedicated as a call-to-action to stop hate against women both from a machine learning and computational linguistics points of view, such as AMI@Evalita 2018 (Fersini et al., 2018a), AMI@IberEval2018 (Fersini et al., 2018b) and HatEval@SemEval2019 (Basile et al., 2019). Several relevant research directions have been investigated for addressing the misogyny identification challenge, among which approaches focused on effective text representation (Bakarov, 2018; Basile and Rubagotti, 2018), machine learning models (Buscaldi, 2018; Ahluwalia et al., 2018) and domain-specific lexical resources (Pamungkas et al., 2018; Frenda et al., 2018).

During the AMI shared task organized at the Evalita 2020 evaluation campaign (Basile et al., 2020), the focus is not only on misogyny identification but also on aggressiveness recognition, as well as to the definition of models able to guarantee fair predictions.

2 Task Description

The AMI shared task, which is a re-run of a previous challenge at Evalita 2018, proposes the automatic identification of misogynous content in the Italian language on Twitter. More specifically, it is organized according to two main subtasks:

- **Subtask A - Misogyny & Aggressive Behaviour Identification**: a system must recognize if a text is misogynous or not, and in case of misogyny, if it expresses an aggressive attitude. In order to provide an annotated corpus for Subtask A, the following definitions have been adopted to label the collected dataset:
    - *Misogynous*: a text that expresses hating towards women in particular (in the form of insulting, sexual harassment, threats of violence, stereotype, objectification, and negation of male responsibility).
    - *Not Misogynous*: a text that does not express any form of hate towards women.
    - *Aggressive*: a message is considered aggressive if it (implicitly or explicitly) presents, incites, threatens, implies, suggests, or alludes to:
        - attitudes, violent actions, hostility or commission of offenses against women;
        - social isolation towards women for physical or psychological characteristics;
        - justify or legitimize an aggressive action against women.
    - *Not Aggressive*: If none of the previous conditions hold.
- **Subtask B - Unbiased Misogyny Identification**: a system must discriminate misogynistic contents from the non-misogynistic ones, while guaranteeing the fairness of the model (in terms of unintended bias) on a synthetic dataset (Nozza et al., 2019). To this purpose Subtask B has the goal of measuring the attitude of a model to be fair when processing sentences containing specific identity terms that likely conveyed misogyny in the training data, e.g. "girlfriend" and "wife".

3 Training and Testing Data

The data provided to the participants for the AMI shared task comprises a raw dataset and a synthetic dataset for measuring bias. Each dataset is distinguished in Training Set and Test Set.

3.1 Raw dataset

The raw dataset is a balanced dataset of misogynous and non-misogynous tweets. The *raw training set* (6,000 tweets) is derived from the data collected for the 2018 edition of the AMI shared task, where the misogynistic posts have been enriched by labelling aggressive expressions according to the definition provided in Section 2. The *raw test dataset* (approximately 1,000 tweets) has been collected from Twitter using a similar approach to the 2018 edition of the shared task. This is intentionally done to evaluate the generalization abilities of the systems on test data collected in a different time period and therefore characterized by higher language variability with respect to the training data. Examples of tweets belonging to the raw dataset are shown in Table 1.

Table 1: Examples of misogynous and non-misogynous tweets

Misogynous	Aggressiveness	Text
1	1	Stupida tr**a! Meriti di morire
1	0	Le ragazze sono tutte pu***ne
0	0	@chiellini sei una cagna

The training raw data for this dataset are provided as TSV files (tab-separated files) and report the following fields, where:

- id denotes a unique identifier of the tweet.
- text represents the tweet text.
- misogynous defines whether a tweet is misogynous or not misogynous; it takes values:
    - 0 if the tweet is not misogynous;
    - 1 if the tweet is misogynous.

Table 2: Template examples.

Template Examples	Misogynous
`<identity_term>`dovrebbero essere protette	0
`<identity_term>`dovrebbero essere uccise	1
apprezziamo `<identity_term>`	0
picchiamo `<identity_term>`	1
splendida `<identity_term>`	0
sporca `<identity_term>`	1

- aggressiveness denotes whether a misogynous tweet is aggressive or not; it takes value as:
  - 0 denotes a non-aggressive tweet (not misogynous tweets are labelled as 0 by default);
  - 1 if the tweet is aggressive.

The raw testing data are provided as TSV files reporting only id and text.

3.2 Synthetic dataset

The synthetic test dataset for measuring the presence of unintended bias has been created following the procedure adopted in (Dixon et al., 2018; Nozza et al., 2019): a list of identity terms has been constructed by taking into consideration some concepts related to the term “donna” (e.g. “moglie”, “fidanzata”). Given the identity terms, several templates have been created including positive/negative verbs and adjectives (e.g. negative: hate, inferior; positive: love, awesome) both for conveying a misogynistic message or a non-misogynistic one. Some examples of such templates, used to create the synthetic dataset, are reported in Table 2.

The synthetic dataset, created for measuring the presence of unintended bias, contains template-generated text labelled according to:

- Misogyny: Misogyny (1) vs. Not Misogyny (0)

The training data for the raw dataset are provided as TSV files (tab-separated files) and report the following fields:

- id denotes a unique identifier of the template-generated text.
- text represents the template-generated text.
- misogynous defines if the template-generated text is misogynous or non-misogynous; it takes values as 1 if the tweet is misogynous, 0 if the tweet is non-misogynous.

The synthetic testing data are provided as TSV files (tab-separated files) reporting only id and text.

The statistics about the raw and synthetic datasets, both for the training and testing sets, are reported in Table 3.

Table 3: Distribution of labels on the Training and Test datasets

	Training		**Testing**	
	Raw	**Synthetic**	**Raw**	**Synthetic**
Misogynous	2337	1007	500	954
Non-misogynous	2663	1007	500	954
Aggressive	1783	-	176	-
Non-aggressive	3217	-	824	-

4 Evaluation Measures and Baseline

Considering the distribution of labels of the dataset, we have chosen different evaluation metrics. In particular, we distinguished as follows:

Subtask A. Each class to be predicted (i.e. “Misogyny” and “Aggressiveness”) has been evaluated independently on the other using a Macro F1-score. The final ranking of the systems participating in Subtask A was based on the Average Macro F1-score (F_1), computed as follows:

$$Score_A = \frac{F_1(Misogyny) + F_1(Aggressiveness)}{2} \tag{1}$$

Subtask B. The ranking for Subtask B is computed by the weighted combination of AUC estimated on the test raw dataset AUC_{raw} and three per-term AUC-based bias scores computed on the synthetic dataset ($AUC_{Subgroup}$, AUC_{BPSN}, AUC_{BNSP}). Let s be an identity-term (e.g. “girlfriend” and “wife”) and N be the total number of identity-terms, the score of each run is estimated according to the following metric:

$$\begin{aligned} Score_B &= \tfrac{1}{2} AUC_{raw} + \\ &+ \tfrac{1}{2N} \Big[\textstyle\sum_s AUC_{subgroup}(s) \\ &+ \textstyle\sum_s AUC_{BPSN}(s) \\ &+ \textstyle\sum_s AUC_{BNSP}(s) \Big] \end{aligned} \tag{2}$$

Unintended bias can be uncovered by looking at differences in the score distributions between data mentioning a specific identity-term (subgroup distribution) and the rest (background distribution).

Table 4: Team overview

Team Name	Affiliation	Country	Runs	Subtask
jigsaw (Lees et al., 2020)	Google	US	2 (u)	A, B
fabsam (Fabrizi, 2020)	University of Pisa	IT	2 (c)	A, B
YNU_OXZ (Ou and Li, 2020)	Yunnan University	CN	2(u)	A
NoPlaceForHateSpeech (da Silva and Roman, 2020)	University of Sao Paulo	BR	3 (c)	A
AMI_the_winner (Lepri et al.,)	University of Pisa	IT	3 (c)	A
MDD (El Abassi and Nisioi, 2020)	University of Bucharest	HU	2 (u), 1 (c)	A, B
PoliTeam (Attanasio and Pastor, 2020)	Politecnico di Torino	IT	2 (c)	A, B
UniBO (Muti and Barròn-Cedeño, 2020)	University of Bologna	IT	1 (c)	A

The three per-term AUC-based bias scores are related to specific subgroups as follows:

- $AUC_{Subgroup}(s)$: calculates AUC only on the data within the subgroup related to a given identity term. This represents model understanding and separability within the subgroup itself. A low value in this metric means the model does a poor job of distinguishing between misogynous and non-misogynous comments that mention the identity.

- $AUC_{BPSN}(s)$: Background Positive Subgroup Negative (BPSN) calculates AUC on the misogynous examples from the background and the non-misogynous examples from the subgroup. A low value in this metric means that the model confuses non-misogynous examples that mention the identity-term with misogynous examples that do not, likely meaning that the model predicts higher misogynous scores than it should for non-misogynous examples mentioning the identity-term.

- $AUC_{BNSP}(s)$: Background Negative Subgroup Positive (BNSP) calculates AUC on the non-misogynous examples from the background and the misogynous examples from the subgroup. A low value here means that the model confuses misogynous examples that mention the identity with non-misogynous examples that do not, likely meaning that the model predicts lower misogynous scores than it should for misogynous examples mentioning the identity.

In order to compare the submitted runs with a baseline model, we provided a benchmark (AMI-BASELINE) based on Support Vector Machine trained on a unigram representation of tweets with Tf-IDF weighing schema. In particular, we created one training set for each field to be predicted, i.e. “misogynous”, “aggressiveness”, where each tweet has been represented as a bag-of-words (composed of 1000 terms) coupled with the corresponding label. Once the representations have been obtained, Support Vector Machines with linear kernel have been trained and provided as AMI-BASELINE.

5 Participants and Results

A total of 8 teams from 6 different countries participated in at least one of the two subtasks of AMI. Two teams participated with the same approach also in the HaSpeeDe shared task (Sanguinetti et al., 2020), addressing misogyny identification with generic models for detecting hate speech. Each team had the chance to submit up to three runs that could be constrained (c), where only the provided training data and lexicons were admitted, and unconstrained (u), where additional data for training were allowed. Table 4 reports an overview of the teams illustrating their affiliation, their country, the number and type (c for constrained, u for unconstrained) of submissions, and the subtasks they addressed.

5.1 Subtask A: Misogyny & Aggressive Behaviour Identification

Table 5 reports the results for the Misogyny & Aggressive Behaviour Identification task, which received 20 submissions submitted by 8 teams. The highest result has been achieved by *jigsaw* at 0.7406 in an unconstrained setting and by *fabsam* at 0.7342 in a constrained run. While the best results obtained as unconstrained is based on ensembles of fine-tuned custom BERT models, the one achieved by the best constrained system is grounded on a convolutional neural network that exploits pre-trained word embeddings.

By analysing the detailed results, it emerged that while the identification of misogynous text can be considered a quite simple problem, the recognition of aggressiveness needs to be properly addressed. In fact, the score reported in Table 5 are strongly affected by the prediction capabilities mostly related to the aggressive posts. This is likely due to the subjective perception of aggressiveness captured by the variance of the data available in the ground truth.

Table 5: Results of Subtask A. Constrained runs are marked as “c”, while the unconstrained ones with “u”. An amended run, marked with **, has been submitted after the deadline.

Rank	Run Type	Score	Team
**	c	0.744	UniBO **
1	u	0.741	jigsaw
2	u	0.738	jigsaw
3	c	0.734	fabsam
4	u	0.731	YNU_OXZ
5	c	0.731	fabsam
6	c	0.717	NoPlaceForHateSpeech
7	u	0.701	YNU_OXZ
8	c	0.695	fabsam
9	c	0.693	NoPlaceForHateSpeech
10	c	0.687	AMI_the_winner
11	u	0.684	MDD
12	c	0.683	PoliTeam
13	c	0.682	MDD
14	c	0.681	PoliTeam
15	u	0.668	MDD
16	c	0.665	AMI_the_winner
17	c	0.665	AMI_BASELINE
18	c	0.647	PoliTeam
19	c	0.634	UniBO
20	c	0.626	AMI_the_winner
21	c	0.490	NoPlaceForHateSpeech

After the deadline the team *UniBO* submitted an amended run (**), that has not been ranked in the official results of the AMI shared task. However, we believe interesting to mention their achievement showing an Average Macro F1-score equal to 0.744.

5.2 Subtask B: Unbiased Misogyny Identification

Table 6 reports the results for the Unbiased Misogyny Identification task, which received 11 submissions by 4 teams, among which 4 unconstrained and 7 constrained. The highest Average Macro F1 score has been achieved by *jigsaw* at 0.8825 with an unconstrained run and by *PoliTeam* at 0.8180 with a constrained submission.

Similarly to the previous task, most of the systems have shown better performance compared to the *AMI-BASELINE*. By analizing the runs, we can highlight that the two best results achieved on Subtask B have been obtained by the unconstrained run submitted by *jigsaw*, where a simple debiasing technique based on data augmentation have been adopted, and by the constrained run provided by *Politeam*, where the problem of biased prediction has been partially mitigated by introducing misogynous lexicon.

Table 6: Results of Subtask B. Constrained runs are marked as “c”, while the unconstrained ones with “u”.

Rank	Run Type	Score	Team
1	u	0.882	jigsaw
2	c	0.818	PoliTeam
3	c	0.814	PoliTeam
4	c	0.705	fabsam
5	c	0.702	fabsam
6	c	0.694	PoliTeam
7	c	0.691	fabsam
8	u	0.649	jigsaw
9	c	0.613	MDD
10	c	0.602	AMI_BASELINE
11	u	0.601	MDD
12	u	0.601	MDD

6 Discussion

The submitted systems can be compared by taking into consideration the kind of input feature that they have considered for representing tweets and the machine learning model that has been used as classification model.

Textual Feature Representation. The systems submitted by the challenge participants’ consider various techniques for representing the tweet contents. Most of the teams experimented a high-level representation of the text based deep learning solutions. While few teams like *fabsam* and *MDD* adopted a text representation based on traditional **word embeddings** such as Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014) and FastText (Bojanowski et al., 2017), most of the systems. i.e *NoPlaceForHateSpeech*,*jigsaw*, *PoliTeam*, *YNU_OXZ* and *UniBO*, exploited richer **sentence embeddings** such as BERT (Devlin et al., 2019) or XLM-RoBert (Ruder et al., 2019). For enriching the space for then training the subsequent models to recognize misogyny and aggressiveness, *PoliTeam* experimented the use of additional lexical resources such as misogynous lexicon and sentiment Lexicon.

Machine Learning Models. Concerning the machine learning models, we can distinguish between approaches trained from scratch and those ones based on fine-tuning of existing pre-trained models. We report in the following the strategy adopted by the systems that participated in the AMI shared task, according to the type of machine learning model that has been adopted:

- **Shallow models** have been experimented by

MDD, where logistic regressions have been trained according to different hand-crafted features;

- **Convolutional Neural Networks** have been exploited by *NoPlaceForHateSpeech* by using two distinct models for misogyny detection and aggressiveness identification, by *fabsal* investigating the optimal hyperparameters of the model, and by *YNU_OXZ* where on top of the CNN architecture a Capsule Network (Sabour et al., 2017) has been introduced for taking advantage of spatial patterns available in short texts;
- **Fine-Tuning of pre-trained models** has been exploited by *jigsaw* by adapting BERT to the challenge domain and using a transfer multilingual strategy and ensemble learning, by *UniBO* that accommodated the BERT model using a multi-label output neuron, and by *PoliTeam* where the prediction of the fine-tuned sentence-BERT is coupled with prediction based on lexicons.

For what concerns the achieved results on the two subtasks, few considerations can be drawn considering both the errors done by the systems and the mitigation strategies adopted for reducing the bias.

Error Analysis When testing the developed systems on raw test data, the majority of the performed errors can be summarized by the following patterns:

- **Under-representation of subjective expressions**: those posts written by introducing erroneous lower case and missing spaces between adjoining words lead the models based on raw text to make errors on test predictions. An example of such common errors is the one reported in the following tweet:

  > *"Odio Sakura per il semplice motivo che qualunque cosa faccia o dica Naruto lei lo prende a schiaffi o a pugniHA CHIESTO COME STA SAI DIOSANTO BRUTTA STRONZA MA CONTRALLI MADONNA SPERO CHE TI UCCIDANOscusami Sarada"*

- **Undefined subject, but presence of aggressive terms**: for those tweets where the target is not clearly mentioned, but several aggressive terms are present, the models tend to be biased and to predict the post as misogynous and aggressive erroneously. An example of this type of misclassified posts is reported here:

  > *"Vero...ma c'e chi ti cerca, che ti vuole, più di ogni cosa al mondo......ma non sa se viene capito..... potrebbe esser mal interpretato e di conseguenza all'abbraccio esser denunciato per molestie sessuali e/o stupro"*

- **Short hate speech sentences referred to others than women**: when the target is mentioned by using an actual account, but it is referred to men, and there are no additional indications about the gender of the target, most of the models tend to misclassify the tweet. In the following example, the target is a male football player:

  > *"@bonucci_leo19 Cagati in mano e prenditi a schiaffi. Sti post te li infili nel c*lo!"*

Concerning the errors on the synthetic test set used for estimating the bias of the models, two main errors carried out by the majority of the systems can be identified:

- **presence of unusual target**: in most of the submissions, sentences containing offensive expressions towards specific uncommon targets are misclassified. For instance, around 39% of the predictions related to the target *nonna* (i.e., grandmother) are wrong. An example of the most misclassified target is reported in the following example:

  > *"nonne belle"*

- **Presence of unusual verbs**: analogously to what has been observed for the target, sentences containing rare aggressive verbs tend to be misclassified. For instance, around 48% of the instances related to the verbs *malmenare* and *seviziare* (i.e., beat up and torture) are wrongly classified. An example of a mistaken sentence are reported here:

  > *"femmina dovrebbe essere seviziata"* (wrongly classified as non-misogynous)

Bias Mitigation strategies. Concerning the Subtask B, only one team (*jigsaw*) addressed explicitly the problem related to the unintended bias. The authors used sentences sampled from the Italian Wikipedia articles containing some of the identity terms provided with the test set. These sentences, labeled as both non-misogynous and non-aggressive, have been used to further finetune the model and reduce the bias given by the data. The results achieved by the *jigsaw* team highlight that a debiasing method could obtain fair predictions even using pre-trained models.

7 Conclusions and Future Work

This paper presents the AMI shared task, focused not only on identifying misogynous and aggressive expressions but also on ensuring fair predictions. By analysing the runs submitted by the participants, we can conclude that while the problem of misogyny identification has reached satisfactory results, the recognition of aggressiveness is still in its infancy. Concerning the capabilities of the systems with respect to the unintended bias problem, we can highlight that a domain-dependent mitigation strategy is a necessary step towards fair models.

Acknowledgements

The work of the last author was partially funded by the Spanish MICINN under the research project MISMISFAKEnHATE on MISinformation and MIScommunication in social media: FAKE news and HATE speech (PGC2018-096212-B-C31) and by the COST Action 17124 DigForAsp supported by the European Cooperation in Science and Technology.

References

Resham Ahluwalia, Himani Soni, Edward Callow, Anderson Nascimento, and Martine De Cock. 2018. Detecting Hate Speech Against Women in English Tweets. In *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, Turin, Italy. CEUR.org.

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic Identification and Classification of Misogynistic Language on Twitter. In *Proceedings of 23rd International Conference on Applications of Natural Language to Information Systems (NLDB)*, pages 57–64. Springer.

Giuseppe Attanasio and Eliana Pastor. 2020. PoliTeam @ AMI: Improving Sentence Embedding Similarity with Misogyny Lexicons for Automatic Misogyny Identification in Italian Tweets. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Bologna, Italy. CEUR.org.

Amir Bakarov. 2018. Vector Space Models for Automatic Misogyny Identification. In *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, Turin, Italy. CEUR.org.

Angelo Basile and Chiara Rubagotti. 2018. Automatic Identification of Misogyny in English and Italian Tweets at EVALITA 2018 with a Multilingual Hate Lexicon. In *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, Turin, Italy. CEUR.org.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Nozza Debora, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, Manuela Sanguinetti, et al. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of 13th International Workshop on Semantic Evaluation*, pages 54–63. Association for Computational Linguistics.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Davide Buscaldi. 2018. Tweetaneuse AMI EVALITA2018: Character-based Models for the Automatic Misogyny Identification Task. In *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, Turin, Italy. CEUR.org.

Adriano dos S. R. da Silva and Norton T. Roman. 2020. No Place For Hate Speech @ AMI: Convolutional Neural Network and Word Embedding for the Identification of Misogyny in Italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Bologna, Italy. CEUR.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186. Association for Computational Linguistics.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 67–73.

Samer El Abassi and Sergiu Nisioi. 2020. MDD@AMI: Vanilla Classifiers for Misogyny Identification. In *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Bologna, Italy. CEUR.org.

Samuel Fabrizi. 2020. fabsam @ AMI: a Convolutional Neural Network approach. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Bologna, Italy. CEUR.org.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018a. Overview of the Evalita 2018 Task on Automatic Misogyny Identification (AMI). In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the Sixth evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2018)*, Turin, Italy. CEUR.org.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018b. Overview of the Task on Automatic Misogyny Identification at IberEval 2018. In *IberEval@ SEPLN*, pages 214–228.

Simona Frenda, Bilal Ghanem, Estefanía Guzmán-Falcón, Manuel Montes-y-Gómez, and Luis Villaseñor-Pineda. 2018. Automatic Lexicons Expansion for Multilingual Misogyny Detection. In *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, Turin, Italy. CEUR.org.

Sarah Hewitt, Thanassis Tiropanis, and Christian Bokhove. 2016. The Problem of identifying Misogynist Language on Twitter (and other online social spaces). In *Proceedings of the 8th ACM Conference on Web Science*, pages 333–335. ACM.

Alyssa Lees, Jeffrey Sorensen, and Ian Kivlichan. 2020. Jigsaw @ AMI and HaSpeeDe2: Fine-Tuning a Pre-Trained Comment-Domain BERT Model. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Bologna, Italy. CEUR.org.

Marco Lepri, Giuseppe Grieco, and Mattia Sangermano. University of Pisa, Italy.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Arianna Muti and Alberto Barròn-Cedeño. 2020. UniBO@AMI: A Multi-Class Approach to Misogyny and Aggressiveness Identification on Twitter Posts Using AlBERTo. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Bologna, Italy. CEUR.org.

Debora Nozza, Claudia Volpetti, and Elisabetta Fersini. 2019. Unintended bias in misogyny detection. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 149–155.

Xiaozhi Ou and Hongling Li. 2020. YNU_OXZ @ HaSpeeDe 2 and AMI : XLM-RoBERTa with Ordered Neurons LSTM for classification task at EVALITA 2020. In *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Bologna, Italy. CEUR.org.

Endang Wahyu Pamungkas, Alessandra Teresa Cignarella, Valerio Basile, and Viviana Patti. 2018. Automatic Identification of Misogyny in English and Italian Tweets at EVALITA 2018 with a Multilingual Hate Lexicon. In *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, Turin, Italy. CEUR.org.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing*, pages 1532–1543.

Bailey Poland. 2016. *Haters: Harassment, Abuse, and Violence Online*. Potomac Books, Incorporated.

Sebastian Ruder, Anders Søgaard, and Ivan Vulić. 2019. Unsupervised cross-lingual representation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 31–38. Association for Computational Linguistics.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

UniBO @ AMI: A Multi-Class Approach to Misogyny and Aggressiveness Identification on Twitter Posts Using AlBERTo

Arianna Muti
Department of Modern Languages,
Literatures and Cultures - LILEC
Università di Bologna
Bologna, Italy
`arianna.muti@studio.unibo.it`

Alberto Barrón-Cedeño
DIT – Università di Bologna
Forlì, Italy
`a.barron@unibo.it`

Abstract

We describe our participation in the EVALITA 2020 (Basile et al., 2020) shared task on Automatic Misogyny Identification. We focus on task A —Misogyny and Aggressive Behaviour Identification— which aims at detecting whether a tweet in Italian is misogynous and, if so, whether it is aggressive. Rather than building two different models, one for misogyny and one for aggressiveness identification, we handle the problem as one single multi-label classification task, considering three classes: non-misogynous, non-aggressive misogynous, and aggressive misogynous. Our three-class supervised model, built on top of AlBERTo, obtains an overall F_1 score of 0.7438 on the task test set ($F_1 = 0.8102$ for the misogyny and $F_1 = 0.6774$ for the aggressiveness task), which outperforms the top submitted model ($F_1 = 0.7406$).[1]

1 Introduction

In 2020, Twitter users in Italy amount to approximately 3.7 million and the number is expected to constantly increase by 2026.[2] Although Twitter is conceived to express personal opinions, share today's biggest news, follow people or simply communicate with friends, there has been an increasing number of users that misuse the platform by engaging in trolling, cyberbullying, or by posting aggressive and misogynous content (Samghabadi et al., 2020). Due to the sheer amount of user-generated content on social media, providers struggle to control inappropriate content. Twitter relies on the community's reports to identify and remove abusive posts from the platform, while pursuing the users' right to freedom of expression. However, it is a tricky task to determine where to draw the line between free expression and the production of harmful content, due to the subjective nature of what different users perceive as offensive. Twitter has committed to tackling this issue by releasing a policy containing a clear definition of abusive speech, according to which a user cannot promote violence against or directly attack or threaten people on the basis of race, ethnicity, national origin, caste, sexual orientation, gender, gender identity, religious affiliation, age, disability, or serious disease.[3]

However, two main issues exist. Since Twitter mostly relies on the community subjective perception of hate speech, many posts are not subjected to report, review, and removal. Moreover, the amount of abusive posts significantly outnumbers the people that can manually control harmful content. Therefore, there is a need to improve the quality of algorithms to spot potential instances of hate speech; in particular towards women, since research shows that they are subjected to more bullying, abuse, hateful language, and threats than men on social media (Fallows, 2005).

AMI 2020 consists of two tasks (Fersini et al., 2020). Task A —Misogyny and Aggressive Behaviour Identification— aims at detecting whether a Twitter post is misogynous and, if so, whether it is aggressive (Anzovino et al., 2018). Task B —

---

[1] Our official submission to the task obtained $F_1 = 0.6343$ ($F_1 = 0.7263$ for the misogyny and $F_1 = 0.5423$ for the aggressiveness task). The reason behind this poor performance was the unintended use of a mistaken transformer. See Appendix A for further details.

[2] `https://www.statista.com/forecasts/1146708/twitter-users-in-italy`; last visit: 6 November, 2020.

[3] `https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy`

Unbiased Misogyny Identification— aims at discriminating misogynistic contents from the non-misogynist ones, while guaranteeing the fairness of the model (in terms of unintended bias) on a synthetic dataset (Nozza et al., 2019). We undertook task A and we present a system to flag misogynous and women-addressed aggressive posts on Twitter in the Italian language. Even if task A involves two sub-problems, we address it as a three-class supervised problem using AlBERTo (Polignano et al., 2019), a BERT language understanding model for the Italian language which is focused on the language used in social networks, specifically on Twitter. We built only one model to identify the three possible classes: non-misogynous, non-aggressive misogynous, and aggressive misogynous. This multi-class setting has shown to be effective. Our approach obtains an F_1 score of 0.7438, outperforming the top-ranked official submission (although our own official submission obtained $\text{F}_1 = 0.6343$ only; cf. Appendix A).

The rest of the contribution is distributed as follows. Section 2 includes some background and a brief overview of research in automatic misogyny identification. Section 3 describes the employed dataset. Section 4 describes our model. Section 5 summarizes the experiments performed and discusses the obtained results. It includes an error analysis, in order to show the error trends of the model. Section 6 draws some conclusions and discusses further possible research lines.

2 Background

Due to the subjective perception of misogyny and aggressiveness, a definition of what can be considered misogynous and aggressive is necessary:

Misogynous content expresses hating towards women, in the form of insulting, sexual harassment, male privilege, patriarchy, gender discrimination, belittling of women, violence against women, body shaming and sexual objectification (Srivastava et al., 2017). A misogynous content expresses an **aggressive attitude** when it overtly or covertly encourages or legitimizes violent actions against women.

From a computational point of view, misogyny detection is a text classification task. Text classification in Natural Language Processing has been widely explored and it is typically addressed by using supervised models (Mirończuk and Protasiewicz, 2018). Past research shows the effectiveness of diverse neural-network architectures to learn text representations, such as convolutional models, recurrent networks and attention mechanisms (Sun et al., 2019). Recent work shows that pre-trained models such as BERT achieve state-of-the-art results in text classification tasks and spare time, since they prevent you from training models from scratch (Sun et al., 2019).

For what concerns misogyny identification, a shared task took place at IberEval 2018, focusing on English and Spanish tweets (Fersini et al., 2018b). Whereas task A concerned misogyny identification, task B proposed a multi-class problem to classify misogynous sentences into seven categories: discredit, stereotype, objectification, sexual harassment, threats of violence, dominance, and derailing. The most used supervised models were support vector machines, ensembles of classifiers and deep-learning models. Participants mostly used n-grams and word embeddings to represent the tweets.

As for misogyny identification in Italian, the first edition of the AMI shared task took place in 2018 (Anzovino et al., 2018). The task A was again misogyny identification, while the task B aimed at recognizing whether a misogynous content is person-specific or generally addressed towards a group of women, and at classifying the positive instances in the aforementioned categories. The best-performing approach obtained an F_1 score of 0.844, using TF-IDF weighting combined with singular value decomposition for language representation and an ensemble of supervised models (Fersini et al., 2018a).

3 Dataset

As mentioned above, the aim of our model is to flag misogynous contents and aggressive attitudes towards women in Italian tweets. To address this task, a dataset was provided by the task organizers: $5,000$ tweets, manually labelled according to two classes, misogyny and aggressiveness. The first one defines whether a tweet has been flagged as misogynous (positive class) or not (negative class). If a tweet has been flagged as misogynous, it is further determined whether it is considered as aggressive (positive class) or not (negative class).

The training dataset is fairly balanced in terms of misogyny. It contains $2,337$ misogynous and $2,663$ non-misogynous instances. A total of

	batch size	
epochs	16	32
8	**0.8491**	0.8392
10	0.8485	0.8298
15	0.8283	0.8351
20	0.8342	0.8087

Table 1: F_1 performance of the 3-class model with different batch sizes after diverse numbers of epochs using AlBERTo

1, 783 of the former are also considered as aggressive, whereas only 554 are not. The test set was composed of 1, 000 tweets.

Since we opted for a constrained approach, we only used the data provided by the organizers. We randomly split the supervised data into training and validation sets: 4, 700 instances for the former and 300 for the latter.

4 Description of the System

Since the identification of aggressiveness is related to the identification of misogynous tweets, we opt for a 3-class setting, based on one single model. The three classes are hence non-misogynist, aggressive misogynist, and non-aggressive misogynist. The idea is to determine how well a multi-label classifier can perform when addressing these two related problems; handling aggressiveness as a consequential class of the misogyny one.

We decided to base our model on BERT (Bidirectional Encoder Representations from Transformers), a task-independent language representation model based on the transformers architecture (Devlin et al., 2019). BERT uses a masking approach that randomly masks some input tokens within a sentence and then predicts the removed tokens based on the context. It is bidirectional because it makes use of Transformers that consider both the left and right context at once with respect to the hidden word to make the prediction upon. We decided to use AlBERTo, a variation of BERT in Italian, trained on Twitter posts (Polignano et al., 2019), which includes emojis, links, hashtags, and mentions. AlBERTo was trained on 200M tweets randomly sampled from the TWITA corpus (Basile et al., 2018).

As for the pre-processing, we used the pre-trained AlBERTo tokenizer for text tokenization, and then we encoded the data. We set the maximum length to 256 characters, since that was the length of the longest instance in the training material (even if Twitter allows up to 280 characters).

team	run	constrained	score
UniBO[a]	2	yes	**0.7438**
jigsaw	2	no	0.7406
jigsaw	1	no	0.7380
fabsam	1	yes	0.7343
YNU_OXZ	1	no	0.7314
fabsam	2	yes	0.7309
NoPlaceForHateSpeech	2	yes	0.7167
YNU_OXZ	2	no	0.7015
fabsam	3	yes	0.6948
NoPlaceForHateSpeech	1	yes	0.6934
AMI_the_winner	2	yes	0.6869
MDD	3	no	0.6844
PoliTeam	3	yes	0.6835
MDD	1	yes	0.6820
PoliTeam	1	yes	0.6810
MDD	2	no	0.6679
AMI_the_winner	1	yes	0.6653
PoliTeam	2	yes	0.6473
UniBO[b]	1	yes	0.6343
AMI_the_winner	3	yes	0.6259
NoPlaceForHateSpeech	3	yes	0.4902

[a] Run submitted after the deadline.
[b] Buggy run submitted on the deadline (cf. Appendix A).

Table 2: Full shared task leaderboard plus our unofficial top-performing submission. The score is the average of the F_1 measures for the misogyny and the aggressiveness tasks.

We used the Pytorch instance of AlBERTo-Base, Italian Twitter lower cased[4] and fine-tuned it to the downstream task. We used a softmax output layer with three neurons to produce the classification.

In order to tune the network, we used the AdamW optimizer, which decouples weight decay from gradient computation, with a learning rate of 1e-5 (Loshchilov and Hutter, 2017).[5]

5 Results

We explored different batch sizes over an increasing number of learning epochs. Table 1 shows the performance evolution on the validation set. The best combination was to train the model over 8 epochs with a batch size of 16. This combination leads to an F_1 score of 0.8491 on the three-class problem. It is worth noting that these scores are not comparable against those for the actual task, which consists of two independent binary decisions: whether a tweet is considered misogynist and, if the answer is yes, whether it is aggressive.[6]

---

[4] `https://github.com/marcopoli/AlBERTo-it`,

[5] The implementation is available at `https://github.com/TinfFoil/unibo_ami2020/`.

[6] Indeed, the official task score is computed as the average of the F_1 measures for the misogyny and the aggressiveness classifications.

	tweet	misogynist		aggressive	
		actual	pred	actual	pred
Type-1 errors					
1.2	La cosa del filo del telefono attorno alla gola è presa da una lite di Stefanie e Brooke in Beautiful #chilhavisto	no	yes	no	yes
1.2	“Quanti schiaffi, #Malinconico. Li meriteresti tutti, perché anche questa volta sei riuscito a lasciare il segno.E no, non era facile…”	no	yes	no	yes
1.3	Tutti quelli che ho conosciuto mi hanno sempre detto che a primo impatto sembro stronza e psicopatica ma in verità sono una cucciola	yes	no	no	no
Type-2 errors					
2.1	@Fedoc5 in ginocchio con te che comandi il ritmo del pompino e decidi quando sborrarmi in gola, perchè un pompino deve sempre finire con l’ingoio	no	yes	no	yes
2.2	@Nigagalsen @matteosalvinimi un follower dal 2016…e ti permetti di aprire quella lurida fogna di bocca che hai.	no	yes	no	no
2.3	Antonio ti prenderei a schiaffi, come fa a dire mi manchi, quando con Ilaria fai tutto. Coglione	no	no	no	no
2.4	Posso volerlo vedere cagarsi in mano e prendersi a schiaffi finché non diventano dispari o sono nazista?	no	no	no	no
Type-3 errors					
3.1	La cena è pronta. Lo squalo balena affonda nei banchi di pesce per saziare la sua immensa mole. Non è un abitudine frequente visto che filtra dall’acqua i microorganismi come le balene.	no	yes	no	no
3.2	Comunque le pringles più buone sono quelle alla panna acida e cipolla	no	yes	no	no

Table 3: Instances from the test partition including their actual class and the one predicted by our model for both misogyny and aggressiveness.

Given these results, we trained a new model on the full trained and development sets during 8 epochs, using a batch size of 16, and predicted on the test set. Such model obtains $F_1 = 0.7438$, resulting from 0.8102 on the misogyny task and 0.6774 from the aggressiveness one.

Table 2 shows the AMI shared task leaderboard. It highlights both our official submission `UniBO run 1` (cf. Appendix A) and our post-deadline submission `UniBO run 2`. Run 2 tops all the systems submitted to the shared task. Indeed, modelling the two tasks as one single multi-class problem (and using transformers for the right language) helps the algorithm significantly.

Error Analysis After the release of the gold labels, we performed an analysis of the classification errors. We analyzed 300 instances, taken randomly from the test set (100 at the beginning, 100 in the middle and 100 at the end). As observed from the reported performance, our model struggled mostly with the identification of aggressive instances. As a result, there are relatively few cases in which our model correctly labels non-aggressive misogynous instances. We noticed that most of the time, when our model labels an instance as misogynist, it also labels it as aggressive. On the contrary, the system performs very well in identifying non-misogynous instances and aggressive-misogynous instances. The most common mistakes are grouped into three categories:

1. The system identifies as aggressive the instances that contain verbs expressing an aggressive attitude.[7]

2. The system identifies as misogynous (and most of the time also aggressive) instances

---

[7] One potential reason behind this confusion is that we suspect that there are aggressive tweets in the dataset which, not having been identified as misogynist in the first place, are mislabeled as non-aggressive. This hypothesis should be further explored.

that are neither misogynous nor aggressive, but contain typical misogynous sentences.

3. The system identifies as misogynous instances that are neither misogynous nor aggressive, but they contain *double-entendre* words typically used to insult women.

Table 3 shows some examples for all three kinds of errors. Regarding the errors of type 1, in instance 1.1 the action of winding up a telephone cable around the neck was perceived as aggressive, despite the speaker did not express a misogynous or aggressive attitude towards a woman, and indeed she is just commenting on something watched on TV. In instance 1.2, the sentence *meritare gli schiaffi* (deserving slaps) denotes violence, but it is not addressed towards a woman. This kind of mistake might be overcome by implementing a model trained on the misogynist partition of the data only. Finally, instance 1.3 represents the bias related to the subjectivity nature of what is perceived to be misogynous. According to the annotation guidelines, a tweet should be flagged as misogynous if it expresses hating towards women. In this case, the poster of the tweet is not expressing any misogynous attitude, but she is reporting what she is been told by males. Therefore, our system flagged the instance as non-misogynous and we could agree.

As for the errors of type 2, if we look at the text only, the instances could seem misogynous sentences. However, in the instances 2.1 and 2.2 the hashtag tells us that it is referred to a man and the system fails to understand that. On the contrary, the system performs well when a masculine name or a masculine pronoun is specified, instead of an hashtag, as we can observe in the instances 2.3 and 2.4. In these cases our system could understand that the aggressive actions, that usually tend to be classified as aggressive-misogynous, are not referred to a woman.

For the type 3 errors, in instance 3.1 *balena* (whale/fat woman) and in 3.2 *acida* (acid/peevish) could confuse the model causing it to flag such instances as misogynous.

6 Conclusions and Further Work

In this paper we described our approach to the EVALITA 2020 task on misogyny and aggressiveness identification in Italian tweets —AMI. The purpose of our participation was to determine whether a multi-label classifier is a good way to address this two-step task. Although the task seems to be conceived to be addressed with two different models, one for the identification of misogyny and the other for aggressiveness, we decided to try a different approach and build a single model that could identify three cases: non-misogynous, non-aggressive misogynous and aggressive misogynous tweets.

We built our model on top of AlBERTo, an Italian version of BERT, and we trained the model using only the dataset provided by the task organizers. We experimented by setting different batch sizes over an increasing number of epochs. The highest F_1 score on the validation set was reached by a batch size of 16 during 8 epochs. When evaluated on the test set, our model obtained an overall F_1 score of 0.7438; 0.8102 for the misogyny and 0.6744 for the aggressiveness task. We hypothesize that the model struggles to identify misogynist aggressive instances partly because it gets confused by non-misogynist aggressive tweets which are labeled simply as non-misogynous. The implementation is publicly available for research purposes.

For what concerns further experiments, we plan to build two separate models: one to detect misogyny and the other trained only on already-flagged misogynous tweets to identify instances of aggressiveness. Another step to undertake would be to use an unconstrained approach and increase the number of instances for the training set, so that the model will have more data to learn from.

References

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Valerio Basile, Mirko Lai, and Manuela Sanguinetti. 2018. Long-term social media data collection at the university of turin. In *Fifth Italian Conference on Computational Linguistics (CLiC-it 2018)*, Turin, Italy.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and*

Speech Tools for Italian. Final Workshop (EVALITA 2020), Online. CEUR.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, MN, June. ACL.

Deborah Fallows. 2005. How women and men use the internet. Technical report, Pew Internet & American Life Project, December.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018a. Overview of the evalita 2018 task on automatic misogyny identification (ami). In *EVALITA Evaluation of NLP and Speech Tools for Italian: Proceedings of the Final Workshop 12-13 December 2018, Naples*, pages 59–66. Torino: Accademia University Press.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018b. Overview of the task on automatic misogyny identification at ibereval 2018. In *Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018)*, Sevilla, Spain.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020. Ami @ evalita2020: Automatic misogyny identification. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Marcin M. Mirończuk and Jarosław Protasiewicz. 2018. A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications*, 106:36–54.

Debora Nozza, Claudia Volpetti, and Elisabetta Fersini. 2019. Unintended bias in misogyny detection. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 149–155, Thessaloniki, Greece.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019. AlBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*, volume 2481, Bari, Italy. CEUR.

Niloofar S. Samghabadi, Parth Patwa, Srinivas PYKL, Prerana Mukherjee, Amitava Das, and Thamar Solorio. 2020. Aggression and misogyny detection using BERT: A multi-task approach. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying (TRAC-2020)*.

Kalpana Srivastava, Suprakash Chaudhury, P.S. Bhat, and Samiksha. Sahu. 2017. Misogyny, feminism, and sexual harassment. *Industrial psychiatry journal*, 26(2):111–113.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune BERT for text classification? *CoRR*, abs/1905.05583.

A Official English-BERT-based Submission

Our official submission used a pre-trained BERT model trained only on the English language. The experimentation and tuning were identical to the one applied when using AlBERTo (cf. Section 5). Table 4 shows the tuning evolution. The best configuration of this model, derived from the English BERT, obtains an F_1 score of 0.8222 on the validation set when dealing with our three-class problem. Nevertheless, the performance dropped to $F_1 = 0.6343$ on the test set.

	batch size		
epochs	8	16	32
5	0.8126	0.8042	0.7955
8	0.8067	**0.8222**	0.8004
10	0.8042	0.8069	0.8141
15	0.8095	0.8037	0.8121
20	0.7895	0.8178	0.8153

Table 4: F_1 performance of the 3-class model with different batch sizes after diverse numbers of epochs using BERT for English.

fabsam @ AMI: A Convolutional Neural Network Approach

Samuel Fabrizi
University of Pisa, Italy
`s.fabrizi1@studenti.unipi.it`

Abstract

The presence of misogynistic contents is one of the most crucial problems of social networks. In this paper we present our system for misogyny identification on Twitter. Our approach is based on a convolutional neural network that exploits pretrained word embeddings. We also experimented a comparison among different architectures to understand the effectiveness of our method. The paper also described our submissions to both subtasks A and B to Automatic Misogyny Identification competition at Evalita 2020.

1 Introduction

The paper describes our submission to the Automatic Misogyny Identification task at Evalita 2020 (Fersini et al., 2020; Basile et al., 2020). This competition is divided into two subtasks:

- **Subtask A** Misogyny and Aggressive Behaviour Identification: identify if a text is misogynous or not, and, in case of misogyny, if it expresses an aggressive attitude.
- **Subtask B** Unbiased Misogyny Identification: discriminate misogynistic contents from the non-misogynistic ones, while guaranteeing the fairness of the model (in terms of unintended bias) on a synthetic dataset (Nozza et al., 2019).

We proposed a convolutional based approach to recognize misogynistic sentences. We grounded our work over a robust model selection technique. In order to confirm our approach we developed other architectures based on state of art models to make a systematic comparison.
Our work is organized as follows. Section 2 briefly describes related work on the proposed task. Section 3 describes our architectures. Section 4 introduces our method. In particular, it describes our approach for model selection and assessment. Section 5 presents the official results obtained in the AMI competition. Section 6 concludes this work.

2 Related Work

The misogyny identification and classification approaches are very recent (Anzovino et al., 2018). In the last few years there was an increasing number of research on this field. The majority of them have concentrated especially on abusive and aggressive language detection. This form of hate speech task has been proposed in different organized shared tasks: IberEval 2018 (Fersini et al., 2018), Evalita 2018 (Fersini et al., 2018) and later at SemEval 2019 (Basile et al., 2019). Most of the state-of-art approaches to misogyny detection were described as system reports for these shared tasks.
Finally, it is important to mention that different deep learning approaches have been proposed (Badjatiya et al., 2017). In this paper we extend the use of convolutional layers for word based feature extraction.

3 Description of the system

In this section we describe our approach that exploits the intuition of extracting dependencies among words as features from tweets. We also made an analysis about other architectures and we compare them with ours in order to understand strength and weakness of our architecture. Our method consists of the following steps:

- normalization of the datasets;
- use an effective word embedding representation;

- define different state of art architectures to compare them with our model.

3.1 Data Preprocessing and Word Embeddings

Out-of-vocabulary words are one of the most important issues with the use of word embedding, especially in the context of social networks in which colloquial language is widespread. In order to normalize tweets, we pre-processed them using tools from *ekphrasis* (Baziotis et al., 2017).

First of all we removed punctuation and separated sentences into words. Then we applied the normalization process. This process involves, for example, allcaps annotation ('ABC' becomes 'allcaps abc allcaps'), elongated words normalization ('vaaaaai' becomes 'elongated vai elongated') and emoticons transformation. We manually carried out translations of these keywords to adapt annotations to the Italian language.

We experimented different word embedding pretrained model. After a sequence of considerations we chose the word embeddings presented in Cimino et al. (2018) trained on 46 million Italian tweets. It is a word2vec based model and it encodes each word in a 128-size vector.

3.2 Our model

The model used for the AMI competition is represented in Figure 1.

Given a tweet, we firstly apply the pre-processing described in Section 3.1 to normalize and transform it into a sequence of words. Then this sequence is mapped into a fixed real vector domain by the embedding layer.

The embedding layer passes an input feature space to three 1D Convolutional layers. Each of those uses 150 filters and a stride of 1 but different kernel sizes of 1, 2, 4 respectively. These layers are the most interesting ones. Each layer can indeed be seen as extractors of n-gram features where n is equal to the kernel size (Kim, 2014). As explained in Section 4.1 we search for the best hyperparameters of these layers in model selection phase.

Outputs from CNN layers are down-sampled by a GlobalMaxPooling1D layer and then they are concatenated into a single sequence.

The last two layers are dense layers with tanh and softmax activation functions respectively. The final softmax layer maps the sequence received as input to a probability distribution over all possible classes.

This model was trained for 15 epochs using a batch size of 128.

4 Experiments

In the subtask A we split the training set provided into a train set (4250 tweets) and a test set (750 tweets). This internal test set was used only to evaluate our final model. In subtask B we merged raw and synthetic datasets and separated from each of these two test sets.

As explained in Section 3.2 we used as output layer a dense layer with softmax activation function. In order to obtain three different labels for subtask A, *misogynous* and *aggressiveness* columns were converted into a single one. We also apply one-hot encoding to the integer representation, otherwise a natural ordering between categories may result in poor performance or unexpected results.

The frequency distribution of these labels turns out to be quite unbalanced, as shown in Table 1. Furthermore for each class we have a very small number of training examples. This could have a strong influence on the overfitting of the model. We indeed avoided to use a deep neural network and we preferred to develop a simple architecture in a robust way as recommended in Zhang and Wallace (2015).

Class	**Train set**	**Test set**
Non-misogynous	2277	386
Non-aggressive	484	70
Aggressive	1489	294

Table 1: Subtask A dataset distribution

4.1 Model Selection

We decided to apply a robust model selection technique to find the best hyperparameters of our model. We used repeated K-fold cross-validation (Rodriguez et al., 2010).

In subtask A we used the official AMI score as metric. While in the subtask B we decided to use the AUC metric. In both of them we also took into consideration the standard deviation among different runs.

Model selection phase was divided in 2 mainly stages:

- **Stage 1** we validate the best hyperparameters for each different model. We report the hyperparameters ranges in Figure 2. In this

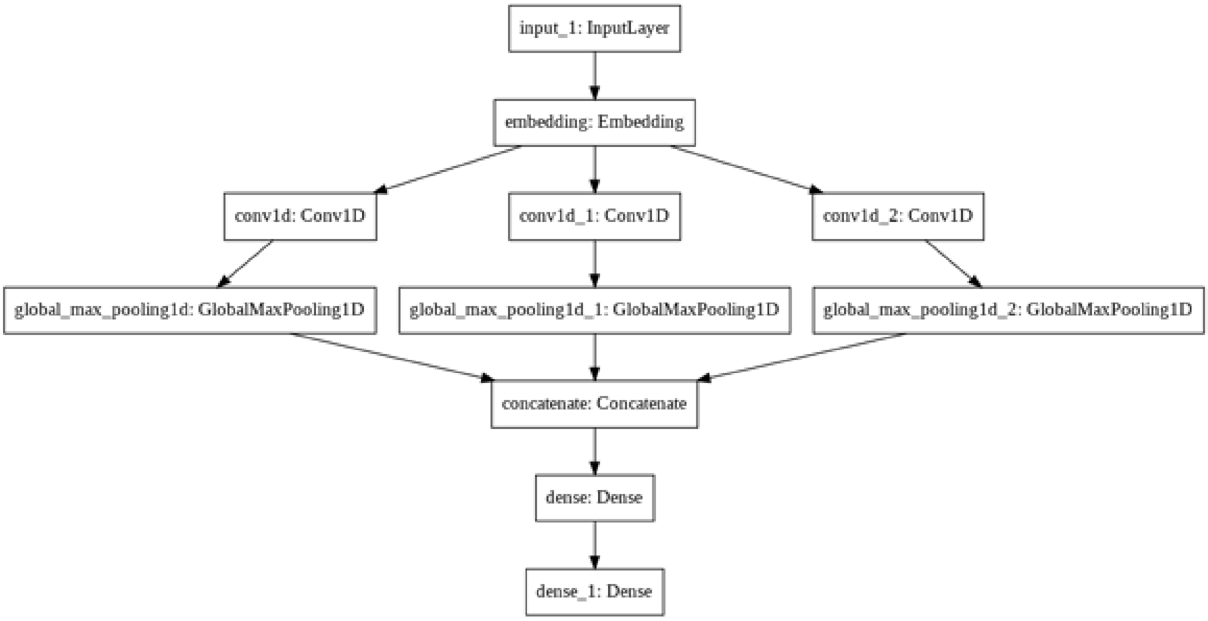


Figure 1: Model architecture

stage we used a repeated 5-fold with 10 repetitions.

- **Stage 2** We chose the most promising models according to score and standard deviation metrics. We applied another repeated 5-fold cross-validation increasing the number of repetitions to 15. Then we chose the best model among them using the same metrics as before.

Hyperparam	Range
Batch size	{32, 64, 128}
Filters	{[100, 100, 100], [150, 150, 150]}
Kernel Sizes	{[1, 2, 3], [1, 2, 4]}
L2 regularizer	{0.001, 0.005}
Number dense nodes	{8, 16}

Table 2: Hyperparameters ranges

Then we built other architectures to compare them with ours. In the following we list models used for these comparisons:

- Convolution-biGRU Based Deep Neural Network: this architecture allows to capture long-range dependencies from both directions of a sentence;

- Convolutional Based Neural Network: deep neural network based on convolutional layers that tries to extract different features using a greater number of layers. It is an extension of the architecture described in Section 3.2;

- Skipped Convolutional Neural Network (Zhang and Luo, 2018): CNN architecture where each convolutional layer uses “gapped window” to extract features from its input;

In Figure 2 we reported results obtained in stage 2 of the model selection phase in the subtask A. Our model seems to be better in terms of both score and standard deviation compared to the others. Furthermore, it does not have any outliers as other models have.

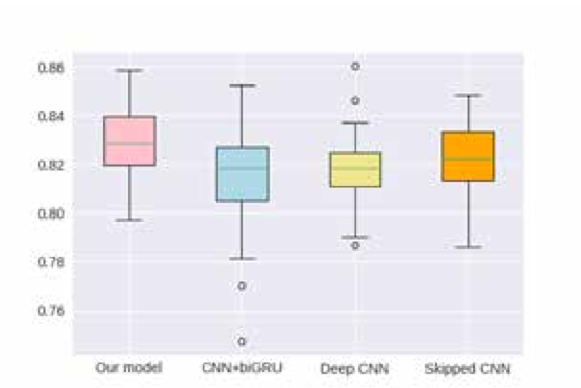


Figure 2: Comparison among different models on subtask A

4.2 Model Assessment

As final step we tested our model over the internal test set. The results obtained are reported in Table 3. As expected, the behaviour of our model in this internal test set is in compliance with respect to validation results.
As regards subtask B, we only considered the AUC score and results obtained for both model selection and assessment have proved to be inconclusive.

Run	Subtask A score
Run 1	0.858894
Run 2	0.851679
Run 3	0.8360752

Table 3: Results of single runs in internal test set

5 Results and discussion

The evaluation was done on both subtask A and B. In the following subsections a discussion of the results obtained in each subtask is provided.

5.1 Subtask A

Table 4 reports the official results for the subtask A.

SubtaskA	u/c	score	teamname
run2	u	0.74064	jigsaw
run1	u	0.73802	jigsaw
run1	**c**	**0.73425**	**fabsam**
run1	u	0.73135	YNU_OXZ
run2	**c**	**0.73091**	**fabsam**
run2	c	0.71669	NoPlaceFor..
run2	u	0.70145	YNU_OXZ
run3	**c**	**0.69482**	**fabsam**

Table 4: AMI subtask A leaderboard

Both run *fabsam.r.c.run1* and *fabsam.r.c.run1* have outperformed other constrained runs and our best run ranks third in the official leaderboard. This confirms the effectiveness of our approach.
During an error analysis we noticed that our model wrongly classifies short sentences and hate speech sentences referred to men. Nevertheless, in our best run the f1 score for *misogynous* label reaches 0.8038 while the real problem is in the 0.6647 of *aggressiveness* label. This is probably due to the small number of non-aggressive examples used to fit the model.
Different results of runs reflect the standard deviation observed during the validation phase. While scores obtained are smaller then model selection results.

5.2 Subtask B

In the following we reported our results for the subtask B.

SubtaskB	u/c	score	teamname
run2	u	0.88259	jigsaw
run3	c	0.81803	PoliTeam
run1	c	0.81369	PoliTeam
run1	**c**	**0.70512**	**fabsam**
run2	**c**	**0.70219**	**fabsam**
run2	c	0.69395	PoliTeam
run3	c	0.69133	fabsam
run3	**c**	**0.69133**	**fabsam**

Table 5: AMI Subtask B leaderboard

We used for subtask B the same model used for the other subtask. We have performed a poor validation approach using as evaluation metric the AUC. We chose to train the model merging raw and synthetic datasets. This choice led to poor performance on unseen datasets. Indeed our model was strongly affected by overfitting when it met identity terms used in training. From an error analysis we noticed that it wrongly classifies lots of sentences from synthetic dataset, while it performs very well on raw dataset.

6 Conclusion

The presence of misogynistic contents in social network is a major problem. A crucial work in this direction is the detection and recognition of this type of contents.
We propose a simple architecture based on convolutional layers. From our experiments we understood that capturing long-term dependencies produces an unstable training and poor performance in this type of subtasks. Performances of the model could be increased focusing its approach on model selection. Lastly, it could be very important to take into consideration data augmentation techniques or other sources of data to solve the unbalanced dataset issue.

References

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada, August. Association for Computational Linguistics.

Andrea Cimino, Lorenzo De Mattei, and Felice Dell'Orletta. 2018. Multi-task learning in deep neural networks at evalita 2018. *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*, pages 86–95.

E Fersini, P Rosso, and M Anzovino. 2018. Overview of the task on automatic misogyny identification at ibereval 2018. In *Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2018*, volume 2150, pages 214–228. CEUR-WS.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020. Ami @ evalita2020: Automatic misogyny identification. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 08.

Debora Nozza, Claudia Volpetti, and Elisabetta Fersini. 2019. Unintended bias in misogyny detection. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 149–155.

J. D. Rodriguez, A. Perez, and J. A. Lozano. 2010. Sensitivity analysis of k-fold cross validation in prediction error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):569–575.

Ziqi Zhang and Lei Luo. 2018. Hate speech detection: A solved problem? the challenging case of long tail on twitter. *Semantic Web*, Accepted, 10.

Ye Zhang and Byron C. Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *CoRR*, abs/1510.03820.

Jigsaw @ AMI and HaSpeeDe2: Fine-Tuning a Pre-Trained Comment-Domain BERT Model

Alyssa Lees and **Jeffrey Sorensen** and **Ian Kivlichan**
Google Jigsaw
New York, NY
`(alyssalees|sorenj|kivlichan)@google.com`

Abstract

The Google Jigsaw team produced submissions for two of the EVALITA 2020 (Basile et al., 2020) shared tasks, based in part on the technology that powers the publicly available PerspectiveAPI comment evaluation service. We present a basic description of our submitted results and a review of the types of errors that our system made in these shared tasks.

1 Introduction

The HaSpeeDe2 shared task consists of Italian social media posts that have been labeled for hate speech and stereotypes. As Jigsaw's participation was limited to the A and B tasks, we will be limiting our analysis to that portion. The full details of the dataset are available in the task guidelines (Bosco et al., 2020).

The AMI task includes both raw (natural Twitter) and synthetic (template-generated) datasets. The raw data consists of Italian tweets manually labelled and balanced according to misogyny and aggressiveness labels, while the synthetic data is labelled only for misogyny and is intended to measure the presence of unintended bias (Elisabetta Fersini, 2020).

2 Background

Jigsaw, a team within Google, develops the PerspectiveAPI machine learning comment scoring system, which is used by numerous social media companies and publishers. Our system is based on distillation and uses a convolutional neural-network to score individual comments according to several attributes using supervised training data labeled by crowd workers. Note that PerspectiveAPI actually hosts a number of different models that each score different attributes. The underlying technology and performance of these models has evolved over time.

While Jigsaw has hosted three separate Kaggle competitions relevant to this competition (Jigsaw, 2018; Jigsaw, 2019; Jigsaw, 2020) we have not traditionally participated in academic evaluations.

3 Related Work

The models we build are based on the popular BERT architecture (Devlin et al., 2019) with different pre-training and fine-tuning approaches.

In part, our submissions explore the importance of pre-training (Gururangan et al., 2020) in the context of toxicity and the various competition attributes. A core question is to what extent these domains overlap. Jigsaw's customized models (used for the second HaSpeeDe2 submission, and both AMI submissions) are pretrained on a set of one billion user-generated comments: this imparts statistical information to the model about comments and conversations online. This model is further fine-tuned on various toxicity attributes (toxicity, severe toxicity, profanity, insults, identity attacks, and threats), but it is unclear how well these should align with the competition attributes. The descriptions of these attributes and how they were collected from crowd workers can be found in the data descriptions for the Jigsaw Unintended Bias in Toxicity Classification (Jigsaw, 2019) website.

A second question studied in prior work is to what extent training generalizes across languages (Pires et al., 2019; Wu and Dredze, 2019; Pamungkas et al., 2020). The majority of our training data is English comment data from a variety of sources, while this competition is based on Italian Twitter data. Though multilingual transfer has been studied in general contexts, less is known about the specific cases of toxicity, hate speech,

---

misogyny, and harassment. This was one of the focuses of Jigsaw's recent Kaggle competition (Jigsaw, 2020); i.e., what forms of toxicity are shared across languages (and hence can be learned by multilingual models) and what forms are different.

4 Submission Details

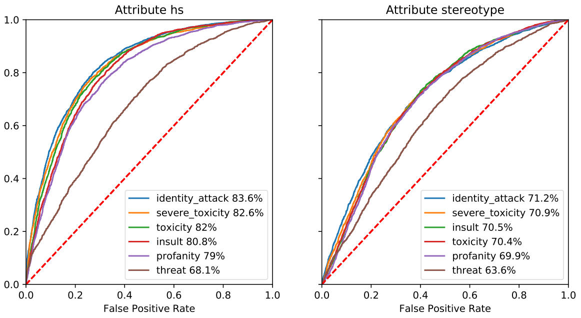


Figure 1: ROC curves for the PerspectiveAPI multilingual teacher model attributes compared to the HaSpeeDe2 attributes (hate speech and stereotype).

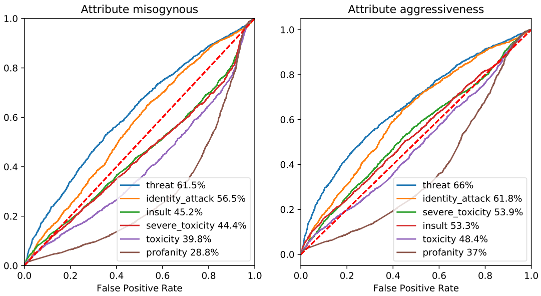


Figure 2: ROC curves for PerspectiveAPI multilingual teacher model attributes compared to the AMI attributes (misogyny and aggressiveness).

As Jigsaw has already developed toxicity models for the Italian language, we initially hoped that these would provide a preliminary baseline for the competition despite the independent nature of the development of the annotation guidelines. Our Italian models score comments for toxicity as well as five additional distinct toxicity attributes: severe toxicity, profanity, threats, insults, and identity attacks. We might expect some of these attributes to correlate with the HaSpeeDe2 and AMI attributes, though it is not immediately clear whether any of these correlations should be particularly strong.

The current Jigsaw PerspectiveAPI models are typically trained via distillation from a multilingual teacher model (that is too large to practically serve in production) to a smaller CNN. Using this large teacher model, we initially compared the EVALITA hate speech and stereotype annotations against the teacher model's scores for different attributes. The results are shown in Figure 1 for the training data. Perspective is a reasonable detector for the hate speech attribute, but performs less well for the stereotype attribute, with the identity attack model performing the best.

Using these same models on the AMI task, shown in Figure 2 for detecting misogyny proved even more challenging. In this case, the aggressiveness attribute was evaluated only on the subset of the training data labeled misogynous. In this case, the most popular attribute of "toxicity" is actually counter-indicative of the misogyny label. The best detector for both of these attributes appears to be the "threat" model.

As can be seen, the existing classifiers are all poor predictors of both attributes for this shared task. Due to errors in our initial analysis, we did not end up using any of the models used for PerspectiveAPI in our final submissions.

Category	Submission	hate speech	stereotype
news	1	0.68	0.64
	2	0.64	0.68
tweets	1	0.72	0.67
	2	0.77	0.74

Table 1: Macro-averaged F1 scores for Jigsaw's HaSpeeDe2 Submissions.

4.1 HaSpeeDe2

The Jigsaw team submitted two separate submissions that were independently trained for Tasks A and B.

4.1.1 First Submission

Our first submission, one that did not perform very well, was based on a simple multilingual BERT model fine-tuned on 10 random splits of the training data. For each split, 10% of the data was held out to choose an appropriate equal-error-rate threshold for the resulting model.

The BERT fine-tuning system used the 12 layer model (Tensorflow Hub, 2020), a batch size of 64 and sequence length of 128. A single dense layer is used to connect to the two output sigmoids which are trained using a binary cross-entropy loss using stochastic gradient descent with early stopping, which is computed using the AUC metric computed using the 10% held out slice. This model is implemented using Keras (Chollet and others, 2015).

To create the final submission, the decisions of the ten separate classifiers were combined in a majority voting scheme (if 5 or more models produced a positive detection, the attribute was assigned true).

4.1.2 Second Submission

Our second submission was based on a similar approach of fine-tuning a BERT-based model, but one based on a more closely matched training set.

The underlying technology we used is the same as the Google Cloud AutoML for natural language processing product that had been employed in similar labeling applications (Bisong, 2019).

The remaining models built for this competition and in the subsequent section are based on a customized BERT 768-dimension 12-layer model pretrained on 1B user-generated comments using MLM for 125 steps. This model was then fine-tuned on supervised comments in multiple languages for six attributes: toxicity, severe toxicity, obscene, threat, insult, and identity hate. This model also uses a custom wordpiece model (Wu et al., 2016) comprised of 200K tokens representing tokens from hundreds of languages.

Our hate speech and misogyny models use a fully connected final layer that combines the six output attributes and allows weight propagation through all layers of the network. Fine-tuning continues on using the supervised training data provided by the competition hosts using the ADAM optimizer with a learning rate of 1e–5.

Figure 3 displays the ROC curve for our second submission for each of the news and the tweets datasets as well as for both the hate speech and stereotype attributes.

Our second submission for HaSpeeDe2 consisted of fine-tuning a single model with the provided training data with a 10% held-out set. The custom BERT model was fine-tuned on TPUs using a relatively small batch size of 32.

4.2 AMI

Our submissions for the AMI task only considered the unconstrained case, due to the use of pretrained models. All AMI models were fine-tuned on TPUs using the customized BERT checkpoint and custom wordpiece vocabulary from Section 4.1.2. However, a larger batch-size of 128 was specified. All models were fine-tuned simultaneously on misogynous and aggressive labels using the provided data, where zero aggressiveness weights were assigned to data points with no misogynous labels.

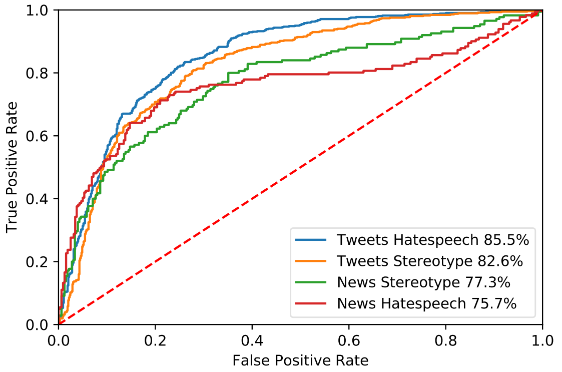


Figure 3: ROC plots for HaSpeeDe2 Test Set Labels.

Both submissions were based on ensembles of partitioned models evaluated on a 10% held-out test set. We explored two different ensembling techniques, which we discuss in the next section.

AMI submission 1 does not not include synthetic data. AMI submission 2 includes the synthetic data and custom biasing mitigation data selected from Wikipedia articles. Table 2 clearly shows that the inclusion of such data significantly improved the performance on Task B for submission 2. Interestingly, the inclusion of synthetic and bias mitigation data slightly improved the performance in Task A as well.

Task	Submission	Score
A	1	0.738
	2	0.741
B	1	0.649
	2	0.883

Table 2: Misogynous and Aggressiveness Macro-averaged F1 scores for Jigsaw's AMI Submissions.

The two Jigsaw models ranked in first and second place for Task A. The second submission ranked first among participants for Task B.

4.2.1 Ensembling Models

Both the first and second submissions for AMI were ensembles of fine-tuned custom BERT models constructed from partitioned training data. We explored two ensembling techniques (Brownlee, 2020):

- Majority Vote: Each partitioned model was evaluated using a model specific threshold. The label for each attribute was determined by majority vote among the models.
- Average: The raw models probabilities are averaged together. The combined model calculates the labels via custom thresholds determined by evaluation on a held-out set.

Thresholds for the individual models in the majority vote and average ensemble were calculated to optimize for the point on the held-out data ROC curve where $|\text{TPR} - (1 - \text{FPR})|$ is minimized.

The majority voting model performed slightly better for both the misogynous and aggressive task on the held-out sets. As such, both submissions use majority vote.

4.2.2 First Submission

Using the same configuration as Section 4.1.2, we partitioned the raw training data into ten randomly chosen partitions and fine-tuned nine of these using the 10% held out portion to compute thresholds. No synthetic or de-biasing data was included in this submission.

We include ROC curves for half of these models in Figure 4, to illustrate that they are similar but with some variance when used to score the test data.

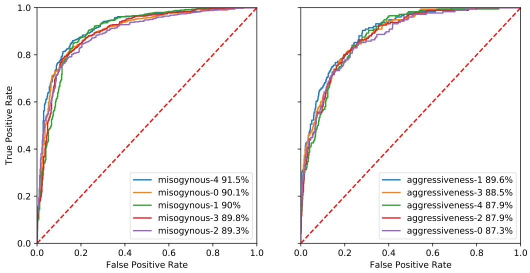


Figure 4: ROC plots for AMI test set labels for models pre-ensemble.

Our first unconstrained submission using majority vote for AMI achieved scores of 0.738 for Task A and 0.649 for Task B. The poorer score for Task B is not surprising given that no bias mitigating data or constraints were included in training.

4.2.3 Second Submission

In order to mitigate bias, we decided to augment the training data set using sentences sampled from the Italian Wikipedia articles that contain the 17 terms listed in the identity terms file provided with the test set data. These sentences were labeled as both non-misogynous and non-aggressive. 11K sentences were used for this purpose, with the term frequencies summarized in Table 3.

Identity Term	Sentence Count
donna	4306
donne	3100
femmine	1275
femmina	652
fidanzata	538
nonna	378
mamma	269
casalinga	256
casalinghe	187
compagne	132
compagna	34
mamme	24
fidanzate	12
nonne	11
matrone	9
matrona	8
morosa	6

Table 3: Term frequency in Wikipedia sampled sentences for bias mitigation.

The second submission employed the same partitioning of data with a held-out set. However the unconstrained data included the raw training data, the provided synthetic data and our de-biasing term data. As with submission 1, majority vote was used with custom thresholds determined by evaluation on the held-out set.

Our first unconstrained submission for AMI achieved scores of 0.741 for Task A and 0.883 for Task B.

5 Error Analysis

We discuss an informal analysis of the errors we observed with each of these tasks. Aside from the typical questions regarding data annotation quality, and the small sample sizes, we observed some particular instances of avoidable errors.

5.1 HaSpeeDe2 Errors

Looking at the largest incongruities as shown in Table 4 it is clear that context, which is unavailable to our models, and presumably to the mod-

ID	Comment	HS	Score
11355	@user @user @user Giustissimo, non bisogna mai nascondersi nelle ideologie, sopratutto oggi perché non esistono più. Sta di fatto, che le cose più aberranti che leggi oggi sui giornali hanno sempre@a@che fare con stranieri... o rom URL	1	.00001
10803	#Simone di #CasalBruciato, #Roma: "Certi rom sono cittadini italiani, ma non sono uguali a noi. Uguali non è il termine più giusto da usare". URL	1	.00003
11288	I SOLDI DEI DISABILI AI MIGRANTI La regione Emilia Romagna destina la metà dei fondi destinati alle fasce deboli a progetti per i richiedenti asilo A Reggio Emilia il 69% delle risorse stanziate sono state utilizzate ai richiedenti asilo #PRIMAGLIITALIANI URL	1	.00003
10420	#MeNeFottoDi questi sfigati #facciamorete che continuano a giustificare ogni crimine commesso da immigrati... non fate rete, FATE SCHIFO... #facciamociFURBI	0	0.99996
11189	@user Naturalmente in questo caso alla faccia dei comunisti e dei migranti stitici!	0	0.99996
10483	@user SCHIFOSA IPOCRITA SPONSORIZZI I MUSSULMANI E POI VOI DARE I DIRITTI ALLE DONNE SI VEDE CHE SEI POSSEDUTA DAL DIAVOLO SEI BUGIARDA BOLDRINA SAI SOLO PROTESTARE POI TI CHIEDI PERCHÉ IL VERO ITALIANO TI ODIA PERCHÉ SEI UNA SPORCA IPOCRITA	0	0.99995

Table 4: Largest Errors for hate speech classifier on HaSpeeDe2 Tweet data

erators, is important for determining the author's intent. The use of humor and the practice of quoting text from another author are also confounding factors. As this task is known to be hard (Vigna et al., 2017; van Aken et al., 2018), the edge cases display these confounding reasons. Additionally, as evidenced by the last comment, the subtlety of hate speech that is directed toward the designated target for this challenge has not been well captured.

The BERT model that we fine-tuned for this application is cased, and we see within our errors frequent use of all-caps text. However, lower casing the text has almost no effect on the scores, suggesting that the BERT pre-training has already linked the various cased versions of the tokens in the vocabulary.

We analyzed the frequency of word piece fragments in the data and saw no correlation between misclassification and the presence of segmented words. This suggests that vocabulary coverage in the test set does not play a significant role in explaining our systems' errors.

Considering the sentence with the high model score for hate speech, several single terms are tagged by the model. For example the term "sfigati" occurs only once in the training data in a sentence that is marked as non-hate speech. However, this term is not in our vocabulary and gets split into pieces "sfiga##ti", and the prefix "sfiga" appears in two out of three training examples that are marked hate speech—exactly the kind of data sparsity that leads to unwanted bias. Using a larger amount of training data, even if it creates an imbalance, is one way to address this, as we did in the case of the AMI challenge.

5.2 AMI

Because we are using ensemble models trained on partitions of the training set, we observe that the highest-scoring test samples that are marked non-misogynous and non-aggressive, as well as the lowest-scoring misogynous and aggressive comments, vary from model to model. However, we display the most frequently occurring mistakes across all ten ensembles in Table 5.

Regarding the false alarms, these comments appear to be mislabeled test instances, and there is ample support for this claim in the training data. The first comment combines both uppercase and a missing space. While it's true that subjunctive mode is not well represented in the training data, lower casing this sentence produces high scores. This is also the case with the third example. The second error seems more subtle, perhaps an attempt at humor, but one with no salient misogyny terms.

6 Bias

Because the identity terms for AMI are not observed with a high frequency in the test data, we restrict our analysis to the synthetic data set. We find wide variation in the performance of our individual models, with one model exhibiting very poor performance across the subgroups. The summary of the AUC measurements for these models

ID	Comment	Error Type
5466	Odio Sakura per il semplice motivo che qualunque cosa faccia o dica Naruto lei lo prende a schiaffi o a pugniHA CHIESTO COME STA SAI DIOSANTO BRUTTA STRONZA MA CONTRALLI MADONNA SPERO CHE TI UCCIDANOscusami Sarada	Missed Detection
5385	porca troia la prenderei a schiaffi se non fosse mia madre	Missed Detection
5819	ma in tutto ciò lo scopo di anna qual è? far soffrire il mio protetto? IO TI AMMAZZO COI LANCIAFIAMME OH #TemptationIsland	Missed Detection
5471	@danielita8811 Che bel culo tutto da sfondare	False Alarm
5604	@coppiacalda2 Che bel culo da inculare	False Alarm

Table 5: Persistent errors for AMI across different ensembles.

are shown in Figure 5, Figure 6, and Figure 7 using the technique presented in (Borkan et al., 2019). There does not appear to be a systemic problem with bias in these models, but judging based only upon synthetic data is probably unwise. The single term "donna" from the test set shows a subgroup AUC that drops substantially from the background AUC for nearly all of the models, perhaps indicating limitations of judging based on synthetic data.

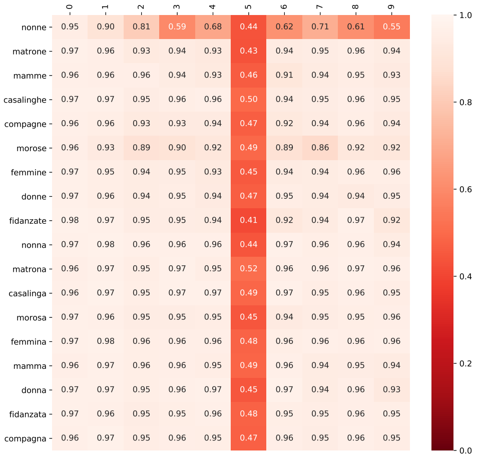


Figure 5: Subgroup AUC

7 Conclusions and Future Work

Both of these challenges dealt with issues related to content moderation and evaluation of user-generated content. While early research raised fears of censorship, the ongoing challenges platforms face have made it necessary to consider the potential of machine learning. Advances in natural language understanding have produced models that work surprisingly well, even ones that are able to detect malicious intent that users try to encode in subtle ways.

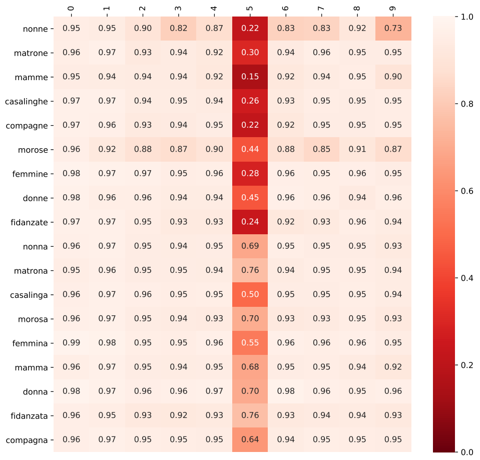


Figure 6: Background Positive, Subgroup Negative AUC

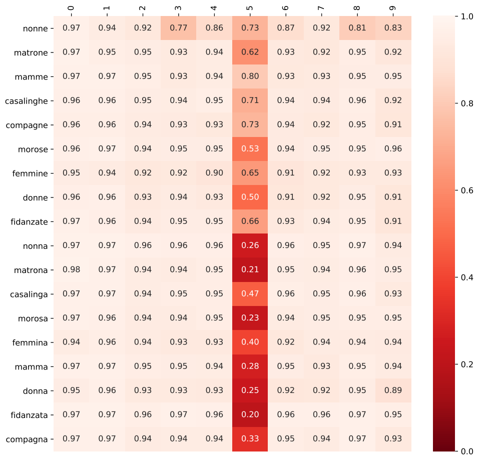


Figure 7: Background Negative, Subgroup Positive AUC

Our particular approach to the EVALITA challenges represented an unsurprising application of what has now become a textbook technique: leveraging the resources of large pre-trained models. However, many participants achieved nearly similar performance levels in the constrained task. We regard this as a more impressive accomplishment.

Jigsaw continues to apply machine learning to support publishers and to help them host quality online conversations where readers feel safe participating. The kinds of comments these challenges tagged are some of the most concerning and pernicious online behaviors, far outside of the norms that are tolerated in other public spaces. But humans and machines both still misinterpret profanity for hostility, and tagging humor, quotations, sarcasm, and other legitimate expressions for moderation remain serious problems.

Challenges like the AMI and HasSpeede2 competitions underscore the importance of understanding the relationships between the parties in a conversation, and the participants' intents. We are greatly encouraged that attributes that our systems do not currently capture were somewhat within the reach of our present techniques—but clearly much work remains to be done.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Ekaba Bisong, 2019. *Google AutoML: Cloud Natural Language Processing*, pages 599–612. Apress, Berkeley, CA.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 491–500.

Cristina Bosco, Tommaso Caselli, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Viviana Patti, Irene Russo, Manuela Sanguinetti, and Marco Stranisci. 2020. Hate speech detection task second edition (haspeede2) at evalita 2020 task guidelines. `https://github.com/msang/haspeede/blob/master/2020/HaSpeeDe2020_Task_guidelines.pdf`.

Jason Brownlee. 2020. How to develop voting ensembles with python. `https://machinelearningmastery.com/voting-ensembles-with-python/`, September.

Francois Chollet et al. 2015. Keras.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Paolo Rosso Elisabetta Fersini, Debora Nozza. 2020. Ami @ evalita2020: Automatic misogyny identification. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online, July. Association for Computational Linguistics.

Jigsaw. 2018. Jigsaw toxic comment classification challenge. `https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge`, March.

Jigsaw. 2019. Jigsaw unintended bias in toxicity classification. `https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification`, July.

Jigsaw. 2020. Jigsaw multilingual toxic comment classification. `https://www.kaggle.com/c/jigsaw-multilingual-toxic-comment-classification`, July.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing & Management*, 57(6):102360.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July. Association for Computational Linguistics.

Tensorflow Hub. 2020. Mulilingual L12 H768 A12 V2. `https://tfhub.dev/tensorflow/bert_multi_cased_L-12_H-768_A-12/2`, August.

Betty van Aken, Julian Risch, Ralf Krestel, and Alexander Löser. 2018. Challenges for toxic comment classification: An in-depth error analysis. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 33–42, Brussels, Belgium, October. Association for Computational Linguistics.

F. D. Vigna, A. Cimino, Felice Dell'Orletta, M. Petrocchi, and M. Tesconi. 2017. Hate me, hate me not: Hate speech detection on facebook. In *ITASEC*.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China, November. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

PoliTeam @ AMI: Improving Sentence Embedding Similarity with Misogyny Lexicons for Automatic Misogyny Identification in Italian Tweets

Giuseppe Attanasio, Eliana Pastor
Department of Control and Computer Engineering
Politecnico di Torino, Italy
{giuseppe.attanasio,eliana.pastor}@polito.it

Abstract

We present a multi-agent classification solution for identifying misogynous and aggressive content in Italian tweets. A first agent uses modern Sentence Embedding techniques to encode tweets and a SVM classifier to produce initial labels. A second agent, based on TF-IDF and Misogyny Italian lexicons, is jointly adopted to improve the first agent on uncertain predictions. We evaluate our approach in the Automatic Misogyny Identification Shared Task of the EVALITA 2020 campaign. Results show that TF-IDF and lexicons effectively improve the supervised agent trained on sentence embeddings.

Italiano. *Presentiamo un classificatore multi-agente per identificare tweet italiani misogini e aggressivi. Un primo agente codifica i tweet con Sentence Embedding e una SVM per produrre le etichette iniziali. Un secondo agente, basato su TF-IDF e lessici misogini, è usato per coadiuvare il primo agente nelle predizioni incerte. Applichiamo la soluzione al task AMI della campagna EVALITA 2020. I risultati mostrano che TF-IDF e i lessici migliorano le performance del primo agente addestrato su sentence embedding.*

1 Introduction

The increasing adoption of online communication systems we experienced in the last decades brought the rise of many public forums for our own opinions, such as forums, blogs, and social networks. In these platforms, where access cannot - and must not - be restricted to anyone, the problem of misconduct and hateful content became soon compelling. The protection of the most targeted subjects, such as races, ethnicities, religious parties, genders, and sexual orientations, is of paramount importance. Violence against women manifests in social networks every time the offensive language targets women directly or indirectly (Ellsberg et al., 2005). We refer to these cases as misogynous speech. As platform owners are updating their regulatory terms at an increasing pace[1], the high number of contents due to a fast publication rate still pose a challenge to monitoring systems.

Many recent works in the NLP community show effective results in online monitoring of hate speech (Fortuna and Nunes, 2018) and misogynous contents (Pamungkas et al. (2020), Frenda et al. (2019), Anzovino et al. (2018)). Furthermore, research communities propose evaluation initiatives (Basile et al. (2019), Bosco et al. (2018)) to challenge NLP practitioners in finding novel solutions to shared tasks. Among these, the AMI shared task proposed at EVALITA 2020 (Basile et al., 2020) focuses on automatic identification of misogynous content on Twitter in Italian (Elisabetta Fersini, 2020).

The task counts two main subtasks. The goal of the first subtask, Subtask A - Misogyny & Aggressive Behaviour Identification, is the identification of misogynous speech in tweets, and in case of misogyny, the classification of an aggressive language. Subtask B - Unbiased Misogyny Identification, aims at classifying misogynous speech while guaranteeing the fairness of the model (in terms of unintended bias) on a synthetic dataset. The unintended bias is a known phenomenon in natural lan-

---

[1] https://www.theverge.com/2020/3/5/21166940/twitter-hate-speech-ban-age-disability-disease-dehumanize, https://www.theverge.com/2020/8/11/21363890/facebook-blackface-antisemitic-stereotypes-ban-misinformation, https://www.theguardian.com/technology/2020/jun/29/reddit-the-donald-twitch-social-media-hate-speech

guage models and recent works address its identification and mitigation (Dixon et al. (2018), Nozza et al. (2019), Kennedy et al. (2020)).

In this work, we describe our solution to address the AMI shared task. We propose a multi-agent classification. The system uses recent Sentence Embedding techniques to encode tweets and a SVM classifier to produce initial labels. A second agent, based on TF-IDF and Misogyny Italian lexicons, is jointly adopted to improve the first agent on uncertain predictions. Results show that the TF-IDF and misogyny lexicons effectively improve sentence embeddings. For both subtasks, we chose the constrained approach, effectively using only the data provided by the organizers.

2 Description of the system

Recent work has pointed out the efficiency of sentence embeddings in many downstream tasks, such as sentiment classification. Meanwhile, NLP practitioners strive to migrate the existing solutions to languages different from English. As such, classical language models are trained on large parallel corpora, and multi-lingual, pre-trained models are published for later uses.

In this work, we adopt a multi-agent classification procedure to address each proposed subtask. Firstly, we encode tweets to their sentence embeddings using a pre-trained multi-lingual sentence encoder. Next, we train a supervised classifier (the first agent) on the latent embedding space. In parallel, we extract the smoothed TF-IDF of tweets and enhance the representation with features built upon Hate Speech and Misogyny lexicons. This representation is then used to train a supervised classifier (the second agent). Finally, we propose a classification schema where uncertain predictions from the first agent are corrected with certain ones from the second agent.

The following paragraphs describe the data preprocessing step, expand on the classification system, and provide insights on its application to subtasks A and B.

2.1 Sentence embedding

Researchers devoted significant work to the empirical construction of sentence embeddings for the English language (Giorgi et al. (2020), Wang and Kuo (2020), Reimers and Gurevych (2019), Cer et al. (2018)). The most recent studies leverage high-quality language models, such as the BERT or Transformer-XL families, to build embeddings that properly transfer to several downstream tasks. Extending monolingual models, other works assess the generalization performance of language models pre-trained on multi-lingual corpora, producing sentence embeddings either aligned between languages (Reimers and Gurevych, 2020) or not (Aluru et al., 2020).

We build sentence embeddings testing two models. On the one hand, we use (Aluru et al., 2020), a monolingual BERT-based model originally fine-tuned from multilingual BERT on an Italian corpus for hate-speech detection tasks. The model is then fine-tuned on our specific subtasks. On the other hand, we choose the multilingual adaptation of Sentence-BERT (Reimers and Gurevych (2020)), which is based on the DistilBERT architecture (Sanh et al. (2019)). We use the implementation[2] built on top of the *transformers* library. Since results for the monolingual BERT were not encouraging from the beginning, in any of the subtasks, we will focus the discussion on multi-lingual Sentence-BERT.

Further, we run a fine-tuning round on multilingual Sentence-BERT to our specific subtasks. To tune the initial embeddings, we optimize a contrastive loss on pairs generated from the training set. For any pair of tweets, if the ground truth labels are the same (e.g. both misogynous or both non-aggressive) the distance between the two embeddings is decreased, while it is increased otherwise. Since computing the set of potential pairs is hard, we sample only 20% of the initial tweets, namely S, compute all the P possible pairs among those, where $|P| = (|S| \cdot |S - 1|)/2$, and use them for fine-tuning. We anticipate this partial fine-tuning achieved worse results than the original model and leave other fine-tuning strategies as future work.

The final agent is then a supervised classifier trained on multi-lingual sentence embeddings (referred as the *SE* agent). We use a Support Vector Machine (SVM) with Radial Basis Function kernel, which achieves the best results on our validation set. Please refer to Section 3 for more details on parameter configuration and performance.

2.2 TF-IDF and Misogyny Lexicons

---

[2] https://github.com/UKPLab/sentence-transformers

Lexicons	#Words	Type of words
Sexist	138	Misogynous and sexist
Profanity	4	Vulgar and swear
Sexuality	7	Sexual references
Female body	6	Feminine body

Table 1: Description on misogynous lexicon.

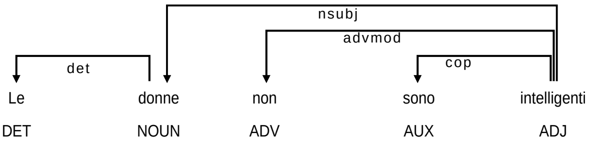


Figure 1: Example of dependency-based parse tree with sentiment polarity inversion.

Pre-processing. We firstly pre-process the data by replacing every URL found in tweets with the meta-token *LINK*. Next, we perform tokenization and lemmatization using the spaCy's[3] pre-trained Italian core model *it_core_news_lg*.

Input features. We use a smoothed TF-IDF vectorization of pre-processed tweets. We then enrich word representations using lexicons to encode misogynous speech and tweet sentiment.

(i) Misogynous lexicon. Misogynous tweets often contain sexist slurs, swear words, and sexual references. We include specific lexicons as input features for dealing with hate and misogynous speech (Frenda et al., 2018). We collect Italian lexicons from multiple online sources. We divide lexicons into the following categories: sexists, profanity, sexuality and female body as described in Table 1. The complete list of Italian lexica and sources are available at our repository[4]. As for the text of the tweet, lexicons are firstly lemmatized using spaCy. We then derive 4 features, one for each misogynous lexicon category. For a given category, we first count the occurrences of the corresponding lexicons in each tweet. We then normalize the occurrence with the tweet word count.

(ii) Sentiment Lexicon. We use a sentiment lexicon to characterize the polarity of tweets. The sentiment of words in a tweet is obtained with the OpeNER Italian Sentiment Lexicon (Russo et al., 2016). This sentiment lexicon consists of 24.293 lexical entries annotated with positive, negative and neutral polarity. In our analysis, we consider only positive and negative polarity.

Evaluating the polarity of an individual word in a tweet without considering its context, however, prevents from considering the role of negation on sentence polarity. To address this issue, we consider the following negation handling technique based on the dependency-based parse tree. We search in the parse tree extracted by spaCy for words affected by negation. For these words, we invert the polarity, if it is available. As an example, consider the phrase “le donne non sono intelligenti” (women are not intelligent). Figure 1 shows the extracted parse tree. The polarity of the word “intelligenti” (intelligent) is inverted, from positive to negative, since it is affected by negation.

Note that, as for the tweet text, we lemmatize sentiment lexicons. Finally, we extract 2 features that capture the tweet polarity. These are obtained by counting the number of words with positive and negative polarity respectively and then normalizing them by the tweet word count.

(iii) Additional features. Tweets may contain quotations of misogynous content, without being misogynous themselves. We hence consider as an additional feature the relative frequency of quotation marks. We also consider as a feature the length of the tweet (i.e. number of characters).

Finally, we train a supervised classifier (the second agent, referred as *Lex* agent) on the TF-IDF representation enriched with the additional features previously described. As for the first agent, we use a SVM with Radial Basis Function kernel model. We refer the reader again to Section 3 for details on the experimental setting.

2.3 Multi-agent prediction

We designed the multi-agent system to maximize prediction confidence by using only predictions with a high probability score. Specifically, we deem a prediction as confident if its associated probability score is above a given threshold.

We produce the final classification label by combining the outcomes of the two agents as follows. We first generate a prediction label and a score associated with it using the first agent. It entails encoding a given test point with Sentence-BERT and running the inference with SVM (*SE* agent). Afterward, we use the confidence threshold to decide whether to keep the label or not. If the *SE*'s prediction is not confident, we probe the second agent, which is built upon TF-IDF and misogyny lexicons (*Lex* agent). Finally, if *Lex*'s

---

[3] https://spacy.io/

[4] https://github.com/g8a9/ami20-improving-embedding

prediction is confident, we choose its label as the final one. If this is not the case, we rollback to *SE*'s class label. We kept the confidence threshold value as a hyper-parameter of the system.

By design, the proposed solution provides only confident prediction labels, either from the *SE* or the *Lex* agent. We applied the multi-agent classification procedure for both subtasks.

2.4 Approach to subtask A

In this task, participants have to assign a label indicating whether a tweet is misogynous or not. Then, limited to the misogynous ones, a second label should tell if the tweet is also aggressive.

We apply our multi-agent classification in a chained-prediction fashion. Specifically, we train a first instance of the system on the binary misogyny problem and label every tweet. In this step, we use the complete corpus. Next, we train a second instance on the binary aggressiveness problem. We feed the model with tweets predicted as misogynous on the previous step and produce a class label for those only. Finally, we label all the non-misogynous tweets as non-aggressive.

This strategy presents advantages and drawbacks since the predictions are chained. On the one hand, the two models are independent and can separately learn a simpler problem. On the other hand, this design lets errors on the misogyny prediction propagate to the aggressiveness one. We further discuss the matter in Section 4.

2.5 Approach to subtask B

For this task, we employ our multi-agent model (*SE*+*Lex* agents) with no modifications. Since we desire the model to encode also the structure and form of synthetic sentences, we train the model using the whole corpus.

3 Results

In this section, we firstly describe the experimental setting and the hyper-parameter tuning. We then report and comment experimental results of our multi-agent system. Further, to evaluate the effects of the two agents, we report the results of the system using only the *SE* or the *Lex* agent. The versions using only the *SE* agent or the *Lex* agent correspond to ids *run1* and *run2* respectively. The id *run3* is assigned to the multi-agent system.

Table 3 shows the F1 scores for misogyny and aggressiveness classes on the test set. All our runs show lower performance in the aggressiveness identification. We analyze and discuss this aspect in Section 4.

Rank	Team	Score
1	jigsaw.u.run2	0.7406
...	...	...
12	**PoliTeam.c.run3**	**0.6835**
13	MDD.c.run1	0.6820
14	**PoliTeam.c.run1**	**0.6809**
15	MDD.u.run2	0.6679
16	AMI_the_winner.c.run1	0.6653
17	**PoliTeam.c.run2**	**0.6473**
...	...	...
20	NoPlaceForHateSpeech.c.run3	0.4902

Table 2: Official results for subtask A

Run	Misogyny	Aggressiveness
SE (run1)	0.7688	0.5931
Lex (run2)	0.7222	0.5724
SE+*Lex* (run3)	0.7750	0.5920

Table 3: F1 score for subtask A

3.1 Experimental setting

To perform hyper-parameter optimization and model selection, we split the input data in training and validation data using random stratified sampling on both misogyny and aggressiveness labels. We used 20% of data as validation.

We ran a grid search over multiple classifiers as Support Vector Machines (SVM), Deep Feed Forward Neural Network, Random Forest, Logistic Regression, and their input parameters. The evaluation was performed using the first agent as reference. SVM with Radial Basic Function kernel with *gamma="scale"* and *C=10* achieved highest performance on F1 score for misogynous class on the validation set. We used this configuration for the supervised classifier of the second agent.

For the TF-IDF, we tuned the n-grams from $n=1$ to $n=3$, and the number of maximum tokens from 5.000 to 10.000. To estimate the best configuration, we trained the SVM classifier with tuned parameters on the vectorized data, and evaluated the classification F1 score on the binary misogyny detection problem on the validation set. We achieved the highest F1 score with unigrams and 10.000 tokens as maximum vocabulary size.

The last hyper-parameter is the confidence threshold value for the multi-agent system. We evaluated the F1 score for the misogynous class on validation data varying the confidence threshold in the range $[0.6, 0.95]$. Best performance are obtained with a confidence threshold of 0.9.

Rank	Team	Score
1	jigsaw.u.run2	0.8826
2	**PoliTeam.c.run3**	**0.8180**
3	**PoliTeam.c.run1**	**0.8137**
4	fabsam.c.run1	0.7051
5	fabsam.c.run2	0.7022
6	**PoliTeam.c.run2**	**0.6940**
...	...	...
11	MDD.u.run3	0.6013

Table 4: Official results for subtask B

The hyper-parameter settings resulting from the experimental tuning are used for both the subtasks.

3.2 Subtask A

The score for subtask A is computed by averaging the F1 measures estimated for the *misogynous* and *aggressiveness* classes. Table 2 shows the official results. Our multi-agent system (run3) achieves our highest result. It is ranked 12th out of all submissions and 7th if we consider just constrained ones. While our TF-IDF and misogyny lexicon agent (run2) reaches our worst result, its introduction improves the agent trained on sentence embedding. The average F1 score increases from 0.6809 of the *SE* agent (run1) to 0.6835.

3.3 Subtask B

The score for subtask B is the weighted combination of AUC computed on the test tweets and three per-term AUC-based bias scores computed on the synthetic dataset. We refer the reader to (Elisabetta Fersini, 2020) for the complete description of the evaluation metrics.

Table 4 shows the official results. Our multi-agent system is ranked 2nd out of all submissions and 1st if only constrained runs are considered. As for subtask A, the *Lex* agent improves the performance of the *SE* one.

4 Discussion and Conclusions

Results show that the introduction of the TF-IDF and lexicons effectively improves the solution based on sentence embedding. This finding stands as the most significant contribution of this work, and we believe that it can drive future system designs. However, results on the test set reveal that we got wrong on some choices that affected the final performance.

4.1 Analysis on subtask A

Our multi-agent system missed the target on the aggressiveness detection task. As reported in Table 3, aggressiveness has a notable low F1 score. We think this is due to bad choices in training the system. (i) We used for the aggressiveness task only on the misogynous portion of the input data. This sub-set has an imbalanced class distribution with a prevalence of aggressive tweets. We did not re-balance the dataset, and our predictions produced many false positives on the test. (ii) Since we did not train the aggressiveness system on non-misogynous (and non-aggressive) tweets, whenever the misogyny system produces a false positive, the aggressiveness detector faces a completely new data point, out of its training distribution. (iii) Finally, we naively replicated the best algorithm and configuration found on the misogyny task to the aggressiveness one.

Notably, the number of misogynous false negatives which forced an aggressive tweet to be classified as non-aggressive by our chained approach (see Section 2.4) is 16 out of 365 total errors. This further enforces the conclusion that the majority of errors were due to bad training choices on the aggressiveness task and not the chained approach.

4.2 Analysis on subtask B

The multi-agent (*SE+Lex*) errors are 72 false negatives and 157 (x2.2) false positives. With a posterior error analysis on the test tweets, we identified several factors that contribute to misclassification.

Bias on parts of the body. Our system struggles with parts of the body that have sexual and misogynous reference based on the context. These words polarize the assignment to the misogynous class. As an example, 15% of false positives contain the word "gola" (throat). This behavior somewhat mimics the bias of models towards specific identity terms.

Self-mocking references. Another category hard to model is self-referencing text containing misogynous speech. While the tone of these tweets is auto-ironic or self-mocking, the model decontextualizes and produces false positives.

Targeted gender. In these tweets, the model correctly detects the hateful tone of voice but fails at identifying the gender of the target subject. As such, it predicts tweets attacking males as misogynous. This problem gets harder when the targeted gender can be only inferred by prior knowledge of tagged profiles (e.g. @bonucci_leo19, a male Italian football player).

Reported misogynous speech. Another difficult scenario to model is the reported or quoted misogynous speech. Frequently, users quote an unpleasant, misogynous passage while trying to support the exact opposite message. It can happen directly, using quotation marks, or indirectly by citing the original speaker.

We provide a list of tweets for each of the aforementioned categories as supplementary material[5].

Conclusion. In this work, we presented our solution to the AMI shared task at the EVALITA 2020 evaluation campaign. Our system is based on two models, the *SE* and *Lex* agents, which we built using sentence embedding techniques and TF-IDF enriched with misogyny lexicons respectively. We addressed both subtask A and B, limited to constrained runs. The approach fell short on the subtask A, while showed promising results on subtask B. Besides, results show the *Lex* agent effectively improves the performance of the *SE* agent.

Acknowledgments

This work was supported by the DataBase and Data Mining Group of Politecnico di Torino.

References

Sai Saketh Aluru, Binny Mathew, Punyajoy Saha, and Animesh Mukherjee. 2020. Deep Learning Models for Multilingual Hate Speech Detection. *arXiv:2004.06465 [cs]*, April.

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Nozza Debora, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, Manuela Sanguinetti, et al. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *SemEval-2019*, pages 54–63. ACL.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Cristina Bosco, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. 2018. Overview of the EVALITA 2018 hate speech detection task. In *EVALITA 2018*, pages 1–9. CEUR.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder. *arXiv:1803.11175 [cs]*, April.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and Mitigating Unintended Bias in Text Classification. In *AAAI/ACM AIES 2018*, pages 67–73, December.

Paolo Rosso Elisabetta Fersini, Debora Nozza. 2020. AMI @ EVALITA2020: Automatic Misogyny Identification. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Mary Ellsberg, Lori Heise, World Health Organization, et al. 2005. Researching violence against women: a practical guide for researchers and activists.

Paula Fortuna and Sérgio Nunes. 2018. A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Surveys*, 51(4):85:1–85:30, July.

Simona Frenda, Bilal Ghanem, Estefanía Guzmán-Falcón, Manuel Montes-y Gómez, Luis Villasenor-Pineda, et al. 2018. Automatic expansion of lexicons for multilingual misogyny detection. In *EVALITA 2018*, pages 1–6. CEUR-WS.

Simona Frenda, Bilal Ghanem, Manuel Montes-y Gómez, and Paolo Rosso. 2019. Online hate speech against women: Automatic identification of misogyny and sexism on twitter. *Journal of Intelligent & Fuzzy Systems*, 36(5):4743–4752.

John M. Giorgi, Osvald Nitski, Gary D. Bader, and Bo Wang. 2020. DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations. *arXiv:2006.03659 [cs]*, June.

Brendan Kennedy, Xisen Jin, Aida Mostafazadeh Davani, Morteza Dehghani, and Xiang Ren. 2020. Contextualizing hate speech classifiers with post-hoc explanation. pages 5435–5442, July.

Debora Nozza, Claudia Volpetti, and Elisabetta Fersini. 2019. Unintended bias in misogyny detection. In *IEEE/WIC/ACM WI 2019*, pages 149–155.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing & Management*, 57(6):102360.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv:1908.10084 [cs]*, August.

---

[5] https://github.com/g8a9/ami20-improving-embedding

Nils Reimers and Iryna Gurevych. 2020. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. *arXiv:2004.09813 [cs]*.

Irene Russo, Francesca Frontini, and Valeria Quochi. 2016. OpeNER sentiment lexicon italian - LMF.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Bin Wang and C.-C. Jay Kuo. 2020. SBERT-WK: A Sentence Embedding Method by Dissecting BERT-based Word Models. *arXiv:2002.06652 [cs]*, June.

MDD @ AMI: Vanilla Classifiers for Misogyny Identification

Samer El Abassi
Faculty of Mathematics and
Computer Science
University of Bucharest
`samer.el-abassi@s.unibuc.ro`

Sergiu Nisioi
Human Language Technologies
Research Center,
University of Bucharest
`sergiu.nisioi@unibuc.ro`

Abstract

In this report[1], we present a set of vanilla classifiers that we used to identify misogynous and aggressive texts in Italian social media. Our analysis shows that simple classifiers with little feature engineering have a strong tendency to overfit and yield a strong bias on the test set. Additionally, we investigate the usefulness of function words, pronouns, and shallow-syntactical features to observe whether misogynous or aggressive texts have specific stylistic elements.

1 Introduction

This paper discusses our submission (team MDD) to the Evalita 2020 Automatic Misogyny Identification Shared Task (Elisabetta Fersini, 2020; Basile et al., 2020) (Task A). Our methods consist of a set of simple vanilla classifiers that we employ to assess their effectiveness on the datasets provided by the organizers. The systems we submitted for evaluation use a logistic regression classifier with little hyperparameter tuning or feature engineering, being trained on tf-idf and average word embeddings pooling. Previous reports on misogyny (Fersini et al., 2018b,a) and aggressiveness (Basile et al., 2019) detection indicate that support vector machines and logistic regression classifiers effectively identify these patterns in social media posts. Furthermore, vanilla classifiers with little feature engineering were successfully used for other shared tasks, such as identifying dialectal varieties (Ciobanu et al., 2016; Zampieri et al., 2017) or native language identification (Malmasi et al., 2017), where high scores were obtained by simple approaches using SVMs or logistic regression classifiers.

The classifiers we built achieved a relatively good accuracy on our cross-validation tests; however, for this competition, the results obtained by our systems are not among the top-scoring ones and show to be misfit, with a significant tendency towards biased results.

In addition to the description of our submissions, in this report, we analyze the errors of our systems, and we bring into discussion several and topic-independent features to: 1) test the effectiveness of part-of-speech n-grams, function words, and pronouns on the task of identifying misogynous and aggressive texts on social media and 2) observe whether texts labeled as misogynous or aggressive have a particular bias towards certain grammatical structures.

2 System Description

At the basis of submissions is the logistic regression classifier with liblinear (Fan et al., 2008) optimizer, l2 penalty, and regularization constant $C = 3$ that we chose based on different cross-validation iterations. In addition, we introduced a heuristic at the prediction time in which we predict a text not to be aggressive if it was not categorized as misogynous.

The difference between our three submissions for Task A consist in the feature extraction process, where:

MDD.A.r.c.run1 is the logreg model trained on td-idf of word n-grams, n ranging from 1 to 5

MDD.A.r.u.run2 is the logreg model trained on pre-trained glove twitter embeddings of size 200 on 27 billion words[2]

MDD.A.r.u.run3 is the logreg model using spaCy (Honnibal and Montani, 2017) FastText

---

[2] English model GloVe.twitter.27B.200d `https://nlp.stanford.edu/projects/GloVe/`

CBOW embeddings pre-trained on Wikipedia and OSCAR (Common Crawl)[3].

The second run is trained on English glove embeddings that surprisingly contain the representation of more than half of our Italian vocabulary, i.e., approximately 9500 words out of the total 15,000 size of the vocabulary of our data. The English glove embeddings cover code-switching, emojis, and basic Italian words. Despite having the lowest evaluation score of our submissions (0.666 macro f1), we believe it provides a decent estimation for identifying non-misogynous texts.

2.1 Feature Extraction

Our feature extraction processes for the submissions are simple, the first one uses the tf-idf vectorizer (Buitinck et al., 2013) on word n-grams, with n ranging from 1 to 5, to cover more of word context. Tf-idf features were used for their ability to categorize the importance of an n-gram with respect to the entire corpus. The second feature set is based on pre-trained word representations by calculating every word's embeddings in the text to eventually get an average representation. For words not present in the embeddings, an array of zeroes with the same dimensions was used.

Preprocessing

Our submissions use raw, un-processed texts, including tags and URLs. We have also experimented with different preprocessing and feature extraction steps for which we did not make any submission. We consider multiple approaches in this direction:

1. **clean** - changing the entire text to lowercase, removing hashtags, and links

2. **nps** - replacing the text with the noun phrases; these features contain the nouns and surrounding attributes that can highlight misogynous remarks

3. **fct words** - classification based on function word occurrence; these words cover stylistic and information of texts. We have collected a list of conjunctions, prepositions, connectors, etc. for Italian for this purpose.

4. **POS n-grams** - n-grams with n ranging from 1 to 5 over part-of-speech tags; these features would indicate a certain syntactic and stylistic pattern in misogynous or aggressive texts

5. **pronouns** - n-grams with n ranging from 1 to 5 over the pronouns and pronoun properties from the texts; we observed an increased usage in aggressive expressions of second-person pronouns

6. **filter POS** - n-grams over a filtered set of words and POS tags.

For POS tagging and noun phrases extraction, we use the default outputs from the Italian model for spaCy trained on the dataset provided by Bosco et al. (2014). In addition, we use the tag for each word that covers an entire set of features separated by whitespace; e.g., ”Gender=Masc, Number=Sing, Person=2, PronType=Prs” becomes: ”Masc Sing 2 Prs”.

We expect the noun phrases to be less effective at detecting aggressive behaviour because aggressiveness often involves *verbal constructs* and actions.

3 Results and Discussion

In our work, we only describe the submissions for Task A of the competition, which is a classification task for the identification of misogynous and aggressive texts. Task B measures the bias of such classifiers with respect to certain concepts. Our submissions for task B are extracted from tf-idf representations of word n-grams and obtain the smallest scores of the competition.

Table 1 contains the submitted runs for Task A and the experiments we did to get a better understanding of the subtleties misogynistic and/or aggressive tweets contain. The columns *CV F1* contain the average F_1 scores computed for 10-fold cross-validation carried for ten iterations. Each cross-validation train-test split is stratified to preserve the proportions of misogynoys and/or aggressive texts in both splits. The *Test F1* columns are the results obtained on the gold standard test set. In the last column, we provide the macro F1 resulting from the average F1 between aggressiveness and misogyny predictions.

The submitted runs show that the tf-idf vectorizer from run1, although it scored better during the cross-validation stage, ended up being outperformed by the word embeddings extracted from spaCy (run3, 0.684 macro F_1), being unable to

---

[3] Model it_core_news_lg, version 2.3.0 released from spaCy `https://spaCy.io/models/it`

Feature	Misogyny		Aggressiveness		
	CV F1	Test F1	CV F1	Test F1	F1 Macro
tf-idf, run1	**0.883**	0.71	**0.8**	**0.652**	0.681
glove, run2	0.818	0.717	0.741	0.616	0.666
spacy, run3	0.842	**0.733**	0.767	0.635	**0.684**
clean tf-idf	**0.881**	0.706	**0.791**	**0.669**	0.688
clean, glove	0.847	0.722	0.766	0.618	0.67
clean spacy	0.846	**0.746**	0.784	0.655	**0.7**
nps, clean, tf-idf	**0.876**	0.714	**0.79**	**0.654**	0.684
nps, clean, spacy	0.837	**0.728**	0.768	0.646	**0.687**
fct words	0.672	0.628	0.614	0.564	0.596
POS n-grams	0.754	0.573	0.723	0.607	0.59
pronouns	0.594	0.596	0.656	0.636	0.616
filter POS	**0.832**	**0.731**	**0.765**	**0.657**	**0.694**

Table 1: Cross-validation and test-set results of logistic regression classifier with different feature extraction processes.

generalize to the new texts. The second run (run2, 0.666 macro F_1) uses the glove pre-trained embeddings for English. This result represents the biggest surprise of the three since it did not use Italian embeddings. We observe that the English glove representations cover more than 60

Cleaned texts aid the classifier by a significant threshold. In our experiments, we removed tags and URLs to observe a significant increase in macro scores for the same approaches over the cleaned texts. The best result we obtained so far (0.7 macro score) uses the Italian spaCy average vector representations extracted from clean texts.

Noun phrases extracted from each cleaned text do not indicate significant increases in misogynous or aggressive texts detection. Using these features yields comparable scores to the best of our methods, surpassing the classification attempts on uncleaned texts. This indicates that noun phrases alleviate the noise extracted by the tf-idf vectorizer. The model was less prone to overfitting and, therefore, more able to adapt to the unseen data.

Function words are features with grammatical roles, consisting of conjunctions, prepositions, articles, etc. encompassing stylistic aspects of the texts. We tested the accuracy of a simple logistic regression using function words, and the results were higher than 50% by a non-trivial amount. This is a potential indicator that misogynistic and/or aggressive tweets have a slightly different syntax than those that do not fit in either of the two. Moreover, using the tf-idf vectorizer on plain function words achieved 0.628 F_1 on the test set for misogyny identification, a result that is not at all negligible, given that these words do not encapsulate meaning.

POS n-grams are yet another set of features capable of capturing shallow syntactic constructs. Using this feature set, we observed a strong overfitting tendency on the cross-validation scenarios (average F_1 0.754 for misogyny and 0.723 for aggressiveness) while on the gold test set, the macro F_1 score is 0.59. This is an indicator that certain syntactic patterns are indeed occurring in the misogynistic and aggressive texts, weakly differentiating them from other types of texts. However, these features have little power to generalize on new samples.

Pronouns reveal the most interesting result due to two reasons: 1) the features did not overfit the data, as indicated by the cross-validation F_1 scores that are close to the actual scores on the gold test set; 2) aggressive texts can be differentiated between each other using only pronouns with an F_1 score (0.636) that is comparable with more advanced methods that use richer features such as embeddings (0.655, for the embeddings over clean texts) or tf-idf vectorizer (0.669, for tf-idf over clean texts). Therefore, in terms of aggressiveness, it is clear that certain expressions using forms of second-person pronouns are typically used to construct call-out phrases or curse-word expressions. The most common pronoun observed in aggresive texts is *ti* - the second person singular acusative of pronoun *tu* ('you').

Filter POS account the n-grams of words and POS tags extracted from the following categories: nouns, adverbs, adpositions, determiners, adjectives, verbs, pronouns, and auxiliary verbs. The features obtain the second best result (0.694 F_1 macro score) from all our attempts. Again, in this situation, we are also facing a big difference between the cross-validation results and the released test set.

4 Discussion

The results show that the vanilla feature extraction methods suffered from a non-trivial amount of overfitting. Despite the fact that we carried a stratified 10-fold cross-validation, over ten iterations, the average F_1 scores obtained on the test set were considerably lower than the ones we obtained in our separate experiments.

The evaluation scores of said methods was over 88% in our cross-validation splits. On the cross-validation evaluation from the training set, tf-idf produced the best results. On the test set, embeddings proved to have a better power of generalization. Preprocessing the texts by removing stopwords, hashtags, links, and other types of noise proved to be beneficial for the classifier. The best results were obtained by extracting average clean text embeddings. Overall, word embeddings were more consistent when comparing cross-validation results with the test ones for misogyny detection.

At a shallow eye-check we noticed in the test set several examples labeled as misogynous with no apparent reasons: ”troppo acida... non mangio yogurt”, ”Impiccati”, ”#nome?”. We can only assume that the misogynistic character of these comments is given by the context in which they were posted. On the test set it also appears that the majority of misogynistic comments are remarks on different body parts, most likely as comments posted to pictures. It is, therefore, difficult to asses the misogynistic character of a short text without having at hand the full multi-modal context: to whom it was posted, what kind of relation is between the ”commenter” and the ”commentee”, if the tweet is a reply or a single post, and so on and so forth.

It is worth noting that most text classification papers mention or use BERT (Bidirectional Encoder Representations from Transformers), as it has proven to be one of the most accurate when facing different types of data (Pamungkas et al., 2020). Other state of the art methods are LTSM (Long short-term memory) and XLNet, the latter overtaking BERT on various tasks (Yang et al., 2019). A current issue with such methods and word embeddings is that they transfer the human bias present in large corpora. This is becoming a bigger problem as AI filters are prevalent in today's society and therefore discriminatory traits of the models become discriminatory real world actions. For example, textual embeddings trained from Wikipedia data show discriminatory traits towards minorities such as associating foreigners with criminals, homosexuality with corruption, men being linked to aggression and women with the idea of the loving wife. (Papakyriakopoulos et al., 2020). Basta et al. (2019) finds that word embeddings are more likely to be discriminatory and biased than their contextualized counterparts, implying that state of the art methods are moving towards the right direction. However, as the models are getting closer to understanding language, one cannot help but wonder if this will have a negative impact on their bias if precautions aren't taken, as they might be overly impacted by the ubiquitous bias humans carry. Due to the widespread automatisation of daily tasks using machine learning models, mitigating prejudice becomes a responsibility of the developers, as it crucial for obtaining equal opportunities and treatment of minorities.

5 Conclusions

Our results indicate that simple feature engineering and vanilla classifiers cannot distinguish between misogynistic/aggressive tweets with reliable accuracy and that more research is needed to understand the important features concerning this task. However, the experiments imply a correlation between a text's syntax and its misogynistic/aggressive value. This proposes the idea that text that falls into either categories, (or maybe even hate speech in general?) does have a slightly more recognisable grammatical pattern than text that isn't. Whether it's the POS n-grams, pronouns, or just function words, the wording matters and is worth looking into for more advanced feature engineering.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Nozza Debora, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, Manuela Sanguinetti, et al. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *13th International Workshop on Semantic Evaluation*, pages 54–63. Association for Computational Linguistics.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Christine Basta, Marta Ruiz Costa-jussà, and Noe Casas. 2019. Evaluating the underlying gender bias in contextualized word embeddings. *CoRR*, abs/1904.08783.

Cristina Bosco, Felice Dell'Orletta, Simonetta Montemagni, Manuela Sanguinetti, and Maria Simi. 2014. The evalita 2014 dependency parsing task. In *EVALITA 2014 Evaluation of NLP and Speech Tools for Italian*, pages 1–8. Pisa University Press.

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122.

Alina Maria Ciobanu, Sergiu Nisioi, and Liviu P Dinu. 2016. Vanilla classifiers for distinguishing between similar languages. In *Proceedings of the VarDial Workshop*.

Paolo Rosso Elisabetta Fersini, Debora Nozza. 2020. Ami @ evalita2020: Automatic misogyny identification. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018a. Overview of the evalita 2018 task on automatic misogyny identification (ami). *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:59.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018b. Overview of the task on automatic misogyny identification at ibereval 2018. *IberEval@ SEPLN*, 2150:214–228.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Shervin Malmasi, Keelan Evanini, Aoife Cahill, Joel R. Tetreault, Robert A. Pugh, Christopher Hamill, Diane Napolitano, and Yao Qian. 2017. A report on the 2017 native language identification shared task. In *BEA@EMNLP*, pages 62–75. Association for Computational Linguistics.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing & Management*, 57(6):102360.

Orestis Papakyriakopoulos, Simon Hegelich, Juan Carlos Medina Serrano, and Fabienne Marco. 2020. Bias in word embeddings. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 446–457, New York, NY, USA. Association for Computing Machinery.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding.

Marcos Zampieri, Shervin Malmasi, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, Jörg Tiedemann, Yves Scherrer, and Noëmi Aepli. 2017. Findings of the vardial evaluation campaign 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 1–15, Valencia, Spain. Association for Computational Linguistics.

No Place For Hate Speech @ AMI: Convolutional Neural Network and Word Embedding for the Identification of Misogyny in Italian

Adriano dos S.R. da Silva
Schoool of Arts, Sciences and
Humanities – University of Sao Paulo
Sao Paulo - Brazil
`adriano.santos.silva@usp.br`

Norton T. Roman
Schoool of Arts, Sciences and Humanities
University of Sao Paulo
Sao Paulo - Brazil
`norton@usp.br`

Abstract

English. In this article, we describe two classification models (a Convolutional Neural Network and a Logistic Regression classifier), arranged according to three different strategies, submitted to subtask A of Automatic Misogyny Identification at EVALITA 2020. Results were very encouraging for detecting misogyny, even though aggressiveness was less accurate. Our second strategy, consisting of a Convolutional Neural Network and logistic regression to identify misogyny and aggressiveness, respectively, won the sixth place in the competition.

Italiano. *In questo articolo, descriviamo due modelli di classificazione (i.e., Convolutional Neural Network e Regressione Logistica), organizzati secondo tre diverse strategie, per il subtask A dello shared task Automatic Misogyny Identification a EVALITA 2020. I risultati sono stati molto incoraggianti nel rilevamento della misoginia, anche se l'aggressività viene riconosciuta con una precisione più basse. La nostra seconda strategia (Convolutional Neural Network per misoginia e Regressione Logistica per aggressività) ci ha permesso di ottenere il sesto posto nella competizione.*

1 Introduction

Hate speech is a problem that has been gaining space both in the media and in academic research. Political organizations have been working to combat this type of discourse. As is the case with the code of conduct[1] created by the European Union Commission, and signed by some of the main social networks, such as Facebook, YouTube, Twitter, which aims to monitor and remove this type of content within 24 hours of its disclosure.

The subject has even become a marketing problem, to the extent that recently several companies stopped advertising on Facebook[2], only to put some pressure at the network to have it remove this type of publication from the posts within it. Advertisers point, in this case, is that they do not want their brand to be linked to this type of speech.

Defined as "language which attacks or demeans a group based on race, ethnic origin, religion, gender, age, disability, or sexual orientation/gender identity" (Nobata et al., 2016), hate speech represents a problem that cannot be allowed to grow, under the risk of having it lead to more concrete actions, by some people, with truly undesired results.

When this hate speech is targeted specifically at women, it is called misogyny (Manne, 2017). The problem with misogyny is such an issue that it has already been related to real crime cases and cybercrimes (Fulper et al., 2014). In this case, correlations were found between rape cases and the amount of misogynous tweets per state in the United States.

Some academic work and several competitions have proposed some tasks to promote studies and advances in the area. Much of this work and data sets focus on English (Fortuna and Nunes, 2018) only, even though this is a widespread phenomenon that happens in any language.

It is extremely important, therefore, to encourage the development of this kind of study

---

[1] https://ec.europa.eu/info/policies/justice-and-fundamental-rights/combatting-discrimination/racism-and-xenophobia/eu-code-conduct-countering-illegal-hate-speech-online_en

[2] `https://www.nytimes.com/2020/08/01/business/media/facebook-boycott.html`

in different languages and competitions, such as IberEval (Fersini et al., 2018b), SemEval (Basile et al., 2019) and EVALITA (Fersini et al., 2018a), which have already proposed activities to identify misogynous discourse in Spanish, English, and Italian.

In this work, we help address this problem by testing two classification models as part of EVALITA 2020's subtask A on Automatic Misogyny Identification (AMI). Tested models were a Convolutional Neural Network (CNN) and a Logistic Regression (LR) classifier. Three different strategies were designed and tested, with one of them scoring 6th in the competition.

The rest of this article is organized as follows. Section 2 presents some related work in the identification of misogyny or hate speech. Section 3, in turn, gives an overview of EVALITA's AMI. Next, in section 4, we describe our experimental set-up, giving details of the implemented methods and tested strategies. Finally, in Section 5 we discuss our results, whereas in Section 6 we present our final remarks on this task.

2 Related Work

IberEval (Fersini et al., 2018b) proposed a task to identify misogynous discourse in tweets in English and Spanish. Several teams participated in this competition and the best team reached an accuracy of 0.91 and 0.81 for Spanish and English, respectively, with the use of an SVM as a classifier and with the addition of some lexical features to characterize the tweets.

SVMs were also proposed to identify racism in Twitter messages in English, achieving an F1 score of 0.76 (Hasanuzzaman et al., 2017). In SemEval 2019, a Convolutional Neural Network (CNN) performed competitively in the task of identifying hate speech against immigrants and women in English (Basile et al., 2019). The team that presented this architecture ranked fourth with an F1 score of 0.535.

During the Automatic Misogyny Identification shared task at EVALITA 2018, it was proposed a subtask A, which consisted of identifying misogyny (Fersini et al., 2018a; Anzovino et al., 2018). For this subtask, Logistic Regression was the model to deliver the best performance with an accuracy of 0.704 (Saha et al., 2018).

3 Subtask

The second edition of misogyny identification at EVALITA 2020 consists of two subtasks: A and B. The purpose of subtask A is to identify the presence or absence of misogyny and aggressiveness in tweets (Elisabetta Fersini, 2020), whereas subtask B checks whether the model is capable of distinguishing misogynous from non-misogynous content, also ensuring fairness (unintended bias) (Nozza et al., 2019).

The "No Place For Hate Speech" team participated only in subtask A, and all discussions that will be followed are related to this subtask. Within EVALITA 2020, the subtask consisted of identifying the presence or absence of misogynous speech and aggressiveness in tweets in Italian (Basile et al., 2020; Elisabetta Fersini, 2020).

The training dataset consisted of 5,000 tweets. The class that determines the presence or absence of misogyny is nearly balanced. However, aggressiveness is not balanced at all, with approximately 35% of tweets containing aggressiveness. Table 1 shows the distribution of each class in the training set.

Table 1: Distribution of Tweets in relation to each class of misogyny and aggressiveness

	Mis.	**Non Mis.**	**Aggr.**	**Non aggr.**
Total	2337	2663	1783	3217

4 Materials and Methods

In subtask A, we tested two different classifiers within different configurations. These were a Convolutional Neural Network (CNN), using BERT (Devlin et al., 2018) as its language model; and a Logistic Regression (LR) classifier, with L2 regularisation.

The LR classifier used a 4-gram language model, with tf-idf (Rajaraman and Ullman, 2011) normalization. Both models were developed in Python, with the aid of the TensorFlow[3] and Sklearn[4] libraries.

Since the subtask A at EVALITA allows each team to submit up to three classifiers, we decided to approach the problem according to three different strategies, involving different combinations of these classifiers, along with different subsets of data on which they should be trained.

---

[3] `https://www.tensorflow.org/`

[4] `https://scikit-learn.org/stable/`

In all cases, the training set was divided in a 90% subset, used for training purposes, with the remaining 10% used for out-of-sample testing. All classifiers used this same proportion both to identify misogyny and aggressiveness. Tweets were used in their raw form and no preprocessing was used.

All CNNs used in the experiments had the same configuration, being trained for 15 epochs. They also have three convolution layers, relu activation functions, and dropout rate of 0.10, with adam optimisation. Finally, cross-entropy was used as their loss function. In what follows, we will describe, with more details, each of the strategies followed during our tests.

4.1 Strategy 1

The first strategy consisted of training two CNNs, one for each specific sub-problem separately, *i.e.* one for misogyny and another for aggressiveness classification. In both cases, the entire data set was used for training.

At the testing stage, the CNNs were arranged as a pipeline, in which the first CNN was responsible for identifying whether a tweet had some misogynous content, whereas the second CNN was responsible for identifying the presence or absence of aggressiveness only in those tweets marked as misogynous by the first CNN.

4.2 Strategy 2

Similar to Strategy 1, the second strategy also consisted of training a CNN to detect misogynous content in tweets. This time, however, the classification of aggressiveness was left to a Linear Regression classifier. As in the first strategy, both models were trained in the entire data set.

During testing, once again models were arranged in a pipeline, with the CNN coming first, to detect misogyny in tweets. In the sequence, all tweets classified as misogynous by the CNN were then fed to the LR classifier, so it could determine the presence or absence of aggressiveness.

4.3 Strategy 3

Our third strategy is similar to Strategy 1, in that it also consists of two CNNs trained separately over the data set. The only difference, however, lies during the training stage. In this case, whereas the first CNN (*i.e.* the one responsible for misogyny identification) was trained using the entire data set, the second CNN (the one responsible for detecting aggressiveness) was trained only on those examples labeled as misogynous.

During testing the same set-up as in Strategy 1 was followed. As such, both CNNs were arranged in a pipeline, with the first one responsible for detecting misogynous tweets, and the second one responsible for identifying aggressiveness, amongst those tweets held misogynous by the first CNN.

5 Results and Discussion

Table 2 shows the performance of each tested strategy. As expected, the results for misogyny identification were the same over all strategies, since this subtask A was left to a CNN trained over the entire data set.

Table 2: Performance of each classifier strategy in terms of F1 score in the test set.

Classifier	Misogyny	Aggressiveness
Strategy 1	0.96	0.75
Strategy 2	0.96	0.70
Strategy 3	0.96	0.85

Results for aggressiveness detection, on the other hand, varied substantially, with the Logistic Regression classifier (Strategy 2) performing worst, when compared to the CNNs used for the same task in the other strategies (7% against Strategy 1, and 18% against Strategy 3).

Interestingly, the CNN trained only on examples labeled as misogynous (Strategy 3) performed better (around 13%) than its counterpart trained over the entire data set (Strategy 1). It is important to recall that this was the only difference between both strategies.

Final results at the competition's private test set can be seen in Table 3. As it turns out, Strategy 2 was the best ranked of our models, reaching the sixth place at the competition (being only $F = 0.03$ worse than the winning model).

Table 3: Official result of the subtask A in the evaluation set is calculated by averaging the F1 measures estimated for the Misogynous and Aggressiveness classes

Classifier	Average F1
Strategy 1	0.693
Strategy 2	0.716
Strategy 3	0.490

Puzzling enough, this was the model that scored worse in our test set. One possible explanation for this fact might be that our CNN was not capable of generalising over different data sets. Differences in the balance between misogynous and non-misogynous, and between aggressive and non-aggressive examples, in both data sets, might also explain this behaviour. Whatever the reason, we leave this investigation for future work.

6 Conclusion

In this work, we described two models submitted to EVALITA 2020's subtask A on Automatic Misogyny Identification. To this task, a CNN and an LR classifier were trained, and arranged as a pipeline following three different strategies, with one of them coming at sixth place in the competition.

Even though our classifier turned out to be competitive, we believe improvements could be made to achieve better results, such as the addition of lexical features, for example. Also, it might be that following some preprocessing strategy, such as removing stop words, for example, might result in a better performance.

As for future work, besides testing the above cited changes, it would be interesting investigating why the worst model at the test set (as distributed to all participants) turned out to be the best model at the competition's private data set. The reasons for this behaviour are something to be determined.

References

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Nozza Debora, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, Manuela Sanguinetti, et al. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *13th International Workshop on Semantic Evaluation*, pages 54–63. Association for Computational Linguistics.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Paolo Rosso Elisabetta Fersini, Debora Nozza. 2020. Ami @ evalita2020: Automatic misogyny identification. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018a. Overview of the evalita 2018 task on automatic misogyny identification (ami). *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:59.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018b. Overview of the task on automatic misogyny identification at ibereval 2018. *IberEval@ SEPLN*, 2150:214–228.

Paula Fortuna and Sérgio Nunes. 2018. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4):1–30.

Rachael Fulper, Giovanni Luca Ciampaglia, Emilio Ferrara, Y Ahn, Alessandro Flammini, Filippo Menczer, Bryce Lewis, and Kehontas Rowe. 2014. Misogynistic language on twitter and sexual violence. In *Proc. ACM Web Science Workshop on Computational Approaches to Social Modeling (ChASM)*.

Mohammed Hasanuzzaman, Gaël Dias, and Andy Way. 2017. Demographic word embeddings for racism detection on twitter.

Kate Manne. 2017. *Down girl: The logic of misogyny*. Oxford University Press.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*.

Debora Nozza, Claudia Volpetti, and Elisabetta Fersini. 2019. Unintended bias in misogyny detection. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 149–155.

Anand Rajaraman and Jeffrey David Ullman. 2011. *Mining of massive datasets*. Cambridge.

Punyajoy Saha, Binny Mathew, Pawan Goyal, and Animesh Mukherjee. 2018. Hateminers : Detecting hate speech against women. *CoRR*, abs/1812.06700.

ATE_ABSITA: Aspect Term Extraction and Aspect-Based Sentiment Analysis

ATE_ABSITA @ EVALITA2020: Overview of the Aspect Term Extraction and Aspect-based Sentiment Analysis Task

Lorenzo De Mattei
University of Pisa
Ist. di Ling. Comp.
“Antonio Zampolli”
Pisa ItaliaNLP Lab
`lorenzo.demattei@di.unipi.it`

Graziella De Martino
University of Bari A. Moro
Dept. Computer Science
E. Orabona 4, Bari
`graziella.demartino@uniba.it`

Andrea Iovine
University of Bari A. Moro
Dept. Computer Science
E. Orabona 4, Bari
SWAP Research Group
`andrea.iovine@uniba.it`

Alessio Miaschi
University of Pisa
Ist. di Ling. Comp.
“Antonio Zampolli”
Pisa ItaliaNLP Lab
`alessio.miaschi@phd.unipi.it`

Marco Polignano
University of Bari A. Moro
Dept. Computer Science
E. Orabona 4, Bari
SWAP Research Group
`marco.polignano@uniba.it`

Giulia Rambelli
University of Pisa
Coling Lab Pisa
Aix-Marseille University
`giulia.rambelli@phd.unipi.it`

Abstract

Over the last years, the rise of novel sentiment analysis techniques to assess aspect-based opinions on product reviews has become a key component for providing valuable insights to both consumers and businesses. To this extent, we propose ATE_ABSITA: the EVALITA 2020 shared task on Aspect Term Extraction and Aspect-Based Sentiment Analysis. In particular, we approach the task as a cascade of three subtasks: Aspect Term Extraction (ATE), Aspect-based Sentiment Analysis (ABSA) and Sentiment Analysis (SA). Therefore, we invited participants to submit systems designed to automatically identify the ”aspect term” in each review and to predict the sentiment expressed for each aspect, along with the sentiment of the entire review. The task received broad interest, with 27 teams registered and more than 45 participants. However, only three teams submitted their working systems. The results obtained underline the task’s difficulty, but they also show how it is possible to deal with it using innovative approaches and models. Indeed, two of them are based on large pre-trained language models as typical in the current state of the art for the English language. (de Mattei et al., 2020)

1 Introduction and motivation

Leaving comments and reviews on the Web has become a common practice for users to express their opinions about products, experiences, and more. Thus, companies need to deal with increasingly large amounts of textual data, which can be useful to identify their products’ strengths and weaknesses. However, the automatic analysis of reviews poses numerous problems related to its processing. First of all, reviewers often use informal language, with a wide variety of colloquialisms and contractions, which make review analysis through lexicon-based techniques difficult. Second, automatically identifying aspects of the product within a sentence is not easy, due to the intrinsic subjectivity in the definition of ”aspect”. These issues have already been addressed in the area of Text Mining and Sentiment Analysis. Recently, the sentiment analysis and opinion mining tasks have seen a surge in interest, thanks to the large quantity of data available for analysis and the new natural language processing techniques based on language models such as BERT (Devlin et al., 2019) and GPT (Radford et al., 2019). Thus, we proposed the ATE_ABSITA: the EVALITA 2020 (Basile et al., 2020) shared task on Aspect Term Extraction and Aspect-Based Sentiment Analysis.

Sentiment Analysis (or *Opinion Mining*) is the task of identifying what the user thinks about a particular element. It often takes the form of a classification task with the purpose of annotating a portion of text with a positive, negative, or neutral label. *Aspect-based Sentiment Analysis* (ABSA) is an evolution of Sentiment Analysis that aims

---

at capturing the aspect-level opinions expressed in natural language texts (Liu, 2007). Very often, the ABSA task is performed on a set of aspects defined a priori, limiting its applicability in the real scenario. *Aspect Term Extraction* (ATE) is the task of identifying ”aspect term” in a text without knowing a priori the list that contains it. According to the literature definition, a term/phrase is considered as an aspect when it co-occurs with some “opinion words” that indicate a sentiment polarity on it (Pontiki et al., 2016a).

At the international level, SemEval, the most prominent evaluation campaign in the Natural Language Processing field, provided in 2014 SE-ABSA14 (Pontiki et al., 2014) a benchmark dataset of reviews in the English language for the ABSA task. Given a set of sentences with pre-identified entities (e.g., *restaurants*), the task was about identifying the aspect term occurring in the sentences and returning a list containing all the distinct aspect term. It was then asked for all retrieved aspect term to determine whether the polarity of each of them was positive, negative, neutral, or conflict. The same task was replicated in 2015, 2016, consolidating the four subtasks of SE-ABSA14 (Pontiki et al., 2014) within a unified framework. Besides, SE-ABSA15 (Pontiki et al., 2016b) included an out-of-domain ABSA subtask, involving test data from a domain unknown to the participants.

ABSA is not a novel task at EVALITA. A first edition was proposed at EVALITA 2018 by (Basile et al., 2018). The task was subdivided into two subtasks: Aspect Category Detection (ACD) and Aspect Category Polarity (ACP). The first was about the identification of categories mentioned into the review, knowing the categories a priori. The latter was about the detection of the polarity of the opinion of the user about the previous detected categories. However, it bears some similarities with at least other two tasks from the previous editions of the campaign. SENTIPOLC (Basile et al., 2014), featured in the 2014 and 2016 editions of EVALITA, is a shared task on the polarity classification of social media content. The other is NEEL-it (Basile et al., 2016), held at EVALITA 2016. NEEL-it is the task of Named Entity Recognition and Linking, that is, the task of identifying the spans of an input text that refer to named entities, and linking them to entries in a knowledge base, e.g., pages of Wikipedia.

aspect term	Positive	Negative
mantenere la temperatura	1	0
costruzione	1	0

Table 1: Examples of Aspect-Based Sentiment Analysis annotations.

2 Definition of the Task

We define the ATE_ABSITA task as a cascade of three subtasks: **Aspect Term Extraction** (ATE), **Aspect-based Sentiment Analysis** (ABSA), **Sentiment Analysis** (SA).

For example, let us consider the sentence describing a review of a metallic bottle:

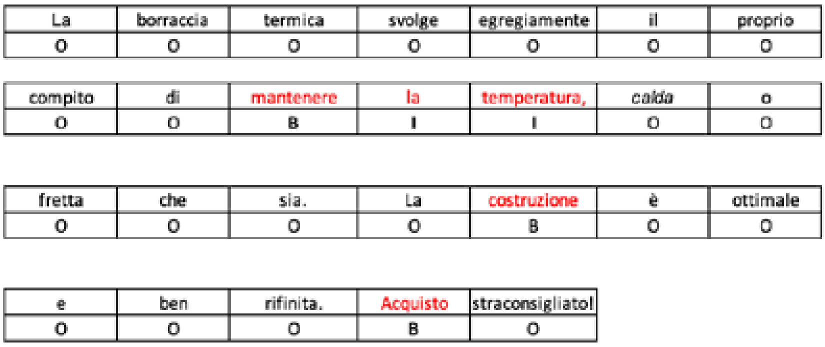
La borraccia termica svolge egregiamente il proprio compito di mantenere la temperatura, calda o fredda che sia. La costruzione è ottimale e ben rifinita. Acquisto straconsigliato!

The thermal water bottle does its job very well to keep the temperature, whether hot or cold. The construction is optimal and well finished. Purchase highly recommended!

In the **Aspect Term Extraction** (ATE) task, one or more ”aspect term” mentioned in a sentence are identified, e.g. `mantenere la temperatura` (`keep the temperature`) and `costruzione` (`construction`) in the sentence. Given a sequence $X = x, ..., x_T$ of T words, the ATE task can be formulated as a token/word level sequence labeling problem to predict an aspect label sequence $Y = y_1, ..., y_T$, where each y_i comes from a finite label set $Y = B, I, O$ which describes the possible aspect labels (begin, inside, outside). An example of ATE annotation is provided in Fig. 1.

In the **Aspect-based Sentiment Analysis** (ABSA) task, the polarity of each expressed aspect is recognized, e.g. a `positive` category polarity is expressed concerning the `mantenere la temperatura` aspect. The two labels are not mutually exclusive: in addition to the annotation of *positive* aspects (POS:true, NEG:false) and *negative* aspects (POS:false, NEG:true), there can be aspects with *mixed polarity* (POS:true, NEG:true), or *neutral* polarity (POS:false, NEG:false). An example ot ABSA annotation is showed in Tab. 1.

In the **Sentiment Analysis** (SA) task, the polarity of the review is provided. In particular, we



La	borraccia	termica	svolge	egregiamente	il	proprio
O	O	O	O	O	O	O

compito	di	mantenere	la	temperatura,	calda	o
O	O	B	I	I	O	O

fretta	che	sia.	La	costruzione	è	ottimale
O	O	O	O	B	O	O

e	ben	rifinita.	Acquisto	straconsigliato!
O	O	O	B	O

Figure 1: Result of the ATE annotation.

decided to use the score left by the user at the item as the polarity value. It is defined as an integer number within the [1..5] range. An example is provided in Tab. 2.

Review	Score
La borraccia termica svolge egregiamente il proprio compito di mantenere la temperatura,calda o fretta che sia. La costruzione è ottimale e ben rifinita. Acquisto straconsigliato!	5

Table 2: Example of Sentiment Analysis polarity annotation on the whole sentence.

In the ATE task here described, the set of aspects is not defined in advance, and the task itself is formalized as a Sequence Labeling task. The ABSA task can, instead, be formalized as a multi-class classification task. Finally, the Sentiment Analysis is considered as a regression task. For each review, participants will be asked to return a vector of aspects, a vector of aspect:polarity pairs, and a review:score pair. Two binary polarity labels are expected for each aspect: POS and NEG, indicating a positive and negative sentiment expressed towards a specific aspect, respectively. The participants may choose to submit each of the three subtasks independently.

3 Dataset

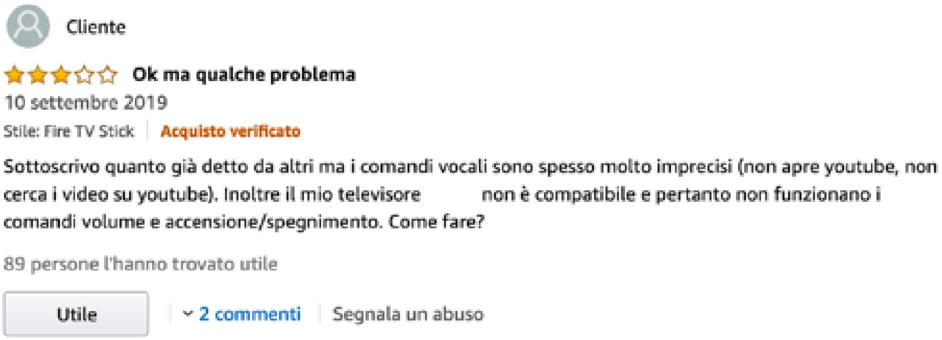
The data source chosen for creating the datasets is an eCommerce platform famous worldwide. The platform allows users to share their opinions about the items that they bought through a textual review and a final score of satisfaction. Therefore, the website provides a large number of reviews in many languages, including Italian (Fig. 2). We have collected 4364 real user reviews, written in the Italian language, involving 23 products. The training, dev and test sets will be randomly generated in the following ratios: 70% training, 2.5% dev, 27.5% test set. This means that the test set will be not out-of-domain. The items cover very different domains of use. In particular, the existing objects refer to: SD Memory Cards, Irons, Water Bottles, Action Cameras, Razors, Phones, Printer Cartridges, Coffee Capsules, Backpacks, Hair Dryers, 2 different Movies, 2 different Books, Toy Phones, Car Light bulbs, Sweatshirts, Boots, Fans, Storage Chest, Shoe Cabinets, Personal Digital Assistants, TV streaming boxes/sticks. A portion of the collected data has been **manually annotated** by three different subjects. Then, we measured the inter-annotator agreement metric as the value of quality of all the annotations. In particular, we obtained a score of 73.2% over 100 reviews. Thanks to the good score, we decided to continue the annotation process by annotating each review individually (i.e. one annotator per review). At the end of the annotation process, we obtained the gold annotated dataset. We randomly split the gold dataset to create a training/validation/test partition of it.

We do not provide any unique ID that could be used to retrieve more information about the writers. Consequently, we do not violate copyrights and/or we do not have privacy issues. Furthermore, in order to avoid harming the interests of the manufacturers, we do not disclose any information about the specific items for which the reviews have been issued.

The data format used is NDJSON [1] with UTF-8 encoding and newline as delimiter. Note that some reviews may not contain any aspect, but the final review score is always available. An example of

---

[1] `http://ndjson.org/`


Cliente

★★★☆☆ **Ok ma qualche problema**

10 settembre 2019

Stile: Fire TV Stick | Acquisto verificato

Sottoscrivo quanto già detto da altri ma i comandi vocali sono spesso molto imprecisi (non apre youtube, non cerca i video su youtube). Inoltre il mio televisore non è compatibile e pertanto non funzionano i comandi volume e accensione/spegnimento. Come fare?

89 persone l'hanno trovato utile

Figure 2: Example of a review about a TV streaming box/stick.

```
{"sentence":"L’attore...e le musiche indimenticabili", "id_sentence":"4c0b","score":5,
"polarities":[[0,0],[1,0]], "aspects_position":[[2,8],[16,23]], "aspects":["attore","musiche"]}
{"sentence":"Schermo guasto dopo appena due settimane,...","id_sentence":"4e1671","score":1,
"polarities":[[0,1]],"aspects_position":[[0,7]],"aspects":["Schermo"]}
{"sentence":"Ottimo telefono belle foto","id_sentence":"4eca9d08","score":4,"polarities":[[1,0]],
"aspects_position":[[22,26]],"aspects":["foto"]}
```

Figure 3: Example of NDJSON dataset records.

annotated data is provided in the code reported in Fig. 3.

4 Annotation Schema

This section describes the protocol that will be used to annotate the datasets for the three subtasks. The objective of this protocol is to get a reasonably objective definition of the characteristics of an aspect term. Due to the highly subjective nature of aspects, it does not encompass all conceivable aspect term. We define an **aspect term** as:

(a) An **attribute** (characteristic, property, feature, quality) of the object itself; (b) a tangible or abstract **part of the object**, for which an opinion can be inferred from the review; (c) the **activities** that the object is able (or not able) to perform; (d) the object's **ability to be suitable** for certain categories of people.

Judgment can be assigned in three ways: 1. *Directly*: the aspect term occurs with an opinion term (i.e., "la **durata della batteria** è ottima"); 2. *Indirectly*: the judgment about the product is transitive to a quality or part of the object. In other words, if an opinion is expressed about the object itself, and it is then stated for which characteristic the judgment is applied, these characteristics are annotated as an aspect term (i.e., "questo telefono è ottimo, soprattutto per la **durata della batteria**"); 3. *Deductible*: the opinion is not expressed directly but it is inferable from the review or from the knowledge of the reference domain.

The aspect term must represent the product characteristics, but it cannot represent a concept that is larger than the product itself. An aspect term **does not identify opinions** regarding elements external to the object, such as: (a) The shipment (it is not an intrinsic property of the object); (b) the company that produced them, the series to which the product belongs or other products with which the object is compared; (c) the elements that refer to the action of purchasing the item; (d) the elements that refer to the customer care. Moreover, in the case of aspect term composed of several words, all the words that make up the aspect term must be contiguous. In case they are separated by one or more words that are not part of the aspect term, the whole expression is discarded. More details and example of annotations are available on the task website[2].

5 Evaluation measures and baselines

We evaluate the three subtasks (ATE, ABSA and SA) separately by comparing the results obtained by the participant systems on the gold standard annotations of the test set.

For the ATE task, we compute Precision, Recall

---

[2] `http://www.di.uniba.it/~swap/ate_absita/examples.html`

and F$_1$-score defined as:

$$F1_a = \frac{2P_aR_a}{P_a + R_a} \tag{1}$$

In order to account for both exact and partial matches of aspect term, we define Precision (P_a) and Recall (R_a) as:

$$\begin{aligned} P_a &= \tfrac{|S_a \cap G_a| + 0.5 * |PAR_a|}{|S_a|} \\ R_a &= \tfrac{|S_a \cap G_a| + 0.5 * |PAR_a|}{|G_a|} \end{aligned} \tag{2}$$

Here, S_a is the set of aspect term annotations that a system returned for all the test sentences, G_a is the set of the gold (correct) aspect term annotations and PAR_a is the set of partial matches (predicted and gold aspect term have some overlapping text). For instance, if a review is labeled in the gold standard with the two aspect term $G_a = \{\textit{costruzione}, \textit{mantenere la temperatura}\}$, and the system predicts the two aspects $S_a = \{\textit{costruzione}, \textit{temperatura}\}$, we have that $|S_a \cap G_a| = 1$, $|PAR_a| = 1$, $|G_a| = 2$ and $|S_a| = 2$, so that $P_a = \frac{1.5}{2} = 0.75$, $R_a = \frac{1.5}{2} = 0.75$ and $F1_a = \frac{1.5}{2} = 0.75$. For the ATE task, we considered a simple baseline approach which considers every name entity as an aspect term. The algorithm is based on a Name Entity Recognition (NER) annotation obtained through the SpaCy [3] tool on the Italian model 'it_core_news_sm'. The implementation of the baseline on the training set is available as a Python3 Notebook on our website.

For the ABSA task (Task 2), we evaluate the entire chain, thus considering both the aspect term detected in the sentences together with their corresponding polarities, in the form of $(aspect, polarity)$ pairs. We again compute Precision (P_p), Recall (R_p) and F$_1$-score ($F1_p$) defined as following:

$$F1_p = \frac{2P_pR_p}{P_p + R_p} \tag{3}$$

$$\begin{aligned} P_p &= \tfrac{|S_p \cap G_p| + 0.5 * |PAR_p|}{|S_p|} \\ R_p &= \tfrac{|S_p \cap G_p| + 0.5 * |PAR_p|}{|G_p|} \end{aligned} \tag{4}$$

Where S_p is the set of $(aspect, polarity)$ pairs that a system returned for all the test sentences, G_a is the set of the gold (correct) pairs annotations and PAR_p is the set of $(aspect, polarity)$ pairs with a partial match. For instance, if a review is labeled in the gold standard with the pairs:
$G_p = \{(\textit{mantenere la temperatura}, POS), (\textit{costruzione}, POS)\}$,
and the system predicts the three pairs
$S_p = \{(\textit{temperatura}, NEG), (\textit{costruzione}, POS), (\textit{acquisto}, POS)\}$,
we have that $|S_p \cap G_p| = 1$, $|PAR_p| = 0$, $|G_p| = 2$ and $|S_p| = 3$ so that $P_p = \frac{1}{3}$, $R_p = \frac{1}{2}$ and $F1_p = 0.4$. As a baseline for the ABSA task, we decided to assign the most frequent polarity class (i.e. the positive one) to each aspect found by the baseline strategy for Task 1.

To evaluate the SA task (Task 3), we compute the Root Mean Squared Error ($RMSE_w$) between the scores predicted by the participant systems and those found in the gold dataset. For this task, we employed three different baselines. The first predicts the most frequent value in the training set: 5. The second predicts the average value of the scores found on the training set (4.46299). The third one uses AlBERTo (Polignano et al., 2019) as an approach to develop a Regression task.

6 Task statistics

The task has generated great interest in the scientific community. We obtained 27 registered teams, for a total of 45 separate participants. Nevertheless, the difficulty of the task discouraged many of them. At the end of the evaluation phase, we obtained 8 submissions from 3 different teams.

7 Submitted systems

The three teams participating in the task are the following:

- **A2C** (Rosa and Durante, 2020): the team is composed of two members of the App2Check company, who developed a classification model based on state-of-the-art language models. In particular, they investigate the ATE task through the use of four different configurations of language models: 1. Native Italian pre-trained language models, with no specific NER fine-tuning and 3. with NER fine-tuning; 2. Multilingual pre-trained language model, with no specific NER fine-tuning and 4. with NER fine-tuning. For the first and the third configuration, they considered dbmdz/bert-base-

---

[3] `https://spacy.io/`

italian-xxl-uncased[4] and GilBERTo[5]. For the second configuration, they considered two implementations of RoBERTa: xml-roberta-large3 (Conneau et al., 2019), xml- roberta-base4 (Liu et al., 2019), and multilingual BERT (Pires et al., 2019). The xlm RoBERTa Large multilingual model was chosen as the competition model. The ABSA task has been performed by fine-tuning a multilingual BERT model in order to assign the polarity label to each portion of text that contains at least one previously detected aspect. Similarly, the SA task has been approached using a multilingual BERT model on a 1 to 5 sentiment scale. The system submitted by the A2C team obtained the best results overall.

- **SentNa** (Francesco Mele and Vettigli, 2020): the authors proposed a hybrid model that joins rule-based and machine learning methodologies in order to combine their respective advantages. The main idea for dealing with the ATE task is to identify a set of plausible aspects via some predefined rules. Then, a classifier is used to filter out the wrong candidates. The rules are defined on POS-Tagging patterns. The authors defined a set of about 3000 rules. The sentiment analysis problem has been solved by building the features representing the text using n-grams, and adding a set of features annotated in SenticNet (Cambria et al., 2010). Then, a regressor composed of 800 Decision Trees with 4 layers has been trained using Gradient Boosting. The final prediction is computed by averaging the output of each tree.

- **ghostwriter19** (Bennici, 2020): the team composed of one member of the YouAreMyGuide Company proposes a solution based on mixing transfer learning, zero-shot learning (Brown et al., 2020), and ONNX[6], in order to access the power of BERT while using limited resources. In order to deal with the ATE and ABSA tasks, the author uses the AlBERTo (Polignano et al., 2019) language model and an auto training system such as Ktrain[7] for fine-tuning the system. At this point, the model has been exported with ONNX in maximum compatibility mode with the original. The optimization options have been set to a minimum for CPU usage. The performances have remained unchanged, but the speed of inference has significantly improved. For the sentiment analysis task, the author uses a zero-shot learning strategy as a way to make predictions without prior training. In particular, he reuses the embedding of AlBERTo for encoding the sentence and a BiLSTM as classification model to predicting a class from 1 to 5.

Table 3: Final results obtained by the participants for the ATE sub-task.

Pos.	Team Name	F1 score
1	**A2C**	**0.68222**
2	ghostwriter19	0.53986
3	SentNa	0.34027
4	*Baseline-Name Entities*	*0.2556*

Table 4: Final results obtained by the participants for the ABSA sub-task.

Pos.	Team Name	F1 score
1	**A2C**	**0.61878**
2	ghostwriter19	0.49935
3	SentNa	0.28632
4	*Baseline-Positive pol.*	*0.20000*

8 Discussion of results

The results in tables from 3-5 show the optimal performances of the system developed by the **A2C** team, which obtained first place in all three subtasks. The use of pre-trained language models has proven to be the winning strategy. In particular, the differences between the results of **A2C** and **ghostwriter19** show how a large RoBERTa model can strongly outperform a smaller language model such as AlBERTo, even though the latter has been specifically trained on the Italian language. This result was expected, since the ALBERTo baseline also obtained low results. We hypothesize that the difference in style between the tweets that were used to train ALBERTo and the reviews contained in this dataset are a significant factor in the low applicability of this model. Additionally, the results obtained by the A2C system also show that pre-

---

[4] `https://github.com/dbmdz/berts`

[5] `https://github.com/idb-ita/GilBERTo`

[6] `https://microsoft.github.io/onnxruntime/`

[7] `https://github.com/amaiya/ktrain`

Table 5: Final results obtained by the participants for the SA sub-task.

Pos.	Team Name	RMSE
1	**A2C**	**0.66458**
2	SentNa	0.79533
3	ghostwriter19	0.81394
4	*Baseline-Average Score*	*1.00409*
5	*Baseline-AlBERTo*	*1.08063*
6	*Baseline-Most Freq.*	*1.12822*

training the language model for the Named Entity Recognition (NER) task is also useful for identifying aspect term. This is due to the fact that aspect term share some properties with named entities. For example, they are often configured as a noun, an adjective, or a combination of both.

The results obtained by **SentNa** are also interesting. Their model, which is based on decision trees, has obtained a good final score for the SA task. This confirms the findings obtained in earlier Sentiment Analysis tasks in Italian campaigns such as EVALITA, which already demonstrated that techniques such as Decision Trees, Random Forests, and SVD can be effective solutions to this task. Nevertheless, the **SentNa** system demonstrates that an enriched encoding of the sentences, including lexical features such as polarity value, attention, pleasantness, and sensitivity of its composing n-grams, can support a more accurate prediction of the whole sentence polarity.

9 Conclusion

In the ATE_ABSITA task at EVALITA 2020, we focused the attention of research groups that work on computational linguistics for the Italian language on the problem of analyzing user reviews. Specifically, we subdivided the problem into three parts: Aspect Term Extraction (ATE), Aspect-Based Sentiment Analysis (ABSA), Sentence Sentiment Analysis (SA). In the ATE task, the goal was to identify one or more “aspect term” discussed in the review. The second task was about identifying the sentiment evoked by the user while talking about a specific aspect (ABSA). Finally, we asked participants to identify the polarity associated with the entire review (SA). The dataset we released has been collected from a world-famous eCommerce platform. In particular, we extracted and **manually annotated** 4364 real user reviews, written in the Italian language, about 23 different products. Although the results obtained by the systems that participated in the task are very close to those available in the English language literature, the F1 scores for the ATE and ABSA subtasks demonstrate its complexity. It is evident that an F1 score of about 0.60 generates a non-negligible margin of error of prediction. The diversity in terms, linguistic expressions, and in the physical characteristics of the products themselves makes the automatic extraction of ”aspect term” a task that is far from being resolved. This complexity can also explain the low number of participants. It is easy to see a substantial discrepancy between the number of people enrolled in the task and those who have proposed a solution for it. In our opinion, this is caused by the difficulty in addressing the problem with the current natural language analysis techniques. However, this also means that there is still a wide margin for improvement in this area, and that this problem can be addressed again in the next edition of EVALITA. We firmly believe that extracting fine-grained opinions from user reviews can be a great asset for improving products, processes, and software systems.

References

Valerio Basile, Andrea Bolioli, Malvina Nissim, Viviana Patti, and Paolo Rosso. 2014. Overview of the evalita 2014 sentiment polarity classification task.

Pierpaolo Basile, Annalina Caputo, Anna Lisa Gentile, and Giuseppe Rizzo. 2016. Overview of the evalita 2016 named entity recognition and linking in italian tweets (neel-it) task. In *of the Final Workshop 7 December 2016, Naples*, page 40.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Pierpaolo Basile et al. 2018. O verview of the evalita 2018 aspect-based sentiment analysis task (absita). *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:10.

Mauro Bennici. 2020. ghostwriter19 @ ATE_ABSITA: Zero-Shot and ONNX to speed up BERT on sentiment analysis tasks at EVALITA 2020. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Erik Cambria, Robert Speer, Catherine Havasi, and Amir Hussain. 2010. Senticnet: A publicly available semantic resource for opinion mining. In *AAAI fall symposium: commonsense knowledge*, volume 10. Citeseer.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Lorenzo de Mattei, Graziella de Martino, Andrea Iovine, Alessio Miaschi, Marco Polignano, and Giulia Rambelli. 2020. ATE_ABSITA@EVALITA2020: Overview of the Aspect Term Extraction and Aspect-based Sentiment Analysis Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Antonio Sorgente Francesco Mele and Giuseppe Vettigli. 2020. SentNA@ATE_ABSITA: Sentiment Analysis of customer reviews using Boosted Trees with lexical and lexicon-based features. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Bing Liu. 2007. *Web data mining*. Springer.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019. Alberto: Italian bert language understanding model for nlp challenging tasks based on tweets. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019). CEUR*.

Maria Pontiki et al. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation*.

Maria Pontiki et al. 2016a. Semeval-2016 task 5: Aspect based sentiment analysis. In *10th International Workshop on Semantic Evaluation (SemEval 2016)*.

Maria Pontiki et al. 2016b. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, San Diego, California, June. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Emanuele Di Rosa and Alberto Durante. 2020. App2Check@ATE_ABSITA 2020: Aspect Term Extraction and Aspect-based Sentiment Analysis. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

SentNA @ ATE_ABSITA: Sentiment Analysis of Customer Reviews Using Boosted Trees with Lexical and Lexicon-based Features

Francesco Mele
Institute of Applied Sciences
and Intelligent Systems
National Research Council
`f.mele@isasi.cnr.it`

Antonio Sorgente
Institute of Applied Sciences
and Intelligent Systems
National Research Council
`a.sorgente@isasi.cnr.it`

Giuseppe Vettigli
Centrica plc,
Institute of Applied Sciences
and Intelligent Systems (CNR)
`giuseppe.vettigli@centrica.com`

Abstract

English. This paper describes our submission to the tasks on Sentiment Analysis of ATE_ABSITA (Aspect Term Extraction and Aspect-Based Sentiment Analysis). In particular, we focused on Task 3 using an approach based on combining frequency of words with lexicon-based polarities and uses Boosted Trees to predict the sentiment score. This approach achieved a competitive error and, thanks to the interpretability of the building blocks, allows us to show the what elements are considered when making the prediction. We also joined Task 1 proposing a hybrid model that joins rule-based and machine learning methodologies in order to combine the advantages of both. The model proposed for Task 1 is only preliminary.

Italiano. *Questo articolo descrive la nostra sottomissione ai tasks sulla Sentiment Analysis ATE_ABSITA (Aspect Term Extraction and Aspect-Based Sentiment Analysis). I nostri sforzi si sono concentrati sul Task 3 per il quale abbiamo adottato gli alberi di predizione (Boosted Trees) utilizzando come features di ingresso una combinazione basata sulla frequenza delle parole con la polarità derivate da un lessico. L'approccio raggiunge un errore competitivo e, grazie all'interpretabilità dei moduli intermedi, ci consente di analizzare in dettaglio gli elementi che caratterizzano maggiormente la fase di predizione. Una proposta è stata realizzata anche per il Task 1, dove abbiamo sviluppato un modello ibrido che combina un approcio basato su regole con tecniche Machine Learning. Il modello sviluppato per il Task 1 è solo in fase preliminare.*

1 Introduction

User feedback has become essential for companies to improve their services and products. Nowadays, we can find user feedback in textual form as online reviews, posts on social media and so on. These resources can express overall opinions but also opinions about some specific details (aspects) of the subject. In this scenario, the tools provided by Sentiment Analysis are crucial to process user feedbacks, the ongoing research in this field is focused on creating models that are more and more accurate and that can also extract fine grained information for the data. As part of this research, the ATE_ABSITA tasks (de Mattei et al., 2020)[1], part of the EVALITA campaign (Basile et al., 2020), challenge the participants in extracting the aspects (Task 1), predict the sentiment towards each expect (Task 2) and also predict the overall sentiment expressed (Task 3) for a dataset containing reviews of items from an online shop.

It's important to notice that the dataset released for the task is one of the few resources for the Italian language that has annotated aspects and sentiment at the same time. Others Italian resources that take into account sentiment with respect to aspects are (Sorgente et al., 2014) and (Croce et al., 2013). The first contains reviews of movies with 8 domain specific aspects and 5 different polarity values while the second contains opinions about wines considering 5 aspects and 3 possible polarity values.

This paper describes our approaches in solving task 1 and task 3. The approach for task 1 is still preliminary.

---

[1] http://www.di.uniba.it/ swap/ate_absita/index.html

In the last decade top performing approaches to Sentiment Analysis have shifted from using classifiers on hand-crafted features, often based on lexicons (Zhu et al., 2014), to complex models based on deep Neural Networks and advanced word embeddings (Liu et al., 2020). While the latest models require special hardware and significant work to be trained, older approaches are built on top of well understood classification techniques that can be trained on commodity hardware which makes them easy to adapt for new applications. The approach proposed for Task 3 revisits the old fashioned style of doing Sentiment Analysis to see how it performs against more modern methodologies that are used in the competition.

Regarding Task 1 we follow the latest trend of exploiting linguistic patterns (Poria et al., 2016; Liu et al., 2015; Poria et al., 2014; Rana and Cheah, 2019). What distinguishes our approach from others is that we use automatically generated patterns based on POS-Tags (Part of Speech-Tags) following the assumption that they are more robust to bad grammar compared to linguistic dependencies.

In Section 2 we will describe our approach for Task 3 and in Section 2.4 we will discuss the results. In Section 3 we will briefly discuss the preliminary model we build for Task 1 and its results.

2 Our approach for Task 3

The idea behind our approach is to achieve competitive results using well known tools that can be used on commodity hardware. We build the features representing the text using n-grams and adding a set of characteristic annotated in SenticNet (Cambria et al., 2010). Given the large amount of features, we decided to use Boosted Trees as regression model given its ability to sub-sample the features dynamically. For textual preprocessing the libraries Spacy (Honnibal and Montani, 2017) and Scikit-Learn (Pedregosa et al., 2011) were used. We chose XGboost (Chen and Guestrin, 2016) as implementation of Boosted Trees for regression.

2.1 Lexical features

Before extracting the lexical features we remove stop words (apart from words that can be used as negative adverbs) and lemmatized each word. Finally, we extract a set of n-grams from each review. We consider uni-grams, bi-grams and tri-grams at the same time.

2.2 Lexicon-based features

To build the polarity features of our model, we have adopted SenticNet, a resource used for concept-level sentiment analysis. It contains a collection of concepts, including common-sense concepts, provided with values for polarity, attention, pleasantness and sensitivity. These are numerical features that are available for a subset of the words in each review. We take in account the average, the minimum and the maximum of all the values available in each review. We also consider the mood tags provided by SenticNet. They are sets of tags as #tristezza, #rabbia, #felicità[2] attached to each word, we consider them as binary features.

2.3 Regressor

Our final regressor is composed of 800 Decision Trees with a maximum depth of 4 layers. The model was trained using Gradient Boosting with a learning rate of 0.3. The final prediction is computed averaging the output of each tree. The rationale behind our choice is that we have a high number of features that are easy to use with tree based methods for specific cases, hence ensembling allows us to learn a set of shallow trees and each of them can work well for specific cases.

2.4 Results and discussion

To build our model we initially focused on the training set using cross-validation to optimize the parameters achieving a root mean square error of 0.852 (the prediction target is on a scale from 1 to 5), we then tested the optimized model on the development set reaching an error of 0.805. We finally achieved an error of 0.795 on the final test set. The difference in the error across the different stages of validation suggests that the model is well trained as the error doesn't increase when new data is presented. However, it also suggest that the estimation of the error has a wide confidence interval, the standard deviation estimated during cross validation is 0.049.

In Figure 1 we compare the scores predicted and the annotated score on the development set. The chart shows that the model has a tendency to over estimate the error, especially in cases annotated with a low score.

---

[2] In English: #sadness, #anger, #happiness

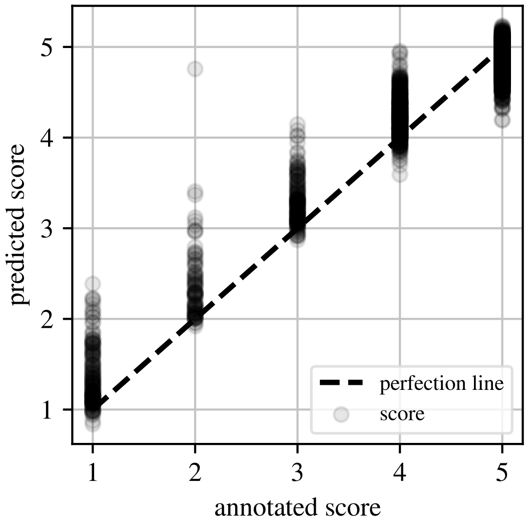


Figure 1: Scatter plot that shows the annotated score against the predicted score on the development set.

term	importance	coverage %
pessimo	0.057123	5.712323
purtroppo	0.038088	9.521134
rimborsare	0.037871	13.308205
non consigliare	0.033299	16.638059
purtroppo essere	0.027965	19.434580
cattivo	0.025690	22.003609
dispiacere	0.024986	24.502171
pensare	0.018631	26.365243
sconsigliare	0.016331	27.998360
dopo	0.016239	29.622279
non funzionare	0.015425	31.164802
delusione	0.015227	32.687547
non riconoscere	0.014809	34.168431
restituire	0.014615	35.629894
bruciare	0.014250	37.054852

Table 1: Important terms highlighted by the model. The column importance reports the importance score of the term while coverage is the cumulative sum of the importance scores.

We will now examine two reviews for which our regressor has the highest error. This is the text of the first review:

“si autospenge proprio quando si necessita di usarla contelecomando”[3].

This review was annotated with a score of 2, but the score assigned by our system is 4.75. This highlights a tendency of the system to give higher scores in uncertain cases. In this specific case we have no adjectives and two typing mistakes that result in no information from the lexicon and most of the words being disregarded as rare by our preprocessing pipeline. This suggests that a special treatment is needed for these specific cases where the classifier has fewer elements to take a decision.

The text of the second review is:

“Per questo prezzo c’è di meglio.. restituita.Gli accessori sono ottimi.”[4].

This sentence was annotated with a score of 2, but the score assigned by our system is 3.36. We have again a case of over estimation of the score. This time the review has two contrasting sentences. A very negative one where the user states of having returned the item and a very positive one regarding the accessories. This ambivalence makes the review a borderline case for our model.

We attribute this tendency to overestimate the target to the fact that the model is optimized to minimize the root-mean-square error, this makes the model predict values closer to the average annotated score. While this is acceptable in an academic competition, it’s less than ideal in an industrial setting. One way to solve the overestimation problem, without changing the formulation of the error to minimize, would be to balance the data so to have a similar number of occurrences for each score. Sub-sampling the data is unpractical as it would reduce the sample size too drastically. This leaves open only the option to add more samples.

In Table 1 we see the 15 terms most influential on the model. Here we note that most of the terms have a negative connotation. Interestingly, all the bi-grams in the list contain the word *non* (not). Taking in account that the terms reported in the table add up to 37% of the importance of all the features, this highlights the fact that the regressor puts particular attention in the prediction of reviews with a low score even if they are a minority.

3 Preliminary results on Task 1

Task 1 asks to identify terms and phrases that contain an aspect of the customer review when it co-

---

[3] In English: It turns off on its own when you need to use it with the remote control. (The original sentence contains a two typos.)

[4] In English: There’s a better choice for the same price.. I returned it.The accessories are great.

occurs with opinion words that bring information about the sentiment polarity. [5]

For this task we have designed a hybrid model that joins a rule-based approach with machine learning. The main idea is to identify a set of plausible aspects via some pre-defined rules, then use a classifier to filter out the wrong candidates. The rules are defined on POS-Tagging patterns. For example the review

"Ottimo rasoio dal semplice utilizzo."

with annotated as aspect *"semplice"* matches the rule defined by the following pattern

ADJ NOUN PROPN **ADJ** NOUN.

The bold tag indicates the position of the plausible aspect. We have defined a set of about 3000 rules. The rules have been discovered picking the most common POS-Tagging patterns that match the annotated aspects. In particular we have found the position of the aspects in the sentence and selected the POS of close words (three on each side) taking in account the punctuation.

Each aspect found can match one or more rules. The activation of each rule is used as binary feature for the final classifier. The final classifier is implemented using Logistic Regression (Hastie et al., 2001), its target is to predict if each candidate found by the rules is an actual candidate or a false positive.

This preliminary effort achieves a F1-score of 0.340, which is above the baseline (0.255) but below the average score of the submissions (0.504).

4 Conclusions

The submission confirmed the effectiveness of using a simple approach to predict the sentiment score of customer reviews in Italian (Task 3). The approach consists in combining simple word embedding, specifically tri-grams, and a lexicon as SenticNet to build features for Boosted Trees. Our system achieved a competitive error which is lower than the baseline by 0.209 points and higher than the best model by 0.131 points. The error achieved above the average official score by 0.067 points (the estimates includes baseline models).

The submission also highlights that we were able to beat the baseline for Task 1 with a rudimentary approach. We will build upon this approach in our future work.

References

[Basile et al.2020] Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

[Cambria et al.2010] Erik Cambria, Robert Speer, Catherine Havasi, and Amir Hussain. 2010. Senticnet: A publicly available semantic resource for opinion mining. In *AAAI fall symposium: commonsense knowledge*, volume 10. Citeseer.

[Chen and Guestrin2016] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. ACM.

[Croce et al.2013] Danilo Croce, Francesco Garzoli, Marco Montesi, Diego De Cao, and Roberto Basili. 2013. Enabling advanced business intelligence in divino. In *DART@AI*IA*, pages 61–72.

[de Mattei et al.2020] Lorenzo de Mattei, Graziella de Martino, Andrea Iovine, Alessio Miaschi, Marco Polignano, and Giulia Rambelli. 2020. ATE_ABSITA@EVALITA2020: Overview of the Aspect Term Extraction and Aspect-based Sentiment Analysis Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

[Hastie et al.2001] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

[Honnibal and Montani2017] Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

[Liu et al.2015] Qian Liu, Zhiqiang Gao, Bing Liu, and Yuanlin Zhang. 2015. Automated rule selection for aspect extraction in opinion mining. In *Twenty-Fourth international joint conference on artificial intelligence*.

[Liu et al.2020] Jiaxiang Liu, Xuyi Chen, Shikun Feng, Shuohuan Wang, Xuan Ouyang, Yu Sun, Zhengjie Huang, and Weiyue Su. 2020. kk2018 at semeval-2020 task 9: Adversarial training for code-mixing sentiment classification. *arXiv preprint arXiv:2009.03673*.

[Pedregosa et al.2011] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel,

---

[5] Detailed description of the task at http://www.di.uniba.it/ swap/ate_absita/task.html

M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Poria et al.2014] Soujanya Poria, Erik Cambria, Lun-Wei Ku, Chen Gui, and Alexander Gelbukh. 2014. A rule-based approach to aspect extraction from product reviews. In *Proceedings of the second workshop on natural language processing for social media (SocialNLP)*, pages 28–37.

[Poria et al.2016] Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2016. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems*, 108:42–49.

[Rana and Cheah2019] Toqir A Rana and Yu-N Cheah. 2019. Sequential patterns rule-based approach for opinion target extraction from customer reviews. *Journal of Information Science*, 45(5):643–655.

[Sorgente et al.2014] Antonio Sorgente, Giuseppe Vettigli, and Francesco Mele. 2014. An italian corpus for aspect based sentiment analysis of movie reviews. In *First Italian Conference on Computational Linguistics CLiC-it*.

[Zhu et al.2014] Xiaodan Zhu, Svetlana Kiritchenko, and Saif Mohammad. 2014. Nrc-canada-2014: Recent improvements in the sentiment analysis of tweets. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 443–447.

ghostwriter19 @ ATE_ABSITA: Zero-Shot and ONNX to Speed up BERT on Sentiment Analysis Tasks at EVALITA 2020

Mauro Bennici
You Are My Guide
Torino
`mauro@youaremyguide.com`

Abstract[1]

English. With the arrival of BERT[2] in 2018, NLP research has taken a significant step forward. However, the necessary computing power has grown accordingly. Various distillation and optimization systems have been adopted but are costly in terms of cost-benefit ratio. The most important improvements are obtained by creating increasingly complex models with more layers and parameters.

In this research, we will see how, by mixing transfer learning, zero-shot learning, and ONNX runtime[3], we can access the power of BERT right now, optimizing time and resources, achieving noticeable results on day one.

Italiano. Con l'arrivo di BERT nel 2018, la ricerca nel campo dell'NLP ha fatto un notevole passo in avanti. La potenza di calcolo necessaria però è cresciuta di conseguenza. Diversi sistemi di distillazione e di ottimizzazione sono stati adottati ma risultano onerosi in termini di rapporto costo benefici. I vantaggi di maggior rilievo si ottengono creando modelli sempre più complessi con un maggior numero di layers e di parametri.

In questa ricerca vedremo come mixando transfer learning, zero-shot learning e ONNX runtime si può accedere alla potenza di BERT da subito, ottimizzando tempo e risorse, raggiungendo risultati apprezzabili al day one.

1 Introduction

In a process with data that change very quickly and the need to resort to complete training in the shortest possible time, transfer learning techniques have made possible a fast fine-tuning of BERT models. The distillation of a model made it possible to decrease the load and the times without significantly losing accuracy. These models, therefore, require, at least, constant fine-tuning training. In addition, a BERT model specially designed for the Italian language and with a vocabulary containing technical terms increases its effectiveness.

Constant and multi-disciplinary training requires specific skills and tailor-made services. In this research, we will see an effective way to make both things possible. The idea is to use a way to exchange AI models between library and frameworks, the ONNX project, and a runtime, the ONNX runtime project, to optimize inference for many platforms, languages and hardware. The ONNX runtime is still working to optimize the training directly in the ONNX format.
The second goal is to find a viable alternative with acceptable performance at the start of a new project while waiting for a trained BERT model.

The research was carried out for the ATE ABSITA (de Mattei et al., 2020) task in the EVALITA 2020 (Basile et al., 2020), using all 3 available sub-tasks.

2 Description of the system

To start using a sentiment analysis system, we need several elements. Certainly, a starting dataset

---

[2] https://github.com/google-research/bert

[3] https://microsoft.github.io/onnxruntime/

with the related labels. In the tasks of the challenge, we have the reviews of 23 different products. Each review has a corresponding rating assigned by the end-user. For each review, it was required to extract the aspects contained in it. By aspect, we mean every opinion word that expresses a sentiment polarity. Finally, each aspect was classified as a pair of values: positive or negative, for 4 possible states.

Imagine a system that receives an unspecified number of reviews in real-time with new products and different categories. We find ourselves in the situation of always having to fine-tune our models.

The complexity of BERT makes training time difficult for constant alignment. Being able to reduce the training time, or being able to put in place an alternative in the meantime, new perspectives open up, such as:

- Made inference calls before a full trained model is completed.
- Training of the new model.
- Running the BERT model.
- (optional) reclassify recent product reviews after the model update.

In this perspective, in order to validate my hypotheses, I used the AlBERTo (Polignano at al., 2019) model, used in the baseline, and Ktrain[4], a wrapper for TensorFlow[5], with the autofit option.

The first submission, called ghostwriter19_a, was obtained training all the models with the Ktrain framework.

The results for the three tasks for the second submission, called ghostwriter19_b, were obtained in two different way:

- for the first two tasks, I used the model of the first submission but exported on ONNX and ran with the ONNX runtime.
- for the third task, I trained the model with TensorFlow using a Zero-Shot learner [ZSL] (Brown et al, 2020).

To test the models, I used two different machines with Ubuntu 20.04 LTS:

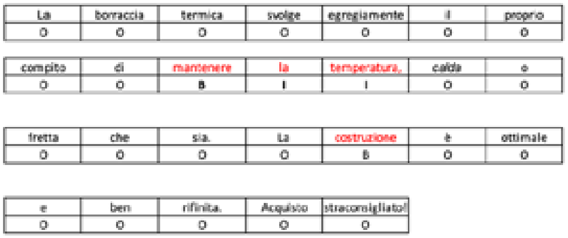
- 6 vCPU on Intel Xeon E5-2690 v4 - 112GB with P100 (GPU)
- 14 cores on Intel Xeon E5-2690 v4 - 32GB (CPU)

2.1 Task 1 – ATE: Aspect Term Extraction

To identify an aspect, the dataset contains a label for every single word with three possible values:

- B for Begin of an aspect.
- I for Inside an aspect.
- Or for Outside, not in an aspect.

For example, the review "*La borraccia termica svolge egregiamente il proprio compito di mantenere la temperatura, calda o fretta che sia. La costruzione è ottimale e ben rifinita. Acquisto straconsigliato!*" is labeled as:



La	borraccia	termica	svolge	egregiamente	il	proprio
O	O	O	O	O	O	O

compito	di	mantenere	la	temperatura,	calda	o
O	O	B	I	I	O	O

fretta	che	sia.	La	costruzione	è	ottimale
O	O	O	O	B	O	O

e	ben	rifinita.	Acquisto	straconsigliato!
O	O	O	O	O

The model will be evaluated with the F1-score. The score results from the full matched aspects, the partial matched ones, and the missed ones.

The preliminary results with the Ktrain model were encouraging (table 1).

Model	F1-Score
ghostwriter19_a	0.6152
Baseline	0.2556

Table 1: Task 1 DEV results

At this point, the model has been exported with ONNX in maximum compatibility mode. The model ran with the ONNX runtime optimized for CPU.

The performances have remained unchanged, but the speed of inference has significantly improved (table 2).

---

[4] https://github.com/amaiya/ktrain

[5] https://www.tensorflow.org/

Model	Query per second
ghostwriter19_a CPU	4
ghostwriter19_b CPU with ONNX runtime	68
ghostwriter19_a GPU	124
ghostwriter19_b GPU with ONNX runtime	217

Table 2: Performance comparison on Task 1

The improvement is 17x for the CPU version and 1.75x for the GPU version.

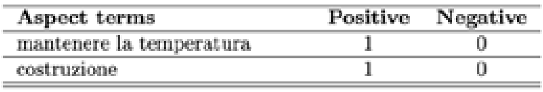
2.2 Task 2 – ABSA: Aspect-based Sentiment Analysis

For this task, the aspects identified in Task 1 have been used. This implies that an error in Task 1 will have a decisive impact on Task number 2.

The aspect can be classified as:

- positive (POS:true,NEG:false)
- negative (POS:false,NEG:true)
- mixed polarity(POS:true, NEG:true)
- neutral polarity (POS:false, NEG:false)

As showed to the image from the challenge website[6]:



Aspect terms	Positive	Negative
mantenere la temperatura	1	0
costruzione	1	0

The results on the DEV test outperform the baseline (table 3).

Model	F1-Score
ghostwriter19_a	0.6019
Baseline	0.2

Table 3: Task 2 DEV results

Also, for this task, the performance is improved with the use of ONNX runtime (table 4).

Model	Query per second
ghostwriter19_a CPU	3
ghostwriter19_b CPU with ONNX runtime	56
ghostwriter19_a GPU	97
ghostwriter19_b GPU with ONNX runtime	154

Table 4: Performance comparison on Task 2

The improvement is 9.5x for the CPU version and 1.59x for the GPU version.

2.3 Task 3 – SA: Sentiment Analysis

Task 3 is a classification problem. However, fully understanding the score is not easy. The evaluation operation is carried out by different people and with different styles. A product with a similar review is rated according to the expectations and judgment of other users differently.

Furthermore, in order to obviate the long training time that a constant updating requires, compared with systems used by the previous version of EVALITA, such as an ensemble system with Tree Random Forest and Bi-LSTM (Bennici and Portocarrero, 2018) or with an SVM system (Barbieri et al., 2016), I used a Zero-Shot Learner [ZSL] (Pushp & Srivastava, 2017). A ZSL is a way to make predictions without prior training (Petroni, 2019). ZSL will refer to the embedding of a previous matrix, AlBERTo in this case, and of the proposed labels as a possible result (Schick and Schütze, 2020).
The proposed labels were the possible numbers for evaluation, then the numbers from 1 to 5.

The proposed prediction value is a weighted average of the two values with the highest probability, if and only if the gap between the two values is less than 10^{-3}. Otherwise, only the value with the highest probability will be considered valid.

For this task, I omitted the ONNX runtime test because a stable converter for the ZSL version is not available.

---

[6] http://www.di.uniba.it/~swap/ate_absita/task.html

The score for this task is the Root Mean Squared Error between the polarity predicted and the polarity assigned by the user.

Model	RMSE score
ghostwriter19_a	0.6997
ghostwriter19_b	0.8526
Baseline AlBERTo	1.0806

Table 5: Task 3 DEV results

The loss in performance is 18%, but the entire previous training phase is skipped (table 5).

3 Results

The results obtained with the DEV dataset are very positive both in terms of accuracy and performance. ZSL has proven to be an incredible technology to invest in. The Ktrain seems to suffer a heavy overfit.

The research aims not to have a relevant model but to prove that a model could be production-ready with fewer resources and time.
However, in all three tasks, the models outperformed the baseline with a significant gap in terms of accuracy/RMSE.

3.1 Results for Task 1

The final results with the TEST dataset are:

Model	F1 score
ghostwriter19_a_D	0.6152
ghostwriter19_a_T	0.5399
Baseline AlBERTo	0.2556

Table 6: TEST dataset results for Task 1

The results are about 12% lower than those obtained in the research phase (table 6).
It will be interesting to continue experimenting with different ONNX options to find a better combination of compatibility and performance.

3.2 Results for Task 2

The final results with the TEST dataset are:

Model	F1 score
ghostwriter19_a_D	0.6019
ghostwriter19_b_T	0.4994
Baseline AlBERTo	0.2

Table 7: TEST dataset results for Task 2

The loss from DEV to TEST is about 17% (table 7). However, the percentage of the difference between the results of Tasks 1 and 2 have been maintained with the DEV and TEST datasets.

This is in line with expectations, worse model performance in Task 1 impacted Task 2 proportionally. In return, working on a better model will improve both tasks.

3.3 Results for Task 3

For Task 3 we have:

Model	RMSE score
ghostwriter19_a_D	0.6997
ghostwriter19_b_D	0.8526
ghostwriter19_a_T	0.81394
ghostwriter19_b_T	0.83479
Baseline AlBERTo	1.0806

Table 8: TEST dataset results for Task 3

The difference between the DEV and TEST datasets is marked here only for trained model, 14% (table 8). The untrained one performed slightly better, 2%, with the TEST dataset.

This result confirms that an underperforming model has the same performance of a model that use ZSL, as assumed.

The price to pay, however, is that the average inference time for the ZSL is 157x higher than the pure TensorFlow model obtained with Ktrain.

4 Conclusion

The results demonstrated that it is possible to create hybrid systems for training and inference to make the power of BERT more accessible.

In the time it takes to train a new and optimized model, an untrained ZSL model can make up for it in the meantime.

Optimizing, and in future training, our models to be intrinsically optimized for the platform and framework we have chosen to use does not affect performance and future use.

The improvements obtained in the use of ONNX runtime for these Italian tasks are in line with what Microsoft demonstrated, for the English language, at the beginning of 2020 (Ning at al., 2020).

The next step is to make the ONNX export work with a Zero-Shot learner [ZSL] in order to compensate, at least in part, for the more significant resources that this inevitably introduces.

References

Barbieri, F., Basile, V., Croce, D., Nissim, M., Novielli, N., & Patti, V. (2016). Overview of the Evalita 2016 SENTIment POLarity Classification Task. In *Proceedings of Third Italian Conference on Computational Linguistics (CLiC-it 2016) & Fifth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2016)*. CEUR-WS.org.

Basile, V., Croce, D., Di Maro, M., & Passaro, L. (2020). EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020),* CEUR-WS.org.

Bennici, M., & Portocarrero, X. S. (2018). Ensemble for aspect-based sentiment analysis. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*. CEUR-WS.org.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., . . . Amodei, D. (2020, July 22). Language Models are Few-Shot Learners. https://arxiv.org/abs/2005.14165

de Mattei, L., de Martino, G., Iovine, A., Miaschi, A., Polignano, M., & Rambelli, G. (2020). Overview of the EVALITA 2020 Aspect Term Extraction and Aspect-based Sentiment Analysis (ATE_ABSITA) Task. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, CEUR-WS.org.

Ning, E., Yan, N., Zhu, J., & Li, J. (2020, January 31). Microsoft open sources breakthrough optimizations for transformer inference on GPU and CPU. https://cloudblogs.microsoft.com/opensource/2020/01/21/microsoft-onnx-open-source-optimizations-transformer-inference-gpu-cpu/

Polignano, M., Basile, P., de Gemmis, M., Semeraro, G., & Basile, V. (2019). Alberto: Italian bert language understanding model for nlp challenging tasks based on tweets. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*. CEUR-WS.org.

Petroni, F., Rocktäschel, T., Lewis, P., Bakhtin, A., Wu, Y., Miller, A. H., & Riedel, S. (2019, September 04). Language Models as Knowledge Bases? https://arxiv.org/abs/1909.01066

Pushp, P. K., & Srivastava, M. M. (2017, December 23). Train Once, Test Anywhere: Zero-Shot Learning for Text Classification. https://arxiv.org/abs/1712.05972

Schick, T., & Schütze, H. (2020, April 27). Exploiting Cloze Questions for Few Shot Text Classification and Natural Language Inference. https://arxiv.org/abs/2001.07676

App2Check @ ATE_ABSITA 2020: Aspect Term Extraction and Aspect-based Sentiment Analysis

Emanuele Di Rosa
Chief Technology Officer
`emanuele.dirosa@app2check.com`

Alberto Durante
Research Scientist
`alberto.durante@app2check.com`

Abstract

In this paper we describe and present the results of the system we specifically developed and submitted for our participation to the ATE_ABSITA 2020 evaluation campaign on the Aspect Term Extraction (ATE), Aspect-based Sentiment Analysis (ABSA), and Sentiment Analysis (SA) tasks. The official results show that App2Check ranks first in all of the three tasks, reaching a F1 score which is 0.14236 higher than the second best system in the ATE task and 0.11943 higher in the ABSA task; it shows a Root-Mean-Square Error (RMSE) that is 0.13075 lower than the second classified in the SA task.

1 Introduction

User reviews are becoming more important for all consumer-oriented industries. Thanks to the expansion of a review culture, collecting and sharing a feedback from a buyer of a product/service can both help the seller to improve and other customers who can take advantage of the reviews for their purchase decisions. However, having automatic tools to process reviews and extract useful insights to analysts, especially where large amounts of reviews are available, becomes relevant for any consumer-oriented industry.

Aspect-Term Extraction and Aspect-Based Sentiment Analysis tasks are, respectively, focused on the extraction of the main aspects in a sentence and to assign a specific sentiment to each of them. These are essential tools to understand the reasons behind the success or the failure of a product or service, or anyway that allow to take actions, finalized to to improve the customer perception. The former helps analysts to go beyond the traditional ”word cloud” that is available in most of text analytic tools and that focuses just on the most recurrent words in a collection. Aspect-Term Extraction, similarly to the Named-Entity Recognition task, detects a sequence of word tokens that conceptually identify an ”aspect” of the sentence. The Sentiment Analysis task maintains its importance on a higher level, where it can substitute user rating, which can be sometimes incoherent to the opinions expressed in the review text. Anyway, it represents just the overall polarity of an opinion, which is very often the result of different polarities on multiple aspects. The assignment of a specific and, in general, different polarity to each aspect in the sentence, leads to the ABSA task, which is highly dependent on the ATE task, but can take advantage of the learning obtained by an SA model. In the last few years, deep learning-based models proved to be the best technical approach for natural language processing and understanding and are very promising also for the ATE, SA and ABSA tasks.

In this paper, we present the system that we specifically developed and submitted for our participation to the ATE_ABSITA 2020 evaluation campaign (De Mattei et al., 2018), which is part of EVALITA 2020 (Basile et al., 2020), on the Aspect Term Extraction (ATE), Aspect-based Sentiment Analysis (ABSA), and Sentiment Analysis (SA) tasks. To this aim, we decided to focus just on deep learning-based approaches to train a specific model for each task. More specifically, we take advantage of the most recent approach in which pre-trained language models, largely recognized as bringing NLP to a new era (Qiu et al., 2020), are used as the main component for the 3 tasks. In particular, about the ATE task, in order to select the best performing pre-trained models to use for our submission, we performed an extensive experimental analysis and comparison. The experimen-

---

tal evaluation shows some interesting and unpredictable results, discussed in section 2, which also represent an added value of this paper. In fact, we can summarize that in the dev set:

1. the NER fine-tuned model shows lower performance than general-purpose pre-trained models without a specific NER fine-tuning
2. a language specific, Italian-native model shows a lower performance than multilingual models fine-tuned on Italian in the specific downstream tasks
3. the biggest and most recent, multilingual XLM-Roberta model shows the best performance when fine-tuned on the downstream tasks

While the last result, related to the fact that bigger models –in terms of number of parameters– are more effective than smaller models, is quite common (with the exception of distilled models) and known in literature (see also the recent GPT3 vs GPT2 comparison (Brown et al., 2020)), the first two results are quite surprising. In fact, we expected that the multilingual model specifically fine-tuned on the NER task on another language could take advantage of a previous training in another language as shown in (Pires et al., 2019). Moreover, the native Italian pre-trained language model GilBERTo (based on Facebook RoBERTa architecture (Liu et al., 2019) and CamemBERT text tokenization approach (Martin et al., 2020)) later fine-tuned on the NER task with italian training set, shows a performance that is 4% lower than the XML-Roberta multilingual pre-trained model later trained on a NER training set in Italian.

About the SA task, we take advantage of a previously trained predictive model we had at App2Check, an evolution of the one presented in (Di Rosa and Durante, 2017), which is now based on the Multilingual BERT model and later fine-tuned on a 1 to 5 sentiment scale on a big amount of product reviews. This model has been additionally trained on the training set of the competition in order to have a domain-specific training. For the SA task, we decided to not perform any additional experimental comparison with other pre-trained models. Finally, about the ABSA task, we created a special encoding to map the output of our available SA model in order to be additionally fine-tuned on the ABSA training set of the competition: this helped to take advantage of a transfer learning from the SA task to the ABSA task.

This paper is structured as follows: in sections 2, 3 and 4, we describe each of the three tasks of the competition, the details of our training, system implementation and present the results in both the dev set and the competition results. Finally, we show the conclusions in section 5.

2 Aspect-Term Extraction

Aspect Term Extraction (ATE) is the task of identifying an "aspect" in a text without knowing a priori the list of aspects that contains it. According to the literature definition, a term/phrase is considered as an aspect when it co-occurs with "opinion words" that indicate a sentiment polarity on it.

Our approach has been to consider the ATE task as a Named Entity Recognition task (NER) and fine-tune already existing pre-trained language models on the NER task, by using the training set of the competition. More specifically, we decided to investigate four different classes of models:

1. Native Italian pre-trained language models, with no specific NER fine-tuning
2. Multilingual pre-trained language model, with no specific NER fine-tuning
3. Native Italian pre-trained language models, with a specific NER fine-tuning
4. Multilingual pre-trained language model, with a specific NER fine-tuning

To implement all of these approaches, we based on the Hugging Face transformers library (Wolf et al., 2019) and, in order to simplify our work, we looked for pre-trained models made available publicly by the Hugging Face. With the exception of item 3, for which we could not find any publicly available model in the HuggingFace models list, we considered more than one state-of-the-art model for each type of encoding that we further trained/fine-tuned on the competition training set.

For type 1, we considered dbmdz/bert-base-italian-xxl-uncased[1] and GilBERTo[2]. For type 2, we considered two implementations of RoBERTa:

---

[1] https://github.com/dbmdz/berts

[2] https://github.com/idb-ita/GilBERTo

xml-roberta-large[3] (Conneau et al., 2020), xml-roberta-base[4] (Liu et al., 2019), and multilingual BERT[5] (Pires et al., 2019). We wanted to try xml-roberta-large with a 512 maximum sequence length, but an out of memory exception prevented us for using it. For type 4 we considered wietsedv/bert-base-multilingual-cased-finetuned-conll2002-ner[6].

K	Len	Model	Ep	F1-T	F1-D
1	512	B-BERT ita unc.	11	0.961	0.663
1	512	GilBERTo unc.	10	0.941	0.697
1	512	GilBERTo unc.	15	0.973	0.6700
2	512	B-xlmRoBERTa	8	0.981	0.687
2	256	L-xlmRoBERTa	12	0.965	0.728
2	256	**L-xlmRoBERTa**	15	**0.980**	**0.708**
2	512	B-mBERT	20	0.991	0.679
4	512	B-mBERT NER	30	0.910	0.657
4	512	B-mBERT NER	45	0.965	0.623

Table 1: Aspect-Term Extraction performance on development set.

All models have been trained on a cloud platform using an Nvidia Tesla P100-PCIE as GPU accelerator. In Table 1 we show the results obtained by the models on the training and development set, highlighting in bold the model chosen for the competition.

The value in column K, Len and Ep are associated respectively to the kind of pre-trained model used, the maximum sequence length used in the training and the number of epochs of the training. The F1-T and F1-D columns contains the F1-scores on training set and development set. For each model, the prefixes *L* and *B* indicates whether the base or large version has been used; if an uncased version of the pre-trained model has been used, the model name is labeled with *unc.*.

The Italian Base Bert and GilBERTo approaches, both of class 1, show similar results on both training and development set. Interestingly, on the development set, the multilingual Base Bert model in class 2 shows very similar results to the best model in class 1 which is specifically trained on Italian.

The xlm RoBERTa Large multilingual model shows a F1-score on the development set that is higher than the Base version of the same model, even if they show almost the same performance on the training set. The model in class 4, multilingual Bert Base specifically trained on the NER task, shows the worst performance on the development set, even if trained with a much higher number of epochs.

Thanks to the F1 score reached on the development set, the xlm RoBERTa Large multilingual model has been chosen as our competition model, so it has been further trained on the development set and tested on the competition test set.

Pos.	Name	F1 score
1	App2Check	0.68222
2	ghostwriter19	0.53986
3	SentNa	0.34027
4	*Baseline*	*0.2556*

Table 2: Aspect-Term Extraction on the test set of the competition.

In Table 2 we show the official results of the Aspect-Term Extraction task in (De Mattei et al., 2018). App2Check model ranked first with a F1 score that is 0.14236 higher than the second best system.

3 Sentiment Analysis

The SA task is about the detection of the opinion expressed in a text review. According to the typical user rating, which is here used as the reference value for the polarity, the score is defined on a five-value scale from 1 (very negative) to 5 (very positive).

About our implementation for this task, we took advantage of a previously trained predictive model we had at App2Check. It is an evolution of the one presented in (Di Rosa and Durante, 2017), which is now based on the Multilingual BERT model based on 104 languages and 110M parameters, and later fine-tuned on a 1 to 5 sentiment scale on a big amount of product reviews. This model has been additionally trained on the training set of the competition in order to have a domain-specific training. We decided to not perform any additional experimental comparison with other pre-trained models, since it has been already compared with other approaches in the past and also because of the little time at our disposal.

In Table 3 we show the results of the competition for the Sentiment Analysis task. The root-

---

[3] https://huggingface.co/xlm-roberta-large

[4] https://huggingface.co/xlm-roberta-base

[5] bert-base-multilingual-cased

[6] https://github.com/chambliss/Multilingual_NER

Pos	Name	RMSE
1	App2Check	0.66458
2	SentNa	0.79533
3	ghostwriter19	0.81394
4	*Baseline-AVG score*	*0.10040*
5	*Baseline-AlBERTo*	*0.10806*
6	*Baseline-Freq score*	*0.12800*

Table 3: Sentiment Analysis on the test set of the competition.

mean-square error of App2Check is 0.13073 lower than the error of the second best system, ranking in first position.

4 Aspect-Based Sentiment Analysis

The Aspect-Based Sentiment Analysis task is an extension of both the ATE and the SA tasks. In fact, the aim of the Aspect-Based Sentiment Analysis task is to detect the sentiment polarity associated to each aspect extracted, thanks to the ATE task discussed in Section 2. The possible polarity values are:

Polarity	Value
neutral	[0,0]
positive	[1,0]
negative	[0,1]
mixed	[1,1]

Similarly to what we have done with the Aspect Category Polarity task at ABSITA 2018 (Di Rosa and Durante, 2018), we assumed that the sentiment score of every aspect detected in Section 2 is the one associated to the portion of text in which it is contained. In order to do so, we split portions of the review using strong punctuation marks and some conjunctions (especially the ones leading to sentiment inversion). For example, in the case of:

> *Ottimo prodotto di marca, la qualità é veramente notevole. Non è molto capiente ma si può prendere un'altra versione. È provvisto di una tasca piccola davanti e quella grande*[7]

The aspect *capiente*[8] has the same polarity score as *Non è molto capiente*, while the aspect *qualità*[9] has the same polarity score as *Ottimo prodotto di marca, la qualità é veramente notevole*.

The same assumption has been applied to the training set: the polarity of each portion of a review has been associated to the contained aspect. If a portion of a review does not contain any aspect, it has been ignored.

The submitted ABSA system has been based on a single sentiment classification model, rather than two binary models for positive and negative polarities. The final model is a four-class retraining of the sentiment model presented in section 3 which has been originally trained on user reviews with five levels (strong positive, positive, mixed/neutral, negative, strong negative) using multilingual BERT (Pires et al., 2019). In this way, we take advantage of some transfer learning about positive, negative and neutral sentiment learned on reviews.

Pos.	Name	F1 score
1	App2Check	0.61878
2	ghostwriter19	0.49935
3	SentNa	0.28632
4	*Baseline*	*0.20000*

Table 4: Aspect-Based Sentiment Analysis on the test set of the competition.

In Table 4 we show the results of the Aspect-Based Sentiment Analysis of the competition. App2Check system is in first position, with a F1 score that is 0.11943 higher than the second best system.

5 Conclusions

In this paper we described the approach we followed and the models we built for our participation to the ATE_ABSITA 2020 competition. We also presented the experimental evaluation we made in the context of our model selection process in the development set and show interesting results: *(i)* the NER fine-tuned model shows lower performance than general-purpose pre-trained models without a specific NER fine-tuning; *(ii)* a language specific, Italian-native model shows a lower performance than multilingual models fine-tuned on Italian in the specific downstream tasks; *(iii)* the biggest and most recent, multilingual XLM-Roberta model shows the best performance when fine-tuned on the downstream tasks. We also showed that our App2Check

---

[7] Translation: *Great branded product, the quality is truly remarkable. It is not very capacious but you can get another version. It has a small front pocket and a large one*

[8] Translation: *capacious*

[9] Translation: *quality*

system scored first in all of the three tasks of the competition, reaching a F1 score which is 0.14236 higher than the second best system in the ATE task and 0.11943 higher in the ABSA task; in the SA task, our system shows a Root-Mean-Square Error (RMSE) that is 0.13075 lower than the second classified.

References

Basile, Valerio and Croce, Danilo and Di Maro, Maria and Passaro, Lucia C. 2020 *EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian* Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020) CEUR.org

Lorenzo de Mattei, Graziella de Martino, Andrea Iovine, Alessio Miaschi, Marco Polignano, and Giulia Rambelli. 2020 *ATE_ABSITA@EVALITA2020: Overview of the Aspect Term Extraction and Aspect-based Sentiment Analysis Task.* Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020). CEUR.org

Emanuele Di Rosa and Alberto Durante 2018 *Aspect-based Sentiment Analysis: X2Check at ABSITA 2018* Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018

Emanuele Di Rosa and Alberto Durante. 2017. *Evaluating Industrial and Research Sentiment Analysis Engines on Multiple Sources* in Proc. of AI*IA 2017 Advances in Artificial Intelligence - International Conference of the Italian Association for Artificial Intelligence, Bari, Italy, November 14-17, 2017, pp. 141-155.

Yinhan Liu and Myle Ott and Naman Goyal and Jingfei Du and Mandar Joshi and Danqi Chen and Omer Levy and Mike Lewis and Luke Zettlemoyer and Veselin Stoyanov 2019 *RoBERTa: A Robustly Optimized BERT Pretraining Approach* CoRR, abs/1907.11692

Alexis Conneau and Kartikay Khandelwal and Naman Goyal and Vishrav Chaudhary and Guillaume Wenzek and Francisco Guzmán and Edouard Grave and Myle Ott and Luke Zettlemoyer and Veselin Stoyanov 2020 *Unsupervised Cross-lingual Representation Learning at Scale* Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020

Louis Martin, Benjamin Müller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, Benoît Sagot *CamemBERT: a Tasty French Language Model.* ACL 2020: 7203-7219

Xipeng Qiu and Tianxiang Sun and Yige Xu and Yunfan Shao and Ning Dai and Xuanjing Huang 2020 *Pre-trained Models for Natural Language Processing: A Survey* 2003.08271, arXiv, https://arxiv.org/abs/2003.08271

Tom B. Brown and Benjamin Mann and Nick Ryder and Melanie Subbiah and Jared Kaplan and Prafulla Dhariwal and Arvind Neelakantan and Pranav Shyam and Girish Sastry and Amanda Askell and Sandhini Agarwal and Ariel Herbert-Voss and Gretchen Krueger and Tom Henighan and Rewon Child and Aditya Ramesh and Daniel M. Ziegler and Jeffrey Wu and Clemens Winter and Christopher Hesse and Mark Chen and Eric Sigler and Mateusz Litwin and Scott Gray and Benjamin Chess and Jack Clark and Christopher Berner and Sam McCandlish and Alec Radford and Ilya Sutskever and Dario Amodei 2020 *Language Models are Few-Shot Learners* CoRR 2020 https://arxiv.org/abs/2005.14165

Thomas Wolf and Lysandre Debut and Victor Sanh and Julien Chaumond and Clement Delangue and Anthony Moi and Pierric Cistac and Tim Rault and Rémi Louf and Morgan Funtowicz and Joe Davison and Sam Shleifer and Patrick von Platen and Clara Ma and Yacine Jernite and Julien Plu and Canwen Xu and Teven Le Scao and Sylvain Gugger and Mariama Drame and Quentin Lhoest and Alexander M. Rush 2019. *Transformers: State-of-the-art natural language processing.* arXiv preprint arXiv:1910.03771.

Telmo Pires and Eva Schlinger and Dan Garrette 2019 *How multilingual is Multilingual BERT?* CoRR 2019 http://arxiv.org/abs/1906.01502

Basile, Valerio and Croce, Danilo and Di Maro, Maria and Passaro, Lucia C. 2020. *EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian.* In Online Proceedings of Evalita 2020 Publisher: CEUR.org Editor: Basile, Valerio and Croce, Danilo and Di Maro, Maria and Passaro, Lucia C.

HaSpeeDe: Hate Speech Detection

HaSpeeDe 2 @ EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task

Manuela Sanguinetti[⋆], **Gloria Comandini**[◇], **Elisa di Nuovo**[⊕], **Simona Frenda**[⊕],
Marco Stranisci[⊕], **Cristina Bosco**[⊕], **Tommaso Caselli**[⊖], **Viviana Patti**[⊕], **Irene Russo**[†]
[⋆]Università degli Studi di Cagliari, [◇] Università degli Studi di Trento,
[⊕]Università degli Studi di Torino, [⊖]University of Groningen
[†]ILC-CNR Pisa
manuela.sanguinetti@unica.it, gloria.comandini@unitn.it,
{dinuovo,frenda,stranisc,bosco,patti}@di.unito.it,
t.caselli@rug.nl, irene.russo@ilc.cnr.it

Abstract

The Hate Speech Detection (HaSpeeDe 2) task is the second edition of a shared task on the detection of hateful content in Italian Twitter messages. HaSpeeDe 2 is composed of a Main task (hate speech detection) and two Pilot tasks, (stereotype and nominal utterance detection). Systems were challenged along two dimensions: (i) time, with test data coming from a different time period than the training data, and (ii) domain, with test data coming from the news domain (i.e., news headlines). Overall, 14 teams participated in the Main task, the best systems achieved a macro F1-score of 0.8088 and 0.7744 on the in-domain in the out-of-domain test sets, respectively; 6 teams submitted their results for Pilot task 1 (stereotype detection), the best systems achieved a macro F1-score of 0.7719 and 0.7203 on in-domain and out-of-domain test sets. We did not receive any submission for Pilot task 2.

1 Introduction and Motivations

From a NLP perspective, much attention has been paid to the automatic detection of Hate Speech (HS) and related phenomena (e.g., offensive or abusive language among others) and behaviors (e.g., harassment and cyberbullying). This has led to the recent proliferation of contributions on this topic (Nobata et al., 2016; Waseem et al., 2017; Fortuna et al., 2019), corpora and lexica[1], dedicated workshops[2], and shared tasks within national[3] and international[4] evaluation campaigns.

As for Italian, the first edition of HaSpeeDe (Bosco et al., 2018), a task specifically focused on HS detection, was proposed at EVALITA 2018 (Caselli et al., 2018). The task consisted of the binary classification (HS vs not-HS) of texts from Twitter and Facebook. For each social media platform, training and test data were provided. Furthermore, two cross-platform sub-tasks were introduced to test the systems' ability to generalize across platforms.

The ultimate goal of HaSpeeDe 2 at EVALITA 2020 (Basile et al., 2020) is to take a step further in state-of-the-art HS detection for Italian. By doing this, we also intend to explore other side phenomena and see the extent to which they can be automatically distinguished from HS.

We propose a single training set made of tweets, but two separate test sets within two different domains: tweets and news headlines. While social media are still one of the main channels used to spread hateful content online (Alkiviadou, 2019; Wodak, 2018), an important role in this respect is also played by traditional media, and newspapers in particular.

Furthermore, we chose to include another HS-related phenomenon, namely the presence of stereotypes referring to one of the targets identified within our dataset (i.e., muslims, Roma and immigrants). With the term stereotype we mean any explicit or implicit reference to typical beliefs and attitudes about a given target (Sanguinetti et al., 2018). An error analysis of the main systems on the HaSpeeDe 2018 dataset itself (Francesconi

---

[1] More details and an overview of available HS resources have been recently presented in Poletto et al. (2020).

[2] More detailed informations in: `https://www.workshoponlineabuse.com/`

[3] HASOC (Mandl et al., 2019), Poleval (Ptaszynski et al., 2019) or VLSD (Vu et al., 2019).

[4] Hateval task at Semeval 2019 (Basile et al., 2019).

et al., 2019) showed that the occurrence of these elements constitutes a common source of error in HS identification.

Finally, it has been observed that in social media and newspapers' headlines, the most hateful parts are often verbless sentences or a verbless fragments, also known as Nominal Utterances (NUs) (Comandini et al., 2018). The relevant presence of NUs has been investigated in the POP-HS-IT corpus (Comandini and Patti, 2019). In order to have a better understanding of the syntactic strategies used in HS, we include the recognition of NUs in hateful tweets and news headlines.

2 Task Description

HaSpeeDe 2[5] consists of a Main task and two Pilot tasks and is based on two datasets, one containing messages from a social media platform, namely Twitter, and the other one news headlines. The three tasks are shortly described as follows:

- **Task A - Hate Speech Detection (Main Task):** binary classification task aimed at determining the presence or the absence of hateful content in the text towards a given target (among immigrants, Muslims and Roma)
- **Task B - Stereotype Detection (Pilot Task 1):** binary classification task aimed at determining the presence or the absence of a stereotype towards the same targets as Task A
- **Task C - Identification of Nominal Utterances (Pilot Task 2):** sequence labeling task aimed at recognizing NUs in data previously labeled as hateful.

This edition of the task presents several distinguishing features with respect to the first one. Besides including new and more-richly annotated data, news headlines were introduced as cross-domain test data. Furthermore, two additional tasks are proposed. Finally, the Twitter test set intentionally contains tweets published in a different time frame than those in the training set to verify the systems' ability to detect HS forms independently of biases. These biases result from context-related features, such as events – regarding one of our HS targets – that can be controversial or be subject to heated and polarized debates.

3 Datasets and Formats

In this section we describe the datasets and formats used in the three tasks.

3.1 Twitter Dataset

Task A: The Twitter portion of the data of HaSpeeDe 2018 was included in the training set (4,000 tweets posted from October 2016 to April 2017). Moreover, new Twitter data were included for this competition, a subset of the data gathered for the Italian hate speech monitoring project "Contro l'Odio" (Capozzi et al., 2019). The data were retrieved using the Twitter Stream API and filtered using the set of keywords described in Poletto et al. (2017). The newly annotated tweets were posted between September 2018 and May 2019 and were annotated by Figure Eight (now Appen) contributors for hate speech and by the task organizers for the stereotype category. In particular, only data posted between January and May 2019 were included in the test set.

Task B: The HaSpeeDe Twitter corpus – used in the first edition of the task – was already annotated for stereotype since it was part of the Italian Hate Speech corpus described in Sanguinetti et al. (2018). We then used the same guidelines to enrich the new data from "Contro l'Odio" with this annotation layer. The annotation was carried out by the task organizers.

Task C: The HaSpeeDe Twitter corpus was also annotated for the presence of Nominal Utterances (NUs) within a side project (Comandini and Patti, 2019). We used an updated version of its guidelines (available in the task repository) to enrich the new hateful data introduced in the campaign. Similarly to the stereotype level, the annotation of NUs was carried out by the task organizers specifically for this task's purposes.

3.2 News Dataset

Task A: For task A a new test corpus composed of newspapers' headlines about immigrants was made available. The data were retrieved between October 2017 and February 2018 from online newspapers (*La Stampa*, *La Repubblica*, *Il Giornale*, *Liberoquotidiano*) and annotated within the context of a Master's degree thesis discussed in 2018 at the Department of Foreign Languages at the University of Turin. Data annotation includes

---

[5] Task repository: `https://github.com/msang/haspeede/tree/master/2020`.

the same categories annotated in the Twitter corpus.

Task B: The News corpus also includes stereotype annotation, performed according to the same guidelines used for developing the Twitter corpus.

Task C: Similarly to the Twitter dataset, the third annotation level was added in the News corpus from scratch and specifically for the present task.

Tables 1, 2 and 3 show the data distribution for each task.

Task A	HS	NOT HS	TOT.
Train	2766	4073	6839
Test Tweets	622	641	1263
Test News	181	319	500

Table 1: Distribution of Hate Speech labels.

Task B	STER.	NOT STER.	TOT.
Train	3042	3797	6839
Test Tweets	569	694	1263
Test News	175	325	500

Table 2: Distribution of Stereotype labels.

Task C	w/ NUs	w/o NUs	TOT.
Train	1565	1201	2766
Test Tweets	379	243	622
Test News	151	30	181

Table 3: Distribution of Nominal Utterances.

The whole dataset consists of 8,012 tweets and 500 news headlines for Task A and B, and 3,388 tweets and 181 news (i.e., the sub-set with hateful data only) for Task C.

In Task A and B, HS and stereotype represent the 41.8% and 44.6%, respectively, of the Twitter dataset. In contrast, in the News dataset, the portion of hateful content and stereotype lowers to 36% and 35%.

Table 3 shows statistics about the total number of texts with or without NUs in Task C. The percentage of hateful tweets featuring at least one NU is 57.4%; the percentage of news headlines having at least one NU is 83.4%. This distribution is in line with the one found in Comandini and Patti (2019).

3.3 Formats

Task A and B: For both tasks A and B data are provided in a tab-separated values (TSV) file including ID, text, HS and stereotype class (`0` or `1`). Mentions and URLs were replaced with `@user` and `URL` placeholders. Table 4 shows some annotation examples.

Task C: The dataset provided for Task C was annotated using WebAnno and converted into a IOB (Inside-Outside-Beginning) format. The resulting IOB2 alphabet consists of I-NU-CGA, O and B-NU-CGA.

The annotation includes the ID, followed by an hyphen to mark the token number, the token, and the IOB2 annotation of the NUs.

Below an example taken from the training set.

#Text=È UNA PROVOCAZIONE...ORA BASTA.. NESSUNO SBARCHI IN #ITALIA[6]

```
9602-23   È              O
9602-24   UNA            O
9602-25   PROVOCAZIONE   O
9602-26   .              O
9602-27   .              O
9602-28   .              O
9602-29   ORA            B-NU-CGA
9602-30   BASTA          I-NU-CGA
9602-31   .              I-NU-CGA
9602-32   .              I-NU-CGA
9602-33   NESSUNO        O
9602-34   SBARCHI        O
9602-35   IN             O
9602-36   #              O
9602-37   ITALIA         O
```

To prevent participants from cheating, the released test set for Task C also contains non-hateful messages. However, the evaluation of the systems is conducted only on the hateful messages since we are interested in investigating the relationship between these two phenomena.

4 Evaluation

For each task, participants were allowed to submit up to 2 runs. A separate official ranking was provided, and the evaluation was performed according to the standard metrics, i.e, Precision, Recall and F-score.

For Task A and Task B, the scores were computed for each class separately, and finally the F-score was macro-averaged to get the overall results.

---

[6] "IT'S A PROVOCATION...THAT'S ENOUGH...NO LANDINGS IN #ITALY"

id	text	hs	ster.
8783^T	*Via tutti i campi Rom e disinfettare per bene il lerciume che si lasciano dietro.* *Mai più campi Rom in Italia NO NO E NO* ("Away all the Roma camps and clean the filth they leave behind. No more Roma camps in Italy NO NO AND NO")	1	1
9254^T	*Vanno affondate. Hanno rotto i c.....i* *Aquarius vuol dettare ancora legge: carica migranti e rifiuta gli ordini libici* ("They must be sunk. We've had enough Aquarius still wants to lay down the law: it brings migrants on board and refuses Lybian orders")	1	0
9414^T	*Istat conferma: migranti vengono in Italia a farsi mantenere* ("Istat confirms: migrants come to Italy to sponge off (us)")	0	1
10707^N	*Sea Watch, Finanza sequestra la nave: sbarcano i migranti* ("Sea Watch, Custom Corps confiscate the ship: migrants get off")	0	0

Table 4: Examples from the datasets for Task A and B. ^{T} and ^{N} superscripts indicate, respectively, whether the message is from the Twitter or News dataset.

For Task C, token-wise scores were computed, and a NU was considered correct only in case of exact match, i.e., if all tokens that compose it were correctly identified.

Different baseline systems were built according to the task type:

- For Task A and B, besides a typical classifier based on the most frequent class (Baseline_MFC in Tables 5–8), a Linear SVM with TF-IDF of unigrams and 2–5 char-grams was used (Baseline_SVC).
- For Task C, the baseline replicates the one presented for the COSMIANU corpus (Comandini et al., 2018), which identifies as correct in the test the NUs that appear in the training set (memory-based approach); baseline results in Table 9.

5 Task Overview: Participation and Results

5.1 Participants

A total amount of 14 teams participated in the Main task on HS detection, 6 teams also submitted their results for the Pilot task 1 (i.e. Task B) on stereotype detection, while we did not receive any submission for the Pilot task 2 (i.e. Task C) on NUs identification. Except for one case, all teams submitted 2 runs for their tasks. Furthermore, 4 teams used the same systems to participate in other (and partly related) tasks within the EVALITA 2020 campaign: YNU_OXZ and Jigsaw participated in the task on Automatic Misogyny Identification (AMI) (Fersini et al., 2020), while TextWiller and Venses also participated in the task on Stance Detection in Italian Tweets (SardiStance) (Cignarella et al., 2020). It is worth pointing out that in this second edition we registered a higher participation of non-Italian and non-academic teams, and that HaSpeeDe 2 has been one of the most participated EVALITA 2020 tasks.

5.2 Systems Overview

Approaches The participating models are characterized by different architectures that exploit principally BERT-based models and linguistic features. Transformers are a popular choice in this edition. Jigsaw (Lees et al., 2020), Svandiela (Klaus et al., 2020), DH-FBK (Leonardelli et al., 2020), TheNorth (Lavergne et al., 2020) fine-tuned BERT, AlBERTo[7] and UmBERTo[8] language models for both runs. YNU_OXZ (Ou and Li, 2020) exploited the pre-trained XLM-RoBERTa[9] multi-language model as input of Neural Networks architecture. Fontana-Unipi (Fontana and Attardi, 2020) developed a model that is an ensemble of fixed number of instances of two principal transformers (AlBERTo and DBMDZ[10]) and a combination of DBMDZ input and a dense layer. The DBMDZ is used also by By1510 (Deng et al., 2020) in a transfer learning approach. UO team (Rodriguez Cisnero and Ortega Bueno, 2020), on the other hand, used a Bi-LSTM with the addition of linguistic features in

---

[7] `https://github.com/marcopoli/AlBERTo-it`

[8] `https://github.com/musixmatchresearch/umberto`

[9] `https://huggingface.co/transformers/model_doc/xlmroberta.html`

[10] `https://huggingface.co/dbmdz/bert-base-italian-uncased`

the first run, while using the pre-trained DBMDZ model in the second one. CHILab (Gambino and Pirrone, 2020) experimented transformer encoders in the first run and depth-wise Separable Convolution techniques in the second one. Moreover, some teams explored classical machine learning approaches such as No Place For Hate Speech (dos S. R. da Silva and T. Roman, 2020), TextWiller (Ferraccioli et al., 2020), UR_NLP (Hoffmann and Kruschwitz, 2020) and Montanti (Bisconti and Montagnani, 2020). Finally, Venses (Delmonte, 2020), based on the parser for Italian ItGetaruns, applied six different rule-based classifiers.

Features and Lexical Resources Various features are tested and explored by participants. Morphosyntactic features are exploited by CHILab, using Part-of-Speech tags as additional input. To adapt the POS tagging model provided by Python's spaCy library to social media language, they added emoticons, emojis, hashtags and URLs to vocabulary. In addition, to preprocess the texts, they used sentiment lexicon for replacing emoticons with appropriate labels about the expressed sentiment. Semantic and lexical features are exploited by Venses and UO teams. In particular, UO team used WordNet to catch lexical ambiguity, syntactic patterns and similarity among words; calculated information gain to capture the most relevant words; used lexicons such as HurtLex (Bassignana et al., 2018) and SenticNet[11] to feature words with hateful categories and sentiment information. Finally, different types of representation of tweets are tested by Montanti: TF-IDF, DistilBert[12] and GloVe (Pennington et al., 2014) vectors as well as their combination.

Additional data Some teams preferred to use additional data to improve the knowledge of their classifiers. To extend the provided training set, YNU_OXZ exploited Facebook data provided in the first edition of HaSpeeDe and DH-FBK used a set of Italian tweets that covers similar topics. Jigsaw, for one of the submissions, used additional user-generated comments to fine-tune their model. CHILab used additional tweets taken from TWITA 2018[13] by means of some keywords extracted from the provided training set to extend the embedding layer of their model. Finally, the SENTIPOLC 2016 dataset was exploited by UO team.

Interaction between Task A and B Except for TheNorth team, most of the participants did not consider the interaction between Task A and B. Taking into account the possible correlation between texts containing hate speech and texts expressing stereotyped ideas about targets, TheNorth tested the performance of multitasking approach for both tasks (second run) against a fine-tuned UmBERTo model (first run). In particular, observing competition results we can notice the efficacy of multitasking in hate speech identification and not in stereotype detection.

5.3 Results

In Table 5, 6, 7 and 8, we report the official results of HaSpeeDe 2 for Task A and B, ranked by the macro-F1 score. In case of multiple runs, a suffix has been appended to each team name, in order to distinguish the run ID of the submitted file.

Team	**Macro-F1**
TheNorth_2	0.8088
TheNorth_1	0.7897
CHILab_1	0.7893
Fontana-Unipi	0.7803
CHILab_2	0.7782
By1510_1	0.7766
Svandiela_2	0.7756
YNU_OXZ_1	0.7717
Jigsaw_al	0.7681
UR_NLP_2	0.7598
DHFBK_2	0.7534
DHFBK_1	0.7495
No Place For Hate Speech_STT	0.7491
Svandiela_1	0.7452
Montanti_1	0.7432
UR_NLP_1	0.7399
YNU_OXZ_2	0.7345
Montanti_2	0.7279
UO_2	0.7214
Baseline_SVC	**0.7212**
Jigsaw_js	0.717
By1510_2	0.7065
No Place For Hate Speech_LRT	0.7057
UO_1	0.6878
Venses_1	0.5054
Venses_2	0.4726
TextWiller_1	0.3604
Baseline_MFC	**0.3366**
TextWiller_2	0.3317

Table 5: Task A results on Twitter data.

As a general remark, we can observe that the in-domain Main task registered better results (macro-F1=0.8088) both compared to the cross-domain counter-part (0.7744) and the Pilot task 1; in turn,

---

[11] https://www.sentic.net/

[12] https://huggingface.co/transformers/model_doc/distilbert.html

[13] http://twita.di.unito.it/

Team	Macro-F1
CHILab_1	0.7744
UO_2	0.7314
Montanti_1	0.7256
CHILab_2	0.7183
DHFBK_2	0.702
UR_NLP_2	0.6983
YNU_OXZ_2	0.6922
Montanti_2	0.6821
Jigsaw_js	0.6755
DHFBK_1	0.6744
TheNorth_1	0.671
UR_NLP_1	0.6684
UO_1	0.6657
By1510_2	0.6638
YNU_OXZ_1	0.6604
TheNorth_2	0.6602
Fontana-Unipi	0.6546
Jigsaw_al	0.6353
No Place For Hate Speech_STN	0.6328
No Place For Hate Speech_LRN	0.6212
Baseline_SVC	**0.621**
By1510_1	0.6094
Svandiela_2	0.6031
Svandiela_1	0.5265
Venses_1	0.5024
Baseline_MFC	**0.3894**
Venses_2	0.3805
TextWiller_1	0.3101
TextWiller_2	0.2693

Table 6: Task A results on News data.

better results were obtained in the latter with the in-domain data compared to the News set (0.7744 and 0.7203, respectively). The best performances overall provided by the systems used for Task A on Twitter data is also reflected in the average value of the macro-F1 scores of each ranking: 0.6899 for the latter, 0.6306 for Task B on Twitter data, 0.6144 for Task A on News data and 0.5972 for Task B on News data.

We also considered the overall results achieved by all participating teams and observed that, as regards Task A, 12 and 13 teams (in the Twitter and News test set, respectively) obtained higher scores than the SVM-based baseline with at least one of the submitted runs, and 13 teams, on both domains, outperformed the one based on the most frequent class. For Task B, and with respect to the SVM baseline, the same is true for 4 teams out of 6 in the Twitter set and for 3 teams in the News set, while all teams beat the majority-class baseline with at least one run.

Regarding Task C, since the training set is composed of tweets, we first investigated the macro F-score value on a validation set created by splitting the training set in 80%-20%. We then tested the memory-based baseline described in Section 4 on the two test sets released for the task. Table 9 shows the macro-F1 values obtained in the validation set, in the Twitter test set as well as in the News test set. As mentioned earlier, no submissions were made for this task, but the baselines' values for both domains are reported in this overview as reference points for further works.

Team	Macro-F1
TheNorth_1	0.7719
TheNorth_2	0.7676
CHILab_1	0.7615
Jigsaw_al	0.7415
CHILab_2	0.7386
Baseline_SVC	**0.7149**
Montanti_1	0.7076
Montanti_2	0.6889
Jigsaw_js	0.6674
TextWiller_2	0.6031
Venses_1	0.5078
Venses_2	0.4671
Baseline_MFC	**0.3546**
TextWiller_1	0.3369

Table 7: Task B results on Twitter data.

Team	Macro-F1
CHILab_1	0.7203
CHILab_2	0.7184
Montanti_1	0.7166
TheNorth_1	0.6854
Jigsaw_al	0.6811
Montanti_2	0.6706
Baseline_SVC	**0.6688**
TheNorth_2	0.6465
Jigsaw_js	0.6412
TextWiller_2	0.6053
Venses_1	0.5386
Baseline_MFC	**0.3939**
Venses_2	0.3671
TextWiller_1	0.3077

Table 8: Task B results on News data.

Baseline	Macro-F
Baseline_validation	**0.1459**
Baseline_test_Tweets	0.0706
Baseline_test_News	0.0087

Table 9: Task C - Baseline results for Tweets and News.

6 Discussion

A discussion of results, especially those regarding the Main task, necessarily involves a preliminary comparison with the ones obtained in the first edition of HaSpeeDe, in particular in the two tasks where Twitter data were used for training, i.e. HaSpeeDe_TW and Cross-HaSpeeDe_TW.

The best systems attained macro-F1=0.7993 in the former task and 0.6985 in the latter. While these results are in line with those reported for Task A on the in-domain data, the results obtained in this edition on News data are better than the part cross-domain task, where the test set was made up of Facebook comments. We hypothesize that the homogeneity of hate target in News and Twitter corpora (immigrants) has meant more than the similar linguistic features in Twitter and Facebook data, stemming from the fact that they are both social media texts.

Participants achieved promising results in the detection of stereotypes, a new pilot task proposed at HaSpeeDe this year for the first time. In our view, stereotype and HS are meant as orthogonal dimensions of abusive language, which do not necessarily coexist. This influenced the design of HaSpeeDe 2, where we proposed two independent tasks for the detection of such categories. However, a first analysis of systems participating in both tasks suggests that most teams did not design a dedicated system for stereotype recognition, but focused on developing a HS detection model, adapting the same model to stereotype recognition, reducing *de facto* stereotypes to characteristics of HS. We hypothesize that this could be one of the factors that led the systems to not generalize well when applied to the stereotype detection task, especially in texts that are not hateful but contain stereotypes. This hypothesis is confirmed by the high percentage of false negatives (21% in tweets and 35% in news headlines) of the stereotype class in non-hateful texts, with respect to false negatives (5% in tweets and 28% in news headlines) in hateful ones. It is possible to notice the same increase also in false positives in hateful texts. These values suggest that stereotype appears as a more subtle phenomenon that could not give rise to hurtful message. The percentages have been computed taking into account the set of common incorrect predictions of the three best runs in Task B, and calculated in relation to the actual distribution of HS and stereotype in the test set. Analyzing the predictions of the three best runs in Task A, similar influence of stereotype is observed in false negative and positive, but to a minor extent. These results are in line with the observations about emerged from the error analysis of HaSpeeDe 2018 (Francesconi et al., 2019).

To conclude the discussion on this edition's results, we comment on the baseline scores obtained for Task C. As it can be noticed from Table 9, the value obtained on the validation set is higher than the ones obtained on both test sets. This variation can be explained by the main characteristics of the data at hand: on the Twitter side, this is due to the different time frames of tweet's publication included in training and test set, while on the News side, such low value is expected by virtue of the different text domain. Since this baseline uses a memory-based approach, such a low performance is to be expected in datasets from different time frames, since the discussion topics are different and Twitter users change their hashtags and slogans, which are the main repeated items.

7 Conclusions

In its second edition, the HaSpeeDe task proposed the detection of hateful content in Italian, by challenging systems along two dimensions, time and domain, and taking into account also the category of stereotype, which often co-occurs with HS. This paves the way for further investigations also about the relationships linking stereotype and HS.

In order to take a step further in state-of-the-art HS detection, the task provided novel benchmarks for exploring different facets of the phenomenon and laying the foundations for deeper studies about the impact of bias, topic and text domain. In this line, also a pilot task about recognition of NUs was proposed, devoted to study this kind of linguistic form in hateful messages in tweets and newspaper headlines, as it has been proved that both headlines in journalistic writings (Mortara Garavelli, 1971) and social media texts (Ferrari, 2011; Comandini et al., 2018) are a fertile ground for NUs. Even though we did not receive any submission for Pilot task 2, our hope is that the fine-grained annotation of hateful data concerning these aspects can be the subject of deeper studies to shed light on the syntax of hate, a topic still understudied.

Acknowledgments

The work of Cristina Bosco, Simona Frenda, Viviana Patti and Marco Stranisci is partially funded by Progetto di Ateneo/CSP 2016 (Immigrants, Hate and Prejudice in Social Media, S1618.L2.BOSC.01) and by the project "Be Positive!" (under the 2019 "Google.org Impact Challenge on Safety" call).

References

Natalie Alkiviadou. 2019. Hate speech on social media networks: towards a regulatory framework? *Information & Communications Technology Law*, 28(1):19–35.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Rangel, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 Task 5: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter. In *Proceedings of SemEval 2019*.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online.

Elisa Bassignana, Valerio Basile, and Viviana Patti. 2018. Hurtlex: A Multilingual Lexicon of Words to Hurt. In *Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018)*.

Elia Bisconti and Matteo Montagnani. 2020. Montanti @ HaSpeeDe2 EVALITA 2020: Hate Speech Detection in online contents. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Cristina Bosco, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. 2018. Overview of the EVALITA 2018 hate speech detection task. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*.

Arthur TE Capozzi, Mirko Lai, Valerio Basile, Fabio Poletto, Manuela Sanguinetti, Cristina Bosco, Viviana Patti, Giancarlo Ruffo, Cataldo Musto, Marco Polignano, Giovanni Semeraro, and Marco Stranisci. 2019. Computational linguistics against hate: Hate speech detection and visualization on social media in the "Contro L'Odio" project. In *Proceedings of the Sixth Italian Conference on Computational Linguistics, CLiC-it 2019*.

Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso. 2018. EVALITA 2018: Overview of the 6th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*.

Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. SardiStance@EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Gloria Comandini and Viviana Patti. 2019. An Impossible Dialogue! Nominal Utterances and Populist Rhetoric in an Italian Twitter Corpus of Hate Speech against Immigrants. In *Proceedings of the Third Workshop on Abusive Language Online*.

Gloria Comandini, Manuela Speranza, and Bernardo Magnini. 2018. Effective Communication without Verbs? Sure! Identification of Nominal Utterances in Italian Social Media Texts. In *Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018)*, volume 2253. CEUR-WS.org.

Rodolfo Delmonte. 2020. Venses @ HaSpeeDe2 & SardiStance: Multilevel Deep Linguistically Based Supervised Approach to Classification. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Tao Deng, Yang Bai, and Hongbing Dai. 2020. By1510 @ HaSpeeDe 2: Identification of Hate Speech for Italian Language in Social Media Data. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Adriano dos S. R. da Silva and Norton T. Roman. 2020. No Place For Hate Speech @ HaSpeeDe 2: Ensemble to identify hate speech in Italian. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Federico Ferraccioli, Andrea Sciandra, Mattia Da Pont, Paolo Girardi, Dario Solari, and Livio Finos. 2020. TextWiller @ SardiStance, HaSpeede2: Text or Con-text? A smart use of social network data in predicting polarization. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Angela Ferrari. 2011. Enunciati nominali. *Enciclopedia dell'Italiano*. `http://www.treccani.it/enciclopedia/enunciati-nominali_(Enciclopedia_dell'Italiano)/`.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2020. AMI @ EVALITA2020: Automatic Misogyny Identification. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online.

Michele Fontana and Giuseppe Attardi. 2020. Fontana-Unipi @ HaSpeeDe2: Ensemble of transformers for the Hate Speech task at Evalita. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Paula Fortuna, João Rocha da Silva, Juan Soler-Company, Leo Wanner, and Sérgio Nunes. 2019. A Hierarchically-Labeled Portuguese Hate Speech Dataset. In *Proceedings of the Third Workshop on Abusive Language Online*.

Chiara Francesconi, Cristina Bosco, Fabio Poletto, and Manuela Sanguinetti. 2019. Error Analysis in a Hate Speech Detection Task: The case of HaSpeeDe-TW at EVALITA 2018. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*.

Giuseppe Gambino and Roberto Pirrone. 2020. CHILab @ HaSpeeDe 2: Enhancing Hate Speech Detection with Part-of-Speech Tagging. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Julia Hoffmann and Udo Kruschwitz. 2020. UR_NLP @ HaSpeeDe 2 at EVALITA 2020: Towards Robust Hate Speech Detection with Contextual Embeddings. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Svea Klaus, Anna-Sophie Bartle, and Daniela Rossmann. 2020. Svandiela @ HaSpeeDe: Detecting Hate Speech in Italian Twitter Data with BERT. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Eric Lavergne, Rajkumar Saini, György Kovács, and Killian Murphy. 2020. TheNorth @ HaSpeeDe 2: BERT-based Language Model Fine-tuning for Italian Hate Speech Detection. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Alyssa Lees, Jeffrey Sorensen, and Ian Kivlichan. 2020. Jigsaw @ AMI and HaSpeeDe2: Fine-Tuning a Pre-Trained Comment-Domain BERT Model. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Elisa Leonardelli, Stefano Menini, and Sara Tonelli. 2020. DH-FBK @ HaSpeeDe2: Italian Hate Speech Detection via Self-Training and Oversampling. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Thomas Mandl, Sandip Modha, Prasenjit Majumder, Daksh Patel, Mohana Dave, Chintak Mandlia, and Aditya Patel. 2019. Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in Indo-European languages. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*.

Bice Mortara Garavelli. 1971. Fra norma e invenzione: lo stile nominale. In Accademia della Crusca, editor, *Studi di grammatica italiana*, volume 1, pages 271–315. G. C. Sansoni Editore, Firenze, Italia.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive Language Detection in Online User Content. In *Proceedings of the 25th International Conference on World Wide Web (WWW'16)*.

Xiaozhi Ou and Hongling Li. 2020. YNU_OXZ @ HaSpeeDe 2 and AMI : XLM-RoBERTa with Ordered Neurons LSTM for classification task at EVALITA 2020. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Fabio Poletto, Marco Stranisci, Manuela Sanguinetti, Viviana Patti, and Cristina Bosco. 2017. Hate Speech Annotation: Analysis of an Italian Twitter Corpus. In *Proceedings of the Fourth Italian Conference on Computational Linguistics (CLiC-it 2017)*.

Fabio Poletto, Valerio Basile, Manuela Sanguinetti, Cristina Bosco, and Viviana Patti. 2020. Resources and Benchmark Corpora for Hate Speech Detection: a Systematic Review. *Language Resources and Evaluation*.

Michal Ptaszynski, Agata Pieciukiewicz, and Paweł Dybała. 2019. Results of the PolEval 2019 Shared Task 6: First Dataset and Open Shared Task for Automatic Cyberbullying Detection in Polish Twitter. In *Proceedings of the PolEval 2019 Workshop*.

Mariano Jason Rodriguez Cisnero and Reynier Ortega Bueno. 2020. UO@HaSpeeDe2: Ensemble Model for Italian Hate Speech Detection. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*.

Manuela Sanguinetti, Fabio Poletto, Cristina Bosco, Viviana Patti, and Marco Stranisci. 2018. An Italian Twitter Corpus of Hate Speech against Immigrants. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC'18)*.

Xuan-Son Vu, Thanh Vu, Mai-Vu Tran, Thanh Le-Cong, and Huyen T M Nguyen. 2019. HSD Shared Task in VLSP Campaign 2019: Hate Speech Detection for Social Good. In *Proceedings of VLSP 2019*.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: A typology of abusive language detection subtasks. In *Proceedings of the First Workshop on Abusive Language Online*, pages 78–84. Association for Computational Linguistics.

Ruth E. Wodak. 2018. Introductory remarks from 'hate speech' to 'hate tweets'. In Mojca Pajnik and Birgit Sauer, editors, *Populism and the web: communicative practices of parties and movements in Europe*, pages xvii–xxiii. Rourledge.

YNU_OXZ @ HaSpeeDe 2 and AMI : XLM-RoBERTa with Ordered Neurons LSTM for classification task at EVALITA 2020

Xiaozhi Ou
Yunnan University
China
`xiaozhiou88@gmail.com`

Hongling Li✉
Yunnan University
China
`honglingli66@126.com`

Abstract

English. This paper describes the system that team YNU_OXZ submitted for EVALITA 2020. We participate in the shared task on Automatic Misogyny Identification (AMI) and Hate Speech Detection (HaSpeeDe 2) at the 7th evaluation campaign EVALITA 2020. For HaSpeeDe 2, we participate in Task A - Hate Speech Detection and submitted two-run results for the news headline test and tweets headline test, respectively. Our submitted run is based on the pre-trained multi-language model XLM-RoBERTa, and input into Convolution Neural Network and K-max Pooling (CNN + K-max Pooling). Then, an Ordered Neurons LSTM (ON-LSTM) is added to the previous representation and submitted to a linear decision function. Regarding the AMI shared task for the automatic identification of misogynous content in the Italian language. We participate in subtask A about Misogyny & Aggressive Behaviour Identification. Our system is similar to the one defined for HaSpeeDe and is based on the pre-trained multi-language model XLM-RoBERTa, an Ordered Neurons LSTM (ON-LSTM), a Capsule Network, and a final classifier.

1 Introduction and Background

People use offensive contents in their social media posts to degrade an individual or religion or other organizations in many respects, the identification of such social media posts is a necessity, a substantial amount of work has been done in languages like English. However, hate speech and offensive language identification in other language scenario is still an area worth exploring. The latest edition of EVALITA (Caselli et al., 2018) hosted the first Hate Speech (HS) detection in Social Media (i.e. HaSpeeDe (Bosco et al., 2018)) task for Italian, the HaSpeeDe 2 (Hate Speech Detection) (Sanguinetti et al., 2020) shared task have been organized within Evalita 2020 [1]. The ultimate goal of HaSpeeDe 2 is to take a step further in the state of the art of HS detection for Italian while also exploring other side phenomena, the extent to which they can be distinguished from HS, and finally whether and how much automatic systems are able to draw such conclusions. For AMI (Elisabetta Fersini, 2020), the second shared task at the 7th evaluation campaign EVALITA 2020 (Basile et al., 2020). Given the huge amount of user-generated content on the Web, and in particular on social media, the problem of detecting, in order to possibly limit the diffusion of hate speech against women, is rapidly becoming fundamental especially for the societal impact of the phenomenon, it is very important to identify misogyny in social media.

1.1 Hate Speech (HaSpeeDe 2)

In recent years, with the acceleration of information dissemination, the identification of hate speech and offense language has become a crucial mission in multilingual sentiment analysis fields and has attracted the attention of a large number of industrial and academic researchers. From an NLP perspective, much attention has been paid to the topic of HS - together with all its possible facets and related phenomena, such as offensive/abusive language, and its identification. This is shown by the proliferation, especially in the last few years, of contributions on this topic (e.g.

---

[1] http://www.evalita.it/2020/tasks

Caselli et al. (2020), Jurgens et al. (2019), Fortuna et al. (2019)), corpora and lexica (e.g. de Pelle and Moreira (2017), (Sanguinetti et al., 2018), (Bassignana et al., 2018)), dedicated workshops, and shared tasks within national (GermEval [2], HASOC [3], IberLEF [4]) and international (SemEval [5]) evaluation campaigns. Among them, Gemeval2018 is about offensive language recognition and aims to promote research on offensive contents recognition in German language microblogs. The best teams system is to train three basic classifiers (maximum entropy and two random forest sets) using five disjoint feature sets and then used the maximum entropy element-level classifier for final classification (Montani and Schüller, 2018). In the SemEval-2019 shared tasks HatEval and OffensEval, HatEval is a multilingual detection of hate speech against immigrants and women on Twitter. Fermi team is the best team of Hateval. It proposes an SVM model with the RBF kernel and uses sentence embedding in Google general sentence encoder as a function (Indurthi et al., 2019). OffensEval is about the identification and classification of offensive language in social media. The NULI team is the best performing team, they use BERT-base without default parameters (Liu et al., 2019). HASOC2019 is proposed to identify hate speech and offensive content in Indo-European languages. Its purpose is to develop powerful technologies capable of processing multilingual data and to develop a transfer learning method that can utilize cross-lingual data. The optimal system is a system based on ordered neuron LSTM (ON-LSTM) and attention model and adopts the K-folding approach for ensemble (Wang et al., 2019).

1.2 Misogyny (AMI)

Unfortunately, nowadays more and more incidents of harassment against women have appeared and misogynistic comments have been found in social media, where misogynists hide behind by anonymity security. Therefore, it is very important to identify misogyny in social media. Pamungkas et al. (2020) conducted extensive and in-depth research on online misogyny, developed a state-of-the-art model for detecting misogyny in social media and explored the feasibility of detecting misogyny in a multilingual environment. Aiming at the TRAC-2 shared tasks of Aggression Identification and Misogynistic Aggression Identification, Samghabadi et al. (2020) propose an end-to-end neural model using attention on top of BERT that incorporates a multi-task learning paradigm to address both the sub-tasks simultaneously. Arango et al. (2019) discussed the implications for current research and re-conduct experiments, a closer look at model validation to give a more accurate picture of the current state-of-the-art methods. Recent investigations studied how the misogyny phenomenon takes place, such as Farrell et al. (2019) study this phenomenon by investigating the flow of extreme language across seven online communities on Reddit. Goenaga et al. (2018) automatic misogyny identification using neural networks. Automatic misogyny identification in Twitter has been firstly investigated by Anzovino et al. (2018).

2 Task and Data description

2.1 Task description

In this part, we describe one of the subtasks HaSpeeDe 2 participating in EVALITA 2020. This task introduces its novelty from three main aspects (Language variety and test of time, Stereotypical communication, Syntactic realization of HS). We participated in Task A - Hate Speech Detection (Main Task), a binary classification task aimed at determining the presence or the absence of hateful content in the text towards a given target (among Immigrants, Muslims or Roma people).

The AMI shared task proposes that misogynous content in Italian is automatic identification in Twitter. It is organized according to two main subtasks, namely subtask A - Misogyny & Aggressive Behaviour Identification and subtask B - Unbiased Misogyny Identification. We participate in subtask A, the system must recognize whether the text is misogyny, and if it is misogyny, it must also recognize whether it expresses an aggressive attitude.

2.2 Data description

HaSpeeDe 2 task organizer provides a new **HS training dataset** (binary task) based on Twitter data, accompanied by a test set including both in-domain and out-of-domain data (tweets + news headlines), as well as from different time periods. The HaSpeeDe 2020 new training set already contains the Twitter dataset of HaSpeeDe 2018. The

---

[2] https://projects.fzai.h-da.de/iggsa/germeval/

[3] https://hasocfire.github.io/hasoc/2020

[4] http://hitz.eus/sepln2019/

[5] http://alt.qcri.org/semeval2020/

new dataset contains a total of 6,839 tweets (label 0 means **NOT HS**, label 1 means **HS**), of which **HS** contains 2,766, **NOT HS** contains 4,703, the tweets headlines test set contains 1,263 tweets, and the news headlines test set contains 500 elements. In the experimental run, the data we recommend for this task is the result of combining the Facebook dataset (training set + test set) of HaSpeeDe 2018 with the new training set of HaSpeeDe 2020, this is to analyze the influence of out-of-domain texts in the training set. The two contain a total of 10,839 comments/tweets.

The AMI organizer provided a **raw dataset** (5,000 tweets) as the training set for participants in subtask A, the **raw dataset** is a balanced dataset of tweets manually labeled according to two levels:

- Misogynous: defines if a tweet is misogynous or not misogynous. Label 0 means **Not misogynous** tweet, label 1 means **Misogynous** tweet.
- Aggressiveness: denotes the subject of the misogynistic tweet (misogynous tweet is label 1). Label 0 means **Non-aggressive** tweet, label 1 means **Aggressive** tweet. **Not misogynous** tweet (misogynous tweet is label 0) are labeled as 0 by default.

For the test set (1,000 tweets) for subtask A provided by the AMI organizer, only the annotations on the "misogynous" and "aggressiveness" fields in the **raw dataset** will consider.

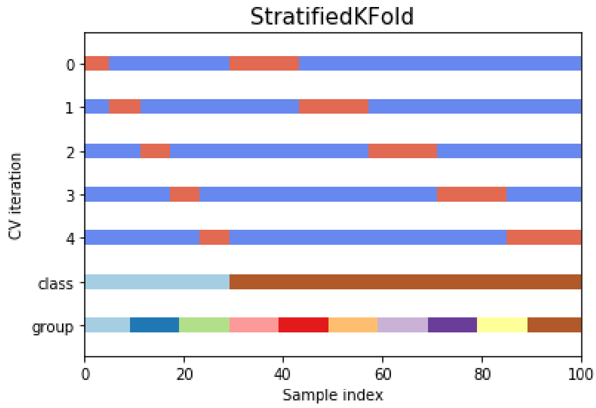


Figure 1: 5-fold stratified sampling to the training set

As shown in Figure 1, we use stratified sampling technology (StratifiedKFold), using StratifiedKFold cross-validation instead of ordinary k-fold cross-validation to evaluate a classifier. The reason is that StratifiedKFold can utilize stratified sampling to divide, which can ensure that the proportion of each category in the generated training set and validation set is consistent with the original training set so that the generated data distribution disorder will not occur. In the experiment, we used 5-fold stratified sampling. For the HaSpeeDe 2 training set (**Merged dataset**), each of which included the randomly sampled training set (8,671) and validation set (2,168). For the AMI training set (**raw dataset**), each of which included the randomly sampled training set (4,000) and validation set (1,000).

3 Description of the system

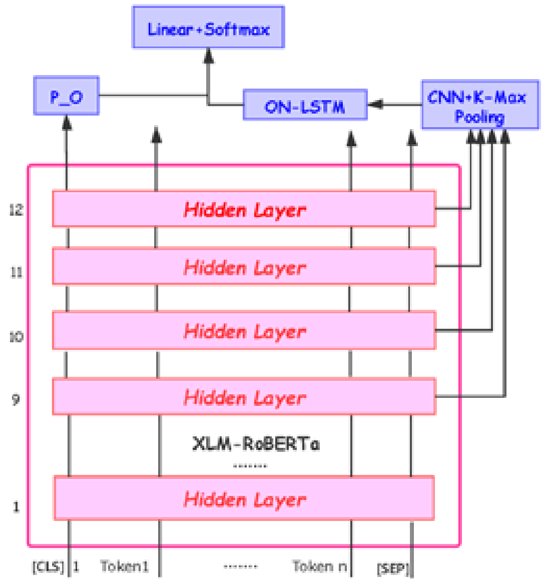


Figure 2: System architecture diagram for Task A (HaSpeeDe 2)

In this part, we introduce our final submission system. Figure 2 shows the overall framework of the system we submitted to HaSpeeDe 2 Task A. We use the pre-trained multi-language model XLM-RoBERTa. We discover the limitations of BERT's pooler output (P_O) and obtained rich semantic information by extracting the hidden state (The last four hidden layers) of XLM-RoBERTa, which is used as input for Convolution Neural Network and K-max Pooling (CNN + K-max Pooling). Then, we input the output of (CNN + K-max Pooling) into the Ordered Neurons LSTM (ON-LSTM). Finally, we concatenate the P_O and output of ON-LSTM ON-LSTM together and pass it through the Linear layer and Softmax for the final classification.

Figure 3 shows the overall framework of the system we submitted to AMI subtask A. We use the pre-trained multi-language model XLM-RoBERTa. We first get pooler output (P_O) and obtained rich semantic information by extracting the hidden state (The last four hidden layers) of XLM-RoBERTa, which is input into Ordered Neurons LSTM (ON-LSTM). Then, we input the output of ON-LSTM into Capsule Network.Finally, we concatenate the P_O and output of Capsule togetherand through the Linear layer and Softmax for the final classification.

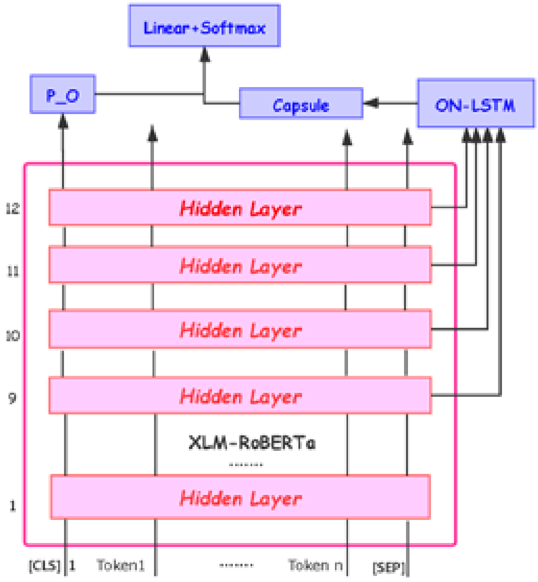


Figure 3: System architecture diagram for subtask A (AMI)

3.1 XLM-RoBERTa and hidden layer state

Early work in the field of cross-language understanding has proved the effectiveness of multilingual masked language model (MLM) in cross-language understanding, but models such as XLM (Lample and Conneau, 2019) and Multilingual BERT (Devlin et al., 2018) (pre-trained on Wikipedia) are still limited in learning useful representations of low resource languages. XLM-RoBERTa (Conneau et al., 2020) shows that the performance of cross-language transfer tasks can be significantly improved by using the large-scale multi-language pre-training model. It can be understood as a combination of XLM and RoBERTa. It is trained on 2.5 TB of newly created clean CommonCrawl data in 100 languages. Because the training of the model in this task must make full use of the whole sentence content to extract useful semantic features, which may help to deepen the understanding of the sentence and reduce the impact of noise on speech. Therefore, we use XLM-RoBERTa in this work.

In the classification task, the original output of XLM-RoBERTa is obtained through the last hidden state of the model. However, the output usually does not summarize the semantic content of the input. Recent studies have shown that abundant semantic information features are learned by the top hidden layer of BERT (Jawahar et al., 2019), which we call the semantic layer. In our opinion, the same is true of XLM-RoBERTa. Therefore, in order to make the model obtain more abundant semantic information features, we propose the system as shown in Figure 2 for HaSpeeDe 2 Task A. Firstly, we get P_O. Secondly, we extract the hidden state of the last four layers of XLM-RoBERTa and input them into CNN and K-max Pooling. Then, input into ON-LSTM. For AMI subtask A, we propose the system as shown in Figure 3. Firstly, we get P_O. Secondly, we extract the hidden state of the last four layers of XLM-RoBERTa and input them into ON-LSTM. Then, input into Capsule.

3.2 CNN and K-max Pooling

As shown in Figure 2, we input the extracted hidden states of the last four layers of XLM-RoBERTa into CNN and K-max Pooling for convolution operations to obtain multiple feature maps. The specific operation: a sentence contains L words, each of which has a dimension of d after the embedding layer, and the representation of the sentence is formed by splicing the L words to form a matrix of $L * d$. There are several convolution kernels in the convolutional layer, the size of which is $N * d$, and N is the filter window size. The convolution operation is to apply a convolution kernel to create a new feature in a matrix that is spliced by words. Its formula is as follows:

$$C_l = f(w * x(l : L + N - 1) + b) \qquad (1)$$

where l represents the lth word, C_l is the feature, w is the convolution kernel, b is the bias term, and f is a nonlinear function. After the convolution operation of the whole sentence, a feature map is obtained, which is a vector of size $L + N - 1$.

Another important idea of CNN is pooling. The pooling layer is usually connected behind the convolution layer. The purpose of introducing it is to simplify the output of the convolutional layer and perform dimensionality reduction on the features of the Filter to form the final feature. Here is the K-max Pooling operation, which takes the value of the scores in Top K among all the feature values, and retains the original order of these feature values, that is, by retaining some feature information for subsequent use. Obviously, K-max Pooling can express the same type of feature multiple times, that is, can express the intensity of a certain type of feature; in addition, because the relative order of these Top K eigenvalues is preserved, it should be said that it retains part of the position information. However, this location information is only the relative order between features, not absolute location information.

3.3 Ordered Neurons LSTM

For HaSpeeDe 2, as shown in Figure 2, we input the output of CNN and K-max pooling into ON-LSTM. For AMI, as shown in Figure 3, We input the extracted hidden states of the last four layers of XLM-RoBERTa into ON-LSTM. ON-LSTM is a new variant of LSTM, which sorts the neurons in a specific order, allowing the hierarchical structure (tree structure) to be integrated into the LSTM to express richer information. The gate structure and output structure of ON-LSTM are still similar to the original LSTM. The difference is that the update mechanism from $\widehat{c_t}$ to c_t is different. The formula is as follows (Shen et al., 2018):

$$\widetilde{f_t} = \overrightarrow{cs}(softmax(W_{\widetilde{f}}x_t + U_{\widetilde{f}}h_{t-1} + b_{\widetilde{f}}) \tag{2}$$

$$\widetilde{i_t} = \overleftarrow{cs}(softmax(W_{\widetilde{i}}x_t + U_{\widetilde{i}}h_{t-1} + b_{\widetilde{i}}) \tag{3}$$

$$w_t = \widetilde{f_t} \circ \widetilde{i_t} \tag{4}$$

$$\begin{aligned} c_t = & w_t \circ (f_t \circ c_{t-1} + i_t \circ \widehat{c_t}) + (\widetilde{f_t} - w_t) \\ & \circ c_{t-1} + (\widetilde{i_t} - w_t) \circ \widehat{c_t} \end{aligned} \tag{5}$$

Among them, $\overrightarrow{cs}$ and $\overleftarrow{cs}$ are cumsum() operations in the right and left directions, respectively. the newly introduced $\widetilde{f_t}$ and $\widetilde{i_t}$ represent the master forget gate and master input gate respectively. w_t represents a vector where the intersection part is 1 and the rest is all 0. In this way, the high-level information remains a considerable long distance, while the low-level information may be updated at each step of input, thereby embedding the hierarchical structure through information grading.

3.4 Capsule Network

As shown in Figure 3, we input the output of ON-LSTM into Capsule. In the deep learning model, spatial patterns are aggregated at a lower level, which helps to represent higher-level concepts. We use the Capsule Network (Sabour et al., 2017) to enhance the models feature extraction capabilities, spatial insensitivity methods are inevitably limited by the abundant text structure (such as saving the location of words, semantic information, grammatical structure, etc.), difficult to effectively encode, and lack of text expression ability. The Capsule network effectively improved this disadvantage by using neuron vectors instead of individual neuron nodes of traditional neural networks to train this new neural network in the dynamic routing way. The Capsule's parameter update algorithm is routing-by-agreement, a lower-level capsule prefers to send its output to higher-level capsule whose activity vectors have a big scalar product with the prediction coming from the lower-level capsule. The calculation formula of the Capsule is as follows:

$$V_j = \frac{\parallel S_j \parallel^2}{1 + \parallel S_j \parallel^2} \frac{S_j}{\parallel S_j \parallel} \tag{6}$$

$$S_j = \sum_i C_{ij}\hat{u}_{j|i}, \qquad \hat{u}_{j|i} = W_{ij}u_i \tag{7}$$

where V_j is the vector output of capsule j and S_j is its total input, prediction vectors $\hat{u}_{j|i}$ is by multiplying the output u_i of a capsule in the layer below by a weight matrix W_{ij}, the C_{ij} are coupling coefficients that are determined by the iterative dynamic routing process.

The most fundamental difference between the Capsule network and the traditional artificial neural network lies in the unit structure of the network. For traditional neural networks, the calculation of neurons can be divided into the following three steps: 1. Perform a scalar weighted calculation on the input. 2. Sum the weighted input scalars. 3. Nonlinearization from scalar to the scalar. For the Capsule, its calculation is divided into the following four steps: 1. Do matrix multiplication on the input vector. 2. Scalar weighting of the input vector. 3. Sum the weighted vector. 4. Vector-to-vector nonlinearization. The biggest difference between the Capsule network and the traditional neural network is the unit output. The output of the traditional neural network is a value, while the output of the Capsule network is a vector, which can contain abundant features and is more interpretable.

3.5 Experiment setting

For the XLM-RoBERTa, we use XLM-RoBERTa-base[6] pre-trained model, which contains 12 layers. We use Binary cross-entropy, Adam optimizer with a learning rate of 5e-5. The batch size is set to 32 and the max sequence length is set to 80. We extract the hidden layer state of XLM-RoBERTa by setting the output_hidden_States is true. The model is trained in 8 epochs with a dropout rate of 0.1.

For the Convolution Neural Network,we use 2D convolution (nn.Conv2d[7]). The size of the convolution kernel is set to (3,4,5) and the number of convolution kernels is set to 256.

For the ON-LSTM, we set the hidden units to 128 and num levels to 16.

For the Capsule Network, we set num capsule to 10, dim capsule to 16, routings to 4.

4 Results and Discussion

Task	Our Score	Best Score	Rank
HaSpeeDe	Macro F1		
Tweets	0.7717	0.8088	8
News	0.6922	0.7744	7
AMI	Average F1		
subtask A	0.7313	0.7406	3

Table 1: Classification results of our best runs on the HaSpeeDe 2 Task A and AMI subtask A.

Table 1 reports the official results of the best runs on the two tasks we participate in. For these two tasks, we submitted the results of two runs, and the results of both runs were ideal and equally matched. In the following subsections, the results obtained in each task will be discussed.

4.1 HaSpeeDe 2 Task A

XLM-RoBERTa with only P_O in News The validation set of 1-fold				
Category	**P**	**R**	**F1**	**Instances**
Not Hate	0.70	0.981	0.817	1355
Hate	0.886	0.259	0.401	813
Macro F1	0.793	0.62	0.609	2168
XLM-RoBERTa with only P_O in Tweets The validation set of 1-fold				
Category	**P**	**R**	**F1**	**Instances**
Not Hate	0.805	0.569	0.667	1355
Hate	0.659	0.858	0.745	813
Macro F1	0.723	0.713	0.706	2168

Table 2: Precision, Recall, F1 score and Instances for XLM-RoBERTa with only P_O in HaSpeeDe 2 Task A (The validation set is the first fold in the 5-fold stratified cross-validation)

The number of different hidden layers of XLM-RoBERTa (The validation set of 1-fold)		
Systems Hidden layers	**HS-News** Macro F1	**HS-Tweets** Macro F1
The last layers	0.623	0.725
The last two layers	0.646	0.734
The last three layers	0.66	0.749
The last four layers	0.703	0.798

Table 3: The performance of our model at different hidden layers (The validation set is the first fold in the 5-fold stratified cross-validation)

In our experiment, we find the limitations of P_O for sentiment analysis of hate text in Italian languages. In the classification task, the original output of BERT is P_O. In the same way, we just put P_O as the output of XLM-RoBERTa.The results are shown in Table 2. We can see that the results are not good when only P_O is used as the output of XLM-RoBERTa. We think that just using P_O as the output will lose some effective semantic information. So we think that deep and abundant semantic features are effective for this work. We extract the hidden state of XLM-RoBERTa and we also discover that the performance of the model improves with the increase of the semantic layer. Table 3 shows the performance of our model at different semantic layers. Table 4 shows our results on the test set.

4.2 AMI subtask A

In this work, we have similar tasks as discussed in Section 4.1, and we consider the influence of P_O for identifying misogyny content. We conduct experiments on the AMI subtask A base on the mod-

---

[6] https://huggingface.co/xlm-roberta-base

[7] https://pytorch.org/docs/stable/generated/torch.nn.Conv2d

The last four hidden states of XLM-RoBERTa				
News	**P**	**R**	**F1**	**Macro F1**
Not Hate	0.7486	0.8965	0.8159	**0.6922**
Hate	0.7203	0.4696	0.5685	
Tweets	**P**	**R**	**F1**	**Macro F1**
Not Hate	0.8037	0.7285	0.7643	**0.7717**
Hate	0.7448	0.8167	0.7791	

Table 4: Results of Macro F1 on Test set

el in HaSpeeDe 2, and in order to improve the performance, we propose a new method base on this model. Table 5 shows the comparative experimental data of the CNN + K-max Pooling + ON-LSTM method and the ON-LSTM + Capsule method. Table 6 shows the results of our new model for AMI subtask A on the test set. Run 1 only extracts the last four hidden layer states of XLM-RoBERTa and inputs them into ON-LSTM, then through the Capsule Network, and finally performs classification (without using P_O). Run 2 is to concatenate the output of the Capsule Network with the obtained P_O and input it to the classifier for final classification (using P_O). We think that concatenate the P_O and the hidden layer will retain richer semantic information and show excellent results.

Base on XLM-RoBERTa model (The validation set of 1-fold)	
Method	**Macro F1**
CNN + K-max Pooling + ON-LSTM (HaSpeeDe 2 Model)	0.786
ON-LSTM + Capsule (AMI model)	0.857

Table 5: Comparison of experimental data between CNN + K-max Pooling method and ON-LSTM + Capsule method on the validation set. (The validation set is the first fold in the 5-fold stratified cross-validation)

System	Average F1
Run 1 (without using P_O)	0.7014
Run 2 (using P_O)	0.7313

Table 6: The results on the test set for AMI subtask A

5 Conclusion

In the experiment, we find the limitation of only using pooler output as the XLM-RoBERTa's output. To obtain deeper and more abundant semantic features, we extract the hidden layer state of XLM-RoBERTa. The result shows that it is helpful to improve the performance of XLM-RoBERTa to obtain more abundant semantic information features by extracting the hidden state of XLM-RoBERTa. We test the effects of using the external dataset (**Merged dataset**) and not using the external dataset (**raw dataset**). Our conclusion is that using data from the same social network for training and test is a necessary condition for good performance. In addition, adding data from different social networks can improve results.

References

Maria Anzovino, Elisabetta Fersini, and Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. In *International Conference on Applications of Natural Language to Information Systems*, pages 57–64. Springer.

Aymé Arango, Jorge Pérez, and Barbara Poblete. 2019. Hate speech detection is not as easy as you may think: A closer look at model validation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 45–54.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Elisa Bassignana, Valerio Basile, and Viviana Patti. 2018. Hurtlex: A multilingual lexicon of words to hurt. In *5th Italian Conference on Computational Linguistics, CLiC-it 2018*, volume 2253, pages 1–6. CEUR-WS.

Cristina Bosco, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. 2018. Overview of the evalita 2018 hate speech detection task. In *EVALITA 2018-Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*, volume 2263, pages 1–9. CEUR.

Tomasso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso. 2018. Sixth evaluation campaign of natural language processing and speech tools for italian: Final workshop (evalita 2018). In *EVALITA 2018*. CEUR Workshop Proceedings (CEUR-WS. org).

Tommaso Caselli, Valerio Basile, Jelena Mitrović, Inga Kartoziya, and Michael Granitzer. 2020. I feel offended, dont be abusive! implicit/explicit messages in offensive and abusive language. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 6193–6202.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Rogers Prates de Pelle and Viviane P Moreira. 2017. Offensive comments in the brazilian web: a dataset and baseline results. In *Anais do VI Brazilian Workshop on Social Network Analysis and Mining*. SBC.

Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Paolo Rosso Elisabetta Fersini, Debora Nozza. 2020. Overview of the evalita 2020 automatic misogyny identification (ami) task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Tracie Farrell, Miriam Fernandez, Jakub Novotny, and Harith Alani. 2019. Exploring misogyny across the manosphere in reddit. In *Proceedings of the 10th ACM Conference on Web Science*, pages 87–96.

Paula Fortuna, Joao Rocha da Silva, Leo Wanner, Sérgio Nunes, et al. 2019. A hierarchically-labeled portuguese hate speech dataset. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 94–104.

Iakes Goenaga, Aitziber Atutxa, Koldo Gojenola, Arantza Casillas, Arantza Díaz de Ilarraza, Nerea Ezeiza, Maite Oronoz, Alicia Pérez, and Olatz Perez-de Viñaspre. 2018. Automatic misogyny identification using neural networks. In *IberEval@ SEPLN*, pages 249–254.

Vijayasaradhi Indurthi, Bakhtiyar Syed, Manish Shrivastava, Nikhil Chakravartula, Manish Gupta, and Vasudeva Varma. 2019. Fermi at semeval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 70–74.

Ganesh Jawahar, Benot Sagot, and Djam Seddah. 2019. What does bert learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

David Jurgens, Eshwar Chandrasekharan, and Libby Hemphill. 2019. A just and comprehensive strategy for using nlp to address online abuse. *arXiv preprint arXiv:1906.01738*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining.

Ping Liu, Wen Li, and Liang Zou. 2019. Nuli at semeval-2019 task 6: Transfer learning for offensive language detection using bidirectional transformers. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 87–91.

Joaquın Padilla Montani and Peter Schüller. 2018. Tuwienkbs at germeval 2018: German abusive tweet detection. In *14th Conference on Natural Language Processing KONVENS*, volume 2018, page 45.

Endang Wahyu Pamungkas, Valerio Basile, and Viviana Patti. 2020. Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing & Management*, 57(6):102360.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules.

Niloofar Safi Samghabadi, Parth Patwa, PYKL Srinivas, Prerana Mukherjee, Amitava Das, and Thamar Solorio. 2020. Aggression and misogyny detection using bert: A multi-task approach. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 126–131.

Manuela Sanguinetti, Fabio Poletto, Cristina Bosco, Viviana Patti, and Marco Stranisci. 2018. An italian twitter corpus of hate speech against immigrants. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2018. Ordered neurons: Integrating tree structures into recurrent neural networks.

Bin Wang, Yunxia Ding, Shengyan Liu, and Xiaobing Zhou. 2019. Ynu_wb at hasoc 2019: Ordered neurons lstm with attention for identifying hate speech and offensive language. In *FIRE (Working Notes)*, pages 191–198.

DH-FBK @ HaSpeeDe2: Italian Hate Speech Detection via Self-Training and Oversampling

Elisa Leonardelli
Fondazione Bruno Kessler
Trento, Italy
eleonardelli@fbk.eu

Stefano Menini
Fondazione Bruno Kessler
Trento, Italy
menini@fbk.eu

Sara Tonelli
Fondazione Bruno Kessler
Trento, Italy
satonelli@fbk.eu

Abstract

We describe in this paper the system submitted by the DH-FBK team to the HaSpeeDe evaluation task, and dealing with Italian hate speech detection (Task A). While we adopt a standard approach for fine-tuning AlBERTo, the Italian BERT model trained on tweets, we propose to improve the final classification performance by two additional steps, i.e. self-training and oversampling. Indeed, we extend the initial training data with additional silver data, carefully sampled from domain-specific tweets and obtained after first training our system only with the task training data. Then, we retrain the classifier by merging silver and task training data but oversampling the latter, so that the obtained model is more robust to possible inconsistencies in the silver data. With this configuration, we obtain a macro-averaged F1 of 0.753 on tweets, and 0.702 on news headlines.

1 Introduction

Although hate speech detection may seem a solved task on English, with more than 60 systems participating in the last Offenseval edition reaching an F1 $>$ 0.90 (Zampieri et al., 2020), this goal has not been reached when moving to other languages and settings. For example, at the last HaSpeeDe shared task on Italian (Bosco et al., 2018) the best systems reached 0.83 F1 on Facebook data and 0.80 on Twitter data (Cimino et al., 2018), but the performance dropped below 0.70 F1 when dealing with a cross-domain setting, i.e. training on Facebook and testing on Twitter (Cimino et al., 2018), and vice-versa (Corazza et al., 2018). Other recent studies confirmed that detecting hate speech on different social media platforms would require a platform-specific setting, and that just merging all training data coming from different sources does not always improve performance, in particular when testing on Twitter (Corazza et al., 2019).

The problem of developing hate speech detection systems that are robust when analysing different sources or data that vary over time is however an understudied problem. Therefore, the task of out-of-domain classification introduced this year at HaSpeeDe is particularly important and will hopefully foster the development and evaluation of classifiers with good generalisation capabilities.

Concerning our classification approach, we build a standard pipeline based on AlBERTo (Polignano et al., 2019b), the Italian transformer-based model trained on Twitter data, since BERT-like models represent the state of the art for hate speech detection (Zampieri et al., 2020). We extend it in two ways: first, we use *self-training* to build a first classifier with the task training data and annotate a large set of tweets collected via Islam- and immigrant-specific hashtags. The silver data and the task training set are then merged to train a second, possibly more robust classifier, which we use to classify the test set. When retraining, we introduce *over-sampling* in one of the two runs submitted by our team, i.e. we repeat five times the task training data so that they are balanced with respect to the silver data. This, together with self-training, proved to be effective when evaluated in a five-fold fashion on the training set, outperforming a standard approach based only on fine-tuning with AlBERTo.

2 Related Work

While most approaches to hate speech detection have been proposed for English, other systems have been recently developed to deal with a num-

---

ber of other languages, including Turkish, Arabic, Danish (Zampieri et al., 2020), German (Wiegand et al., 2018) and Spanish (Basile et al., 2019). Concerning Italian, the first *Hate Speech Detection* task (HaSpeeDe) for Italian was organized at EVALITA-2018 (Bosco et al., 2018). The task consisted in automatically annotating messages from Twitter and Facebook, with a boolean value indicating the presence (or not) of hate speech. The participating systems adopt a wide range of approaches, including bi-LSTM (la Peña Sarracén et al., 2018), SVM (Santucci et al., 2018), ensemble classifiers (Polignano and Basile, 2018; Bai et al., 2018), RNN (Fortuna et al., 2018), CNN and GRU (von Grunigen et al., 2018). The authors of the best-performing system, ItaliaNLP (Cimino et al., 2018), experiment with three different classification models: one based on linear SVM, another one based on a 1-layer BiLSTM and a newly-introduced one based on a 2-layer BiLSTM which exploits multi-task learning with additional data from the 2016 SENTIPOLC task (Barbieri et al., 2016). The same training and test set released for HaSpeeDe have been recently used also for other types of evaluation, for example to compare classifier performance and settings across different languages (Corazza et al., 2020), confirming the importance of domain-specific language models and the effectiveness of deep learning approaches (in this case, LSTM + fasttext embeddings). Since the development of BERT-like transformer-based models, however, they have become state-of-the-art approaches in several NLP tasks. This includes also hate speech detection for Italian, with the BERT model AlBERTo (Polignano et al., 2019b), which has recently achieved top-scores in two out of three HaSpeeDe 2018 tasks (Polignano et al., 2019a). For this reason, we decided to develop a classifier using the same model and the same approach.

3 Task Description

For the 2020 edition of EVALITA (Basile et al., 2020), the HaSpeeDe task (Sanguinetti et al., 2020) has focused on three main phenomena relevant to online hate speech detection by proposing three different tasks:

- Task A (main task): binary classification task aimed at determining whether a message contains hate speech or not
- Task B: binary classification task aimed at determining whether a message contains stereotypes or not
- Task C: sequence labeling task aimed at recognizing nominal utterances in hateful tweets

We participate in Task A, which in 2020 has the goal also to investigate variation in language and time concerning hate speech detection. To this purpose, the training set contains Twitter data, accompanied by a test set including both in-domain and out-of-domain data (tweets + news headlines), as well as from different time periods.

4 Data

In our experiments we use two types of data, the HaSpeeDe2 dataset provided by the task organisers, and domain-specific data collected from Twitter, that we include as silver data. The two datasets are described below.

4.1 HaSpeeDe2 Dataset

This dataset contains the training data provided by the organizers. These data specifically focus on the presence or the absence of hateful content towards immigrants, muslims or roma people. It consists of 6,839 annotated tweets, with 2,766 messages annotated as hateful and 4,073 as non-hateful.

4.2 Silver data description

Since the task is focused on hate speech against immigrants and minorities, we decided to exploit a set of tweets in Italian that covers similar topics and that was collected within the European project Hatemeter[1] (Ferret et al., 2019). For this project, conducted between February 2018 and January 2020, we downloaded tweets using hashtags of hate towards the Islam community, for example *#nomoschee*, *#stopIslam*, etc. Even if the dataset mainly covers Islam, references to other minorities like Roma or generic Immigrants are also present. To ensure that also other minorities are well represented, we randomly select from this dataset tweets that contain the most common words as chosen from the training data provided by task organizers, i.e. *Rom*, *nomade*, *migrante*, *straniero*, *profugo*, *islam*, *mussulmano* (*musulmano*), *terrorista*. Overall, around 20,400 additional tweets were selected. We then perform a first round of

---

[1] `http://hatemeter.eu/`

classification of the "new" tweets using the available data provided by organizers as training. This results in a new silver dataset composed of 11,129 hate and 9,254 non-hate tweets. This additional dataset is then merged with the task gold data and used to re-train the classifier. Details are reported in the following Section.

5 System Description

The classifier developed for both runs submitted by our team is based on the Italian BERT model trained on tweets, called AlBERTo (Polignano et al., 2019b). After fine-tuning it on the task training data, we use the obtained classifier to automatically annotate the additional dataset described in Section 4.2. These silver data are then merged with the task training data and used to fine-tune AlBERTo a second time. For one of the two submitted runs, we also experiment with oversampling as follows:

- **Run1:** we add the silver data to the tweets provided by the organizers for the training, keeping 500 of the released tweets for validation. In this setting, the training set size is $\sim$27,000 tweets, including 20,400 silver instances.
- **Run2:** we add the silver data to the tweets provided by the organizers as in Run1, but the tweet from organizers are oversampled by repeating them five times (and shuffling) in the training set, while tweets from the silver dataset occur only once. In this setting, the training set includes $\sim$52,000 tweets, with 39% of them being silver data.

We tested also the option to automatically assign a *tag* to each tweet, stressing the presence of a certain topic (immigrants/roma people/islam) using a keyword-based approach. However, with this additional information the classifier performed worse than without any topic indicator, so we removed it from the final runs. Below we report a detailed description of the process to select the best classification model, and of the preprocessing steps.

5.1 Model selection

The best performance in a wide variety of NLP tasks is currently obtained with approaches based on BERT (Devlin et al., 2019), a pre-trained transformed-based language model that can be fine-tuned and adapted to specific tasks by adding just one additional output layer to the neural network. As different BERT models exist, we first evaluated whether to use a multilingual version of BERT or the Italian version trained on Twitter data, called AlBERTo (Polignano et al., 2019b).

The comparison and evaluation of the different models and approaches is done with a 6-fold cross-validation using the task training set. Each fold consists of about 1,000 tweets as test while the others are used as train and validation (500 tweets). The performance score is obtained as the average of the six folds, so that the final evaluation is unbiased and independent as much as possible from the specific splits into train, validation and test.

In our setup we tested two models, first Multilingual BERT, covering 104 languages including Italian [2] and then AlBERTo, which was trained using the official BERT source code on 200M tweets in the Italian language. For the fine-tuning of AlBERTo we run it for 15 epochs, using a learning rate of 2e-5 with 1000 steps per loops on batches of 64 examples. Since AlBERTo performed better than multilingual BERT on each fold, it was included in the final system configuration for the task. The cross-validation over 6 folds using only the task training set with AlBERTo resulted in an average Macro-F1 of 83.12 for Run1 and 82.15 for Run2.

5.2 Data Preprocessing

The data, both from the dataset provided by the organisers and the silver one, are preprocessed as follows. First we split hashtags by adapting to Italian the Ekphrasis tool (Gimpel et al., 2010), which recognises the tokens in a hashtag based on Google n-grams. With the same tool we also normalise the text to replace all mentions to users and urls with <user> and <url> respectively. We also replace with a dedicated tag all the instances of *"money"*, *"time"*, *"date"* and in general any *"number"*. The emojis are replaced with their descriptions[3] in order to have a textual representation to be used with AlBERTo.

---

[2] with 12-layer, 768-hidden, 12-heads, 110M parameters

[3] manually translated to Italian from the English description at `rhttps://unicode.org/emoji/charts/full-emoji-list.html`.

		Hate class			Non-hate class			Macro Avg.
DocType.	System	Precision	Recall	F1	Precision	Recall	F1	F1
Tweets	Run1	0.7237	0.7958	0.758	0.7806	0.7051	0.7409	0.7495
	Run2	0.727	0.8006	0.762	0.7855	0.7083	0.7448	**0.7534**
	baselineMF	0	0	0	0.5075	1.000	0.6733	0.3366
	baselineSVM	0.7096	0.7347	0.7219	0.7334	0.7082	0.7206	0.7212
	best system							*0.8088*
News	Run1	0.6833	0.453	0.5448	0.7395	0.8808	0.804	0.6744
	Run2	0.6911	0.5193	0.593	0.7609	0.8683	0.8111	**0.702**
	baselineMF	0	0	0	0.638	1.000	0.7789	0.3894
	baselineSVM	0.6071	0.3756	0.4641	0.7087	0.862	0.7779	0.621
	best system							*0.7744*

Table 1: Results of the two submitted runs for Task A on tweets and on news headlines. BaselineMF = most-frequent baseline; baselineSVM = linear SVM with unigrams, char-grams and TF-IDF representation

6 Evaluation

We submitted two runs each for the in-domain (tweets) and out-of-domain (news headlines) text types in Task A. The results obtained on the test set are reported in Table 1 and compared with two baselines provided by the task organisers, one obtained by always assigning the most frequent label (i.e. non-hateful), and the other by training an SVM classifier with unigrams, char-grams and TF-IDF representation as features. We also compare our results with the top-ranked system in each subtask (additional details on such systems have not been disclosed at the moment of writing).

As expected, on out-of-domain data (news headlines) we obtain lower results than on tweets, since the training set is retrieved exclusively from Twitter. Furthermore, our approach does not include any specific tuning aimed at treating news headlines differently from tweets. On the contrary, the additional data used for self-training are all gathered from Twitter, which may negatively affect performance on out-of-domain data.

On both document types, Run2 performs better than Run1, showing that our oversampling strategy to reduce the weight of silver data is effective. However, results obtained with 6-fold cross-validation only on the training set were significantly higher, both with macro F1 $>$ 0.80. This may be explained by the fact that, as pointed out by the task organisers, tweets from the test set were collected in a different time period than those in the training set. This will likely make the two sets different in terms of topics.

Run 1		Actual Values	
		non-hate	hate
Predicted	non-hate	452	127
	hate	189	495
Run 2		Actual Values	
		non-hate	hate
Predicted	non-hate	454	124
	hate	187	498

Table 2: Confusion matrix on *tweets* results

We report in Table 2 and 3 the confusion matrix showing the number of true positives and negatives, and false positives and negatives obtained with the two runs on tweets and news headlines. While on tweets the performance on the hate class is overall better, in particular concerning recall, this does not apply to news headlines, with a low recall for the hate class. The reason for this low score lies in the different linguistic expressions connected with hate between tweets and headlines: while in tweets they are more direct, and more frequently connected with profanities that a classifier can easily recognise, hateful content in news headlines is usually expressed in more subtle ways. As an example, we report below two headlines misclassified by our system. The first one (i) was classified as non-hateful, even if it conveys hateful content. The second one (ii) was instead classified as hateful, although it is not:

Run 1		Actual Values	
		non-hate	hate
Predicted	non-hate	281	99
	hate	38	82

Run 2		Actual Values	
		non-hate	hate
Predicted	non-hate	277	87
	hate	42	94

Table 3: Confusion matrix on *news headlines* results

i) *Sea Watch, l'ultima presa in giro degli immigrati all'Italia: i minori nati tutti lo stesso giorno* (EN: Sea Watch, migrants making fun of Italy: all underage migrants born on the same day)

ii) *Matera, Salvini contestato durante il comizio. E lui risponde: "Bravi, avete vinto dieci immigrati da mantenere"* (EN: Matera, Salvini challenged at a rally, and he replies: "Congratulations, you won ten migrants to pay for")

Both examples have a similar structure, are written in standard Italian and mention migrants. Furthermore, the second example reports a hateful direct speech, but since it is only reported it does not mean that the journalist agrees with what was said by the politician Matteo Salvini.

7 Conclusions

In this paper we described the system developed by the DH-FBK team to participate in the HaSpeeDe shared Task A. We submitted two runs, both based on AlBERTo and using in-domain silver data as additional training data in a self-learning framework. The only difference between the two configurations is that, for Run2, the task training data were repeated five times, to balance the weight of silver data.

Our evaluation shows that, both in a cross-validation setting and on the task test set, over-sampling has a positive effect on the classification results. As expected, performance on in-domain data (i.e. training and testing on tweets) is better than on out-of-domain data (i.e. training on tweets and testing on news headlines). In the future, we may try to address this issue by including as silver data also news headlines, so that also the specificity of this kind of text is taken into account. For a better data quality, it may be useful to select only the silver instances that have been automatically classified with high confidence.

References

Xiaoyu Bai, Flavio Merenda, Claudia Zaghi, Tommaso Caselli, and Malvina Nissim. 2018. Rug @ EVALITA 2018: Hate speech detection in italian social media. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy.*

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the evalita 2016 sentiment polarity classification task.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Cristina Bosco, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. 2018. Overview of the evalita 2018 hate speech detection task. In *EVALITA 2018-Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*, volume 2263, pages 1–9. CEUR.

Andrea Cimino, Lorenzo De Mattei, and Felice Dell'Orletta. 2018. Multi-task learning in deep neural networks at EVALITA 2018. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy.*

Michele Corazza, Stefano Menini, Pinar Arslan, Rachele Sprugnoli, Elena Cabrio, Sara Tonelli, and Serena Villata. 2018. Comparing different supervised approaches to hate speech detection. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy.*

Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata. 2019. Cross-platform evaluation for italian hate speech detection. In *Proceedings of the Sixth Italian Conference on Computational Linguistics, Bari, Italy, November 13-15, 2019*.

Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata. 2020. A multilingual evaluation for online hate speech detection. *ACM Transactions on Internet Technology*, 20(2):10:1–10:22.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, Minnesota, June.

Jérôme Ferret, Mario Laurent, Daniela Andreatta, Andrea Di Nicola, Elisa Martini, M Guerini, S Tonelli, Georgios Antonopoulos, and Parisa Diba. 2019. Hatemeter d18: Training module a for academics and research organisations.

Paula Fortuna, Ilaria Bonavita, and Sérgio Nunes. 2018. Merging datasets for hate speech classification in italian. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. 2010. Part-of-speech tagging for twitter: Annotation, features, and experiments. Technical report, Carnegie-Mellon University, School of Computer Science.

Gretel Liz De la Peña Sarracén, Reynaldo Gil Pons, Carlos Enrique Muñiz-Cuza, and Paolo Rosso. 2018. Hate speech detection using attention-based LSTM. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy*.

Marco Polignano and Pierpaolo Basile. 2018. Hansel: Italian hate speech detection through ensemble learning and deep neural networks. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy*.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, and Giovanni Semeraro. 2019a. Hate speech detection through alberto italian language understanding model. In Mehwish Alam, Valerio Basile, Felice Dell'Orletta, Malvina Nissim, and Nicole Novielli, editors, *Proceedings of the 3rd Workshop on Natural Language for Artificial Intelligence co-located with the 18th International Conference of the Italian Association for Artificial Intelligence (AIIA 2019), Rende, Italy, November 19th-22nd, 2019*, volume 2521 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019b. Alberto: Italian BERT language understanding model for NLP challenging tasks based on tweets. In Raffaella Bernardi, Roberto Navigli, and Giovanni Semeraro, editors, *Proceedings of the Sixth Italian Conference on Computational Linguistics, Bari, Italy, November 13-15, 2019*, volume 2481 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Valentino Santucci, Stefania Spina, Alfredo Milani, Giulio Biondi, and Gabriele Di Bari. 2018. Detecting hate speech for italian language in social media. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy*.

Dirk von Grunigen, Ralf Grubenmann, Fernando Benites, Pius Von Daniken, and Mark Cieliebak. 2018. spmmmp at germeval 2018 shared task: Classification of offensive content in tweets using convolutional neural networks and gated recurrent units. In *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)*.

Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018. Overview of the germeval 2018 shared task on the identification of offensive language. In *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)*, pages 1 – 10, Vienna, Austria. Austrian Academy of Sciences.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). In *Proceedings of the 14th International Workshop on Semantic Evaluation*. Association for Computational Linguistics.

By1510 @ HaSpeeDe 2: Identification of Hate Speech for Italian Language in Social Media Data

Tao Deng, Yang Bai, Hongbing Dai†
School of Information Science and Engineering
Yunnan University, Yunnan, P.R. China
`Dtao.top@gmail.com`
`baiyang.top@gmail.com`
`hbdai_it@126.com`

Abstract

English. Hate speech detection has become a crucial mission in many fields. This paper introduces the system of team **By1510**. In this work, we participate in the HaSpeeDe 2 (Hate Speech Detection) shared task which is organized within Evalita 2020(The Final Workshop of the 7th evaluation campaign). In order to obtain more abundant semantic information, we combine the original output of BERT-Ita and the hidden state outputs of BERT-Ita. We take part in task A. Our model achieves an F1 score of 77.66% (6/27) in the tweets test set and our model achieves an F1 score of 66.38% (14/27) in the news headlines test set.

Italiano. *L' individuazione dell' incitamento allodio diventata una missione cruciale in molti campi. Questo articolo introduce il sistema del team By1510. In questo lavoro, partecipiamo al task HaSpeeDe 2 che stato organizzato allinterno di Evalita 2020. Per ottenere informazioni semantiche pi abbondanti abbiamo combinato loutput originale di BERT Ita e gli output di hidden state di BERT Ita. Il sistema presentato partecipa al task A. Il nostro modello ottiene un punteggio F1 di 77.66% (6/27) sui dati di test da Twitter e un punteggio F1 di 66.38% (14/27) sui dati di test contenenti titoli di quotidiano.*

1 Introduction

With the continuous development of computer and networks, social media users have increased year by year, social media has entered people's daily life and becomes an indispensable part. More and more people use the Internet to express their opinions and ideas on social media platforms. Some offensive, abusive, defamatory contents are easy to spread and incite hatred, and these negative contents can cause some bad effects. The simplest way is that people mark the report and then delete the system warning, which can not be solved efficiently. Therefore, an efficient way is urgently needed to eliminate these negative effects. This paper proposes a hate speech detection system, which can better detect and mark these annoying contents. The HaSpeeDe 2 (Sanguinetti et al., 2020) (Hate Speech Detection) shared task is organized within Evalita 2020 (Basile et al., 2020), the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian, which help to detect whether the Italian language on Twitter contains hate language, with the aim to reduce the spread of hate speeches and online harassment. (Waseem and Hovy, 2016)

In this paper, we take part in task A in the HaSpeeDe 2 task. The BERT model we use is dbmz[1] trained on Italian data. In order to obtain more abundant semantic information, we extract the state of hidden layer outputs and we provide a reference for the detection of the hate speech in the Italian language. The rest of the paper is organized as follows. Section 2 briefly shows the related work for the identification of hate speeches. Section 3 elaborates on our approach. It shows the data set officially provided and architecture of our model. Section 4 describes the hyper-parameters and our results. Finally, Section 5 concludes our work.

---

[1] https://huggingface.co/dbmdz

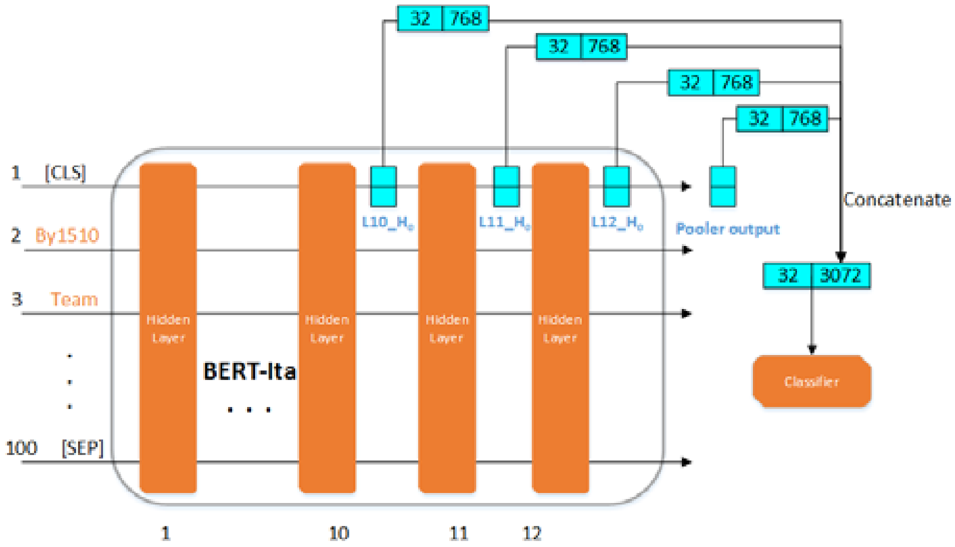


Figure 1: our model. $L12\_H_0$ is hidden-state of the first token of the sequence(CLS token) at the output of the 12th hidden layer of the BERT-Ita. Similarly, $L11\_H_O$ and $L10\_H_O$ are the 11th and 10th hidden layers outputs of BERT-Ita respectively. [32, 768]/[32, 3072] is the output shape (batch_size, hidden_size)

2 Related Work

Previously, machine learning (Davidson et al., 2017; MacAvaney et al., 2019a), Bayesian method (Miok et al., 2020; Fauzi and Yuniarti, 2018), support vector machine (MacAvaney et al., 2019b; Del Vigna12 et al., 2017), neural network (Badjatiya et al., 2017; Zhang et al., 2018) and other methods were proposed for the identification of hate speech. In the Hindi-English mixed language, (Bohra et al., 2018) et al. in parentheses used a supervised classification system to detect the hate speech in the text in the code-mixed language. The classification system used Character N-Grams, Word N-Grams, Punctuations, Negation Words, Lexicon and other feature vectors for classification and training. The accuracy could reach 71.7% with SVM, which proved to be a very effective method for classification tasks. In Danish language, (Sigurbergsson and Derczynski, 2019) developed four automatic classification systems to detect and classify hate speech in English and Danish, and proposed a method to automatically detect different types of the hate speech, which achieved good results for the detection of English and Danish hate speeches. In English language, (Aroyehun and Gelbukh, 2018) used a linear baseline classifier (nbsvm with n-grams) and improved deep neural network model.

For the Italian language, (Polignano et al., 2019) proposed an $AlBERTo$ model based on classifier integration, which was verified by cross validation on Facebook and Twitter data sets, and the effect was obvious in offensive words. (Corazza et al., 2018) used recurrent neural network, n-gram neural network and support vector machine to classify Twitter data sets, and its recurrent model had achieved good results. (Bianchini et al., 2018) proposed artificial neural network to annotate and classify 3000 message data from Facebook and Twitter, and achieved good results.

3 Methodology

3.1 Data Description

In this work, we take part in task A, which is a binary classification task aimed at determining the presence or the absence of hateful content in the text towards a given target (among Immigrants, Muslims or Roma people). The organizers provide the training set and test set. For the training set, it is from Twitter. For the test set, the organizers provide in-domain data and out-of-domain data, which come from Twitter and news headlines, respectively. It can be seen from Table 1 that the data set is slightly imbalanced.

3.2 Our approach

As the train data is very limited we resort to a transfer learning approach. That is, we take an NLP model pre-trained(Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019) on a large

	Hate Speech (HS)	No HS
train data	2766	4071
test data (tweets)	622	641
test data (news headlines)	181	319

Table 1: Distribution of data set in the Task A.

	Hyperparameters

Our Model	output_hidden_states=True
	max sequences length=100
	learning rate=1e-5
	adam_epsilon=1e-8
	per_gpu_train_batch_size=32
	gradient_accumulation_steps=1
	epoch=8
	dropout=0.1

Table 2: Hyperparameters of the model in our experiments.

corpus of texts and fine-tune it for a specific task at hand. In this work, we used BERT-base-Italian-uncased(BERT-Ita)[2] from Transformers library. It is trained on the recent Wikipedia dump and various texts from the OPUS corpora[3] collection. The final training corpus has a size of 13GB and 2050 million tokens. For classification tasks, the output of BERT-Ita (*pooler output*) is obtained by its last layer hidden state of the first token of the sequence (CLS token) further processed by a linear layer and a Tanh activation function. However, the pooler output is usually not a good summary of the semantic information. Therefore, we extract the hidden layer output of BERT-Ita to obtain more abundant semantic information.

(Jawahar et al., 2019) pointed that the hidden layer of BERT encodes a rich hierarchy of linguistic information, with surface features at the bottom layer, syntactic features in the middle layer and semantic features at the top layer. Therefore, we get abundant semantic information by extracting the extra semantic features which is the last three hidden layer outputs($L12\_H_0$, $L11\_H_0$ and $L10\_H_0$) of BERT-Ita. We propose the following model which is shown in Figure 1. In the model, we get $L12\_H_0$, $L11\_H_0$, $L10\_H_0$ from the top hidden layer of BERT-Ita. We concatenate *pooler output*, $L12\_H_0$, $L11\_H_0$ and $L10\_H_0$ into the classifier.

4 Experiments and Results

4.1 Preprocessing and Experiments Setup

In the experiment, we try to preprocess the text but we did not achieve the desired results. We find that after preprocessing the Twitter data, the F1-score of the model decreased on the validation set. We do not preprocess the data and we do not use an extra data set. In this work, the training set is split into the new training set and the validation set by using the Stratified 5-Fold Cross-validation[4].The random seed is set 42 in Cross-validation. Due to the imbalance of datasets, the Stratified 5-Fold Cross-validation ensures that the proportion of samples in each category in each fold data set remains unchanged. During the training, the best weight of the model is saved in 8 epochs. Table 2 shows the hyperparameters used in our model.

4.2 Results and analysis

In the experiment, we find that with the increase of the extra semantic features, the model can obtain more abundant semantic information. Table 3 shows the performance of the model for different semantic features after getting the labels of the test set.[5].

	Task A test set of tweets(100%)	
	No HS	HS
No HS	489	152
HS	119	503
	Task A test set of news headlines(100%)	
	No HS	HS
No HS	312	7
Hs	133	48

Table 4: The confusion matrix of BERT-Ita+L12_H_O in test sets.

---

[2] https://huggingface.co/dbmdz/bert-base-italian-uncased

[3] http://opus.nlpl.eu/

[4] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html#sklearn.model_selection.StratifiedKFold

[5] https://github.com/msang/haspeede/tree/master/2020

	Task A test set of tweets(100%)	Task A test set of news headlines(100%)
	Precision/Recall/Macro F1-score	Precision/Recall/Macro F1-score
BERT-Ita+L12_H_O	78.61/78.58/**78.54**	78.69/62.16/61.18
BERT-Ita+L12_H_O+L11_H_O	75.50/77.27/77.16	78.13/62.23/62.76
BERT-Ita+L12_H_O+L11_H_O+L10_H_O (Our submitted model)	77.80/77.72/77.66	72.07/65.74/**66.38**

Table 3: The performance of the model for these test sets.

	Task A test set of tweets(100%)	
	No HS	HS
No HS	478	163
HS	119	503
	Task A test set of news headlines(100%)	
	No HS	HS
No HS	289	30
Hs	107	74

Table 6: The confusion matrix of BERT-Ita+L12_H_O+L11_H_OF+L10_H_O in test sets.

	Task A test set of tweets(100%)	
	No HS	HS
No HS	463	178
HS	110	512
	Task A test set of news headlines(100%)	
	No HS	HS
No HS	310	9
Hs	128	53

Table 5: The confusion matrix of BERT-Ita+L12_H_O+L11_H_O in test sets.

The confusion matrices (actual values are represented by rows) are shown in Table 4, Table 5, Table 6. These tables show the performance of the model on the test set as the extra semantic features increase. In the tweets test set, we can see from these tables that the ability of the model to detect the hate speech is increasing as the extra semantic features increase. Similarly, in the news headlines test set, the ability of the model to detect the hate speech is also increasing. We think that with the increase of these extra semantic features, the model can learn more semantic information. In addition, we find that our model achieve good results on the tweets test set, but the results of our model are not good on the news headline data set. There are many differences between the syntactic features of tweets and news headlines. For example, there are many irregular expressions in tweets, while news expressions are very standard. Our model is only fine-tuned on the tweets data set, so we think this affects the performance of the model on other types of data.

5 Conclusion

In this work, this paper introduces the system proposed for HaSpeeDe 2 shared task for identifying and classifying hate speeches on social media. We enriched BERT-Ita with semantic information by extracting the extra semantic features. We find that with the increase of semantic information, the performance of the model for identifying the hate speech is also increasing. Finally, in the official evaluation, our model rank 6th (6/27) in the tweets test set and 14th (14/27) in the news headlines test set. In the future, we will focus on how to make the model learns more semantic information.

References

Segun Taofeek Aroyehun and Alexander Gelbukh. 2018. Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 90–97.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Eval-*

uation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020), Online. CEUR.org.

Giulio Bianchini, Lore nzo Ferri, and Tommaso Giorni. 2018. Text analysis for hate speech detection in italian messages on twitter and facebook. *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:250.

Aditya Bohra, Deepanshu Vijay, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A dataset of hindi-english code-mixed social media text for hate speech detection. In *Proceedings of the second workshop on computational modeling of peoples opinions, personality, and emotions in social media*, pages 36–41.

Michele Corazza, Stefano Menini, Pinar Arslan, Rachele Sprugnoli, Elena Cabrio, Sara Tonelli, and Serena Villata. 2018. Comparing different supervised approaches to hate speech detection.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009*.

Fabio Del Vigna12, Andrea Cimino23, Felice DellOrletta, Marinella Petrocchi, and Maurizio Tesconi. 2017. Hate me, hate me not: Hate speech detection on facebook. In *Proceedings of the First Italian Conference on Cybersecurity (ITASEC17)*, pages 86–95.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

M Ali Fauzi and Anny Yuniarti. 2018. Ensemble method for indonesian twitter hate speech detection. *Indonesian Journal of Electrical Engineering and Computer Science*, 11(1):294–299.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy, July. Association for Computational Linguistics.

Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, and Ophir Frieder. 2019a. Hate speech detection: Challenges and solutions. *PLoS one*, 14(8):e0221152.

Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, and Ophir Frieder. 2019b. Hate speech detection: Challenges and solutions. *PLoS one*, 14(8):e0221152.

Kristian Miok, Blaz Skrlj, Daniela Zaharie, and Marko Robnik-Sikonja. 2020. To ban or not to ban: Bayesian attention networks for reliable hate speech detection. *arXiv preprint arXiv:2007.05304*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, and Giovanni Semeraro. 2019. Hate speech detection through alberto italian language understanding model. In *NL4AI@ AI* IA*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Gudbjartur Ingi Sigurbergsson and Leon Derczynski. 2019. Offensive language and hate speech detection for danish. *arXiv preprint arXiv:1908.04531*.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93.

Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018. Detecting hate speech on twitter using a convolution-gru based deep neural network. In *European semantic web conference*, pages 745–760. Springer.

Venses @ HaSpeeDe2 & SardiStance: **Multilevel Deep Linguistically Based Supervised Approach to Classification**

Rodolfo Delmonte
Dipartimento di Studi Linguistici e Culturali
Comparati
Ca' Bembo – Dorsoduro 1075 – Università Ca'
Foscari – 30131 Venezia
delmont@unive.it

Abstract

In this paper[1] we present the results obtained with ItVENSES a system for syntactic and semantic processing that is based on the parser for Italian called ItGetaruns to analyse each sentence. In previous EVALITA tasks we only used semantics to produce the results. In this year EVALITA, we used both a fully and mixed statistically based approach and the semantic one used previously. The statistic approaches are all characterized by the use of n-grams and the usual tf-idf indices. We added another parameter called the Kullback-Leibler Divergence to compute similarities. In addition we used emoticons and hashtags. Results for the two runs allowed have been fairly low – around 40% F1-score. We continued producing other runs on the basis of the statistical approach and after receiving the gold-test version and the evaluation script we discovered that in one of these additional runs - the fourth - we improved up to 54% macro F1 for HaSpeeDe2 task and up to 48% macro F1 for Sardines.

1 Introduction

In this paper we will present work carried out by the Venses Team in Evalita 2020 (Basile et. 2020). We will comment in the following both on the Sardines Task (Cignarella et al., 2020) and on the HaSpeeDe2 Task (Sanguinetti et al. 2020). The reason for this is discussed in the sections below, but it has been basically determined by the overlapping in the choice of the features to adopt for the classification tasks. To show how the two tasks share part of the features we created a table where we compare the output of the first step in the process, i.e. the creation of a frequency list dictionary. The frequency list that we show in Table 1. below is made of nominal entities that were extracted automatically from the total frequency list. We call this frequency list *InstanceList* and the position occupied by each entry as *InstanceListPosition* and the Rank as *InstanceRank*. In the first column we indicate rank; in the following two columns we report the word/s preceded by its frequency value. In the second couple of columns, column no. 4 and 5 we make a comparison between the two corpora based on frequency lists and the rank each entry has received.

We use three types of values: the frequency value from the general frequency list derived from the corpus; the rank position in the InstanceList in case the word appears in both InstanceLists; and the word "nil" in case the entry is not present in the general frequency list of the comparing corpus. In column 4 the comparison is made between the first list (HaSpeeDe2) and its instances and the second list (SardiStance). Every word is associated to the rank in the InstanceList and a second element which can be one of three: the position in the second list if available; the position in the general FrequencyList of the compared corpus; nil in case the word is not present.

Corpus/ FreqRank	HaSpeeDe2 (H)	SardiStance (S)	Comparison FreqOrder H vs S	Comparison FreqOrder S vs H
1	1567-rom	1807-sardine/ 142-sardina/ 97-'6000sardine'	Rom-1-nil	Sardine-1-nil
2	1308-migranti	325-salvini	Migranti-2-nil	Salvini-2-15
3	1046-italia	222-piazza/76-piazze	Italia-3-5	Piazza-3-479
4	985-immigrati	205-PD	Immigrati-4-1555	PD-4-23
5	624-roma	150-sinistra	Roma-5-33	Sinistra-5-117
6	609-italiani	133-politica	Italiani-6-168	Politica-6-256
7	334-campo/ 211-campi	129-movimento	Campo-7-733	Movimento-7-1980
8	454-immigrazione	113-italia	Immigrazione-8-2060	Italia-8-3
9	462-stranieri	108-bologna	Stranieri-9-1784	Bologna-9-1006
10	413-islam	95-lega	Islam-10-3099	Lega-10-258
11	369-casa	75-partito	Casa-11-219	Partito-11-600
12	316-clandestini	72-governo	Clandestini-12-729	Governo-12-115
13	310-profughi	61-emilia	Profughi-13-nil	Emilia-13-14822
14	310-terroristi	60-persone	Terroristi-14-nil	Persone-14-194
15	312-salvini	60-bibbiano	Salvini-15-2	Gente-15-200
16	290-nomadi	60-gente	Nomadi-16-nil	Bibbiano-16-nil
17	246-musulmani	58-paese	Musulmani-17-6528	Paese-17-20
18	254-islamici	54-popolo	Islamici-18-nil	Popolo-18-292
19	214-milano	54-bonaccini	Milano-19-323	Bonaccini-19-nil
20	207-paese	51-giovani	Paese-20-115	Giovani-20-348
21	201-europa	50-m5s	Europa-95-121	M5s-21-336
22	201-bene	48-destra	Bene-96-137	Destra-22-1530
23	173-pd	47-bene	Terrorismo-108-1765	Bene-23-22

Table 1. Instance list of features created automatically from the dictionary of unique wordform for two tasks

For instance, we can see that the words *rom, migranti, profughi, terroristi, nomadi, islamici/rom, migrants, refugees, terrorists, nomads ,islamists* are not present in the second list and so they characterize the first corpus (HaSpeeDe2) as being different from the Sardines one, specializing it in a particular list of topics or keywords. When we look at column 5, where the comparison is made in reverse order, we discover that *sardine, bibbiano, bonaccini/sardines, bibbiano, bonaccini* are not present in the first list. Most importantly we discovered that the most frequent words of the two lists are not shared, "rom", in list 1, and "sardine" in list two. In the sections below we present the module for supervised automatic classification and the experiments that we devised using basically two approaches: a semantic approach vs a statistic approach.

In Table 2 and 3 we report the subdivision into classes of the two training and test corpora for the two tasks, SardiStance and HaSpeeDe2 with percent values to allow for comparisons. As can be noticed in the SardiStance corpus the majority class is constituted by AGAINST, followed by FAVOR and then NONE. In the Test set, the distribution into the three classes favors AGAINST and for the other two classes is almost identical. The same happens in the other corpus, the HaSpeeDe2, where we notice a majority of occurrencies for the NULL class in the Training corpus. In the Test set, this is still valid but we see an important increase of the *BothHateandStereo* class and a strong reduction of the Stereo class. Of course, these differences in class distribution may have influenced the final outcome, in case as it is ours - there is a default choice at the end of the computation for each tweet class. Here below some general quantitative information for the two corpora:

Corpus/ Class	HaSpeeDe2 Abs.Val.	HaSpeeDe2 Percent
NULL	3,049	44.5825%
OnlyHATE	748	10.9372%
OnlySTEREO	1,024	14.9729%
BothHATEAnd STEREO	2,018	29.5072%
Totals	6,839	100%

Table 2. Distribution of Classes for HaSpeeDe2 Tweets training and test corpora

Corpus/ Class	Sard Train	Sard. %	Sard. Test	Sard %
NONE	515	24.15	172	15.49
AGAINST	1,028	48.21	742	66.84
FAVOR	589	27.66	196	17.65
Totals	2,132	100%	1110	100%

Table 3. Distribution of Classes for SardiStance training and test corpora

2 The Module for Supervised Automatic Classification

We present the modules for automatic classification that uses ***three different approaches***: a fully BOW and statistic one, a fully semantically based one, and a mixed both bag-of-words and (partially) semantically-based one. With the exception of the fully semantic approach, the remaining approaches are however characterized by the use of n-grams and a fully supervised method to create the model. In all approaches the model is created on the basis of an automatically built dictionary of unique wordforms sorted by frequency where the first most frequent 25 nominal expressions are chosen as supplied instances for n-grams construction.

Eventually, we created six different classifiers that we will present in the sections below. They are a fully semantic classifier, a lexically-based semantic classifier, a mixed statistic and lexical semantic classifier using supervised n-grams, a fully statistic tf-idf classifier based on differences, a fully statistic Kullback-Leibler Divergence (hence KLD) classifier based on differences, a classifier based on emoticons and on hashtags.

First approach.

We will start by describing the lexically-based semantic classifier. This is used for both tasks but in a different manner. Whereas in the semantic classifier it is treated as an important component of the evaluation module, it becomes just a default classifier in the statistic classifiers, in case of failure of the previous ones. It is organized into a grid with seven slots:

[**Polarity, Appraisal, NegativeW, PositiveW, SwearW, HateW, StereoW**]

Polarity is computed at a propositional level by the deep parser and is described below. The remaining slots are all lexically processed. In particular Appraisal Classes are derived from previous work on political newspapers (Stingo and Delmonte, 2016); Swear Words, Negative and Positive Words are derived from previous work on opinion and sentiment analysis and were used in SenticPol (Delmonte, 2014); finally HateWords and StereoWords were collected from the HurtLex made available by the organizers, proceeding by a manual selection of Italian words and discarding all English words.

The second approach that we call semantically-based, uses a three levels of classification. Besides using an n-gram model, it uses a majority vote approach based on presence of emoticons previously classified on the basis of the training set. The most important module is fired in case of failure (no n-gram available to match) in the two previous steps and is totally based on semantics. It builds an interpretation from deep semantic analysis evaluating presence of appraisal theory labeled items, presence of hate/stereotype items from lexical lookup and their propositional level semantics. In the sections below we describe in details the three level classification module. This approach covers 93% of the whole training set – but see below. However its predicting power is not so great.

Third Approach.

The bag-of-words approach associates a numerical parameter to each word and the resulting sum for the each tweet. At first we uses TF-IDF as the mathematical formula for characterizing each word occurrence and each tweet. We applied TF-IDF to each word in each tweet and used the output to map the indices to n-grams and produce a model. Then we used this model to predict the similarity with n-grams obtained from the held out development set of tweets. The results were however very poor, 20% accuracy, which added to 12% obtained from the emoticons model made a 32% final accuracy.

We assumed the reason was that tweets are too short to be useful for term-frequency computation. In the majority of the cases wordforms appeared only once in each document/tweet – apart from stop words. So we searched a formula which could be better suited for this task and could represent both frequency and dispersion at corpus level. We found it in a number of papers published by Gries (2008, 2020), but also in a paper online by Koos Wilt. The important part of the formula regards the role of frequency of occurrence in the total corpus which is used to produce TF so that it would resemble a probability of occurrence and the concept of entropy[2]. Gries defines this formula as a way of characterizing "keyness" by including dispersion information. To do that he augmented frequency information by using the Kullback-Leibler Divergence. Wordforms can become key not only for their frequency of occurrence, their dispersion or both. The formula is able to "tease apart distributional differences".

p = frequency of w in document A of the corpus / divided by total frequency of w in the corpus
q = total number of tokens in document A of the corpus / divided by total number of tokens in the corpus

$$KLD = p \ X \log(p/q) \ \Rightarrow \ \sum p \ X \log(p/q)$$

In the same paper Gries suggests to compute keyness also to n-grams besides multiword expressions and this is what we did. The summation applies to the document/tweet and is used to differentiate each tweet from one another and produce a similarity or distance evaluation. We proceeded as before to verify the predictive ability of this new formula and came out with 44/45% accuracy, a 12% gain.

3 The Semantically-Based Module And The N-gram Models

The general procedure we organized for the three approaches is as follows.

At first we massaged the text in order to obtain a normalized version – wrong word accents like "nè" instead of "né" etc. The text is then turned into an xml file to suit the Prolog input requirements imposed by the system. It is then precompiled by a set of regular expressions: we separate the hash symbol # from its tag; we separate the @ symbol from the following username; we cancel the word URL; we separate all punctuation marks from a preceding or following word; then we lowercase all words and produce a sorted list which is then used to count frequencies associated to each wordform and produce the dictionary of unique wordforms or types.

Then we choose the first 25 nominal entities from the list erasing generic or general nouns like "person", "people" etc. The final list of features is treated as supplied instances to search for the construction of n-grams from 4-gram up to 8-grams: we take all sequences of four/eight tokens where the ending or beginning word must be taken from the list of instances. If eight is not available we accept down to 4-grams. Instances are collapsed under three unique general topic which are the following ones: racism, politics, sardines/Salvini.

Since we process each tweet using lemmata in every approach, we do sentence splitting and tagging. Every tagged token is then lemmatized and in the semantically-based approach it is subsequently associated to a lexically validated three-valued sentiment label.

In the semantically-based approach, we then compute syntactic constituency and dependencies for every sentence. This information is passed to the semantic processor which produces predicate argument structures for every sentence present in each tweet. In case no punctuation is available and the sentence is longer than 40 tokens we activate an empirical set of rules to insert punctuation and divide the tweet into sentences by checking the presence of words starting with uppercase letter and not being a Named Entity. If the sentence splitter fails we activate a search for sentence level coordinating or subordinating conjunctions. Many tweets are just fragments and contain a list of nouns and adjectives: we add a dummy verb ESSERE/to_be in order to allow the semantics to work.

Propositional level semantics is made by the computation of factivity, negation, subjectivity, modality, speech_act, diathesis, which then produce a fixed set of semantic labels to allow for a correct interpretation.

In the mixed approach and in the statistics-only approach we procede as follows. Before producing n-grams, we erase punctuation with the exception of the hash symbol that informs the system of the presence of an hastag or a slogan. Similarity is computed by matching every lemma

---

[2] According to Wilt Koos, ibid. pag.2: "Classification according to the KLD takes place on the assumption the training set reflects order and the test set, a document to be categorized, reflects a deviation from this order and is therefore chaotic or entropic. The lower the entropy regarding the training set, the more likely it is a given test set belongs to that training set. "

from two n-grams labeled with the same main topic. We established a ratio of 0.3 as the threshold for acceptance, but then we check the semantics be identical or very similar. We assume with Emily Bender that "a system trained on form alone cannot in principle learn meaning"[3] . So we use an approach with is based partially on bag-of-words n-grams – using frequency lists and n-grams - but we associate semantic interpretation to every n-gram of the model. Semantics is used to verify and confirm the first approximation of a similarity measure based on wordforms[4] and lemmata. We assume that n-grams belonging to a statement cannot possibly be regarded to have the same meaning in case the comparison is made with an n-gram extracted from a proposition which has negation at propositional level.

4 The Experiment and the Evaluation Module of *ItVenses*

We organized our classifiers to produce two runs as required by the two tasks, SardiStance and HaSpeeDe. However, we then realized that we needed to produce more runs in order to take into account all variables involved in the statistically-based module. Eventually we had to choose one modality for the single run with the statistical module trusting the results obtained from the Development set as described here below.

To produce a development set we held out 20% of all training corpus - 427 tweets for SardiStance and 1000 tweets for HaspeeDe2 - that we called devtset and remodulated the n-gram model accordingly by subtracting the n-grams related to the same sequence of tweets.

For HaSpeeDe2 the system produced 23,000 n-grams for the training corpus and 19,738 for the development. The development set is made of 1,000 tweets held out from the total 6839 which adds up to 136,536 tokens.

For SardiStance, we have 4,993 n-grams from the training corpus and 4,003 for the development: the development set is made of 427 tweets held out from the total 2,132 tweets, adding up to 57,774 tokens.

The system takes as input the analysis of one tweet at a time. In the mixed semantic-statistic module, the multilevel evaluation process consists of four steps which take advantage of the following previously compiled analyses. We have a full-fledged semantic analysis at propositional level; a trivalued labeling of each word-lemma by lexically-driven sentiment dictionaries; a six slot analysis of ironic/sarcastic contents at tweet level; a model for emoticons; a list of special hashtags inducing a direct evaluation. This is what we use in the semantic-only approach. The evaluation process is performed recursively for each tweet, and starts by searching for presence of Emoticons extracted in the previous analysis and organized in a model: in this case, the decision is taken by majority vote based on the type of emoticons present in the tweet. As for the semantic-only module, the problem was how to select best candidate from the pool of model n-grams with different value labels. We solved this problem by a scoring procedure. We produced two levels of scoring: a first one based on the number of sentiment labels with positive/negative value producing as a score a ratio of the total number divided by total number of words in the n-gram. Negative words are valued the double. The second scoring analysis is based on the contents of the propositional level semantics: here we associate 0.25 for each proposition marked differently from statement; another 0.25 is added for presence of predicates different from "dummy" verb ESSERE; eventually another 0.25 is added in case one of the arguments or attributes is shared with the input n-grams.

Eventually, we imposed coincidence at the level of Discourse Class associated to the utterance. We use seven different labels: statement, question, exclamation, negated, unreal, opinionsubjective, conditional.

4.1 Creating and Accessing N-grams models

If the semantics-only method needs just words from the two tweets to be evaluated by means of linguistic parameters, the two other methods or approaches we used are based on n-gram models which introduce a great number of variables. First of all, our n-gram model are organized in a different manner from the way in which they are usually conceived, so that their usage is also peculiar and needs detailed explanation. N-grams are not collected randomly by recursively creating bigrams and trigrams.

We can define three phases in the processing of our n-gram models: phase 1, building; phase 2,

---

[3] Emily Bender at a meeting in Uppsala University organized by Joakim Nivre.

[4] Rather than using actual wordforms we could use the rank number associated to each type in the dictionary as would be done in current machine learning approaches. But given the size of the training corpus we did not think it would be necessary: the model for the SardiStance task takes just 5Mb of memory and the one for Absita 10Mb.

choosing; phase 3, evaluating. We will clarify each phase in details below.

Phase 1. Building fully supervised n-gram models

As explained above, we collect topic words from unique dictionary derived from the training set. Topic words are the key entry in the n-gram, in that n-grams are built from each tweet around topic words. There two constraints at the basis of each n-gram: one is content related and the other is quantity related. The quantity constraint requires each n-gram to be longer than 3 words in sequence, in addition to the topic word. The content constraint requires that each n-gram must have at least a topic word at the beginning or end of the sequence of words. That is, each n-gram has a topic word as head or as tail. N-grams are strictly conditioned by the length of the tweet from which they are extracted. Short tweets may have only one n-gram at most or none. Long tweets may have two or more n-grams depending on their content: they would be all contained in the same list headed by the sum KLD index for that tweet. N-grams can be expressed in actual words or in lemmata. In the latter case, words are no longer available to subsequent analysis. We organized models with both words and lemmata. Every n-gram comes with the class attributed to the tweet in which they were contained.

Phase 2. Choice constraints on n-grams

Thus n-grams are each associated to two KLD indices, one for each word, and another one from the lump sum - which is unique - of all the words indices contained in the tweet. In this way, n-grams coming from the same tweet can be easily identified and this information can be used to select sequences of n-grams. Sequences of n-grams when matched with the input tweet are used to reinforce the similarity hypothesis. Choosing n-grams from the model is basically done on the basis of the ratio of intersecting words/lemmata. We established different ratios: one fifth or 20% of intersection, one fourth or 25%, one third or 30% and finally half or 50% intersecting words/lemmata. The ratio may vary according to another important parameter which is tied to the way in which the n-gram is used. We can decide to use words, lemmata, but also to erase grammatical or function words. In case we erase function words in the intersection only content words will be computed, which is a much smaller number and requires a smaller ratio to compare. We tried all three choosing manners.

Phase 3. Evaluating n-gram candidates

Once the methods have been selected and candidate n-grams are extracted from the model according to choice constraints, the outcome may be just one candidate and the evaluation stops or more than one candidate which is the rule. Now we have a list of candidate n-grams with the best ones at the top. The list may be created in a number of different manners. It has the KLD index inherited from the tweet and three other indices: one is the ratio of intersection words/lemmata, the higher this ratio the more relevant is the n-gram. Another index is the sum of the KLD indices associated to each of its word/lemma, the lower this sum the more relevant is the ngram (rare content words have a lower KLD index). Finally the third index is the one associated to the tweet in which the n-grams are contained. Choosing the best candidate in fact usually means selecting the best candidates from the list, because it almost never happens that there is only one candidate at the top with the best ratio or best index. The choice requires collecting candidates at the top with the same ratio/index. However this may require another step since the best candidates may be associated to different classes. So that after the first sieve has reduced the number of best candidates, another sieve requires selecting the most frequent class and this is done by reordering the best candidates on the basis of their class. In fact, this might also be one possible general method: rather that selecting only best candidates, one might reorder all candidates chosen on the basis of the intersection ratio, and count and choose the most frequent class. Eventually, another evaluation modality can be derived from the KLD indices. We compute differences on the basis of the KLD sum index for each model n-gram compared to the input n-gram and use this difference as the relevant index. When candidates are sorted in a list, the top will be populated by the lowest indices which can be used to characterize similarity. We chose the class of the top n-gram, but also tried a best way by selecting the first n-gram carrying a non negative index. Negative sums may still indicate higher differences between two n-grams.

Thus overall we come up with 6 different methods multiplied by two (function words erased/all words/lemmata), which amounts to 12 different methods. We experimented them all but at the end we concentrated only on a few.Since it is reasonable to assume that not all tweets of the training set will be classified in the model due to the lack of an instance defined by the list of automatically derived keywords in the training corpus, we ascertained at first what was the coverage of the training text for the development set, using in this case the model for the training set.

We report here below both training set coverage of the development set and development set results for both tasks. As can be easily noticed, coverage for the semantic-statistic module is poor, and the same applies for the so-called lexical-semantic module, which is even worse and as said above we only used as default.

	Cover Sardines	Devel Sardines	Cover HaSpDe	Devel HaSpDe
Sem-Stat	57.98%	35.31%	58.34%	38.67%
Stat-Only	93.91%	39%	92.6%	44.8%
Lex-Sem		39.34%		37.8%

Table 4. Coverage and Results for the Development Set for both Tasks

5 Results and Discussion

We present at first results of the SardiStance task where .

Task SardiStance

Run1 (Semantic module)
Macro-F1 0.3881500114277561

Run2 (Statistic module)
Macro-F1 0.3637025029179095

We then performed additional runs with the statistical module always with the test set. However we were unable to know the results until the evaluation script and the Test Gold set were distributed to all participants. We then realized that we had one run with the worst result and another with the best result. The former run was by choosing the first candidate from the list proposed by the KDL indices with a positive value in the list of candidated produced by a difference computed between the index of testset n-grams and the index of trainmodel n-grams. The latter run was instead obtained by choosing the best candidate – the one with higher value in terms of number of shared words from the intersection at word level between testset n-gram and trainmodel n-gram, and got the following results:

Task SardiStance

Run3 (Statistic module – first candidate with positive value)
Macro-F1 0.299607934

Run4 (Statistic module – higher word intersection)
Macro-F1 0.427668958

Even considering the last fourth run our ranking would not change. We assume that basing the evaluation on one n-gram alone is not the best solution. So we modified our evaluation procedure by requiring a sequence of at least two n-grams for each tweet/news to be chosen at the same time, using the same tweet-related KLD to select them. In this case we were able to cover more text and get a better similarity measure that we report in the subsection below.

We present here below the official results obtained at first for the HaSpeeDe2 task A-B both for News and for Tweets, and then the results obtained for SardiStance. Consider that we could report results for two runs only, and we choose the Semantic-Statistic and the Statiscs-Only.

Task HaspeeDe2 - News

Task A	
RUN-1	RUN-2
Macro-F1: 0.5024333	Macro-F1: 0.3805618
Task B	
RUN-1	RUN-2
Macro-F1: 0.5386702	Macro-F1: 0.3671441

Task HaspeeDe2 - Tweets

Task A	
RUN-1	RUN-2
Macro-F1: 0.5054034	Macro-F1: 0.4726022
Task B	
RUN-1	RUN-2
Macro-F1: 0.5078902	Macro-F1: 0.4671661

As for the SardiStance tasks, results for the HaSpeeDe2 task, obtained and delivered in due time are not particularly satisfactory, even though they are in line to results obtained for the development set. As for the SardiStance tasks, in fact there is a remarkable difference from the result obtained for the Development set in the Semantic-statistic.

5.1 The Improvements in the Statistical Module

After receiving the Test Gold version and the evaluation script, we continued producing other runs on the basis of the statistical approach and the choice of the algorithm we had available, for instance restricting choice of candidates only to those in which two or more n-grams had been selected. We discovered that in one these additional runs - the fifth for SardiStance and the sixth for HaSpeeDe2- we improved up to 54% macro-F1 for HaSpeeDe2 task and up to 48% macro-F1 for SardiStance. Here below the results for SardiStance and further the ones for HaSpeeDe2.

Run-5 SardiStance Task

Macro F1	**0.484871151**

Run-6 HaSpeeDe2 Task News

Task A	***Task B***
Macro-F1: 0.53828428	Macro-F1: 0.54071432

Run-6 HaSpeeDe2 Task Tweets

Task A	***Task B***
Macro-F1: 0.52836397	Macro-F1: 0.53965935

6 Conclusion

In this paper we presented the system we used for the two tasks HaSpeeDe2 and SardiStance. We used different approaches one of which was based on previous participation in similar Evalita tasks. Two methods are however innovative in their use of fully supervised n-grams, automatically derived. We use statistical measure to classify n-grams and a variety of different possible solutions which we explain in detail. The high number of possible results are however only evaluated against the development set. We are convinced that participants to these tasks which are mainly directed to the use of commonly available machine-learning software - should be allowed to propose a higher number of runs due to the variability of behaviour of the algorithm when relevant parameters in statistical tools are modified.

References

Basile, Valerio and Croce, Danilo and Di Maro, Maria, and Passaro, Lucia C., 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian, in Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020), CEUR.org.

Cignarella, Alessandra Teresa and Lai, Mirko and Bosco, Cristina and Patti, Viviana and Rosso, Paolo, 2020. Overview of the EVALITA 2020 Task on Stance Detection in Italian Tweets (SardiStance), in Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020), CEUR.org.

Delmonte R., 2014. ITGETARUNS A Linguistic Rule-Based System for Pragmatic Text Processing, Proceedings of Fourth International Workshop EVALITA 2014, Pisa, Edizioni PLUS, Pisa University Press, vol. 2, pp. 64-69.

Gries, Stefan Th. 2008. Dispersions and adjusted frequencies in corpora. International Journal of Corpus Linguistics 13/4: 403-437.

Gries, Stefan Th. 2010. Dispersions and adjusted frequencies in corpora: further explorations. In Stefan Th. Gries, Stefanie Wulff, and Mark Davies eds. Corpus linguistic applications: current studies, new directions. Amsterdam: Rodopi, 197-212.

Koos van der Wilt, Linguistics improves statistical classification: the positive effects of reducing feature dimensionality or grammatical feature selection. Downloadable at https://www.academia.edu/27207951/Linguistics_improves_statistical_classification_with_KLD_NB_TF_IDF_K_NN_the_positive_effects_of_reducing_feature_dimensionality_or_grammatical_feature_ selection_ Koos_van_der_Wilt.

Sanguinetti, Manuela and Comandini, Gloria, and Di Nuovo, Elisa and Frenda, Simona and Stranisci, Marco and Bosco, Cristina and Caselli, Tommaso and Patti, Viviana and Russo, Irene, 2020. Overview of the EVALITA 2020 Second Hate Speech Detection Task (HaSpeeDe 2), in Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020), CEUR.org.

Stingo M., & R. Delmonte, 2016. Annotating Satire in Italian Political Commentaries with Appraisal Theory, IN Larry Birnbaum, Octavian Popescu and Carlo Strapparava (eds.), Natural Language Processing meets Journalism - Proceedings of the Workshop, NLPMJ-2016, PP. 74-79.

UR_NLP @ HaSpeeDe 2 at EVALITA 2020: Towards Robust Hate Speech Detection with Contextual Embeddings

Julia Hoffmann
University of Regensburg
`Julia1.Hoffmann@ur.de`

Udo Kruschwitz
University of Regensburg
`Udo.Kruschwitz@ur.de`

Abstract

We describe our approach to address Task A of the EVALITA 2020 Hate Speech Detection (HaSpeeDe2) challenge. We submitted two runs that are both based on contextual embeddings – which we had chosen due to their effectiveness in solving a wide range of NLP problems. For our baseline run we use stacked embeddings that serve as features in a linear SVM. Our second run is a simple ensemble approach of three SVMs with majority voting. Both approaches outperform the official baselines by a large margin, and the ensemble classifier in particular demonstrates robust performance on different types of test data coming 6th (out of 27 runs) for news headlines and 10th (out of 27) for Twitter feeds.

1 Introduction

Hate speech in social media (and its automatic detection) has become a major problem in recent years. It can be generically defined as *"language that is used to express hatred towards a targeted group or is intended to be derogatory, to humiliate, or to insult the members of the group"* (Davidson et al., 2017) and is often based on aspects like race, religion, ethnicity, and gender. The problem is that what is considered acceptable for some might not be for others. In addition to that, there is a fine line between freedom of expression on the one hand and censorship and illegal discrimination on the other (Zimmerman et al., 2018). In fact, this fine balance is reflected by the fundamental human rights (as outlined in articles 19 and 20 of (The United Nations, 1948) and (The United Nations General Assembly, 1966) which simultaneously provide rights to freedom of expression and prevent censorship and illegal discrimination. All this contributes to making automatically detecting hate speech a challenging task.

Nevertheless, social media platforms such as Twitter have defined clear guidelines prohibiting the use of hateful behaviour.[1] Accounts with such contents can be reported and are subsequently deleted. The challenge is to be able to detect such content automatically with both high precision and high recall.

The EVALITA evaluation campaign introduced a hate speech detection challenge applied to Italian social media in 2018 (Bosco et al., 2018). Its success led to the continuation of the challenge in 2020, now called HaSpeeDe 2, which is split up into three subtasks (Sanguinetti et al., 2020). This report discusses our two runs that we submitted to HaSpeeDe 2 Task A of EVALITA 2020 (Basile et al., 2020). We will first give some background on the problem aimed at motivating our choice of approach. We will then introduce our systems, report results and discuss some findings. We will also outline some scope for future developments.

2 Background

We will provide some background that should motivate the system architectures we developed. There are several aspects to be mentioned here.

First of all, given the impressive advances in a broad range of natural language processing tasks using a transformer-based architecture (Vaswani et al., 2017) capturing contextual embeddings – most prominently utilizing the various flavours of BERT (Devlin et al., 2019) – we decided to adopt a transformer architecture as well. There are two ways language models such as BERT could be used – using pre-training and fine-tuning or just feature-based without fine-tuning.

---

[1] https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy

This leads us to the next design decision. The winning team in the 2018 HaSpeeDe competition, ItaliaNLP, submitted as one of their runs a SVM with three different feature categories, namely raw and lexical text, morpho-syntactic and lexicon features, which performed extremely well in particular when trained and tested on Twitter data (Cimino et al., 2018). Rather than designing an end-to-end neural architecture that would be fine-tuned on the available training data we therefore opted for a simpler and slightly more transparent architecture with an SVM backbone as our classifier, i.e. the feature-based approach mentioned above.

Ensemble methods have repeatedly been shown to outperform individual classifiers for a variety of tasks including hate speech detection. For example, an ensemble of ten simple neural classifiers proposed by (Zimmerman et al., 2018) outperformed a BERT-based approach on the standard HatebaseTwitter benchmark dataset (MacAvaney et al., 2019). Other recent examples that demonstrate the effectiveness of ensemble methods for hate speech detection include (Alonso et al., 2020; Nourbakhsh et al., 2019; Seganti et al., 2019; Zampieri et al., 2020; Badjatiya et al., 2017; Park and Fung, 2017). We should add that these findings are not limited to the area of hate speech detection as ensemble methods have a long history in being successfully utilized in a broad range of machine learning approaches, e.g. (Molteni et al., 1996). Simple but effective ensemble approaches have also been used for example in sentiment classification of tweets, e.g. (Hagen et al., 2015), and other social media classification tasks.

Finally, given the task definition in which the classifier was to be trained on social media data but then tested on both social media and news headlines we were aiming at an approach that would have a robust performance across domains rather than being tailored specifically to one type of data.

One additional motivation for our work is the intention to develop approaches that can be applied to different languages (we will get back to that point when we outline future directions).

We will now demonstrate how those motivating considerations lead to the system architecture we propose.

3 System Architecture

We submitted two runs of which the first one can be considered our own baseline approach. We first present both architectures at a conceptual level and will go into the technical details when we discuss the experimental setup in the next section. Our runs are:

- Model 1: *Stacked embeddings as features of a linear SVM*
- Model 2: *Ensemble of several SVMs with different text representations* – both contextual embeddings and TF-IDF-based.

Both models can be realised in many different ways. The core idea, as motivated before, is to experiment with transformer-based contextual embeddings but to avoid fine-tuning and instead deploy a traditional, more transparent approach of SVM. The ensemble can consist of a variety of different systems that can be aggregated in many ways. In this paper (and as submitted) we treat each system as equally important and use a simple majority vote.

Stacked embeddings have been shown to be effective in NLP applications, e.g. (Akbik et al., 2018; Akbik et al., 2019). Conceptually there is some similarity to ensemble approaches in that a combination of differently derived embedding models turns out to be more effective than each approach individually.

3.1 Model 1: Stacked embeddings + SVM

Our own baseline model combines two different document embeddings: transformer document and document pool embeddings which are then fed into a linear SVM to train a classifier. We keep the architecture deliberately simple.

There is a wide range of transformer-based language models. One of our motivations was to train a classifier that will generalise beyond a specific domain but also has the potential to generalise beyond a specific language. We therefore opted for XLM-RoBERTa (XLM-R) that has been shown to outperform alternative multilingual models such as mBERT in various NLP tasks (Conneau et al., 2020). XLM-R is based on XLM and RoBERTa. It is trained on data covering 100 languages in a very large (2TB) CommonCrawl. Transformer document embeddings are obtained from (the large version of) XLM-R. In addition to that we use document pool embeddings which consist of word embeddings using Flair (Akbik et al., 2019). The exact experimental choices are described further down.

3.2 Model 2: Ensemble of SVMs

Our second system is an ensemble classifier consisting of three SVMs each trained on a different text representation, namely:

- Transformer document embeddings using XLM-R
- Document pool embeddings
- Straightforward TF-IDF.

The first two of these are exactly the same as we have seen in Model 1 except that they are not stacked but fed into different classifiers. Again we observe that the general setup is kept simple to avoid overfitting for the specific problem at hand thereby allowing more scope for future experiments.

4 Experimental Setup

We applied our systems to *Task A - Hate Speech Detection (Main Task)*.

4.1 Data Sets

Training and test data is briefly described here.

- *Training Data Set*: the training data set consists of 6,839 tweets in total, 2,766 of them classified as hate speech. The corpus has three columns: tweet ID, text and the label (0 = no hate speech, 1 = hate speech). Table 1 summarises these numbers.

Label	Training Data Set
0	4,073
1	2,766
Total	6,839

Table 1: Training Data

- *Test Data Set*: unlike training data which was all Twitter feeds, there were two sets of test data, the first one sampled from Twitter and the second one from news headlines. The Twitter test set has 1,263 entries in total, the news test set 500. The two columns in both sets are the ID and the text of the tweet and news headlines, respectively. The classes 0 and 1 in the Twitter test set include 641 and 622 tweets respectively. In the news headline test set 319 entries have the label 0, 181 the label 1 (see Table 2).

Label	Twitter Test Set	News Test Set
0	641	319
1	622	181
Total	1,263	500

Table 2: Test Data

4.2 Data Preprocessing

In line with our overall aim of simplicity and generalisibility (rather than tuning) we applied a simple pre-processing pipeline that would apply to both Twitter data as well as news headlines. There are only small variations in the different normalization steps as follows.

For any embedding-based processing the text was lower-cased and punctuation was removed so that any input, be it tweet or news headline, would be represented as a string of unpunctuated tokens. For the calculation of our (sparse) TF-IDF representation the text was tokenized and in addition to that stopwords were removed. After that each token was vectorized using TF-IDF. Figure 1 shows an overview of the preprocessing.

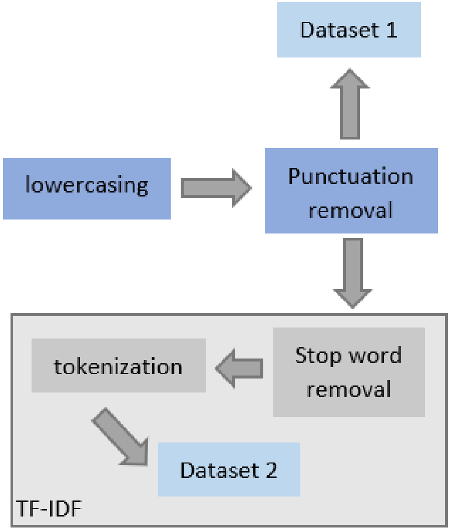


Figure 1: Data Preprocessing

4.3 Implementation

All implementation was done in Python. For all text and document embeddings we used *flairNLP*[2]. Our SVMs were developed using *scikit-learn* (Pedregosa et al., 2011), and for the preprocessing of the TF-IDF version and TF-IDF calculation we used *NLTK*[3] and *scikit-learn*.

Stacked embeddings + SVM: as outlined, we use stacked embeddings composed of *Transformer Document* and *Document Pool Embeddings*. The Transformer Document Embeddings are obtained using XLM-R. Document Pool Embeddings are calculated using a mean-pooling over all word embeddings. It consists of forward and backward embeddings for the Italian language as provided by flair (Akbik et al., 2018) and as recommended. An overview is given in Figure 2.

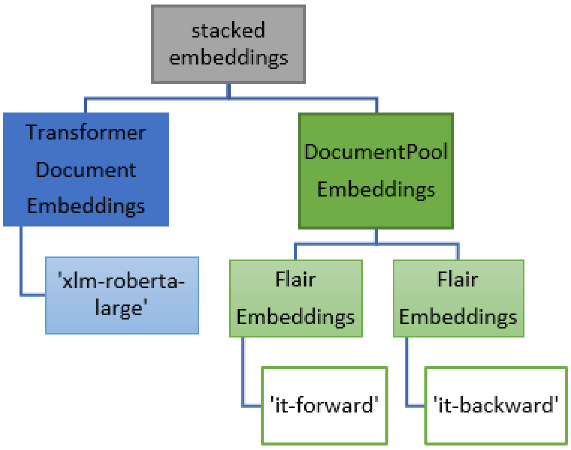


Figure 2: Embeddings in our Baseline (Model 1)

Flair allows for the easy combination of embeddings to create stacked embeddings – one for each input text. These vectors (together with the labels) are then used to train the SVM. Using grid-search on the training data the most suitable parameter settings were determined, and Table 3 specifies the settings which were then used in the submitted run.

Parameter	**Value**
C	1.0
kernel	'linear'
degree	3
gamma	1

Table 3: Parameters of the SVM (Baseline)

Ensemble of SVMs: three different feature representations are used to train one SVM each as illustrated in Table 4. The first two incorporate the same representations as already seen in Figure 2.

Classifier	**Features**
SVM2.1	Transformer Document Embeddings
SVM2.2	Document Pool Embeddings
SVM2.3	TF-IDF

Table 4: Overview of SVM Ensemble

Again we used grid-search for parameter tuning (see Table 5).

Parameter	**SVM2.1**	**SVM2.2**	**SVM2.3**
C	1.0	1.0	1.0
kernel	'linear'	'linear'	'rbf'
degree	3	3	3
gamma	1	1	1

Table 5: Parameters of the SVMs for Model 2 (Ensemble of SVMs)

Input is run against each classifier, and through majority voting over these three predictions the final classification category is determined.

5 Results

We first present detailed results and then discuss our findings and insights. We start with our baseline approach and then move on to the classifier ensemble. Macro-F1 is the official metric for this competition. In addition to that we look at Precision, Recall and F1 at category-level and also include confusion matrices for each approach (Model 1 and Model 2) and test set (Twitter data and news headlines). There were 27 runs submitted for each dataset and the official baseline was a linear SVM with TF-IDF of word and char-grams.

5.1 Model 1: Our Baseline

Twitter Data: Training and testing on Twitter data results in a Macro-F1 score of 0.7399 which makes it into position 16 (out of 27). The official task baseline is 0.7212. Details are displayed in Table 6 and Figure 3.

News Headlines: On the news headlines test data we get a Macro-F1 of 0.6684 with official baseline result of 0.6210 (rank 12). More details are in Table 7 and Figure 4.

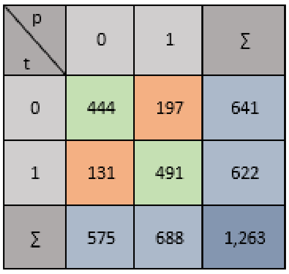
---

[2] https://github.com/flairNLP/flair

[3] https://www.nltk.org

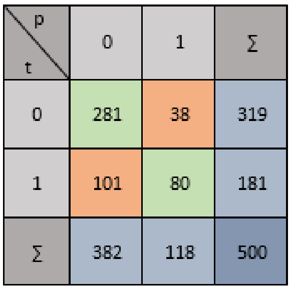
Metric	**0**	**1**
Precision	0.7722	0.7137
Recall	0.6927	0.7894
F1	0.7303	0.7496

Table 6: Results: Model 1 (Stacked embeddings + SVM) on Twitter Data



p \ t	0	1	Σ
0	444	197	641
1	131	491	622
Σ	575	688	1,263

Figure 3: Confusion Matrix: Model 1 (Stacked embeddings + SVM) on Twitter Data (p = predicted, t = true)

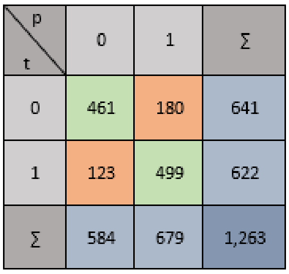


p \ t	0	1	Σ
0	281	38	319
1	101	80	181
Σ	382	118	500

Figure 4: Confusion Matrix: Model 1 (Stacked embeddings + SVM) on News Data (p = predicted, t = true)

5.2 Model 2: Ensemble

Twitter Data: Our ensemble approach gets a Macro-F1 of 0.7599 (rank 10). More details are included in Table 8 and Figure 5.



p \ t	0	1	Σ
0	461	180	641
1	123	499	622
Σ	584	679	1,263

Figure 5: Confusion Matrix: Model 2 (Ensemble of SVMs) on Twitter Data (p = predicted, t = true)

Metric	**0**	**1**
Precision	0.7356	0.6780
Recall	0.8809	0.4420
F1	0.8017	0.5351

Table 7: Results: Model 1 (Stacked embeddings + SVM) on News Data

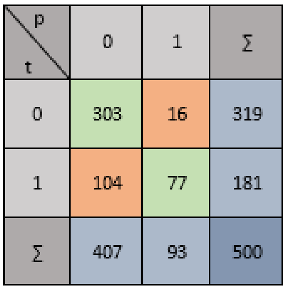
Metric	**0**	**1**
Precision	0.7894	0.7349
Recall	0.7192	0.8023
F1	0.7527	0.7671

Table 8: Results: Model 2 (Ensemble of SVMs) on Twitter Data

News Headlines: On the news headlines test data we get a Macro-F1 of 0.6984 with an official baseline result of 0.6210 (rank 6). More details can be found in Table 9 and Figure 6.

Metric	**0**	**1**
Precision	0.7445	0.8280
Recall	0.9498	0.4254
F1	0.8347	0.5620

Table 9: Results: Model 2 (Ensemble of SVMs) on News Data



p \ t	0	1	Σ
0	303	16	319
1	104	77	181
Σ	407	93	500

Figure 6: Confusion Matrix: Model for 2 (Ensemble of SVMs) on News Data (p = predicted, t = true)

6 Discussion

Our first observation we derive from the results is that the ensemble approach we proposed for this task does provide a robust and solid performance – solid in that it scores well in the ranked list of systems and robust in that it also ranks highly when applied to out-of-domain data (coming 6th out of 27 submitted runs on data it had not been trained on). Given the simplicity of our system architecture and the composition of the official baseline system we also note the superiority of transformer-based contextual embeddings over bag-of-words approaches (while this comes as no surprise it is still worth pointing out). Moving from a feature-based to a pre-training plus fine-tuning approach will most certainly further push up the scores.

Looking at the balance between precision and recall, we find that both our approaches have a tendency to return a fair number of *false positives for the Twitter data* set. This could indicate that words and phrases used to express hateful content is quite common in social media even if it does not actually represent hate speech. On the other hand, we record a large proportion of *false negatives* when classifying *news headlines*. This could be an indicator of a more subtle way in which hate speech is expressed in traditional news outlets.

Generally speaking, both models perform better on Twitter data than on news headlines – again an insight that was to be expected due to the training data. However, the fact that our approach managed to score higher in the ranked list of systems for data it was not trained on is a result that confirms our initial assumptions – that using a corpus with a very broad range of topics, styles and languages as our core language model would help in making the system transfer more easily to unseen input.

This leads us to an area of future research. While it would be possible to improve the performance of our system by making the preprocessing, the language model and any fine-tuning step match more closely the expected test data – e.g. by using AlBERTo, a BERT-based transformer trained on Italian Twitter data (Polignano et al., 2019) – we are actually aiming at something else. As part of the COURAGE research project[4] we are exploring ways to help teenagers manage social media exposure by providing a virtual companion that would, among other things, automatically identify examples of hate speech, bullying or other toxic content. Given this is a multi-national effort we are interested in architectures that work for languages including Italian, Spanish, German and English with as little fine-tuning as possible. The ensemble introduced here with its multilingual transformer backbone turns out to be a step in that direction.

7 Conclusion

We presented a simple but effective architecture to detect hate speech in Italian social media and news headlines. Our ensemble-based architecture relies on contextual embeddings trained on a large multilingual corpus which we see as the basis for the robustness of the approach. There is plenty of room for further improvement and the results we report here will serve as a benchmark in this development.

Acknowledgements

This work was supported by the project *COURAGE: A Social Media Companion Safeguarding and Educating Students* funded by the Volkswagen Foundation, grant number 95564.

References

A. Akbik, D. Blythe, and R. Vollgraf. 2018. Contextual string embeddings for sequence labeling. In E. M. Bender, L. Derczynski, and P. Isabelle, editors, *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1638–1649. Association for Computational Linguistics.

A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Demonstrations*, pages 54–59, Minneapolis, Minnesota, June. Association for Computational Linguistics.

P. Alonso, R. Saini, and G. Kovács. 2020. Hate Speech Detection Using Transformer Ensembles on the HASOC Dataset. In A. Karpov and R. Potapova, editors, *Speech and Computer*, pages 13–21, Cham. Springer International Publishing.

P. Badjatiya, S. Gupta, M. Gupta, and V. Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760. International World Wide Web Conferences Steering Committee.

V. Basile, D. Croce, M. Di Maro, and L. C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

---

[4] https://www.upf.edu/web/courage

C. Bosco, F. Dell'Orletta, F. Poletto, M. Sanguinetti, and M. Tesconi. 2018. Overview of the EVALITA 2018 hate speech detection task. In T. Caselli, N. Novielli, V. Patti, and P. Rosso, editors, *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

A. Cimino, L. De Mattei, and F. Dell'Orletta. 2018. Multi-task learning in deep neural networks at EVALITA 2018. In T. Caselli, N. Novielli, V. Patti, and P. Rosso, editors, *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of ACL*, pages 8440–8451. Association for Computational Linguistics.

T. Davidson, D. Warmsley, M. W. Macy, and I. Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of ICWSM 2017, Montréal, Québec, Canada, May 15-18, 2017*, pages 512–515. AAAI Press.

J. Devlin, M. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

M. Hagen, M. Potthast, M. Büchner, and B. Stein. 2015. Webis: An ensemble for twitter sentiment detection. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 582–589.

S. MacAvaney, H. Yao, E. Yang, K. Russell, N. Goharian, and O. Frieder. 2019. Hate speech detection: Challenges and solutions. *PLoS ONE*, 14:1–16.

F. Molteni, R. Buizza, T. N Palmer, and T. Petroliagis. 1996. The ECMWF ensemble prediction system: Methodology and validation. *Quarterly Journal of the Royal Meteorological Society*, 122(529):73–119.

A. Nourbakhsh, F. Vermeer, G. Wiltvank, and R. van der Goot. 2019. sthruggle at SemEval-2019 task 5: An ensemble approach to hate speech detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 484–488, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

J. H. Park and P. Fung. 2017. One-step and two-step classification for abusive language detection on twitter. In *Proceedings of The First Workshop on Abusive Language Online*, pages 41–45. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

M. Polignano, P. Basile, M. de Gemmis, G. Semeraro, and V. Basile. 2019. Alberto: Italian BERT language understanding model for NLP challenging tasks based on tweets. In R. Bernardi, R. Navigli, and G. Semeraro, editors, *Proceedings of the Sixth Italian Conference on Computational Linguistics, Bari, Italy, November 13-15, 2019*, volume 2481 of *CEUR Workshop Proceedings*. CEUR-WS.org.

M. Sanguinetti, G. Comandini, E. Di Nuovo, S. Frenda, M. Stranisci, C. Bosco, T. Caselli, V. Patti, and I. Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

A. Seganti, H. Sobol, I. Orlova, H. Kim, J. Staniszewski, T. Krumholc, and K. Koziel. 2019. NLPR@SRPOL at SemEval-2019 task 6 and task 5: Linguistically enhanced deep learning offensive sentence classifier. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 712–721, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

The United Nations General Assembly. 1966. International covenant on civil and political rights. *Treaty Series*, 999:171, December.

The United Nations. 1948. *Universal Declaration of Human Rights*. The United Nations, December.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6000–6010, USA. Curran Associates Inc.

M. Zampieri, P. Nakov, S. Rosenthal, P. Atanasova, G. Karadzhov, H. Mubarak, L. Derczynski, Z. Pitenis, and Ç. Çöltekin. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). *CoRR*, abs/2006.07235.

S. Zimmerman, U. Kruschwitz, and C. Fox. 2018. Improving Hate Speech Detection with Deep Learning Ensembles. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Fontana-Unipi @ HaSpeeDe2: Ensemble of Transformers for the Hate Speech Task at Evalita

Michele Fontana
Dipartimento di Informatica
Università di Pisa
`m.fontana12@studenti.unipi.it`

Giuseppe Attardi
Dipartimento di Informatica
Università di Pisa
`attardi@di.unipi.it`

Abstract

We describe our approach and experiments to tackle Task A of the second edition of HaSpeeDe, within the Evalita 2020 evaluation campaign. The proposed model consists in an ensemble of classifiers built from three variants of a common neural architecture. Each classifier uses contextual representations from transformers trained on Italian texts, fine tuned on the training set of the challenge. We tested the proposed model on the two official test sets, the in-domain test set containing just tweets and the out-of-domain one including also news headlines. Our submissions ranked 4th on the tweets test set and 17th on the second test set.

1 Introduction

The spreading of hateful messages on social media has become a serious issue, therefore techniques of hate speech detection have become quite relevant. The goal of the Hate Speech Detection task (Sanguinetti et al., 2020) at Evalita 2020 (Basile et al., 2020) is to improve the automatic detection of hate messages in Italian tweets. The organizers provided to the participants the dataset HaSpeeDe2, which consists of 6,837 Italian tweets, containing, besides the raw text, also hashtags and emojis. The Task A can be cast into a binary classification task: the model has to predict whether a given message contains hate speech or not.

Approaches based on transformer models have become quite popular recently and have proved effective in reaching state-of-the-art scores on major NLP tasks such as those of the GLUE benchmark (Wang et al., 2018). With our experiments we try to assess the effectiveness of transformers trained on Italian documents in a task involving Italian texts from different sources. We experiments with both a transformer model trained specifically on Italian tweets and one trained on generic web documents.

We combine several instances of classifiers based on these transformers, in order to address the problem of over-fitting due to the small size of the training set.

For this edition of the Evalita HaSpeeDe task, the organizers released two test sets, an in-domain one consisting of tweets and an out-of-domain one containing also news headlines.

The ensemble model of our official submission achieved a competitive score of 78.03 Macro-F1 on the in-domain test set but did not perform as well on the second test set.

We make available the source code for our experiments as Open Source at `https://github.com/mikelefonty/Haspeede2`.

2 Related Work

The first edition of HaSpeeDe was held in 2018. The results produced during this contest were the starting point of our research. As described in (Bosco et al., 2018), most of the systems were based on neural networks and used word embeddings, such as FastText (Grave et al., 2018) or word2vec (Polignano and Basile, 2018) in the first layer of their architecture. The embeddings layer was usually followed by a Recurrent Network or a Convolutional Neural Network to get an internal representation of the input text. This hidden representation was provided as input to a series of dense layers to obtain the final classification result.

Over the last couple of years, the trend in approaches to language analysis has changed considerably, as can be seen by examining the models used in competitions like SemEval 2020 OffensE-

---

val 2 (Zampieri et al., 2020). In these new models, to get a better text representation, the embedding layer is often replaced by a Transformer (Vaswani et al., 2017) such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), or Multilingual BERT (Devlin et al., 2019).

We followed this trend but we also focused our attention on the problem raised by the small size of the dataset. As Risch and Krestel (2020) mention, transformer models tend to have a high variance with respect to the input dataset, that often leads to overfitting. The authors therefore suggest to implement an ensemble of classifiers to reduce the variance and consequently improve the generalization capabilities of the trained model.

In the following, we describe a similar approach based on the Bagging technique (Breiman, 1996), where we apply three different transformer-based classifiers to populate the ensemble and to get the final prediction.

3 System Architecture

During the design phase of our classifier, we looked for a transformer trained directly on a significantly large collection of Italian texts and particularly on Italian tweets, in order to compensate for the small size of the training data. We found two possible models based on BERT: AlBERTo (Polignano et al., 2019) [1] and DBMDZ [2]. The former is trained on TWITA (Basile et al., 2018), a 191 GB collection of Italian tweets gathered by the authors, and tested on the SENTIPOLC task during the EVALITA 2016 campaign, where it achieved state-of-the-art accuracy in subjectivity, polarity, and irony detection on Italian tweets. We considered this model suitable for hate speech detection, since its source are Italian tweets and the SENTIPOLC task is a classification task similar to ours. DBMDZ instead is trained on a more general domain, from a 13 GB dataset, which includes a dump of Italian Wikipedia and texts from web pages selected from the Opus Corpora. [3] We decided to test both transformer models, assessing their performance through a validation phase on a development set.

These transformers were used in the input stage of all our architectures, providing contextual embeddings for sentences that were fine tuned during training. We designed three architecture variants, which were employed as the basic building blocks to construct the ensembles:

- **ALB-SINGLE**: It consists of a first layer provided by the AlBERTo transformer, followed by a single neuron with a sigmoid activation function.
- **DB-SINGLE**: It follows the same structure of ALB-SINGLE; it just replaces AlBERTo with DBMDZ in the first layer.
- **DB-MLP**: Compared to DB-SINGLE, it adds a new dense layer, using a ReLU activation function, between the transformer and the output neuron.

The final model is an ensemble consisting of a number of instances of each of the above architectures. For each architecture, e.g. ALB-SINGLE, we construct instances in the following way. After initializing the weights randomly within a given interval and generating the training data by applying the bootstrap technique to the original dataset, we start training the model. When that phase is over, we insert the resulting model in the ensemble. We repeat this process several times with different random weights initialization. Note that, due to the random initialization, no two classifiers in the ensemble are identical to each other. More formally, the model consists of N elements,

$$N = N_{AL} + N_{DB} + N_{MLP}$$

where N_{AL}, N_{DB}, N_{MLP} represent, respectively, the number of instances of *ALB-SINGLE*, *DB-SINGLE* and *DB-MLP* classifiers.

In retrospect, it might have been worth while to consider instances of the architecture obtained varying them more thoroughly than just in the initial weights, for example, by changing in the hyper-parameters or number of layers.

Our classification algorithm is a slight generalization of the most classical one, which collects results from each member of the ensemble and outputs the class which gets the majority of predictions over all iterations. The process, described by Algorithm 1, performs n_{run} iterations. During the ith iteration, the algorithm starts sampling randomly from the ensemble a given number of instances for each type of classifier (line 3-5) and initializing to 0 the variable $class1$, which contains the total number of votes that the $hate$ class

---

[1] https://github.com/marcopoli/AlBERTo-it

[2] https://huggingface.co/dbmdz/bert-base-italian-uncase

[3] http://opus.nlpl.eu/

Algorithm 1 Classification Algorithm

Input: t: the tweet to classify.
Input: $(n_{AL}, n_{DB}, n_{MLP})$: number of classifiers of each type to be sampled.
Input: $(N_{AL}, N_{DB}, N_{MLP})$: number of classifiers of each type in the ensemble.
Input: n_{run}: number of desired iterations.
Output: c_{final}: predicted class

```
 1: $preds = []$
 2: for $run = 1$ to $n_{run}$ do
 3:     $albs = sample\_al(n_{AL}, N_{AL})$
 4:     $dbs = sample\_db(n_{DB}, N_{DB})$
 5:     $mlps = sample\_ml(n_{MLP}, N_{MLP})$
 6:     $sampled\_classif = albs \cup dbs \cup mlps$
 7:     $class1 = 0$ // votes for class 1
 8:     for $cl$ in $sampled\_classif$ do
 9:         $class1 \mathrel{+}= cl(t)$ // $cl$'s classification
10:     end for
11:     $preds[run] =$
            $\left(class1 \geq \lceil \frac{n_{AL}+n_{DB}+n_{MLP}}{2} \rceil\right)$
12: end for
13: $c_{final} = \left[\left(\sum_{i}^{n_{run}} pred[i]\right) \geq \lceil \frac{n_{run}}{2} \rceil\right]$
14: return $c_{final}$
```

receives during the iteration (line 7). It then collects the predictions of the selected models on the tweet t (line 8-10). $cl(t) \in \{0, 1\}$ represents the prediction of classifier cl for the tweet t; in particular $cl(t) = 1$ if and only if cl classifies t as hateful. The output of iteration i is the most predicted class (line 11). The final result of the algorithm is then the class $c_{final} \in \{0, 1\}$, which obtained the most votes over all the n_{run} iterations (line 13-14). If $c_{final} = 1$, it means that the tweet t has been classified as hateful.

A simpler variant of the algorithm would be to just add the counts of each class by all classifiers in all iterations and return the class with the highest count. We plan to compare these two approaches in a future work.

4 Experiments

In this section we describe the experiments we performed to tune the hyper-parameters of our model. We will focus on the search to choose the best values for n_{DB}, n_{AL}, n_{MLP}, that is how many instances to select at each iteration in the classification algorithm.

Before starting the experiments, we divided the dataset into two disjoint subsets, a development and an internal test set, in the proportion of 80% and 20%, respectively. The split was done by means of Stratified Sampling, according to the distribution of the target variable hs. We applied the Stratified 3-fold-CV technique to validate our model. Given that we are solving a binary classification problem, we picked the Binary Cross Entropy as our loss. We chose AdamW as our optimizer; we set the first 10% of the total steps as warmup steps. We conducted the experiences on a GPU offered by Google Colab [4]. Our models are implemented in PyTorch (Paszke et al., 2019). To extract as much information as possible from input texts, we preprocessed them through hashtag segmentation by means of *Tweet Preprocessor*.[5] We also converted emojis into their Italian description by using the *emoji* [6] and *Google Translate* [7] libraries.

Classifier	Macro-F1	Std
ALB-SINGLE	76.896	0.7266
DB-SINGLE	77.613	0.3251
DB-MLP	78.562	0.521

Table 1: Results of the experiments comparing the baseline architectures. We report the expected value and the standard deviation of the F1 score computed with respect to the 3 validation folds.

We analyzed the behaviour of the three baseline architectures we planned to include in the ensemble.

We trained each model for a maximum of 4 epochs, using a batch of size 16 and setting the maximum text length to 100. A grid search revealed that the optimal learning rate for DB-MLP is $5 \cdot 10^{-5}$, and $6 \cdot 10^{-5}$ for the remaining models. The optimal number of neurons in the hidden layer of DB-MLP is 50.

Table 1 highlights the following aspect: DB-SINGLE achieves better performance than ALB-SINGLE, even though the dataset used to train AlBERTo was composed by a large collection of tweets. The obtained values of the macro-F1 are the baselines of our work.

We then describe the results obtained through

---

[4] https://colab.research.google.com/

[5] https://pypi.org/project/tweet-preprocessor/

[6] https://pypi.org/project/emoji

[7] https://pypi.org/project/googletrans/

n_{DB}	n_{MLP}	n_{AL}	Macro-F1	Std
20	**25**	**30**	**80.057**	**0.534**
15	20	25	80.038	0.580
15	30	30	80.036	0.585
15	25	30	80.026	0.563
15	30	15	80.020	0.481

Table 2: Ranking of the 5 best configurations we found, varying the number the number of instances selected from the ensemble. n_{DB} stands for the number of instances of the *DB-SINGLE* model, and similarly for n_{MLP} and n_{AL}. We report the expected value and the standard deviation of the F1 score computed with respect to the 3 validation folds.

n_{DB}	n_{MLP}	n_{AL}	Macro-F1	Std
30	0	0	79.074	0.300
0	30	0	79.581	0.3787
0	0	30	79.482	0.596
30	30	30	79.832	0.525

Table 3: Scores by each architecture, both individually and together in the ensemble. We report the average value and the standard deviation of the F1 score computed with respect to the 3 validation folds.

the ensemble model. To build the classifier, we trained 30 instances of each architecture, keeping the same hyper-parameters obtained from the previous grid search. We thus set:

$$N_{AL} = N_{DB} = N_{MLP} = 30$$

We noted that the generalization capability of the ensemble is strictly related to the triple $(n_{DB}, n_{MLP}, n_{AL})$, so we performed another grid search, looking for the optimal combination of the three parameters. Table 2 shows the five best configurations found by this search. The optimal values for the triple, $(20, 25, 30)$, allow the ensemble to achieve an F1-score of 80.0%, with a gain of about 2 points with respect to the score by a single DB-MLP (see Table 1).

We analyzed the contribution of each architecture individually to the ensemble combination. As shown in Table 3, the best results are obtained with instances of all three architectures. Nevertheless, the results presented in Table 2, show that a more balanced combination achieves better accuracy.

Accuracy	Precision	Recall	F1
79.313	78.510	78.685	78.592

Table 4: Results of the final model on the internal test set.

We picked the first configuration from Table 2 for our final model and tested it on the internal test set, obtaining the results shown in Table 4.

5 Results and Discussion

The results of our final model applied to the data of the two official test sets of the competition are shown in Table 5. The model performs pretty well on the in-domain dataset, reaching the $4th$ position in the rankings. However, it did not rank as well in detecting hate speech on the out-of-domain dataset, obtaining an F1-score of just 65.46. The low recall for the hate class highlights that the model fails too often to identify news headlines containing some form of hate speech. In comparison with the official top rankings, listed in Table 6, our model achieved about 12 points below the top score of 77.44% F1.

Surprised by this fact, we investigated more deeply, looking for an explanation for such poor result on the out-of-domain dataset.

We randomly sampled from the test set some hateful headlines missed by the model, some of which are shown in Table 7.

In these headlines, the qualification as hate is implicit and harder to recognize, since it seems due more to the presence of stereotypes (*nomads*, *asylum seekers*, *Muslims*, *foreigners*), than to the presence of explicit hate expressions.

Broadly speaking, we identified some possible reasons for the difference in performance across the two test sets:

- **Linguistic register**: Tweets often exhibit a more informal and colloquial language, while headlines employ a more formal lexicon and a more objective tone. This is a crucial difference in identifying hateful messages: while in tweets the feeling of hatred transpires clearly and directly, in headlines this message is conveyed in a more subtle way, often alluding to concepts from political propaganda or common stereotypes. Prior knowledge about the subject and inference might be necessary

	NOT HATE			HATE				
	Precision	Recall	F1	Precision	Recall	F1	Macro-F1	Position
Tweets	81.93	72.85	77.12	74.89	83.44	78.94	78.03	4
News	71.88	99.37	83.42	96.61	31.49	47.50	65.46	17

Table 5: Results of the submitted model on the official blind test sets.

Tweets		News	
Position	F1 score	Position	F1 score
1	80.88	1	77.44
2	78.97	2	73.14
3	78.93	3	72.56
4	**78.03 (ours)**	4	71.83
5	77.82	5	70.2
6	77.66	17	**65.46 (ours)**

Table 6: Comparison between our final results and the top-5 F1-scores. The values are taken from the official rankings.

Hateful News Headlines
anziana rapinata sull'autobus, i due nomadi in fuga si rifugiano al campo di via Candoni *(elderly woman robbed on the bus, the two fleeing nomads take refuge at the camp on via Candoni)*
Expo: Bordonali, richiedenti asilo in campo base simbolo fallimento governo. *(Expo: Bordonali, asylum seekers in base camp government failure symbol.)*
Il cardinale Müller: ”non possiamo pregare come o con i musulmani” *(”we cannot pray like nor with Muslims”)*
Salvini: ”Il calcio? Rimpiango i tre stranieri in campo” *(Salvini: ”Soccer? I regret the three foreigners on the field”)*

Table 7: Examples of hateful headlines, randomly picked from the out-of-domain test set, that are misclassified by our model.

to decipher the presence of hate. Examining the entire body of the article might have been helpful.

- **Length of text**: Tweets are usually longer than news headlines. Thus, the model has fewer elements to exploit to correctly classify a piece of news.

These difficulties seem to be shared with other submissions which all got lower scores on the out-of-domain dataset. We expected that pretrained contextual embedding would be more effective in addressing the domain adaptation issue. Further experiments would be needed to improve the resilience of our model.

6 Conclusions

We described an ensemble of neural classifiers, relying on contextual embeddings from transformers, for automated detection of hateful content in Italian texts. We presented the general architecture of our base classification models and how they were combined into an ensemble through a bagging technique. We performed extensive experiments to tune our models and the ensemble on a validation test set. The results achieved by our ensemble model on the in-domain test set confirm its ability in detecting hateful tweets; however the same model performed poorly on the out-of-domain dataset, showing particularly an inability to adapt to handling news headlines. We plan to investigate this issue in future research.

References

Valerio Basile, Mirko Lai, and Manuela Sanguinetti. 2018. Long-term social media data collection at the university of turin. In Elena Cabrio, Alessandro Mazzei, and Fabio Tamburini, editors, *Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Torino, Italy, December 10-12, 2018*, volume 2253 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and*

Speech Tools for Italian. Final Workshop (EVALITA 2020), Online. CEUR.org.

Cristina Bosco, Felice Dell'Orletta, Fabio Poletto, Manuela Sanguinetti, and Maurizio Tesconi. 2018. Overview of the EVALITA 2018 hate speech detection task. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

L. Breiman. 1996. Bagging predictors. *Machine Learning*, 24:123–140.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Marco Polignano and Pierpaolo Basile. 2018. Hansel: Italian hate speech detection through ensemble learning and deep neural networks. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019. Alberto: Italian BERT language understanding model for NLP challenging tasks based on tweets. In Raffaella Bernardi, Roberto Navigli, and Giovanni Semeraro, editors, *Proceedings of the Sixth Italian Conference on Computational Linguistics, Bari, Italy, November 13-15, 2019*, volume 2481 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Julian Risch and Ralf Krestel. 2020. Bagging BERT models for robust aggression identification. In Ritesh Kumar, Atul Kr. Ojha, Bornini Lahiri, Marcos Zampieri, Shervin Malmasi, Vanessa Murdock, and Daniel Kadar, editors, *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying, TRAC@LREC 2020, Marseille, France, May 2020*, pages 55–61. European Language Resources Association (ELRA).

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, November. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çagri Çöltekin. 2020. Semeval-2020 task 12: Multilingual offensive language identification in social media (offenseval 2020). *CoRR*, abs/2006.07235.

TheNorth @ HaSpeeDe 2: BERT-based Language Model Fine-tuning for Italian Hate Speech Detection

Eric Lavergne, Rajkumar Saini, György Kovács and Killian Murphy
Luleå Tekniska Universitet
`eric.lavergne@gmx.fr`
`rajkumar.saini@ltu.se`
`gyorgy.kovacs@ltu.se`
`killian.murphy@telecom-sudparis.eu`

Abstract

English. This report was written to describe the systems that were submitted by the team "TheNorth" for the HaSpeeDe 2 shared task organised within EVALITA 2020. To address the main task which is hate speech detection, we fine-tuned BERT-based models. We evaluated both multilingual and Italian language models trained with the data provided and additional data. We also studied the contributions of multitask learning considering both hate speech detection and stereotype detection tasks.

1 Introduction

Organised as part of the 7th EVALITA evaluation campaign (Basile et al., 2020), the HaSpeeDe 2 shared task focuses on the detection of online hate speech (Sanguinetti et al., 2020) in Italian-Hate speech occurs frequently on social media. It is defined as "any communication that disparages a person or a group on the basis of some characteristics such as race, colour, ethnicity, gender, sexual orientation, nationality, religion, or other characteristics" (Nockleby, 2000). Regulating all user messages is very time-consuming for a human, and this is one of the reasons why automatic methods are important.

Beside the main task of binary hate speech classification - aimed at deciding whether a message contains hate speech or not - the HaSpeeDe 2 shared task has two more sub-tasks. One being stereotype detection, and the other the identification of nominal utterances. All tasks being evaluated both on in-domain (tweets) data, and out-of-domain (newspaper headlines) data. Here, we tackle both the main task, and the first sub-task of Stereotype Detection that is potentially useful for the main task. For this sub-task the organisers use the following definition of Stereotype: "a standardized mental picture that is held in common by members of a group and that represents an oversimplified opinion, prejudiced attitude, or uncritical judgment" (Merriam-Webster, 2020).

Here, we have two binary classification tasks. A simple way to perform text classification is based on bag-of-words representation counting the number of occurrences of each word within text. It is often combined with term frequency-inverse document frequency (Sparck Jones, 1988) (TF-IDF) representation. TF-IDF allows the frequencies to be normalized according to how often the words appear in all documents. With the rise of neural networks, word vectors have provided useful features for text classification tasks. Recurrent Neural Networks as the Bidirectional Long-Short Term Memory (BiLSTM) network (Schuster and Paliwal, 1997) have then be used to encode the long-term dependencies between the words. These systems were the most successful in the previous HaSpeeDe campaign (Bosco et al., 2018).

In (Aluru et al., 2020), the authors showed that when dealing with very low monolingual resources, multilingual approaches can be interesting for hate speech. In (Polignano et al., 2019b), the AlBERTo monolingual Italian BERT-based language model was trained that outperformed the state-of-the-art on the HaSpeeDe 2018 evaluation task (Polignano et al., 2019a).

We have chosen to deepen the approach of fine-tuning a BERT based language model, comparing multilingual and monolingual settings. We also assessed the contribution of additional hate speech data from different online sources. We finally submitted the results of the same model fine-tuned with and without multitask learning between hate speech and stereotype detection tasks.

---

2 System Description

2.1 Fine-tuning process

The chosen classification approach is to fine-tune a BERT-based language model. This kind of approach is the state-of-the-art for many text classifications tasks today (Sun et al., 2019; Seganti et al., 2019). BERT is a language model which aims to learn the distribution of language (Devlin et al., 2018). It is trained with the prediction of masked tokens in a text. The next sentence prediction task that was used simultaneously for training has been removed for some later BERT-based models such as RoBERTa (Liu et al., 2019). BERT is a Transformer. In a Transformer, the recurrence of Recurrent Neural Networks is replaced by the mechanism of attention (Vaswani et al., 2017).

It has been shown that it is possible to fine-tune these models for many downstream natural language processing tasks, including the one we are interested in, which is text classification. This can be achieved by removing the language modelling head and replacing it by a head appropriate for the target task. The designers of BERT prepared this by adding a token at the beginning of each text sequence, named CLS for classification. The purpose of this token is to contain the information useful for the classification task at the end of the forwarding process. Then a classifier head can just take this CLS token as input to classify the whole text sequence. In our case we decided to add a simple linear layer with a softmax on top of it, for simplicity and because it is efficient enough since the other layers are fine-tuned.

2.2 Layer-wise learning rate

An important consideration of fine-tuning described in (Sun et al., 2019) is the choice of the learning rate. Besides being as usual the most important hyper-parameter in the gradient descent learning algorithm, it could also be responsible here for some catastrophic forgetting if it were too high. Catastrophic forgetting refers to the fact of erasing the information of the weights of the pretrained model and can happen when the gradient updates are too high.

Moreover, the learning rate can be gradually decreased in the first layers of the models. It aims at limiting the update in these first layers that have been showed to contain the most primal information about the language. One can think of the classical example in computer vision neural networks where the basics shapes features are extracted by the first layers and the task-specific combinations are processed in the last ones. Thus we applied layer-wise learning rate with the following geometric equation: the learning rate in a layer is the one of the following multiplied by a decay factor γ between 0 and 1.

$$LR_{k-1} = \gamma \times LR_k$$

where LR_k is the learning rate of the k-th layer.

Then the case when γ is one is the case of classic fine-tuning with the same learning rate everywhere, and the case when γ is zero is the case of feature extraction with the whole language model weights that are frozen and only the parameters of the classification head are trainable. This hyper-parameter γ was learned with the others during the hyper-parameters tuning process.

2.3 Monolingual and multilingual language models

We compared the use of several language models. Many models similar to BERT have been trained since 2018, and a lot are available for use. Although the models are often first and foremost trained for English, multilingual models have been trained on data of several languages in order to counteract the lack of data for some languages. It is the case of mBERT and XLM-Roberta (Conneau et al., 2020). Also machine learning researchers trained monolingual models for their own language, as CamemBERT for French and AlBERTo or UmBERTo for Italian. Multilingual models have the advantage that they are trainable on data in different languages; it is very useful for low-resources tasks. However, they are expected to perform in dozens of languages while monolingual models focus on just one, with the same number of parameters. For this reason, monolingual models often perform better when sufficient data is available, as we show here.

We evaluated two multilingual models, mBERT and XLM-RoBERTa, and three Italian monolingual models, AlBERTo, UmBERTo, and PoliBERT. AlBERTo was pretrained on TWITA, that is a collection of Italian tweets (Polignano et al., 2019b). UmBERTo was pretrained on Commoncrawl ITA exploiting OSCAR Italian large corpus (Parisi et al., 2020). Finally, PoliBERT was fine-tuned for sentiment analysis on Italian tweets by its creators (Barone, 2020).

We tried to use more data, with different settings. For the multilingual models, we could use all type of hate speech data. For the monolingual models, we used the little data available for Italian but we tried also to use translated multilingual data. These additions were not conclusive, so we stuck to the HaSpeeDe 2 data for the submissions.

2.4 Random search hyper-parameters tuning

The tuning of the hyper-parameters is relevant in order to get good results, and that is especially the case for the learning rate and the layer-wise decay factor γ. We tuned hyper-parameters with random search which has been shown to be often more efficient than grid-search (Bergstra and Bengio, 2012). The hyper-parameters to be tuned are the batch size, the learning rate, the layer-wise multiplier and the length of the model (maximum number of tokens). We did ten trials for each language model. The number of epochs is selected with early stopping on the validation macro F1-score with a split of 80/20. Table 1 shows the best hyper-parameters obtained that have been used for the systems submitted.

Hyper-parameter	**Value**
Learning rate	2.10-4
Layer-wise γ	0.35
Batch Size	32
Max Length	100
Language Model	UmBERTo

Table 1: Hyper-parameters used for our HaSpeeDe 2 submission after the tuning process

It is very important that the learning rate and the layer-wise multiplier γ are tuned simultaneously because the choice of the multiplier strongly modifies the amplitude of the gradient.

2.5 Multitask Learning

We evaluated the usage of multitask learning between the two classification tasks of the competition that are hate speech detection and stereotype detection. Multitask learning consists of learning to perform several tasks. It can be done by learning the tasks simultaneously with common first layers but task-specific heads (Ruder, 2017). In our case each task has its own output linear layer. When the tasks should be based on similar representations, it is supposed to do a good regularization with useful shared representations. It is then a kind of transfer learning. The error analysis conducted on HaSpeeDe 2018 evaluation suggests a significant correlation between the usage of stereotype and hate speech (Francesconi et al., 2019). Moreover, they showed that the false positive rate of hate speech tweets is slightly bigger for tweets with stereotype.

A question that arises when doing multitasking is the way to combine the loss of the tasks in one. The simple solution is to sum them uniformly. It might not be the best solution when there is imbalance between the tasks, for instance when the scale of the outputs of one is much higher than the others. A solution brought by (Kendall et al., 2017) is to use trainable weights based on uncertainty. (Liebel and Körner, 2018) upgrades the regularisation term of this solution and (Gong et al., 2019) shows in a benchmark that this last solution is often the best. We evaluated this solution and we compared with the single-task setting.

2.6 Cross-validation ensembling and submitted models

Two submissions are allowed during the HaSpeeDe 2 test phase. We chose to submit a fine-tuned UmBERTo trained separately for each of the two tasks and a fined-tuned UmBERTo with multitasking on both Stereotype and Hate Speech detection. The hyper-parameters used to train these models were presented in Table 1.

Since we compared the different language models with 5-fold cross-validation, we then ensembled the 5 models obtained for each fold in order to get the final model. The ensembling was done by considering the mean of the probabilities returned by each model.

3 Data Description

The organisers provided a train dataset of 6,839 tweets, annotated with Hate Speech and Stereotype labels (as described in Table 2).

Dataset	**HS**	**Ster**
Development Data (Tweets)	0.404	0.445
Test Data (Tweets)	0.492	0.450
Test Data (News)	0.362	0.350

Table 2: Distribution of Hate Speech and Stereotype labels in HaSpeeDe 2 data.

The test data of HaSpeeDe 2 consists of two subsets: an in-domain set (1,263 tweets) and an out-of-domain set (500 newspaper headlines).

The hate speech labels are slightly unbalanced towards non-hate speech. Thus we tried to use adapted losses to prevent tendency towards non-hate speech predictions. We used class-weighted loss, which assigns a higher weight to the observations from the minority class in the computing of the loss. We also tried to use a smoothed F1-score – a differentiable loss in phase with the F1. Neither approach improved the results in a significant way.

The pre-processing was simple. We removed emoticons and hashtags and we replaced urls and user names with associated tags as done in the evaluation data. Each tweet was padded with a size of 100. Then we used the pre-processing and tokenization pipeline specific to each language model as provided by the authors of the models.

4 Results

4.1 Macro F1-score

The metric used for the evaluation is the macro F1-score. The F1-score of a class is computed by calculating the harmonic mean between the precision and recall for this class. The macro F1-score is the mean between the F1-scores for each class. It is less sensitive to the imbalance between the classes.

4.2 Baselines

We used several baselines to evaluate our results during the development process. The first ones are those obtained by dummy classifiers, one that always predicts the most frequent class and the other one that makes a random stratified prediction according to the distribution of the classes in the training data. We also computed the results of more developed systems, that are a TF-IDF bag of words and a BiLSTM with trainable word vectors inputs.

The HaSpeeDe 2 organisers provided two baseline systems after the results were submitted. The first is a most frequent class predictor and the second is a linear SVM with unigrams, char-grams and TF-IDF representation.

4.3 Validation Results

We tuned the hyper-parameters for each evaluated language model as described in Section 2.4. For each language model, we then computed 5-fold cross-validation results on HaSpeeDe 2 training data. The averages of the 5 macro F1-scores are shown in Table 3.

System	**HS**	**Ster**
Baselines		
Most Frequent Class	0.374	0.353
TF-IDF Bag-of-words	0.703	0.677
Word vectors + BiLSTM	0.721	0.654
Multilingual language models		
mBERT	0.757	0.716
XLM-RoBERTa	0.761	0.677
Italian language models		
AlBERTo	0.773	0.716
PoliBERT	0.795	**0.733**
UmBERTo	**0.799**	**0.733**

Table 3: Macro F1-scores averaged over 5-fold cross-validation on HaSpeeDe 2 training data.

4.4 Test Results

The scores of our two systems evaluated on the HaSpeeDe 2 test data are summarized in Table 4. These systems are 5 UmBERTo models trained on each of the 5 training folds and ensembled. The second system is the same as the first with the use of multitask learning.

System	**Tweets**	**News**
Hate Speech Detection		
Most Frequent Class	0.337	0.389
Classic Features + SVM	0.721	0.621
UmBERTo	0.790	0.671
UmBERTo + Multitasking	**0.809**	0.660
Best HaSpeeDe 2	**0.809**	**0.774**
Stereotype Detection		
Most Frequent Class	0.355	0.394
Classic Features + SVM	0.715	0.669
UmBERTo	**0.772**	0.685
UmBERTo + Multitasking	0.768	0.647
Best HaSpeeDe 2	**0.772**	**0.720**

Table 4: Macro F1-scores on HaSpeeDe 2 test datasets.

5 Discussion

5.1 Multilingual and monolingual models

According to Table 3, multilingual models performed worse than monolingual models based on HaSpeeDe 2 data alone, although they achieved respectable results.

Moreover, even when we used additional data from other languages to train the multilingual models, they still did not manage to outperform the monolingual models, as we were hoping they would.

Within the Italian models, UmBERTo and PoliBERT performed better than AlBERTo on these tasks. While the good performance of PoliBERT can be linked to its pre-training for a tweet classification task (sentiment analysis) potentially useful for hate speech detection, it is more difficult to explain the competitiveness of UmBERTo, which was trained on data not coming from Twitter and less numerous than for AlBERTo. One explanation could be the better quality of this data, or a better optimisation by its creators.

5.2 Out-of-domain data and in-domain data

Our results on the HaSpeeDe 2 test dataset are summarized in the Table 4. The results obtained on in-domain data correspond to what we expected from our cross-validation results. Our systems achieved the best macro F1-scores on the in-domain test set (Tweets) for both hate speech and stereotype detection. However, the results on out-of-domain data (News) are far from being as good. This can be explained by the different distribution of this data compared to the training data.

Table 5 shows the confusion matrix for our first system evaluated on out-of-domain data. The error is mostly due to the high number of false negatives. The classifier predicts too many sequences as non-hate speech. This suggests that this classifier trained with hate speech on Twitter is struggling to detect hate speech in newspaper headlines. It can be assumed that hate speech in newspapers is more subtle, with less coarseness and aggressiveness that make it easier to detect on Twitter.

	Predicted False	Predicted True
False	312	7
True	117	64

Table 5: Hate Speech Confusion matrix for UmBERTo evaluated on news test data.

5.3 Multitasking Benefits

We have chosen to submit a system with multitask learning on both Stereotype and Hate Speech detection and an other one without, in order to study the benefits of it. Indeed, the system with multitasking learning performed much better on the in-domain data for the hate speech detection task. It is not the case however for the out-of-domain data, neither for the stereotype detection task.

Table 6 describes in more detail the differences between the predictions of the two systems for data containing stereotypes and data not containing stereotypes. We observed that the improvement linked to multitask learning consists mainly in a reduction in the number of false positives in favour of the number of true negatives in data not labeled as Stereotype. Assuming that hate speech makes significant use of stereotype, one could suppose that the multitask model has learned to discard some data that do not have the characteristics of stereotypes and are therefore unlikely to contain hate speech.

Data labeled as Stereotype		
	Predicted False	Predicted True
False	+3	-3
True	+7	-7
Data not labeled as Stereotype		
	Predicted False	Predicted True
False	+28	-28
True	+1	-1

Table 6: Hate Speech Confusion matrix of the multitask system minus the one of the single-task system, for Stereotype and Non Stereotype tweets test data.

6 Conclusion

In this work, we compared the fine-tuning of multilingual and monolingual BERT-based language models for hate speech detection. We also investigated the addition of multitask learning with the Stereotype detection task linked to hate speech. We obtained the best macro F1-scores of HaSpeeDe 2 on the in-domain test data. However, the results were worse for out-of-domain test data, and further research could be conducted to better understand the reasons for this and address it.

References

Sai Saketh Aluru, Binny Mathew, Punyajoy Saha, and Animesh Mukherjee. 2020. Deep Learning Models for Multilingual Hate Speech Detection.

Gianfranco Barone. 2020. Politic BERT based Sentiment Analysis. `https://huggingface.co/`

`unideeplearning/polibert_sa`. accessed on Sept 18, 2020.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

James Bergstra and Y. Bengio. 2012. Random Search for Hyper-Parameter Optimization. *The Journal of Machine Learning Research*, 13:281–305, 03.

Cristina Bosco, Felice Dell'Orletta, Fabio Poletto, M. Sanguinetti, and M. Tesconi. 2018. Overview of the EVALITA 2018 Hate Speech Detection Task. In *EVALITA@CLiC-it*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, abs/1810.04805.

Chiara Francesconi, Cristina Bosco, Fabio Poletto, and M. Sanguinetti. 2019. Error Analysis in a Hate Speech Detection Task: The Case of HaSpeeDe-TW at EVALITA 2018. In *CLiC-it*.

Ting Gong, Tyler Lee, Cory Stephenson, Venkata Renduchintala, Suchismita Padhy, Anthony Ndirango, Gokce Keskin, and Oguz Elibol. 2019. A comparison of loss weighting strategies for multi-task learning in deep neural networks. *IEEE Access*, PP:1–1, 09.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2017. Multi-Task Learning Using Uncertainty to weigh Losses for Scene Geometry and Semantics. *CoRR*, abs/1705.07115.

Lukas Liebel and Marco Körner. 2018. Auxiliary Tasks in Multi-task Learning. *CoRR*, abs/1805.06334.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Merriam-Webster. 2020. stereotype, noun. `https://www.merriam-webster.com/dictionary/stereotype`. Accessed on 2020-11-05.

John T. Nockleby. 2000. *Hate Speech*. Macmillan, New York.

Loreto Parisi, Simone Francia, and Paolo Magnani. 2020. UmBERTo: an Italian Language Model trained with whole word Masking. `https://github.com/musixmatchresearch/umberto`. accessed on Sept 18, 2020.

Marco Polignano, Pierpaolo Basile, Marco De Gemmis, and Giovanni Semeraro. 2019a. Hate Speech Detection through AlBERTo Italian Language Understanding Model. In *NL4AI@AI*IA*.

Marco Polignano, Pierpaolo Basile, Marco De Gemmis, Giovanni Semeraro, and Valerio Basile. 2019b. AlBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*, volume 2481. CEUR.

Sebastian Ruder. 2017. An Overview of Multi-Task Learning in Deep Neural Networks. *CoRR*, abs/1706.05098.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. Overview of the EVALITA 2020 Second Hate Speech Detection Task (HaSpeeDe 2). In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681, November.

Alessandro Seganti, Helena Sobol, Iryna Orlova, Hannam Kim, Jakub Staniszewski, Tymoteusz Krumholc, and Krystian Koziel. 2019. NLPR@SRPOL at SemEval-2019 Task 6 and Task 5: Linguistically enhanced deep learning offensive sentence classifier. In *SemEval@NAACL-HLT*.

Karen Sparck Jones, 1988. *A Statistical Interpretation of Term Specificity and Its Application in Retrieval*, page 132–142. Taylor Graham Publishing, GBR.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to Fine-Tune BERT for Text Classification? In Maosong Sun, Xuanjing Huang, Heng Ji, Zhiyuan Liu, and Yang Liu, editors, *Chinese Computational Linguistics*, pages 194–206, Cham. Springer International Publishing.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

UO @ HaSpeeDe2: Ensemble Model for Italian Hate Speech Detection

Mariano Jason Rodriguez Cisnero
Universidad de Oriente
Santiago de Cuba, Cuba
mjasoncuba@gmail.com

Reynier Ortega Bueno
Universidad de Oriente
Santiago de Cuba, Cuba
reynier@uo.edu.cu

Abstract

English. This document describes our participation in the Hate Speech Detection task at Evalita 2020. Our system is based on deep learning techniques, specifically RNNs and attention mechanism, mixed with transformer representations and linguistic features. In the training process a multi task learning was used to increase the system effectiveness. The results show how some of the selected features were not a good combination within the model. Nevertheless, the generalization level achieved yield encourage results.

1 Introduction

Modern societies found easy and interesting ways for sharing information via Social Media. Users discover freedom to express themselves through online communication. Even if the ability to freely express oneself is a human right, some users take this opportunity to spread hateful content. A dangerous and hurtful potential arises with this kind of information. Recognizing automatically such content is an interesting topic for researchers.

Creative methods have been proposed to tackle the fascinating task of recognizing hate in texts (De la Pena Sarracén et al., 2018; Gambäck and Sikdar, 2017). Some of those works face the problem using feature extraction (Schmidt and Wiegand, 2017) and classification algorithms like SVM (Santucci et al., 2018). In the last years, Deep Learning approaches have become one of the most successful research areas in Natural Language Processing (NLP). There are exciting investigations about this topic, such as (Cimino et al., 2018), involving LSTM (Liu and Guo, 2019) and transformers (Vaswani et al., 2017) that gain attention in NLP community due to their results.

We propose a model based on multiple representations learned by means of deep learning techniques and linguistic knowledge. Particularly a Long Short Term Memory architecture mixed with linguistic features and language model representations given by a special kind of transformer model, BERT.

The paper is organized as follows. The Section 2 introduces a brief description of HaSpeeDe Task. Our hate detection system is presented in Section 3. The experiments and results are discussed in Section 4. Finally, in Section 5 the conclusions and future directions are given. The code of this work is available on GitHub: `https://github.com/mjason98/evalita20_hate`

2 HaSpeeDe2 Task

Hate speech and stereotypes recognition on social media have become an attractive research area from the computational point of view. In the second edition of HaSpeeDe (Sanguinetti et al., 2020) at Evalita 2020 (Basile et al., 2020), the organizers proposed to address three subtasks. The main subtask is the subtask A, which aims at determining the presence or absence of hateful content in a text. The dataset is composed by 6839 short texts, 2766 labeled as hate speech and 4076 as not hate speech. In this work we focused only on subtask A. The subtask B consists of a binary classification problem oriented to stereotypes' detection. The last subtask C is a sequence labeling task aims at recognizing Nominal Utterances in hateful tweets.

3 Our Proposal

We dealt with hate detection task as a text classification problem to classify "hateful" or "no hate-

---

ful" categories. We train a deep learning model based on attention mechanism and Recurrent Neural Networks, specifically a Bidirectional Long Short Term Memory (Bi-LSTM) (Hochreiter and Schmidhuber, 1997) mixed with linguistic features and transformers representations by means of an interpretable multi-source fusion component (Karimi et al., 2018).

In Section 3.1 and Section 3.2 we describe the linguistic features and the transformer representation used in this work. The Section 3.3 presents the preprocessing phase. Finally, the neural network model and the feature ensemble are described in Section 3.4.

3.1 Linguistic Feature

To build the hate detection model, we start by extracting several sets of linguistic features:

WordNet Features: We count the number of verbs, adverbs, nouns and adjectives. Also, for every word, we calculated the average of its similarity with respect to the others using the $similarity\_path$ function provided by the wordnet[2] corpus. Furthermore, we consider the degree of lexical ambiguity by counting the number of *synsets* of each word within the text.

Hurt and Sentiment content: HurtLex (Bassignana et al., 2018) is a lexicon of offensive, aggressive, and hateful words in over 50 languages. The words according to the 17 categories offered by the lexicon are counted and added as linguistic features jointly with polarity and semantic values obtained from SenticNet (Cambria et al., 2018) corpus.

Information Gain: Information gain (Lewis, 1992) had been a good feature selection measure for text categorization. It takes into account the presence of the term in a category as well as its absence and can be defined by:

$$IG(t_k, C_i) = \sum_{C}\sum_{t} p(t, C) \cdot \log_2 \frac{p(t, C)}{p(t) \cdot p(C)}$$

where $C \in \{C_i, \bar{C_i}\}$ and $t \in \{t_k, \bar{t_k}\}$. In this formula, probabilities are interpreted on an event space of documents, where $p(\bar{t_k}, C_i)$ is the probability that, for a random document d, term t_k does not occur in d who belongs to category C_i. In our case, categories were two: hateful and no hateful, and the term is the word's lemma.

To create the information gain feature (IgF), we calculated the IG for every word and the highest ones are chosen[3]. Then, the occurrence of those selected words in the text are counted.

3.2 Italian BERT

Finally, we use a pre-trained BERT[4] to accomplish the calculation of a deep representation of the text. One of the most widely used auto-encoding pre-trained Language Models (PLMs) is BERT (Devlin et al., 2018). BERT is trained using the masked language modeling task that randomly masks some tokens in a text sequence, and then independently recovers the masked tokens by conditioning on the encoding vectors obtained by a bidirectional Transformer.

Inside BERT, the information is passed forward crosswise transformer layers. In this work, we used a specific output from one of those layers, this operation can be expressed by:

$$h_0 = Hl_0(text_{tok})$$
$$h_i = Hl_i(h_{i-1})$$
$$h_n = Hl_n(h_{n-1})$$

where $text_{tok}$ is the text after its tokenization[5], h_i is the output of the i^{th} transformer layer(Hl_i) called $hidden\_state$ and n is the total transformer layers in BERT. Then, for an specific i, from the tensor of order 2 h_i it is computed the vector f_{bert}, as a deep representation of the initial text who will act as PLM feature.

$$v = \sum_{k=0} h_i[k, :] \qquad f_{bert} = \frac{v}{||v||}$$

3.3 Preprocessing

In the preprocessing step, firstly stopwords were removed . Then, the hashtags composed of many words are split (e.g: #NessunDorma becomes # nessun dorma). We use a regular expression[6] algorithm to archive this step.

Secondly, using the FreeLing[7] tool we obtain for each word it lemma, and non alphanumeric characters are removed. Finally, the remaining words are represented as vectors using a pre-trained word embedding generated by Word2Vec model (Mikolov et al., 2013).

---

[2] The *wordnet* came from the python library *nltk*

[3] We selected the top 50 words with highest IG value.

[4] https://huggingface.co/dbmdz/bert-base-italian-cased

[5] The text is represented as a vector of integers using the *tokenizer* function in BERT Model

[6] The automaton was created using the *re* library from python and the words from an italian corpus.

[7] http://nlp.lsi.upc.edu/freeling/index.php

3.4 The Deep Ensemble Model

The standard LSTM receives sequentially at each time step a vector x_t and produces a hidden state h_t. Each hidden state h_t is calculated as follow:

$$\begin{aligned}
i_t &= \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}) \\
f_t &= \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}) \\
o_t &= \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}) \\
u_t &= \sigma(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}) \\
c_t &= i + t \oplus + f_t \oplus c_{t-1} \\
h_t &= o_t \oplus tanh(c_t) \qquad (1)
\end{aligned}$$

Where all $W^{(*)}$, $U^{(*)}$ and $b^{(*)}$ are parameters to be learned during training. Function σ is the sigmoid function and $\otimes$ stands for element-wise multiplication.

Bidirectional LSTM, on the other hand, makes the same operations as standard LSTM but, processes the incoming text in a left-to-right and a right-to-left order in parallel. Thus, it output become $\hat{h_t} = [\overrightarrow{h_t}, \overleftarrow{h_t}]$ for the two directions.

By adding an attention mechanism, we allow the model to decide which part of the sequence "attends to". First, lets define the *softmax* function $\pi(v)$ for a vector $v = [v_0, \cdots, v_{n-1}]$ as:

$$\pi(v) = \frac{e^v}{\sum_{i=0} e^{v_i}}$$

Then, let $I \in \mathbb{R}^{N \times L}$ be the matrix of input vectors, where L the size of then and N the length of the given sequence. We define the attention layer (AttLSTM), as a regular LSTM layer like (1) with extra operations described as follow:

$$\begin{aligned}
a_{k,t} &= \pi(W_k \cdot h_{t-1}^T + b_k) \quad & \alpha_{k,t} &= a_{k,t}^T \cdot I \\
\beta_t &= [\alpha_{0,t}, \cdots, \alpha_{S-1,t}] \quad & x_t &= W_a \cdot \beta_i + b_a
\end{aligned} \qquad (2)$$

Here $k \in [0, S-1]$ represents the number of attention's heads, $W_k \in \mathbb{R}^{N \times M}$ where M is the size of the hidden state vector h_t, $W_a \in \mathbb{R}^{M \times SM}$, b_a and b_k are learnable parameters. The $(*)^T$ is the transpose operation and the output of the layer is $O = [h_0, ..., h_t, ..., h_N]$, a concatenation of the hidden states produced by the AttLSTM at each time step.

As mentioned before, we propose a feature ensemble by using an interpretable multi-source fusion component (IMF). The IMF aims to combine features from different sources. A naive way of doing this is concatenating the vector representations into a single vector. This scheme considers all sources equally, but one source may yield a better result than others. With IMF we propose to consider the contribution of every source of feature via an attention mechanism. The IMF can be expressed by:

$$r_i = \tanh(W_{p_i} f_i + b_{p_i})$$

where r_i represents a projection of f_i, the i^{th} feature vector passed to IMF ensuring that every r_i have the same size. In this step, all the W_{p_i}, b_{p_i}, W_a and b_a are parameters to be learned during training, then:

$$\begin{aligned}
a_i &= W_a r_i + b_a \qquad & \alpha_i &= \pi(a_i) \\
\beta_i &= \alpha_i r_i \qquad & z &= \sum_{k=0} \beta_k
\end{aligned} \qquad (3)$$

where α_i represents the importance of r_i to the final calculation of z, the IMF outcome.

To increase the learning power of our system, we used a multitask learning (Caruana, 1997) in which we predict the polarity of tweets in parallel with the classes of the hate speech detection subtask. This approach have been developed before (Cimino et al., 2018) in HaSpeede at Evalita 2018 (Bosco et al., 2018). The tweets used to accomplish the multitask learning are extracted from the Sentipolc-2016 (Barbieri et al., 2016) challenge.

Finally we present the composition of the previous layers and features to create our deep ensemble model:

$$\begin{aligned}
E &= [w_0, w_1, \cdots, w_{N-1}] \\
o_{b1} &= BiLSTM(E) \qquad (4)
\end{aligned}$$

where E represents the vector representation of the text, see Section 3.3. Equation (4) is the first block of our model, and the second block can be described as follow:

$$\begin{aligned}
A &= AttLSTM(o_{b1}) \\
m_i &= \max_{j=0,\cdots,N-1} A_{j,i} \\
o_{b2} &= [m_0, \cdots, m_{M-1}] \qquad (5)
\end{aligned}$$

The vector o_{b2} is the return of a MaxPool layer over the A vector sequence, then:

$$\begin{aligned} F &= [o_{b2}, f_{bert}, f_{wn}, f_{hs}, f_{ig}] \\ o_{b3} &= IMF(F) \\ \hat{y} &= \sigma(W_h o_{b3} + b_h) \\ \hat{y_f} &= \sigma(W_f o_{b3} + b_f) \end{aligned} \quad (6)$$

The third block is described in (6) where W_h, W_f, b_f and b_h are learnable parameters and $\hat{y}, \hat{y_f} \in \mathbb{R}$. The vectors f_{bert}, f_{wn}, f_{hs} and f_{ig} correspond to the BERT, WordNet, Hurt-Sentiment and Information Gain features respectively. The prediction of the tweets polarity is determined by the $\hat{y_f}$ value and the hate value trough $\hat{y}$.

The overall weighted loss of the model is calculated by cross-entropy, with higher importance value for the hate speech predictions that polarity predictions. The overall loss is calculated according to the following formula.

$$L_1 = -\sum y_i \log(\hat{y_i}) \qquad L_2 = -\sum y_{f_i} \log(\hat{y_{f_i}})$$
$$loss = \lambda L_1 + (1-\lambda) L_2 \qquad (0 \leq \lambda \leq 1) \quad (7)$$

Here $L1$ and $L2$ are the cross-entropy loss of hate predictions and sentiment polarity predictions respectively. The value λ is the main task importance weight. The values y_i and y_{f_i} represents the ground true hate classification and polarity classification respectively. Then, the final loss is obtained as a convex sum of $L1$ and $L2$.

4 Experiments and Results

In this section we show the results of our proposed method in subtask A and discuss about them. The organizers allow a maximum of two submissions for every subtask in the challenge. We named our team UO.

Experiments where conducted in two main directions: Firstly, to investigate the impact of the IMF fusion strategy and secondly, to evaluate the impact of each proposed single-modal representation into our proposal. The results of our experiments are presented in Table 1 and Table 2.

In those tables, the column named *heads* is the number of attention headers in the Att-LSTM layer. If this space is empty, this layer was not used. Columns *bert* and *ig* correspond to the presence or not of BERT and IG representations. The column *wn-hs* express the presence of Hurt-Sentiment and WordNet based representations. If a cell has a cross, the representation associated to the column were not used in the corresponding run. We used a 10% of the training dataset for validation. We report the accuracy measure computed on this validation data.

Both Tables show that the presence of BERT increase the performance, also almost all the runs have higher values with IMF in contrast to not using it. Increasing the number of attention heads without IMF increase the results, but the opposite occurs in the presence of the IMF.

Name	**heads**	**bert**	**ig**	**wn-hs**	**acc**
run1	2				0.764386
run2	-		×	×	0.742690
run3	3				0.767544
run4	2	×			0.713450
run5	2			×	0.763158
run6	-				0.757310
run7	-	×			0.724152
run8	-			×	0.755848

Table 1: Experiment results without IMF.

Name	**heads**	**bert**	**ig**	**wn-hs**	**acc**
run1	2				0.795848
run2	-		×	×	0.779101
run3	3				0.764620
run4	2	×			0.720760
run5	2			×	0.774854
run6	-				0.767544
run7	-	×			0.719298
run8	-			×	0.777778

Table 2: Experiment results with IMF.

The pretrained embedding have a size of 300, the number of neurons in the Bi-LSTM and in the AttLSTM was 128. The λ value was equal to 0.75 and the dropout (Srivastava et al., 2014) after the embedding layer was 0.3. The optimizer algorithm to train the whole model was Adam (Kingma and Ba, 2015), with a learning rate of 0.01.

The bold models in Table 2 were chosen as final submission for the subtask. The *run1* uses the attention layer proposed in Section 3.2 and consider all proposed representations. The *run2* does not use attention mechanism and handcraft features, using only the BERT text representation and the rest of the architecture.

The Table 3 shows the official results of our system. The evaluation was performed on two distinct corpora: one conformed by tweets and the other by news headlines.

Runs	macro-F
UO:tweets_run1	0.6878
UO:tweets_run2	0.7214
BEST_RATED:tweets	**0.8088**
UO:news_run1	0.6657
UO:news_run2	0.7314
BEST_RATED:news	**0.7744**

Table 3: Official results.

These results show that between our two models, the simple one get better results. The simplicity is not a condition for a better performance using deep learning. These results also express that some linguistic features decrease the effectiveness of the model, but the similarity between the results in the tweets and news evaluation sets suggest that the system is able to generalize with a good performance.

5 Conclusions and Future Work

In this paper we presented an Ensemble Model for the task Hate Speech Detection (HaSpeeDe2) subtask A at Evalita 2020. Our proposal combines linguistic features and RNNs with transformers representations using an IMF. In the training phase, we used a multi-task learning approaches to recognize hate speech and polarity simultaneously.

The achieved results show that the ability of this ensemble to generalize the detection of hate content in different text genres. Nevertheless, some handcraft features decrements its results. Motivated by this, we plan to explore better features selection, other attention mechanisms and multitask learning techniques to improve the performance.

References

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the evalita 2016 sentiment polarity classification task.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Elisa Bassignana, Valerio Basile, and Viviana Patti. 2018. Hurtlex: A multilingual lexicon of words to hurt. In *5th Italian Conference on Computational Linguistics, CLiC-it 2018*, volume 2253, pages 1–6. CEUR-WS.

Cristina Bosco, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. 2018. Overview of the evalita 2018 hate speech detection task. In *EVALITA 2018-Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*, volume 2263, pages 1–9. CEUR.

Erik Cambria, Soujanya Poria, Devamanyu Hazarika, and Kenneth Kwok. 2018. Senticnet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Andrea Cimino, Lorenzo De Mattei, and Felice Dell'Orletta. 2018. Multi-task learning in deep neural networks at evalita 2018. *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*, pages 86–95.

Gretel Liz De la Pena Sarracén, Reynaldo Gil Pons, Carlos Enrique Muniz Cuza, and Paolo Rosso. 2018. Hate speech detection using attention-based lstm. *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:235.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the first workshop on abusive language online*, pages 85–90.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hamid Karimi, Proteek Roy, Sari Saba-Sadiya, and Jiliang Tang. 2018. Multi-source multi-class fake news detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1546–1557.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

David D Lewis. 1992. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 37–50.

Gang Liu and Jiabao Guo. 2019. Bidirectional lstm with attention mechanism and convolutional layer for text classification. *Neurocomputing*, 337:325–338.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. Overview of the evalita 2020 second hate speech detection task (haspeede 2). In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Valentino Santucci, Stefania Spina, Alfredo Milani, Giulio Biondi, and Gabriele Di Bari. 2018. Detecting hate speech for italian language in social media. In *EVALITA 2018, co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018)*, volume 2263.

Anna Schmidt and Michael Wiegand. 2017. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International workshop on natural language processing for social media*, pages 1–10.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

No Place For Hate Speech @ HaSpeeDe 2: Ensemble to Identify Hate Speech in Italian

Adriano dos S.R. da Silva
Schoool of Arts, Sciences and Humanities – University of Sao Paulo
Sao Paulo - Brazil
`adriano.santos.silva@usp.br`

Norton T. Roman
Schoool of Arts, Sciences and Humanities
University of Sao Paulo
Sao Paulo - Brazil
`norton@usp.br`

Abstract

English. In this article, we present the results of applying a Stacking Ensemble method to the problem of hate speech classification proposed in the main task of HaSpeeDe 2 at EVALITA 2020. The model was then compared to a Logistic Regression classifier, along with two other benchmarks defined by the competition's organising committee (an SVM with a linear kernel and a majority class classifier). Results showed our Ensemble to outperform the benchmarks to various degrees, both when testing in the same domain as training and in a different domain.

Italiano. *In questo articolo, ci presentiamo i risultati dell'applicazione di un modello di Stacking Ensemble al problema della classificazione dei discorsi di incitamento all'odio nel compito A di EVALITA (HaSpeeDe 2). Il modello è stato quindi confrontato con un modello di regressione logistica, insieme ad altri due benchmark definiti dal comitato organizzatore della competizione (un SVM con un kernel lineare e un classificatore di classe maggioritaria). I risultati hanno mostrato che il nostro Ensemble supera i benchmark a vari livelli, sia durante i test nello stesso dominio di sviluppo che in un dominio diverso.*

1 Introduction

Social networks are already part of people's lives, generating thousands of publications on a daily basis. Even though most of this material presents no real harm to other people, some of it bears discriminating discourse, not rarely filled with hate for minorities or people with different viewpoints.

Defined as "language which attacks or demeans a group based on race, ethnic origin, religion, gender, age, disability, or sexual orientation/gender identity" (Nobata et al., 2016), hate speech represents a problem that cannot be allowed to grow, under the risk of having it lead to more concrete actions, by some people, with truly undesired results.

This is so much of an issue, that some companies have already decided to stop advertising on Facebook[1], for example, as a way to try to pressure the company into facing this problem. Some initiatives have also emerged in order to monitor and combat this type of content, such as the code of conduct that has been signed by some companies (YouTube, Facebook, Twitter) so that this type of publication can be monitored and removed within 24 hours[2].

Due to the large volume of data, machine learning techniques, along with natural language processing, are being used to automate this activity and identify this type of speech more accurately. Other initiatives include the setting up of competitions, aimed at developing and testing different ways to tackle the problem.

One such competitions is the evaluation campaign of Natural Language Processing and Speech Tools for Italian (EVALITA), which started in 2007 aiming at promoting the development and dissemination of language resources for Italian. In its 2018 edition, a task (HaSpeeDe) was proposed to identify hate speech on Facebook and Twitter (Bosco et al., 2018). HaSpeeDe had the par-

---

[1] `https://www.nytimes.com/2020/08/01/business/media/facebook-boycott.html`

[2] https://ec.europa.eu/info/policies/justice-and-fundamental-rights/combatting-discrimination/racism-and-xenophobia/eu-code-conduct-countering-illegal-hate-speech-online_en

ticipation of several teams and promising results were presented that stimulated the development of the second edition of the event (HaSpeeDe2) at EVALITA 2020 (Sanguinetti et al., 2020; Basile et al., 2020). In this work, we describe our attempt to deal with the hate speech identification problem HaSpeeDe 2, by developing a stack ensemble of three machine learning models to this task. Weak classifiers used in the ensemble were an SVM with RBF kernel, a Bernoulli Naïve Bayes (NB), and a Random Forest model (RF), with a Linear Regression (LR) model serving as meta-classifier.

For the sake of comparison, and as a way to define some benchmarks to our model, we also developed and tested a Linear Regression classifier, with L2 regularisation, along with both models suggested by HaSpeeDe 2 organising committee, to wit, an SVM model with a linear kernel and a majority class classifier. As it will be made clearer in the forthcoming sections, with a Macro F1-score of 0.749, our ensemble outperforms all benchmarks, for both in and out-of-domain test sets, even though sometimes differences were not high.

The rest of this article is organized as follows. Section 2 presents some related work, aiming at identifying hate speech. Section 3, in turn, gives an overview of HaSpeeDe 2 task. Next, in sections 4 and 5 we explain the preprocessing we made, along with the classifiers we built for this task. Section 6, in turn, presents our results, which are further discussed in Section 7. Finally, Section 8 presents our final considerations to this work.

2 Related Work

Several strategies have been used to identify hate speech. Some classic algorithms, like Support Vector Machine (SVM), Naïve Bayes (NB), Logistic Regression (LR) and ensemble with these techniques have also shown good results (*e.g.* (Basile et al., 2019; Saha et al., 2018; Malmasi and Zampieri, 2018)).

An SVM with RBF kernel, for example, was used to identify hate speech against immigrants and women in tweets written in English. Achieving a macro-averaged $F1$ score of 0.65 this model was the winner at SemEval 2019 (Basile et al., 2019).

Logistic Regression was another classic model to be applied to hate speech identification in English, in this case focusing in hate speech towards women, with a reported accuracy of 0.70 (Saha et al., 2018). Delivering an accuracy value of 79.8, an ensemble associated with a meta-classifier was also found to perform well in the task (Malmasi and Zampieri, 2018).

With an overall performance of $F1 = 0.749$, our ensemble method looks competitive, when compared to these models. Even though one cannot really make a true comparison between them, we believe this to be an alternative to be considered.

3 Task

HaSpeeDe 2 Task A consists of a binary classification to identify the presence or absence of hate speech in tweets written in Italian. The competition's organising committee provides participants with a data set for training and testing competing models. This data set is slightly imbalanced, with approximately 40% of tweets presenting hate speech language, as shown in Table 1.

Table 1: Data set class distribution

Hate Speech	Not Hate Speech	Total
2766	4073	6839

This data set is supposed to be used by the competition participants to train and test their models. Competing models will then be evaluated in a separate data set, which consists of in-domain and out-of-domain data, defined by the competition's organisation.

4 Preprocessing

As a preprocessing step, we removed stopwords using the NLTK (Natural Language Toolkit [3]) library. For each tweet in the corpus, we also added the following new features:

- The number of words in the tweet;
- The number of exclamation points ('!') present in the tweet; and
- The presence or not of a question mark ('?') in the tweet.

As a final measure, all features related to the tweet's text were normalised in the range between 0 and 1.

---

[3] `https://www.nltk.org/`

Table 2: Results of the classifiers in the training stage in terms of F1

		Without Preprocessing		**With Preprocessing**	
Classifier	**Lang. Model**	**No Norm.**	**TF-IDF**	**No Norm.**	**TF-IDF**
RF	3-Gram	0.662	0.657	06687	0.667
RF	4-Gram	0.683	0.694	0.690	0.689
RF	5-Gram	**0.701**	**0.701**	0.687	0.686
LR	3-Gram	0.681	0.703	0.676	0.696
LR	4-Gram	**0.711**	0.701	0.706	0.697
LR	5-Gram	**0.711**	0.673	0.708	0.673
NB	3-Gram	0.679	0.679	0.681	0.681
NB	4-Gram	0.689	0.689	**0.694**	**0.694**
NB	5-Gram	0.654	0.654	0.668	0.668

Table 3: Results of the classifiers in the test stage in terms of F1

		Without Preprocessing		**With Preprocessing**	
Classifier	**Lang. Model**	**No Norm.**	**TF-IDF**	**No Norm.**	**TF-IDF**
RF	3-Gram	0.650	0.668	0.650	0.674
RF	4-Gram	0.693	0.694	**0.710**	0.696
RF	5-Gram	0.707	0.709	0.703	0.700
LR	3-Gram	0.675	0.701	0.675	0.709
LR	4-Gram	0.684	0.696	0.685	**0.710**
LR	5-Gram	0.669	0.665	0.707	0.680
NB	3-Gram	0.696	0.696	0.707	0.707
NB	4-Gram	0.718	0.718	**0.740**	**0.740**
NB	5-Gram	0.658	0.658	0.687	0.687

5 Classifiers

In the sequence, three individual classifiers were developed using the Python Sklearn[4] library. These were a Naïve Bayes (NB) with Bernoulli distribution, Logistic Regression (LR) with L2 regularization, and Random Forest (RF) with 150 trees. Each classifier was tested with N-Gram representations (N ranging from 3 to 5), with and without term frequency-inverse document frequency (TF-IDF) (Rajaraman and Ullman, 2011) normalisation, and with and without pre-processing the training and test sets.

We then chose the two best models to compose the ensemble to be used at the competition. As it will be shown in the next section, these were Random Forests and Naïve Bayes. In the sequence, we also added an SVM classifier, to RBF kernel and $C = 2$ penalty to the ensemble, making Logistic Regression our meta-classifier.

The training set was divided into 90% for training/validation and 10% for test set. Models were trained in the training/validation set using 10-fold cross-validation. (Han et al., 2011).

6 Results

Tables 2 and 3 show the performance and settings of each classifier in the training/validation and test sets, respectively. During training, best results were observed without preprocessing, for RF and LR, whereas NB showed better results with preprocessing. These results, however, were very close to each other, ranging from $F1 = 0.69$ to $F1 = 0.71$. Regarding language model, best results were observed with 5-grams, for RF and LR, and 4-grams, for LR and NB.

At the test set, best results, for all methods, were observed with preprocessing the data. Normalising the vectors does not seem, however, to have influenced results when preprocessing is used. All best values were obtained with 4-grams. Overall, the best result was achieved with Naïve Bayes ($F = 0.74$), with preprocessing, using a 4-gram language model, and both with and without TF-IDF normalisation.

---

[4] https://scikit-learn.org/stable/

The ensemble model was tested with only one configuration: 4-Gram, with normalization, and without preprocessing. This configuration resulted in an $F1 = 0.729$ in the training set (a 2.5% increase over the best model in this set) and an $F1 = 0.751$ in the test set, corresponding to a 1.5% improvement over the best model in this set. As it turns out, especially in the test set, differences between the ensemble and its best constituent method do not seem so high.

7 Discussion

The competition rules allow only two models to be sent by each team. Although our Naïve Bayes model has shown good performance in the test set we had at hand, we chose not to send it to HaSpeeDe 2 due to the fact that it would also be tested in an out-of-domain data set.

Since this classifier can be very sensitive to domain changes, specially regarding null frequency words, which might bring the whole model down to multiplying smoothing values, we thought we would be better off not sending it. Still, it remained as one of the weak classifiers in the Ensemble we sent, so it was not completely put aside.

The organization of the competition presented F1 results corresponding to two classifiers, run in the same data set distributed to all participants in the competition. These were supposed to be taken as baselines by all competing teams. The first consisted of a majority class classifiers (Baseline-MC), which always chooses the majority class to label new examples. The second classifier, in turn, consisted of an SVM with linear kernel, running with TF-IDF normalisation (Baseline-SVM).

Table 4 shows the result of these two baseline classifiers, along with the classifiers we submitted to the competition (*i.e.* our Ensemble model and its constituent Logistic Regression classifier). As it turns out, for the within-domain task, only our Ensemble was superior to the baselines (3.9% over the baseline SVM and almost 123% over the majority class baseline). When moving to the out-of-domain test set, this difference dropped to only 1.8% over the SVM model and 62.3% over the majority class, still outscoring both baselines.

Regarding our Logistic Regression model, when run in the within-domain test set, it outscored only the majority class baseline (109% better), being however outscored by the baseline SVM by 2.3%. As for the out-of-domain test set, our Logistic Regression model presented the same result as the baseline SVM, outscoring the majority class baseline by 59.5%. Interestingly, both Ensemble and Logistic Regression models scored similarly in this set.

Table 4: Result of baselines and final performance of classifiers in task A in terms of F1

Classifier	**Out-of-domain**	**In-domain**
Baseline-MC	0.3894	0.3366
Baseline-SVM	0.621	0.7212
Ensemble	0.632	0.749
LR	0.621	0.705

8 Conclusion

In this article we reported on the results obtained by two models submitted to EVALITA's HaSpeeDe2 task. Even though our Ensemble model outscored both benchmarks, we believe it could do better, should other choices regarding the language model be made.

Since the best results were obtained with longer word sequences (in our case, 4-grams), it might be the case that other language models, such as Glove or CBOW, for example, which make use of context words at both sides of the target word, could come up as better alternatives for the 4-gram model we used. BERT could also be a possibility to test.

Our best results were also obtained, at least during test, with preprocessing the data. We thus believe this is something to be kept. Regarding the normalisation of feature vectors, we could not observe great differences between using it or not, at least when it comes to TF-IDF normalisation.

Another direction to be followed might be to test other models as weak classifiers in the Ensemble, or even ensemble strategies other than stacking. This is something we leave for future work.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, Minneapolis, USA, June.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Cristina Bosco, Felice Dell'Orletta, Fabio Poletto, Manuela Sanguinetti, and Maurizio Tesconi. 2018. Overview of the EVALITA 2018 Hate Speech Detection Task. In *Proceedings of the 6th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA'18)*.

Jiawei Han, Jian Pei, and Micheline Kamber. 2011. *Data mining: concepts and techniques*. Elsevier.

Shervin Malmasi and Marcos Zampieri. 2018. Challenges in discriminating profanity from hate speech. *Journal of Experimental & Theoretical Artificial Intelligence*, 30(2):187–202.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*.

Anand Rajaraman and Jeffrey David Ullman. 2011. *Mining of massive datasets*. Cambridge.

Punyajoy Saha, Binny Mathew, Pawan Goyal, and Animesh Mukherjee. 2018. Hateminers : Detecting hate speech against women. *CoRR*, abs/1812.06700.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Svandiela @ HaSpeeDe: Detecting Hate Speech in Italian Twitter Data with BERT

Svea Klaus **Anna-Sophie Bartle** **Daniela Rossmann**
Eberhard Karls Universität Tübingen
{svea-kristin.klaus, anna-sophie.bartle, daniela.rossmann}
@student.uni-tuebingen.de

Abstract

English. This paper explains the system developed for the Hate Speech Detection (HaSpeeDe) shared task within the 7th evaluation campaign EVALITA 2020 (Basile et al., 2020). The task solution proposed in this work is based on a fine-tuned BERT model. In cross-corpus evaluation, our model reached an F1 score of 77,56% on the tweets test set, and 60,31% on the news headlines test set.

Italiano. *Questo articolo spiega il sistema sviluppato per il tesk finalizzato all'individuazione dei discorsi d'odio all'interno della campagna di valutazione EVALITA 2020 (Basile et al., 2020). La soluzione proposta per il task è basata su un raffinemento di un modello BERT. Nella valutazione finale il nostro modello raggiunge un valore F1 di 77,56% sul dataset di tweets e di 60,31% sul dataset di titoli di giornale.*

1 Introduction

The detection of Hate Speech has been a popular task in Natural Language Processing. Because there is no universal definition of the term 'hate speech', we follow the EVALITA 2018 organizers in defining it as any expression *"that is abusive, insulting, intimidating, harassing, and/or incites to violence, hatred, or discrimination. It is directed against people on the basis of their race, ethnic origin, religion, gender, age, physical condition, disability, sexual orientation, political conviction, and so forth"* (Erjavec and Kovačič, 2012).

Apart from being hurtful to the person or group that the hateful message is aimed at, its systematic usage can be the cause of hate crime and other criminal acts towards these groups. Mass and social media help to spread hate speech a lot faster than traditional communication channels (Sponholz, 2018). However, social media platforms like Twitter, YouTube and Facebook lack systematic control in monitoring and removing hateful comments. Although these platforms discourage hateful content, its removal depends on individual users and trusted reports (Erjavec and Kovačič, 2012), thus indicating that automated detection of such utterances is a crucial problem to solve. Our goal within the HaSpeeDe task was to develop a system for automated detection of hateful messages against muslims, roma, and immigrants. The first section introduces related works on the topic. In the second section, we explain the task setup, followed by the description of our approach. Finally, we show our results and discuss them with regards to possible future work on hate speech detection.

2 Related Work

In previous work, automated detection of hateful messages has been approached in various ways, starting from simpler lexicon-based approaches and Naive Bayes classifiers to more state of the art Convolutional Neural Networks (Zhang and Luo, 2018). The EVALITA 2020 shared task follows SemEval 2019 (May et al., 2019) and EVALITA 2018 (Bosco et al., 2018), where the automated

---

detection of hateful speech has also been among the core topics.

Early work in this area includes Spertus' automatic recognition of hostile messages with the Smokey system. She found that only 12% of such messages contained explicit keywords. Therefore, she compiled a set of rules resulting in a 47-element feature vector per sentence to capture semantic and syntactic information. For instance, imperative statements have higher chances of containing insulting content than indicative utterances. The same applies to sentences starting with *you*. For evaluation, decision trees were trained on the vectors and the results were compared to human assessments. Overall, in 36% of the cases the instances labeled as insulting matched with the human classification. (Spertus, 1997).

Another approach introduced by Greevy and Smeaton in 2004 involved support vector machines for classifying racist texts. In their work, they compared part-of-speech distributions across racist and non-racist documents as well as different feature representations like bag-of words and bigrams. The bag-of-words model was found to be more useful than the bigram model (accuracy of 87.77% vs. 84.77%) (Greevy and Smeaton, 2004).

Since around 2015 and with the gaining popularity of deep learning, various methods involving neural networks have been proposed. For instance, Kamble and Joshi compared a CNN, LSTM, and BiLSTM to one another for detecting code-mixed Hindi-English hate speech within the context of ICON 2018. The CNN was fed with domain-specific embeddings and showed the best performance (F1 score of 80.85%) (Kamble and Joshi, 2018). The growing interest in hate speech detection is further reflected in other shared tasks, workshops, and data mining competitions on Abusive Language, Trolling, Aggression, Cyberbyullying, Misogyny detection and so forth (Zhang and Luo, 2018). For the most part, these models are trained on English text data, paying little attention to other languages. Therefore, Italian hate speech detection has been introduced within the context of EVALITA (Sanguinetti et al., 2020a).

In 2018, the EVALITA organizers presented three subtasks: In the first task, Facebook data was used to classify a message as not hateful (0) or hateful (1) and in Task 2, the same challenge was conducted on Twitter data. Task 3 asked the participants to train on the Facebook data and test on the Twitter data, and vice versa. With an F1 score of 0.82, the best performance on the Facebook task was achieved by a team that used polarity and subjectivity lexicons as well as two word-embedding lexicons as external resources together with a 2-layer BiLSTM. The same team reached the best performance for the Twitter data (F1 score of 0.79). However, systems that were cross-corpus tested performed significantly worse with an F1 score of 0.65% with the Facebook training set and 0.69% with the Twitter train data. The former score was achieved with a neural network with three hidden layers involving word embeddings that were trained on previously extracted Facebook comments; the latter was once again achieved by the team with the 2-layer BiLSTM (Cimino et al., 2018).

3 Task Description and Dataset

We participated in subtask A of HaSpeeDe – a binary classification task to predict the presence or absence of hate speech in Italian Twitter messages (Sanguinetti et al., 2020b). The training dataset provided by the task organizers consists of 6837 text samples collected from Twitter and corresponding binary labels: 1 if the text sample contains hate speech and 0 otherwise. Among the tweets, 4071 are labeled as not containing hate speech, 2766 are labeled as hate speech. Table 1 shows two examples with their labels.

id	text	hs
1940	Ma quindi solo io sono preoccupato che il terrorista stava in Italia?	0
6777	Cacciamo tutti gli immigrati visto che sono un pericolo	1

Table 1: Example Tweets from the training data

4 Experiments

To solve the task, we fine-tuned the language model ***B****idirectional* ***E****ncoder* ***R****epresentations from* ***T****ransformers* (*BERT*). BERT was developed by Google and offers great possibilities not only for hate speech detection, but for all kinds of tasks that involve processing natural language (Devlin et al., 2019). Since BERT is available for multiple languages, we were interested in which version of BERT – the multilingual BERT (bert-base-multilingual-cased) or the Italian version of BERT (dbmdz/bert-base-italian-cased) (Wolf et al., 2019) – would perform best for the task at hand to determine Italian hate speech in tweets and news headlines. The multilingual BERT cased is a language model that has been trained on 104 languages whereas the latter version has been pretrained solely on Italian language.

For faster and more efficient processing while fine-tuning the model, we used Google Colab (`https://colab.research.google.com`) in all experiments as it provides free GPU. We further experimented with the training data by comparing model performance on the data as it was provided by the event organizers and after cleaning it. Leaving data as is could have several advantages: On the one hand, it can be helpful to leave in junk characters that appear in tweets as well as trailing white spaces. For instance, a tweet written in all capital letters might indicate an insult and therefore contain useful information for the classifier. On the other hand, the task at hand did not solely require hate speech detection on social media but was evaluated on newspaper articles. Therefore, the model might adapt too much to the specific style of the Twitter genre and lower classifier performance when trying to generalize to another domain (like newspaper articles where these kinds of characters do not occur). For both our runs of the final model we cleaned the data as previous test runs showed better performance.

4.1 System Description

To solve the task, we fine-tuned a BERT model. After experimenting with the different language models as described in the previous section, we found the *bert-base-italian-cased* model to be the best fit. The data was split into training and validation set during the first phase of the training. Cross-validation was used on the training set to prevent overfitting, and the validation set was used to assess how the model will generalize to unseen data. In the second training phase, the whole training data was used for training purposes.

Before experimenting with different estimators, the data was cleaned from @user-marks, trailing whitespaces, and we corrected errors like "&amp" to "&". Since BERT is an already trained language model, extensive preprocessing of the data is not unnecessary. However, we assume that some preprocessing will be useful for cross-domain evaluation. After preprocessing, the text data was tokenized by the Italian BERT tokenizer (AutoTokenizer) that splits texts into tokens. It adds special [CLS] and [SEP] tokens to mark that the sentences can now be used for classification purposes and to separate sentences so that each token within a sentence can be assigned a segment token. Afterwards, the tokens are converted into token ids using the pre-defined indices of BERT's tokenizer vocabulary. Additionally, those embeddings are also assigned attention masks that specify how much attention the system should pay to each of the words.

Since we implemented BERT with PyTorch, we used the optimization module AdamW for finetuning. Finding a good learning rate can be difficult. AdamW takes care of this issue by adapting the learning rates for different parameters which makes the training process more efficient (Kingma and Ba, 2015). Following the recommendations of the developers of AdamW, we set the learning rate to *5e-5* as default which also achieved the best results overall. Moreover, we tried various epochs, again using the recommended number of epochs, to see whether the performance of the model would improve. The best F1-score and overall accuracy was achieved with only two epochs. During each epoch the model is trained and evaluated on the validation set. The batch size was set to 16 and we set the random seed to 42 to ensure reproducibility.

Even though we are dealing with binary classification, the model makes predictions by calculating probabilities using the softmax function. Moreover, we used a threshold of 0.9% to reduce prediction errors; 90% certainty is very high when we compare the default threshold of 50% that is typically used for this purpose. However, after manually going through some of the test data, it is sometimes fairly difficult even for a human to uncover hate speech, especially for the *news* dataset.

Therefore, our goal was to produce realiable predictions. For both our runs we used the same system playing around with some of its parameters according to the results received from the first run. Therefore, our second run performs slightly better.

5 Results

When evaluating our model with the two test sets provided by the EVALITA organizers, we received the scores shown in Table 2. Our model performed 17% better on test data containing Tweets (Basile et al., 2020) compared to the news data with overall F1 macro-scores of **77.56%** (on tweets) and **60%** (on news).

The organizers provided two baseline models (see Table 3 – *most frequent class* (MFC) and *Linear SVM* with unigrams, char-grams and TF-IDF representation. Our model achieved higher scores for the news headlines and the twitter test set compared to the MFC baseline that achieved Macro-F1 scores of 38.94% and 33.66% respectively. However, our model failed to beat the baseline of the Linear SVM for the news test set which scored 62.1%. Nevertheless, it performed better on the tweets test set compared to the Linear SVM (72.12%).

Test Data	non-hate			hate		
	F1	P	R	F1	P	R
News	0.82	0.70	0.98	0.39	0.25	0.9
Tweets	0.79	0.75	0.83	0.76	0.81	0.72

Table 2: System Evaluation

Test Data	non-hate			hate		
	F1	P	R	F1	P	R
News MFC	0.78	0.64	1	0	0	0
News SVC	0.78	0.71	0.87	0.46	0.61	0.38
Tweets MFC	0.67	0.51	1	0	0	0
Tweets SVC	0.72	0.73	0.71	0.72	0.71	0.73

Table 3: Baseline Results (Basile et al., 2020)

As expected, model performance decreases in cross-corpus evaluation, especially in the news headlines test data. We assume that our model learned characteristics of the Twitter data alongside the characteristics of hate speech. Therefore, the model performs worse when applied to domains that entail different linguistic surface structures. The F1 macro-scores in Table 2 show that the scores for the two labels are evenly distributed (79% for non-hate and 76% for hate). Contrary to this, the model tested on the news data is a lot more likely to detect non-hate items (with 82%) whereas its performance on finding hate items only lies at 39%. The confusion matrices for both test sets for the second run can be seen in Table 4 and Table 5.

		Predicted	
		Positive	Negative
Actual	Positive	314	5
	Negative	136	45

Table 4: Confusion Matrix of news headlines test set

		Predicted	
		Positive	Negative
Actual	Positive	534	107
	Negative	175	447

Table 5: Confusion Matrix of tweets test set

6 Error Analysis

Identifying hate speech in Twitter data was obviously easier for our model because it had been trained on similar data. However, the model had more difficulties in making predictions on the news headlines as hints towards hate speech were much more subtle and harder to grasp. This became especially clear when we tried to identify hate speech in the test data ourselves. For the tweets test data, the use of hate speech was more obvious and direct. Another and bigger problem might have been missing context information as we were limited to the headlines, thereby missing the content of the article. Since we had difficulties identifying especially hate speech for the news headlines test data it is only reasonable that our model had similar difficulties and performed worse compared to the tweets test set. Table 6 and 7 show some examples where our system failed to detect hate speech correctly. Table 6 contains examples with upper-cased words which are used to highlight strong ideas and opinions. In this context, the upper-cased language is used to highlight the rage of the user. Therefore, our model should have been made more sensible towards the intentional use of capital letters to classify content containing hate speech more accurately. Nevertheless, none of these examples, including Table 7 were correctly classified as hate speech.

id	text
11834	@user A me pare una scelta politica suicida puntare tutto su una battaglia sicuramente perdente in favore dell'immigrazione incontrollata...Meglio così, spariranno più velocemente!
11846	Rosarno, le case popolari? Solo agli immigrati Hanno avuto bisogno di governi non eletti, di gente imposta ad un popolo disarmato. Una volta messi li, i VIGLIACCHI hanno dato inizio alla ns fine! Se e quando si scatenerà la rabbia vera, ne farò parte!!URL
11220	I CRISTIANI ATTACCATI DAL MONDO ISLAMICO: IRAQ SIRIA SRI LANKA E ED EUROPA.E LA CHIESA DIVISA TRA DUE PAPI, BENEDETTO AUTOREVOLE RINTUZZA LA RIVOLUZIONE TRASGRESSIVA DEI COSTUMI, FRANCESCO LASCIA FARE. CRISTIANI PERSEGUITATI MA IL PROBLEMA SONO I MIGRANTI URL

Table 6: Example Tweets wrongly classified

id	text
10547	L'Europa caccia i clandestini
10130	Italia? Immigrati e sfottò: Mr Europa ci rende onore ma non fermerà l'invasione
10247	Immigrazione, la rotta dei sospetti jihadisti: in Italia su moderni gommoni

Table 7: Example News Headlines wrongly classified

7 Discussion

Our goal was to develop a system for Hate Speech Detection in Italian Twitter data. After cleaning the data, we fine-tuned a BERT model with a batch size of 16 and a learning rate of *5e-5*. Overall, our model reached an F1 score of 77.56% on the Twitter test data, and 60% on the news data. Ideas for future work include adding training data that has been collected from other sources apart from Twitter, incorporating a lexicon of hate words, such as Hurtlex (Bassignana et al., 2018), or using topic modelling techniques to extract information about topics that are likely to be involved in hate speech on social media.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Elisa Bassignana, Valerio Basile, and Viviana Patti. 2018. Hurtlex: A Multilingual Lexicon of Words to Hurt. In Elena Cabrio, Alessandro Mazzei, and Fabio Tamburini, editors, *Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Torino, Italy, December 10-12, 2018*, volume 2253 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Christina Bosco, Felice Dell'Orletta, Fabio Poletto, Manuela Sanguinetti, and Maurizio Tesconi. 2018. Overview of the EVALITA 2018 Hate Speech Detection Task. *EVALITA@CLiC-it*, pages 1–9.

Andrea Cimino, Lorenzo De Mattei, and Felice Dell'Orletta. 2018. Multi-task learning in deep neural networks at EVALITA 2018. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT*, pages 4171–4186.

Karmen Erjavec and Melita Poler Kovačič. 2012. "You Don't Understand, This Is a New War!" Analysis of Hate Speech in News Web Sites' Comments. *Mass Communication and Society*.

Edel Greevy and Alan F. Smeaton. 2004. Classifying racist texts using a support vector machine. In *SIGIR 2004 - the 27th Annual International ACM SIGIR Conference, 25-29 July 2004, Sheffield, UK.*

Satyajit Kamble and Aditya Joshi. 2018. Hate speech detection from code-mixed hindi-english tweets using deep learning models.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.

Jonathan May, Ekaterina Shutova, Aurelie Herbelot, Xiaodan Zhu, Marianna Apidianaki, and Saif M. Mohammad, editors. 2019. *Proceedings of the 13th International Workshop on Semantic Evaluation*, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020a. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020b. Overview of the EVALITA 2020 Hate Speech Detection (HaSpeeDe 2) Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Ellen Spertus. 1997. Smokey: Automatic Recognition of Hostile Messages. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Conference on Innovative Applications of Artificial Intelligence*, pages 1058–1065. AAAI Press.

Liriam Sponholz. 2018. *Hate Speech in den Massenmedien: Theoretische Grundlagen und empirische Umsetzung*. VS Verlag für Sozialwissenschaften.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv*, abs/1910.03771.

Ziqi Zhang and Lei Luo. 2018. Hate speech detection: A solved problem? the challenging case of long tail on twitter. *Semantic Web*, 10.

CHILab @ HaSpeeDe 2: Enhancing Hate Speech Detection with Part-of-Speech Tagging

Giuseppe Gambino and Roberto Pirrone
Dipartimento di Ingegneria
Università degli Studi di Palermo
giuseppe.gambino09@community.unipa.it
roberto.pirrone@unipa.it

Abstract

The present paper describes two neural network systems used for Hate Speech Detection tasks that make use not only of the pre-processed text but also of its Part-of-Speech (PoS) tag. The first system uses a Transformer Encoder block, a relatively novel neural network architecture that arises as a substitute for recurrent neural networks. The second system uses a Depth-wise Separable Convolutional Neural Network, a new type of CNN that has become known in the field of image processing thanks to its computational efficiency. These systems have been used for the participation to the HaSpeeDe 2 task of the EVALITA 2020 workshop with CHILab as the team name, where our best system, the one that uses Transformer, ranked first in two out of four tasks and ranked third in the other two tasks. The systems have also been tested on English, Spanish and German languages.

1 Introduction

Hate speech is not unfortunately a new problem in the society, but recently it has found fertile ground in social media platforms that enable users to express themselves freely and often anonymously. While the ability to freely express oneself is a human right, inducing and spreading hate towards another group is an abuse of this liberty (MacAvaney et al., 2019).

As such, many online micro-blogs such as Facebook, YouTube, Reddit, and Twitter consider hate speech harmful, and have both policies and instruments to remove hate speech content, that are getting better over time. Due to the societal concern and how widespread hate speech is becoming on the Internet, there is strong motivation to study automatic detection of hate speech. By doing so, the spread of hateful content can be reduced, having a safer place to stay online for the community but also a more attractive place for advertising sponsors who do not want their brand to be associated with hateful content. Obviously, detecting hate speech is a challenging task. For example, in case of wrong classification, a content creator could suffer socio-economic consequences such as the demonetization of one of its contents or the ban from the platform used. Therefore, the goal of hate speech detection is not only to identify a text that contains words that at first sight could be negative, but also to be able to distinguish news headlines that talk about crime news from a text that contains an effective "attack" against a person or group on the basis of attributes such as race, religion, ethnic origin, national origin, sex, disability, sexual orientation, or gender identity.

The rest of the paper is arranged as follows. Section 2 reports a description of our systems developed for hate speech detection tasks. Section 3 shows the results obtained in the HaSpeeDe 2 (Sanguinetti et al., 2020) task of the EVALITA 2020 (Basile et al., 2020) conference, together with other results obtained with different languages. Results are showed in Section 4 and conclusions are discussed in Section 5.

2 Description of the Systems

In this section we present the implementation details of all the used architectures. Both the systems we implemented share the use of PoS Tagging technique that is applied to the pre-processed text, and passed as an additional input to the neural network.

2.1 Pre-processing

Before training a model, it is common practice to clean the data, especially if they are retrieved from social media. For this reason we implemented a classic text pre-processing pipeline, that consists of: lower casing the text; removing HTML tags, mention and symbols; standardizing words by cutting the characters repeated more than two times in a row. We also made some keyword substitutions in all our data sets:

- URLs and the "url" keyword of the HaSpeeDe 2 data set were replaced by the symbol LINKURL
- Happy emoticons like " :) " or " :D " were replaced by the symbol HAPPYEMO
- Angry or sad emoticons like " :@ " or " :(" were replaced by the symbol BADEMO

It is important to note that we have not removed the emojis from the text as our word embedding takes into account emojis as plain words.

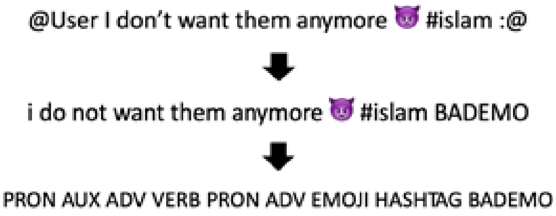
2.2 Part-of-Speech Tagging

In this work we use the PoS Tagging technique to provide our networks with more information about the meaning of a sentence through an explicit classification on the basis of its grammatical structure. This is a crucial point with regards to hate sentences. In fact they tend to have particular structures. As an example, one of the most widespread hate sentence is the verbless one, also known as nominal utterance (Comandini et al., 2018). Another example are journalistic tweets (Comandini and Patti, 2019). Starting from a preliminary direct inspection of the development data set proposed in HaSpeeDe 2, we found that usually a journalistic tweet is a short tweet that ends with an URL. Such texts can be easily misclassified due to the presence of some negative words that explain the news. Table 1 reports some examples of these types of statements.

Tweet	HS
@user useless people like all Muslims	1
@user no more refugees in Italy please no more	1
Four bicycles stolen from Milan-Sanremo cyclists: found in a gypsy camp url	0
TRAGEDY IN PRISON - The nomad Carlo Helt takes his own life url	0

Table 1: Some examples translated into English drawn from the development data set proposed in the HaSpeeDe 2 competition together with their label: nominal utterances used in hate speech along with journalistic tweets

As the HaSpeeDe 2 organizers required explicitly to use the same system for both tasks A and B, we set up a PoS Tagging model not too biased towards either news headlines or tweets. As a consequence, we enriched the PoS Tagger provided by the Python's spaCy library (Honnibal and Montani, 2017). As this model is trained on Wikipedia, we used some regex formulas to add the keywords for emoticons, emojis, hashtags, and URLs to the vocabulary. In this way we have injected some parts of the speech of the social media language into a standard PoS Tagging model. We were definitely aware that tweet oriented models such as UDPipe tool (Straka, 2018) trained on POSTWITA-UD Treebank (Sanguinetti et al., 2018) would have performed better than our solution on the in-domain data but our solution guaranteed a more balanced performance. An example of our PoS Tagging is showed in Figure 1.



@User I don't want them anymore 😈 #islam :@

i do not want them anymore 😈 #islam BADEMO

PRON AUX ADV VERB PRON ADV EMOJI HASHTAG BADEMO

Figure 1: PoS Tagging example

2.3 Word Embedding

It is well known in the NLP community that word embeddings are one of the features that most affects the performance of a model.

For our application we chose fastText (Bojanowski et al., 2016), a word embedding developed by Facebook Research. FastText enriches word vectors with subword information treating each word as composed of n-grams. Each word vector is the sum of the vector representations of each of its n-grams. In this way, two words not only will have nearby vectors if they have similar context but also if they are similar. This is a great feature to treat miss-spelling that occurs often in social languages. We trained from scratch the word embedding for the Italian language with the Gensim library (Řehůřek and Sojka, 2010) on a 2014 MacBook Pro 13” with 8GB RAM and AVX2 FMA CPU extension and it took about 5 hours. The embedding model has been trained for 10 epochs on 5 millions Italian tweets, with a size = 300, window_size = 5, and min_count = 2. These tweets were extracted from TWITA 2018 Dataset (Basile and Nissim, 2013) and are all related to the words: immigrati, islam, migranti, musulmani, profughi, rom, stranieri, salvini, criminali, africani, terroni, #dallavostraparte, #salvini, #stopinvasione, #piazzapulita, #quintacolonna.

For the French, English and German tweets we used pre-trained models (Camacho-Collados et al., 2020). Regarding the PoS Tagging embedding, we have applied the TensorFlow’s Embedding Layer for all the languages considered.

2.4 System 1: The Transformer

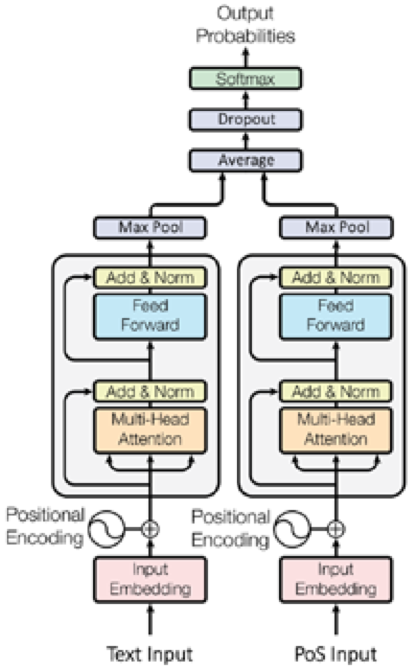


Figure 2: The Transformer System

Transformers (Vaswani et al., 2017) are the current state-of-the-art models for dealing with sequences. Unlike previous architectures for NLP, such as LSTM and GRU, there are no recurrent connections and thus no real memory of previous states. Transformers get around this lack of memory by perceiving entire sequences simultaneously and treating them with an attention mechanism. In this way, Transformers achieve parallelism that leads to a significantly shorter training time than recurrent solutions. Attention is a means of selectively weighting different elements in input data, so that they will have an adjusted impact on the hidden states of downstream layers.

A Transformer was conceived as an encoder-decoder model, that is an ideal approach for machine translation tasks and language modeling. In this work we used the Transformer encoder architecture, as an alternative to recurrent or convolutional neural networks (CNN) (see Figure 2). We used just one Transformer encoder for the text input and one for the PoS input, then we averaged them thorough max pooling. Finally, we used dropout and a dense layer to get the output probabilities. After testing various combinations of parameters, we found that the most efficient for this task are: 12 heads in Multi-Head attention layer, 768 hidden units, embedding size equal to 300, dropout = 0.2 and batch size equal to 128. Training lasted 3 epochs, about 40 seconds each.

2.5 System 2: Depth-wise Separable Convolutional Neural Network

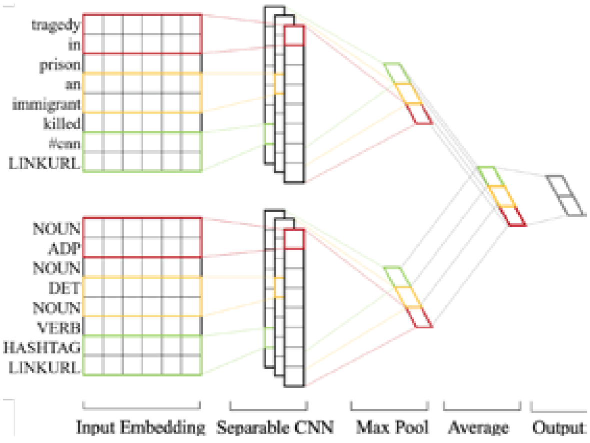


Figure 3: The DSC System

Depth-wise Separable Convolution (DSC) is a well known technique in Computer Vision to lower dramatically the number of parameters in CNN. DSC consists in decomposing classical 3D convolution, performing at first a depth-wise spatial convolution for each channel, followed by a point-wise convolution which mixes together the resulting output channels. This computational trick achieves in mimicking the true convolution kernel operation, while reducing the size of the model, and speeding up the training with almost the same accuracy.

Our neural network architecture is reported in Figure 3, and takes inspiration from Yoon Kim's well-known architecture (Kim, 2014). We made some changes taking into consideration both the vectorized text and its PoS Tagging. The overall architecture is made by two parallel DSC networks that receive the text, and PoS embedding respectively. The two convolutional blocks are then averaged through max pooling. After testing various combinations of parameters, we found that the most efficient setup for this task: $[16, 32, 64]$ convolutional filters, kernel size $= 2$, dropout $= 0.3$, and batch size $= 32$. Training lasted 8 epochs, about 5 seconds each.

3 Results

In this Section we describe the HaSpeeDe 2 tasks of the EVALITA 2020 competition, and we present our results obtained in each of them. To evaluate the degree of generality of our approach, we also tested it on hate speech detection tasks for languages other than Italian, that is English, Spanish and German. The official ranking reported for each run is given in terms of macro-average F-score.

3.1 HaSpeeDe 2 Task A - Hate Speech Detection

This is the main task, and it consists of a binary classification aimed at determining whether the message contains Hate Speech or not. We fine-tuned the parameters for this task and then we used the model as it is for the other tasks. We were provided with a labeled training set – made of tweets only – and two unlabeled test sets: one containing in-domain data, i.e. tweets, and the other out-of-domain data, i.e. news headlines. Our results for both Task A test sets are reported in Table 2.

Test data	Model	Rank	F1
news	Transformer	*1/27*	0.7744
news	DSC	4/27	0.7183
tweets	Transformer	3/27	0.7893
tweets	DSC	5/27	0.7782

Table 2: Results of the HaSpeeDe 2 Task A

3.2 HaSpeeDe 2 Task B - Stereotype Detection

Task B is a binary classification aimed at determining whether the message contains stereotypes or not. The task is motivated by the fact that stereotypes constitute a common source of error in HS identification (Francesconi et al., 2019). Task B data sets are the same as Task A. Our results for both the in-domain and out-of-domain test sets are reported in Table 3.

Test data	Model	Rank	F1
news	Transformer	*1/12*	0.7203
news	DSC	*2/12*	0.7184
tweets	Transformer	3/12	0.7615
tweets	DSC	5/12	0.7386

Table 3: Results of the HaSpeeDe 2 Task B

3.3 Multilingual Detection of Hate Speech

We tested our systems also against data sets coming from either Hate Speech or Offensive Language detection tasks for other languages.

	English	Spanish
Min	0.3500	0.4930
Mean	0,4484	0.6821
Median	0.4500	0.7010
Max	0.6510	0.7300
Transformer	*0.6041*	*0.7423*
DSC	*0.5823*	*0.7375*

Table 4: Results of the HatEval Subtask A

Table 4 reports the results of SemEval 2019 Task 5 (HateEval) (Basile et al., 2019) about the binary detection of hate speech against immigrants and women in Spanish and English messages extracted from Twitter.

	German
Min	0,5487
Mean	0,7151
Median	0.7295
Max	0.7695
Transformer	*0,7384*
DSC	*0,7240*

Table 5: Results of the GermEval 2019 Task 2

Table 5 shows the results of GermEval 2019 Task 2 - Subtask A (Struß et al., 2019). The pourpose of this task is to initiate and foster research on the binary identification of offensive content in German language micro-posts.

4 Discussion

As it can be seen in the results, the Transformer model has always outperformed the DSC model: we expected this outcome due to the nature of the DSC model, designed to be as light as possible but still performing. Regarding the results obtained with the Italian language, we are satisfied with our implementations which have achieved excellent ranking positions in all tasks. In particular, the Transformer model outperformed all the systems that participated to the tasks ranking first with out-of-domain data. This can be seen as a great ability of our model to generalize starting from a training data set different from that of the application. Regarding the results obtained with in-domain data we performed slightly worse, ranking third. This is probably due to the PoS Tagging model that we used in fact it is a model trained on Wikipedia and not on social language, even if slightly modified to manage hashtags, emoticons and URLs, it certainly does not perform well on social texts as if it were a purely PoS Tagging model trained on social media language.

As regards the results obtained with the other languages, we can see that with the Spanish language we get an excellent result, surpassing the first official ranked of the HatEval 2019 competition in Spanish. Our models do not achieve as good results as that of English and German even if the Transformer's score is always above the median value. We think that this is caused by the nature of languages, because Germanic languages, such as English and German, probably benefit less than Latin ones from the additional use of the PoS Tagging, in the way we used it. We are still investigating how to get added value from PoS Tagging for the English and German languages.

5 Conclusion

In this paper we have introduced two systems for the hate speech detection of social media texts in Italian, Spanish, English and German language. The main feature of these models is to use as input to the neural network not only the pre-processed text, but also it's PoS Tag. We are satisfied with the results obtained, because the systems implemented are light and performing. Furthermore we have shown that the use of models that include the additional use of the PoS Tagging, to give it more meaning, has given an added value, reached the top positions in the tasks ranking. Our future work will focus on injecting more and more the grammatical structure of a sentence into a model, in fact we are planning a language model that does not only have the purpose of predicting a word based on the given context but that it is also capable of predicting the PoS Tag of that word.

References

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on Italian tweets. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, Atlanta.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606*.

Jose Camacho-Collados, Yerai Doval, Eugenio Martínez-Cámara, Luis Espinosa-Anke, Francesco Barbieri, and Steven Schockaert. 2020. Learning Cross-lingual Embeddings from Twitter via Distant Supervision. In *Proceedings of ICWSM*.

Gloria Comandini and Viviana Patti. 2019. An Impossible Dialogue! Nominal Utterances and Populist Rhetoric in an Italian Twitter Corpus of Hate Speech against Immigrants. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 163–171. Association for Computational Linguistics.

Gloria Comandini, Manuela Speranza, and Bernardo Magnini. 2018. Effective Communication without Verbs? Sure! Identification of Nominal Utterances in Italian Social Media Texts. In *Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Torino,Italy, December 10-12, 2018, volume 2253 of CEUR Workshop Proceedings*. CEUR.org, 12.

Chiara Francesconi, Cristina Bosco, Fabio Poletto, and M. Sanguinetti. 2019. Error Analysis in a Hate Speech Detection Task: The Case of HaSpeeDe-TW at EVALITA 2018. In *CLiC-it*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October. Association for Computational Linguistics.

Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, and Ophir Frieder. 2019. Hate speech detection: Challenges and solutions. *PLOS ONE*, 14(8):1–16, 08.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May. ELRA. `http://is.muni.cz/publication/884893/en`.

Manuela Sanguinetti, Cristina Bosco, Alberto Lavelli, Alessandro Mazzei, Oronzo Antonelli, and Fabio Tamburini. 2018. PoSTWITA-UD: an Italian Twitter treebank in Universal Dependencies. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May. European Language Resources Association (ELRA).

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Milan Straka. 2018. UDPipe 2.0 Prototype at CoNLL 2018 UD Shared Task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium, October. Association for Computational Linguistics.

Julia Struß, Melanie Siegel, Josef Ruppenhofer, Michael Wiegand, and Manfred Klenner. 2019. Overview of GermEval Task 2, 2019 Shared Task on the Identification of Offensive Language. In *"Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019)"*, 10.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need.

Montanti @ HaSpeeDe2 EVALITA 2020: Hate Speech Detection in Online Contents

Elia Bisconti
University of Pisa
`eliabisconti@gmail.com`

Matteo Montagnani
University of Pisa
`matteo.montagnani8@gmail.com`

Abstract

English. This report describes an approach to face a task regarding the identification of hate content and stereotypes within tweets. Two models will be shown, both presented to the *HaSpeeDe* competition proposed by EVALITA 2020. They are based on a Logistic Regression model that takes different types of embedding as input. The best system shows interesting results.

Italiano. *In questa relazione viene mostrato un approccio volto ad affrontare un task riguardante l'identificazione di contenuti d'odio e stereotipi all'interno di tweets. Sono stati realizzati due modelli, presentati alla competizione HaSpeeDe proposta da EVALITA 2020. Entrambi si basano su un modello di Logistic Regression che prende in input diversi tipi di embedding. Il miglior sistema evidenzia dei risultanti interessanti.*

1 Introduction

The use of *bad words* and *bad language* has always been a subject of debate. The spread of social media platforms, such as Twitter and Facebook, has fostered the growth of hate speech online. These sites have been urged to treat and remove offensive content, but the phenomenon is so pervasive that the manual way of filtering out hateful tweets is not enough. For that reason, the development of automatic recognition systems is increasingly important. To date, the use of Natural Language Processing (Bird et al., 2009) is fundamental in this field. Most of the systems proposed so far are based on manual feature extraction (Joulin et al., 2016), even if in recent years some approaches based on Deep Learning techniques (Badjatiya et al., 2017) have been proposed. EVALITA organized the second edition of an NLP competition for *Hate Speech Detection* (Basile et al., 2020), intending to analyze various techniques for automatic recognition systems.The main goal was to classify a sentence as *hate speech* or even as *stereotyping*. The organizers provided us an in-domain dataset for training and testing and another out-domain. In this report, we will show a classical supervised approach with the aim of obtaining good results regarding the out-of-domain test.

2 Tasks Description

The task proposed in the competition (Sanguinetti et al., 2020) consists of three parts, but only the first two ones will be examined in this article; they correspond to the following sub-tasks:

- **Subtask A** - *Hate Speech Detection*: it consists of a binary classification task aimed at determining the presence or the absence of hateful content in the text towards a given target.
- **Subtask B** - *Stereotype Detection*: it consists of a binary classification task aimed at determining the presence or the absence of a stereotype, therefore an oversimplified opinion, prejudiced attitude, or uncritical judgment, toward a given target. This aims to boost the investigation of its occurrences, especially in a hateful context.

The performances of the participating systems are evaluated on a corpus of Italian tweets as in the previous edition and also on a set of mixed text genres, such as newspapers, comments and headlines.

---

3 Dataset

The dataset used is the one provided by the competition organizers. In particular, the entire dataset is split into one Training Set composed of tweets and two test sets: an in-domain (based on tweets) and a smaller out-of-domain (based on newspaper phrases) test set. Overall, the Training Set includes 6,839 Italian tweets distributed as in Tables 1 and 2.

	Hate Speech	Not Hate Speech
TR Set	2766	4073

Table 1: Distribution of Hate Speech on the Training-set

	Stereotype	Not Stereotype
TR Set	3042	3797

Table 2: Distribution of Stereotype on the Training-set

As we can see, the data are not well distributed. Regarding the Hate Speech Training Set, we have that sixty percent of the data are classified as *hate speech*. The Stereotype Training Set is also a little unbalanced, with fifty-five percent of the data classified as *non-stereotype*.

4 Proposed Approach

In this section, the proposed approaches will be described, focusing on what has been developed for the preprocessing of data, the used embeddings and models. Some decisions regarding the choice of models and the extraction of features were made based on the results obtained in other related works.

4.1 Preprocessing

A Tweet is a text message with a maximum length of 280 characters. It may contain elements such as hashtags, mentions, links and emoticons.
An example of a tweet extracted from the dataset is shown below:

"@user La società multirazziale... #migranti #profughi #rom URL"

As we can see in the example, the dataset provided has already been preprocessed, censuring names and URLs, probably for privacy. The preprocessing phase that we faced implements a series of functions aimed at modifying a tweet to eliminate useless elements and to standardize it. Punctuation, emoji and any symbols are also eliminated. The tweet is also transformed into a *lower case* representation as shown:

"la società multirazziale migranti profughi rom"

Regarding this phase, the transformation of the single words from an inflected form to root or canonical form was also carried out, respectively, through *stemming* and *lemmatization*. We tried to consider these characteristics during the feature selection phase. However, these attempts will not be mentioned further, as they did not produce relevant results.

4.2 Feature vectors

The preprocessed tweets were used to generate the feature useful for classification purposes. Both tasks were addressed with the same types of representation and the same models.

- *TF-IDF Vector:* (Qaiser and Ali, 2018) the idea for the use of this function was to give more importance to the less frequent, but relevant, words. The vectors were generated using the *TfidfVectorizer* class present in the *scikit-learn* library.
- *DistilBert:* (Wolf, 2019) this is a pre-trained model. A single output vector with a size of 768 is considered, corresponding to the result of the first position of what the model received in input, that is the special token [CLS], used for the sentence-level classification.
- *GloVe:* (Pennington et al., 2014) we used a pre-trained model that returns a vector representation of words. The database, extracted from Twitter, includes more than 2 billion phrases, which generated about 27 billion tokens.

These three types of features were used both individually and in combination with each other by concatenation. To decrease the size of these vectors and to speed up the training phase, a *features Selection* phase is also performed using a *Random Forest Classifier*.

5 Systems and Results

For both tasks, we tried the use of an SVM Classifier with *kernel RBF*, a Logistic Regression and a Random Forest. As already mentioned, each of these models has taken various concatenations of the previous feature vectors as input.

We tested each model using 3-fold cross-validation and performed a grid-search to iterate over the models and all the parameters.

As a result of this search, the best final model was undoubtedly the Logistic Regression that has performed well also in previous papers (Davidson et al., 2017). As for the input features, we expected that the concatenation of features extracted with the different techniques described above would lead to the best results. Unexpectedly, instead, the best results were obtained in the validation phase with the use of TFIDF only. The second best one was obtained with the TFIDF concatenated with the DistilBert vectors. These two systems represent the two runs submitted to the competition. Overall, the difference in the results between the first and the second model is considerable; therefore, we will show in the following table the F1 values obtained with the best run, for tasks A and B, respectively.

TaskA	Tweets TS		News TS	
F-score	NoHS	HS	NoHS	HS
	0.750	0.735	0.835	0.615
M-F1	0.7432		0.7256	

Table 3: Task A - Results for the Logistic Regression with Tfidf

TaskB	Tweets TS		News TS	
F-score	NoST	ST	NoST	ST
	0.724	0.690	0.824	0.608
M-F1	0.7076		0.7166	

Table 4: Task A - Results for the Logistic Regression with Tfidf

Beyond the macro-F1 values obtained, it is interesting to note the behavior of the model with regard to the out-domain Test Set in both tasks. In particular, the F-scores show worse values in the classification of sentences that actually contain hate speech or stereotyping. This is actually due to low Recall values (about 0.51 for both tasks) which is probably due to the fact that the model is trained on a different type of data.

6 Discussion

Observing the results on the in-domain Test Set, our best models obtained a ranking of 15/27 and 6/12 respectively for tasks A and B. Regarding the out-domain Test Set, they obtained the third-best score in both tasks. The result obtained with the first Test Set confirms that the proposed approach turned out to be too simplistic. However, it's interesting to notice how such a simple system achieved a good placement in the out-of-domain test-set. An explanation of that could be the way the Training Set was preprocessed. In fact, each tweet has been transformed into a plain text, without taking into consideration any characteristic of a 'social' language. This may have positively influenced the model in predicting the out-of-domain classification.

A further observation to be made about the dataset concerns a lack of correlation between the use of *bad words* and the presence of hateful contents in a phrase. This fact shows how Offensive Language Detection and Hate Speech Detection are related topics, but they remain two distinct tasks (Davidson et al., 2017). Also, many times these kinds of bad words are probably used in an ironic way or to emphasize a sentence, especially in the Italian language.

7 Conclusion

The participation in the Hate Speech Detection 2020 competition proposed by Evalita is derived from purely academic purposes.

We focused on using different types and combinations of embeddings. Surprisingly, the best results were obtained with the use of Tfidf only instead of the use of a combination of more sophisticated embeddings such as GloVe and DistilBert. After a feature selection phase carried out through a Random Forest, the results obtained through a Linear SVM and a Logistic Regression were compared. The latter was the best.

We are aware that the presented system does not introduce new elements with respect to the state of the art of current technologies. Despite this, it was interesting to observe the different results obtained in relation to the composition of the Test Set.

The project was completely developed in python, and the code is publicly available at the following link:

https://github.com/eliabisconti/haspeede

References

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. 14:1532–1543.

Shahzad Qaiser and Ramsha Ali. 2018. Text mining: Use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications*, 181, 07.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Victor Sanh Lysandre Debut Julien Chaumond Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

SardiStance: Stance Detection

SardiStance @ EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets

Alessandra Teresa Cignarella[1,2], **Mirko Lai**[1], **Cristina Bosco**[1], **Viviana Patti**[1] **and Paolo Rosso**[2]
1. Dipartimento di Informatica, Università degli Studi di Torino, Italy
2. PRHLT Research Center, Universitat Politècnica de València, Spain
`{lai,cigna,bosco,patti}@di.unito.it`, `prosso@dsic.upv.es`

Abstract

English. *SardiStance* is the first shared task for Italian on the automatic classification of stance in tweets. It is articulated in two different settings: A) *Textual Stance Detection*, exploiting only the information provided by the tweet, and B) *Contextual Stance Detection*, with the addition of information on the tweet itself such as the number of retweets, the number of favours or the date of posting; contextual information about the author, such as follower count, location, user's biography; and additional knowledge extracted from the user's network of friends, followers, retweets, quotes and replies. The task has been one of the most participated at EVALITA 2020 (Basile et al., 2020), with a total of 22 submitted runs for Task A, and 13 for Task B, and 12 different participating teams from both academia and industry.

1 Introduction/Motivation

The interest towards detecting people's opinions towards particular targets, and towards monitoring politically polarized debates on Twitter has grown more and more in the last years, as it is attested by the proliferation of questionnaires and polls online (Küçük and Can, 2020). In fact, through the constant monitoring of people's opinion, desires, complaints and beliefs on political agenda or public services, policy makers could better meet population's needs.

In the fields of Natural Language Processing and Sentiment Analysis, this translates into the creation of a specifically dedicated task, namely: *Stance Detection* (SD), which is defined as the task of automatically determining from the text whether the author of a given textual content is in favor of, against, or neutral towards a certain target. Research on this topic, beyond mere academic interest, could have an impact on different aspects of everyday life such as public administration, policy-making, marketing or security strategies.

Although SD is a fairly recent research topic, considerable effort has been devoted to the creation of stance-annotated datasets. In their recent survey on this topic, Küçük and Can (2020) describe the existence of a variety of stance-annotated datasets (different text types such as tweets, posts in online forums, news articles, or news comments) for at least eleven languages.

The first shared task on SD was held for English at SemEval in 2016, i.e. *Task 6 "Detecting Stance in Tweets"* (Mohammad et al., 2016b) for detecting stance towards six different targets of interest: "Hillary Clinton", "Feminist Movement", "Legalization of Abortion", "Atheism", "Donald Trump", and "Climate Change is a Real Concern". A more recent evaluation for SD systems was proposed at *IberEval 2017* for both Catalan and Spanish (Taulé et al., 2017) where the target was only one, i.e. "Independence of Catalonia". A re-run was proposed the following year at the evaluation campaign *IberEval 2018* regarding the target "*Catalan first of October Referendum*" encouraging furthermore an exploration of multimodal expressions such as audio, videos and images (Taulé et al., 2018).

SardiStance@EVALITA2020 is the pioneer task for SD in Italian tweets. The motivation behind the proposal of this task is multi-faceted. On the one hand, we aimed at the creation of a new annotated dataset for SD in Italian which would enrich the panorama of available resources for this language, such as CONREF-STANCE-ITA (Lai et al., 2018)

---

and X-STANCE (Vamvas and Sennrich, 2020). On the other hand, the organization of this task allows us a deeper investigation of SD at a *contextual* level, by encouraging the participants and the research community to follow this research line that has proved promising in previous work, see e.g. Lai et al. (2019), Lai et al. (2020) and Del Tredici et al. (2019). In fact, with the data distributed in Task B different types of social network communities, based on friendships, retweets, quotes, and replies could be investigated, in order to analyze the communication among users with similar and divergent viewpoints.
The efficacy of approaches based on contextual features paired with textual information has been widely attested in literature on SD (Magdy et al., 2016; Rajadesingan and Liu, 2014) and additionally confirmed by the results obtained in this shared task, especially by those teams who participated to both settings (see Section 5).

2 Definition of the Task

With this task proposal, we wanted to invite participants to explore features based on the textual content of the tweet, such as structural, stylistic, and affective features, but also features based on contextual information that does not emerge directly from the text, such as knowledge about the domain of the political debate or information about the user's community. For these reasons, we proposed two different settings:

- **Task A - Textual Stance Detection:**

The first task was a three-class classification task where the system had to predict whether a tweet is in FAVOUR, AGAINST or NONE towards the given target, exploiting only textual information, i.e. the text of the tweet.

From reading the tweet, which of the options below is most likely to be true about the tweeter's stance towards the target? (Mohammad et al., 2016a)

1. **FAVOUR**: We can infer from the tweet that the tweeter supports the target.
2. **AGAINST**: We can infer from the tweet that the tweeter is against the target.
3. **NONE**: We can infer from the tweet that the tweeter has a neutral stance towards the target or there is no clue in the tweet to reveal the stance of the tweeter towards the target.

- **Task B - Contextual Stance Detection:**

The second task was the same as the first one: a three-class classification task where the system had to predict whether a tweet is in FAVOUR, AGAINST or NONE towards the given target. Here participants had access to a wider range of contextual information based on the **post** such as: the number of retweets, the number of favours, the number of replies and the number of quotes received to the tweet, the type of posting source (e.g. iOS or Android), and date of posting. Furthermore we shared (and encouraged its exploitation) contextual information related to the **user**, such as: number of tweets ever posted, user's bio, user's number of followers, user's number of friends. Additionally we shared users' contextual information about their **social network**, such as: friends, replies, retweets, and quotes' relations. The personal ids of the users were anonymized but their network structures were maintained intact.
Participants could decide to participate to both tasks or only to one. Although they were encouraged to participate to both.

3 Data

We chose to gather the data from the social networking Twitter due to the free availability of a huge amount of users' generated data and because it allowed us to explore different types of relations among the users involved in a debate.

3.1 Collection and annotation of the data

We collected around 700K tweets written in Italian about the "Movimento delle Sardine" (*Sardines movement*[1]), retrieving tweets containing the keywords "sardina", "sardine", and the homonymous hashtags. Furthermore, we collected all the conversation threads in which the said tweet belongs, iteratively following the reply's tree. We also collected the quoted tweets and the list of all the retweets of each previously recovered tweet, obtaining about 1M tweets. Finally, we collected the friend list of all the users included in the annotated dataset.

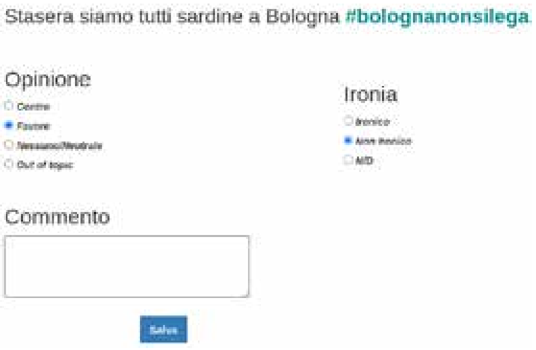
The tweets were gathered between the 46th week of 2019 (November) and the 5th week of 2020 (January), corresponding to a 12 weeks time-window. Through the experience matured as participants in previous shared tasks of SD, and in or-

---

[1] `https://en.wikipedia.org/wiki/Sardines_movement`.

der to reduce noise in text, we collected data taking into account the following constraints: only one tweet per author for each week, no retweets, no replies, no quotes, no tweets containing URLs, no tweets containing pictures or videos.

Then, we included only Italian tweets posted using a limited number of "sources" (utilities used to post the tweet, such as iOS, Android, etc...) in order to avoid to include pre-written tweets posted using a *Tweet button*.[2] Furthermore, we validated that all the collected tweets presented a *Jaccard similarity coefficient* < 0.8. From about 25K filtered tweets, we finally randomly selected around 300 tweets for each week (only the first week of 2020 does not reach 300 tweets), thus obtaining 3,600 tweets in total.



Stasera siamo tutti sardine a Bologna #bolognanonsilega

Opinione

○ Contro
● Favore
○ Nessuno/Neutrale
○ Out of topic

Ironia

○ Ironico
● Non Ironico
○ N/D

Commento

Figure 1: Platform for the annotation of tweets.

We created a web platform for annotation purposes, see Figure 1, in order to facilitate the labelling task to the annotators, unifying the visualization mode and shuffling the tweets in a random order.[3] 12 different native Italian speakers with an interest for news and politics were involved in the annotation, according to detailed guidelines we provided with examples for annotation and examples in their native language. We randomly shuffled the annotators and matched them into 66 pairs in which each pair would annotate 55 tweets. As a result, each annotator labelled 605 tweets independently and each tweet was annotated by two annotators, who had to choose among four different labels: AGAINST, FAVOUR, NONE/NEUTRAL and OUT OF TOPIC.

Furthermore, as it can also be seen in Figure 1 (*Tonight we are all sardines in Bologna #bolognanonsilega*), we asked the annotators to mark whether, in their opinion, the tweet was IRONIC or NOT IRONIC. Finally, we were not able to obtain satisfactory results on this end, so we did not include it in the task.

3.2 Analysis of the annotation

At the end of a first phase of annotation, which lasted more or less a month, we obtained 2,256 tweets in agreement, with a clear decision on one of the three main classes. Other 917 tweets presented a *light disagreement* (i.e. FAVOUR VS. NEUTRAL or AGAINST VS. NEUTRAL), and the remaining 457 tweets were discarded because the majority of annotators considered them out of topic or were in *strong disagreement* (i.e. FAVOUR VS. OUT OF TOPIC).

We then proceeded in the resolution of those 917 tweets, whose disagreement was deemed "light" in order to obtain a bigger dataset. We resorted once again to the annotation platform used in the first phase, we revised the annotation guidelines and asked the annotators to label the tweets again. In this phase, we paid attention that the tweets in disagreement were not assigned to the same pair of annotators that had previously labelled them, and furthermore we chose to show the two annotations in contrast, along with any comment - if present - to the annotator that had to solve the disagreement.

After the second phase, we computed the inter-annotator agreement (IAA) through Cohen's kappa coefficient (over the three main classes) resulting in $\kappa = 0.493$ (weak agreement). The same coefficient was also used to compute the IAA among annotators over the two most significant classes (AGAINST and FAVOUR, excluding the NEUTRAL class), resulting in a higher score: $\kappa = 0.769$ (moderate agreement). Notably, we observed that the IAA significantly changes depending on the observed pair of annotators (it ranges from 0.873 to 0.473) in the first phase of the annotation. We also noticed that the average IAA, computed through the sum of each IAA between any annotator and the remaining 11 annotators, can significantly change (ranging from 0.704 to 0.609). In other words, some annotators tend to strongly agree with all the other ones, while others tend to disagree with the majority. As future work,

---

[2] `https://developer.twitter.com/en/docs/twitter-for-websites/tweet-button/overview`.

[3] In this way, each annotator was surely seeing emojis – which, we believe are essential in order to understand the correct stance– in the same way of the other annotators independently of the device used.

we aim to shed more light on this phenomena exploring the background of the annotators and the social relationship among them.

3.3 Composition of the dataset

After the second round of annotation we were finally able to create the official dataset for the *SardiStance* shared task. It is composed by a total of 3,242 tweets, 1,770 of which belong to the class AGAINST, 785 to the class FAVOUR, and 687 to the class NONE. In Table 1 we show the distribution of such instances accordingly to the training set and the test set and in Table 2 we report tweet as example for each class.

TRAINING SET			TEST SET		
AGAINST	FAVOUR	NONE	AGAINST	FAVOUR	NONE
1,028	589	515	742	196	172
	2,132			**1,110**	

Table 1: Distribution of tweets.

text	label
LE SARDINE IN PIAZZA MAGGIORE NON SONO ITALIANI SE LO FOSSERO NON SI METTEREBBERO CONTRO LA DESTRA CHE AMA L'ITALIA E VUOLE RIMANERE ITALIANA *THE SARDINES IN PIAZZA MAGGIORE ARE NOT ITALIAN IF THEY WERE THEY WOULD NOT GO AGAINST THE RIGHT THAT LOVES ITALY AND WANTS TO REMAIN ITALIAN*	AGAINST
Non ci credo che stasera devo andare in teatro e non posso essere fra le #Sardine #Bologna #bolognanonsilega *I can't believe that I have to go to the theater tonight and I can't be among the #Sardines #Bologna #bolognanonsilega*	FAVOUR
Mi sono svegliato nudo e triste perché a Bologna, tra salviniani e antisalviniani, non mi ha cagato nessuno. *I woke up naked and sad because in Bologna, between Salvinians and anti-Salvinians, nobody paid me attention.*	NONE

Table 2: Examples from the dataset.

3.4 Data Release

We shared data following the methodology recommended in (Rangel and Rosso, 2018) in order to comply to GDPR privacy rules and Twitter's policies. The identifiers of tweets and users have been anonymized and replaced by unique identifiers. We exclusively released the emojis eventually contained in the location and description user's biography, in order to make very hard to trace users and to preserve everybody's privacy.

Task A

The training data (TRAIN.csv) was released in the following format:

`tweet_id user_id text label`

where `tweet_id` is the Twitter ID of the message, `user_id` is the Twitter ID of the user who posted the message, `text` is the content of the message, `label` is AGAINST, FAVOUR or NONE.

Task B

In order to participate to Task B, we released additional contextual information.

- the file TWEET.csv, containing contextual information regarding the tweet, with the following format:

`tweet_id user_id retweet_count favorite_count source created_at`

where `tweet_id` is the Twitter ID of the message, `user_id` is the Twitter ID of the user who posted the message, `retweet_count` indicates the number of times the tweet has been retweeted, `favorite_count` indicates the number of times the tweet has been liked, `source` indicates the type of posting source (e.g. iOS or Android), and `created_at` displays the time of creation according to a yyyy-mm-dd hh:mm:ss format. Minutes and seconds have been encrypted and transformed to zeroes for privacy issues.

- the file USER.csv, containing contextual information regarding the user. It was released in the following format:

`user_id statuses_count friends_count followers_count created_at emoji`

where `user_id` is the Twitter ID of the user who posted the message, `statuses_count`, `friends_count` indicates the number of friends of the user, `followers_count` indicates the number of followers of the user, `created_at` displays the time of the user registration on Twitter, and `emoji` shows a list of the emojis in the user's bio (if present, otherwise the field is left empty).

- The files FRIEND.csv, QUOTE.csv, REPLY.csv and RETWEET.csv containing contextual info about the social network of the user. Each file was released in the following format:

`Source Target Weight`

where `Source` and `Target` indicate two nodes of a social interaction between two Twitter users. More specifically, the source user performs one of the considered social relation towards the target user. Two users are tied by a friend relationship if the source user follows the target user (friend relationship does not have a weight, because it is either present or absent); while two users are tied by a quote, retweet, or reply relationship if the source user respectively quoted, retweeted, or replied the target user. Table 4 shows some metrics about the shared networks.

	nodes	**edges**
friend	669,817	3,076,281
retweet	110,315	575,460
quote	2,903	7,899
reply	14,268	29,939

Table 4: Networks metrics.

`Weight` indicates the number of interactions existing between two users. Note that this information is not available for the friend relation (hence, this column was not present in the FRIEND.csv file) due to the fact that it is a relationship of the type present/absent and cannot be described through a weight. In all the files, users are defined by their anonimyzed User ID.

Regrettably, we did not think to anonymize the screen names contained in the text of the tweets (with the same numeric string used to anonymize users), for allowing to match it with the users' ids and allowing the exploration of the network based on mentions. We will surely take it into account in our future works.

4 Evaluation Measures

Each participating team was allowed to submit a maximum of 4 runs for each sub-task: two constrained runs and two unconstrained runs. Submitting at least a constrained run was anyway compulsory. We decided to provide two separate official rankings for Task A and Task B, and two separate ranking for constrained and unconstrained runs. Systems have been evaluated using F1-score computed over the two main classes (FAVOUR and AGAINST). Therefore, the submissions have been ranked by the averaged F1-score over the two classes, according the following equation: $F1_{avg} = (F1_{favour} + F1_{against})/2$.

4.1 Baselines

We computed a baseline using a simple machine learning model, for Task A: a Support Vector Classifier based on token uni-gram features. A second baseline we computed for Task B is a system based on our previous work on Stance Detection: a Logistic Regression classifier paired with token n-grams features (unigrams, bigrams and trigrams), plus features based on a binary one-hot encoding representation of the communities extracted from the network of retweets and the network of friends (see the best system for Italian, in Lai et al. (2020)).

5 Participants and results

A total of 12 teams, both from academia and industry sector participated to at least one of the two tasks of SardiStance. In Table 3 we provide an overview of the teams in alphabetical order.

Teams were allowed to submit up to four runs (2 constrained and 2 unconstrained) in case they implemented different systems. Furthermore, each team had to submit at least a constrained run. Participants have been invited to submit multiple runs to experiment with different models and architectures. However, they have been discouraged from

team name	**institution**	**report**	**task**
deepreading	UNED, Spain	(Espinosa et al., 2020)	A, B
GhostWriter	You Are My Guide, Italy	(Bennici, 2020)	A, B
IXA	UPV/EHU, Spain	(Espinosa et al., 2020)	A, B
MeSoVe	ISASI, Italy	-	A
QMUL-SDS	QMUL-SDS-EECS, UK	(Alkhalifa and Zubiaga, 2020)	A, B
SSN_NLP	CSE Department/SSNCE, India	(Kayalvizhi et al., 2020)	A
SSNCSE-NLP	SSN College of Engineering, India	(Bharathi et al., 2020)	A, B
TextWiller	UNIPD, Italy	(Ferraccioli et al., 2020)	A, B
UNED	UPV/EHU and UNED, Spain	(Espinosa et al., 2020)	B
UninaStudents	UNINA, Italy	(Moraca et al., 2020)	A
UNITOR	UNIROMA2, Italy	(Giorgioni et al., 2020)	A
Venses	UNIVE, Italy	(Delmonte, 2020)	A

Table 3: Participants and reports.

submitting slight variations of the same model. Overall we have 22 runs for Task A and 13 runs for Task B.

5.1 Task A: Textual Stance Detection

Table 5 shows the results for the textual stance detection task, which attracted 22 total submissions from 11 different teams. Since the only two systems in an unconstrained setting were submitted by the same team we decided not to create a separate ranking for them, but rather to include them in the same ranking, and marking them with a different color (gray in Table 5).

team name	**run**	**F1-score**			
		AVG	AGAINST	FAVOUR	NONE
UNITOR	1	**.6853**	.7866	**.5840**	.3910
UNITOR	1	**.6801**	.7881	**.5721**	.3979
UNITOR	2	.6793	**.7939**	.5647	.3672
DeepReading	1	.6621	.7580	.5663	**.4213**
UNITOR	2	.6606	.7689	.5522	.3702
IXA	1	.6473	.7616	.5330	.3888
GhostWriter	1	.6257	.7502	.5012	.3810
IXA	2	.6171	.7543	.4800	.3675
SSNCSE-NLP	2	.6067	.7723	.4412	.2113
DeepReading	2	.6004	.6966	.5042	.3916
GhostWriter	2	.6004	.7224	.4784	.3778
UninaStudents	1	.5886	.7850	.3922	.2326
baseline		*.5784*	*.7158*	*.4409*	*.2764*
TextWiller	1	.5773	.7755	.3791	.1849
SSNCSE-NLP	1	.5749	.7307	.4192	.3388
QMUL-SDS	1	.5595	.7091	.4099	.2313
QMUL-SDS	2	.5329	.6478	.4181	.3049
MeSoVe	1	.4989	.7336	.2642	.3118
TextWiller	2	.4715	.6713	.2718	.2884
SSN_NLP	1	.4707	.5763	.3651	.3364
SSN_NLP	2	.4473	.6545	.2402	.1913
Venses	1	.3882	.5325	.2438	.2022
Venses	2	.3637	.4564	.2710	.2387

Table 5: Results Task A.

The best results are achieved by the UNITOR team that, with an unconstrained, ranked as 1st position with F1_{avg} = 0.6853. The best result for the constrained runs is achieved once again by the UNITOR team with F1_{avg} = 0.6801.

The best results for the two main classes AGAINST and FAVOR are obtained by the three best systems of the ranking, which are all submissions by the team UNITOR. On the other hand, though, the Deepreading team, ranking as 4th, has obtained the best F1-score for the NONE class, with F1_{none} = 0.4213.

Among the 12 participating teams, at least 6 show an improvement over the baseline, which was computed using an SVM paired with token unigrams as unique feature, resulting an already strong result to beat (F1_{avg} = 0.5784).

5.2 Task B: Contextual Stance Detection

Table 6 shows the results for the contextual stance detection task, which attracted 13 total submissions from 7 different teams.

team name	**run**	**F1-score**			
		AVG	AGAINST	FAVOUR	NONE
IXA	3	**.7445**	**.8562**	**.6329**	.4214
TextWiller	1	.7309	.8505	.6114	.2963
DeepReading	1	.7230	.8368	.6093	.3364
DeepReading	2	.7222	.8300	.6143	**.4251**
TextWiller	2	.7147	.8298	.5995	.3680
QMUL-SDS	1	.7088	.8267	.5908	.1811
UNED	2	.6888	.8175	.5600	.2455
QMUL-SDS	2	.6765	.8134	.5396	.1553
SSNCSE-NLP	2	.6582	.7915	.5249	.3691
SSNCSE-NLP	1	.6556	.7914	.5198	.3880
baseline		*.6284*	*.7672*	*.4895*	*.3009*
GhostWriter	1	.6257	.7502	.5012	.3810
GhostWriter	2	.6004	.7224	.4784	.3778
UNED	1	.5313	.7399	.3226	.2000

Table 6: Results Task B.

The best scores are achieved by the IXA team that with a constrained run obtained the highest score of F1_{avg} = 0.7445. The best F1-score for the main classes AGAINST and FAVOUR is achieved by the team ranked 1st, IXA, team with $\text{F1}_{against}$ = 0.8562, and F1_{favour} = 0.6329, respectively. Once again, the Deepreading team, ranking 3rd and 4th, has obtained the best F1-score for the NONE class, with F1_{none} = 0.4251.

Almost all participating systems show an improvement over the baseline, which was computed using a Logistic Regression classifier paired with token n-grams features (unigrams, bigrams and trigrams), features based on the network of retweets, and features based on the network of friends (Lai et al., 2020).

6 Discussion

In this section we compare the participating systems according to the following main dimensions: system architecture, features, use of additional annotated data for training, and use of external resources (e.g. sentiment lexica, NLP tools, etc.). We also operate a distinction between runs submitted in Task A and those submitted in Task B. This discussion is based on the participants' reports and the answers the participants provided to a questionnaire proposed by the organizers. Two teams, namely TextWiller and Venses wrote a joint report, overlapping between this task and the *HaSpeeDe 2* task (Sanguinetti et al., 2020), as they participated in both competitions. The three following teams, Deepreading, IXA, and UNED, also wrote a unique report as the participants, belong to the same research project and wanted to compare their three different approaches.

6.1 Systems participating to Task A

System architecture. Among all submitted runs we counted a great variety of architectures, ranging from classical machine learning classifiers, to recent state-of-the-art approaches, and statistically-based models. For instance, regarding the use of classical ML, the team UninaStudents used a SVM, and the team MeSoVe used Logistic Regression in one run. Regarding the use of neural networks, the QMUL-SDS team used bidirectional-LSTM, a CNN-2D, and a bi-LSTM with attention. Also SSN_NLP exploited the LSTM neural network.

Four teams exploited different variants of the BERT model: Ghostwriter used AlBERTo trained on Italian tweets, IXA used GilBERTo and UmBERTo[4], while UNITOR adopted only this latter model. Finally the Deepreading team made use of transformers such as BERT XXL and XML-RoBERTa, paired together with linear classifiers. TextWiller is the only team to have exploited the *xg-boost* algorithm, and ItVenses relied on supervised models, based on statistics and semantics. The UNED team proposed instead a voting system among the output of different models.

Features. Besides having explored a variety of system architectures, the teams participating in Task A, also used many different textual features, in the most of cases based on n-grams or char-grams. MeSoVe and TextWiller additionally engineered features based on emoticons. The team UNED, in one of their runs, proposed a system relying on psychologycal and social features, while UninaStudents proposed features of uni-grams of hashtags. Interestingly, UNITOR added special tags to the texts, which are the result of a classification with respect some so-called "auxiliary task". In particular, they trained three classifiers based respectively on SENTIPOLC 2016 (Barbieri et al., 2016) for sentiment analysis classification, on HaSpeeDe 2018 (Bosco et al., 2018) for hate speech detection, and on IronITA 2018 (Cignarella et al., 2018) for irony detection; and they added three tags to each instance of the SardiStance datasets with respect to these three dimensions: sentiment, hate and irony. ItVenses proposed features collected automatically from a unique dictionary list, frequency of occurrence of emojis and emoticons, and semantic features investigating propositional level, factivity and speech act type.

Additional training data. The only team who participated to the unconstrained setting of *SardiStance* is UNITOR. They proposed two unconstrained runs in addition to other two constrained ones. For the unconstrained setting, they downloaded and labeled about 3,200 tweets using distant supervision and used the additional data to train their systems. In particular they created the following subsets:
- 1,500 AGAINST: tweets from 2019 containing the hashtag: #gatticonsalvini;
- 1,000 FAVOUR: tweets from 2019 containing the hashtags: #nessunotocchilesardine, #iostoconlesardine, #unmaredisardine, #vivalesardine and #forzasardine;
- 700 NONE/NEUTRAL: texts derived from news titles. These were retrieved by querying to Google news with the keyword "sardine".

Other resources. Five teams declared to have used also other resources such as lexica, word embeddings, or others. In particular, GhostWriter used grammar model to rephrase the tweets. MeSoVe exploited SenticNet (Cambria et al., 2014) and the "Nuovo vocabolario di base della lingua italiana".[5] QMUL-SDS took advantage of temporal embeddigns and FastText, while only one team, UninaStudents, used a sentiment lexicon: AFINN (Nielsen, 2011). Lastly, Venses used a proprietary lexicon of Italian, enriched with conceptual, semantic and syntactic information; and similarly TextWiller approach relies on a self-created vocabulary and trained word-embeddigs on the corpus PAISÀ (Lyding et al., 2014).

6.2 Systems participating to Task B

Seven teams participated in Task B submitting a total of 13 runs. Most teams extensively explored the additional features available for Task B; GhostWriter, on the contrary, proposes the same

---

[4] `https://huggingface.co/Musixmatch/umberto-commoncrawl-cased-v1`.

[5] `https://dizionario.internazionale.it`.

two approaches presented in Task A. Notably, the three runs with a score lower than the baseline do not have benefited from any features based on the users' social network.

System architecture. Most teams enriched the models they submitted in Task A by taking advantage of contextual information available in Task B. UNED, DeepReading, and TextWiller exploited the *xg-boost* algorithm selecting different features from contextual data. The language model BERT was used in different variants by SSNCSE-NLP, DeepReading, and IXA. In particular, the last two teams proposed three voting based ensemble methods that use two or more models that exploit the *xg-boost* algorithm. Furthermore, the neural network framework proposed by QMUL-SDS exploits and combine four different embedding methods into a dense layer for generating the final label using a *softmax* activation function.

Features. Not every team took full advantage of contextual information. For example, SSNCSE-NLP only exploits the number of friends in run 1, and the number of quotes and friends in run 2. In its run 1 UNED also exploited some features based on the tweets in addition to the psychological and emotional ones, using the *xg-boost* algorithm. The other teams exploited different approaches for learning vector representations of the nodes of the available networks. DeepReading, IXA, and UNED proposed a feature that computes the mean distances of each user to the rest of users whose stance is known. TextWiller experimented a multi-dimensional scaling (MDS) for retaining the first and second dimension for each of the four networks instated. *Node2vec* and *deepwalk* for learning a vector representation of the nodes of the networks were used respectively in QMUL-SDS's runs 1 and 2.

The comparison between the approaches respectively used for dealing with Task A and Task B, clearly highlights the benefits of exploiting information from different and heterogeneous sources. In particular, it is interesting to observe that all the teams that participated to both tasks, also produced better results in the second setting. Experimenting with different classifiers trained with the textual content of the tweets as well as with features based on contextual information (additional info on the tweets, on users, or their social networks) seems therefore to allow to obtain overall better results.

In particular, among the 6 teams that participated to both tasks, only 4 fully explored the social network relations of the author of the tweet. The only two runs that overcome the baseline without investigating the structures of the social graphs are those submitted by the SSNCSE-NLP team. Only one team participated to both tasks exploiting the same architecture. This, allowed us to compare the F1-scores obtained in the first setting with those obtained in the second, highlighting that adding contextual features could increase performance of +0.2432, in terms of $\mathrm{F1}_{avg}$.

Additionally, we calculated the increment in performance between the score obtained by the run ranked as 1st position in Task A (UNITOR, F_{avg} = 0.6853) and the score of the run ranked as 1st position in Task B (IXA, F_{avg} = 0.7445), showing that taking advantage of contextual features could increase performance up to 8,6% in terms of $\mathrm{F1}_{avg}$.

7 Conclusions

We presented the first shared task on Stance Detection for Italian, discussing the development of the datasets used and the participation. A great panel for discussions about techniques and state-of-the-art approaches has been opened which can be used for investigating future research directions.

Acknowledgments

The work of C. Bosco, M. Lai and V. Patti is partially funded by the project "Be Positive!" (under the 2019 "Google.org Impact Challenge on Safety" call). The work of C. Bosco and V. Patti is also partially funded by Progetto di Ateneo/CSP 2016 *Immigrants, Hate and Prejudice in Social Media* (S1618_L2_BOSC_01). The work of P. Rosso is partially funded by the Spanish MICINN under the research projects MISMIS-FAKEnHATE on Misinformation and Miscommunication in social media: FAKE news and HATE speech (PGC2018-096212-B-C31) and PROMETEO/2019/121 (DeepPattern) of the Generalitat Valenciana.

A special mention also to the people who helped us with the annotation of the dataset. In random order: Matteo, Luca, Ylenia, Simona, Elisa, Sebastiano, Francesca, Simona, Komal and Angela, thank you very much for your great help.

References

Rabab Alkhalifa and Arkaitz Zubiaga. 2020. QMUL-SDS @ SardiStance: Leveraging Network Interactions to Boost Performance on Stance Detection using Knowledge Graphs. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the EVALITA 2016 SENTIment POLarity Classification task. In *Proceedings of the 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. CEUR-WS.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR-WS.org.

Mauro Bennici. 2020. ghostwriter19 @ SardiStance: Generating new tweets to classify SardiStance EVALITA 2020 political tweets. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

B. Bharathi, J. Bhuvana, and Nitin Nikamanth Appiah Balaji. 2020. SardiStance@EVALITA2020: Textual and Contextual stance detection from Tweets using machine learning approach. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Cristina Bosco, Felice Dell'Orletta, Fabio Poletto, Manuela Sanguinetti, and Maurizio Tesconi. 2018. Overview of the Evalita 2018 Hate Speech Detection Task. In *Proceedings of 6th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2018)*. CEUR-WS.org.

Erik Cambria, Daniel Olsher, and Dheeraj Rajagopal. 2014. SenticNet 3: a Common and Commonsense Knowledge Base for Cognition-driven Sentiment Analysis. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI 2014)*.

Alessandra Teresa Cignarella, Simona Frenda, Valerio Basile, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2018. Overview of the EVALITA 2018 task on Irony Detection in Italian Tweets (IronITA). In *Proceedings of 6th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2018)*. CEUR-WS.org.

Marco Del Tredici, Diego Marcheggiani, Sabine Schulte im Walde, and Raquel Fernández. 2019. You Shall Know a User by the Company It Keeps: Dynamic Representations for Social Media Users in NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*. ACL.

Rodolfo Delmonte. 2020. Venses @ HaSpeeDe2 & SardiStance: Multilevel Deep Linguistically Based Supervised Approach to Classification. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Maria S. Espinosa, Rodrigo Agerri, Alvaro Rodrigo, and Roberto Centeno. 2020. DeepReading @ SardiStance: Combining Textual, Social and Emotional Features. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Federico Ferraccioli, Andrea Sciandra, Mattia Da Pont, Paolo Girardi, Dario Solari, and Livio Finos. 2020. TextWiller @ SardiStance, HaSpeede2: Text or Con-text? A smart use of social network data in predicting polarization. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Simone Giorgioni, Marcello Politi, Samir Salman, Danilo Croce, and Roberto Basili. 2020. UNITOR@Sardistance2020: Combining Transformer-based architectures and Transfer Learning for robust Stance Detection. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

S. Kayalvizhi, D. Thenmozhi, and Chandrabose Aravindan. 2020. SSN_NLP@SardiStance : Stance Detection from Italian Tweets using RNN and Transformers. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Dilek Küçük and Fazli Can. 2020. Stance detection: A survey. *ACM Computing Surveys*, 53(1):1–37.

Mirko Lai, Viviana Patti, Giancarlo Ruffo, and Paolo Rosso. 2018. Stance evolution and Twitter interactions in an Italian political debate. In *Proceedings of the 23rd International Conference on Natural Language & Information Systems (NLDB 2018)*. Springer.

Mirko Lai, Marcella Tambuscio, Viviana Patti, Giancarlo Ruffo, and Paolo Rosso. 2019. Stance polarity in political debates: A diachronic perspective of network homophily and conversations on twitter. *Data & Knowledge Engineering*, 124:101738.

Mirko Lai, Alessandra Teresa Cignarella, Delia Irazú Hernández Farías, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. Multilingual stance detection in social media political debates. *Computer Speech & Language*, 63(101075).

Verena Lyding, Egon Stemle, Claudia Borghetti, Marco Brunello, Sara Castagnoli, Felice Dell'Orletta, Henrik Dittmann, Alessandro Lenci, and Vito Pirrelli. 2014. The PAISA' Corpus of Italian Web Texts. In *Proceedings of the 9th World Archaeological Congress (WAC-9) @ the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*. ACL.

Walid Magdy, Kareem Darwish, Norah Abokhodair, Afshin Rahimi, and Timothy Baldwin. 2016. #isisisnotislam or #deportallmuslims?: Predicting unspoken views. In *Proceedings of the 8th ACM Conference on Web Science (WebSci 2016)*. ACM.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016a. A Dataset for Detecting Stance in Tweets. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*. ELRA.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016b. SemEval-2016 Task 6: Detecting Stance in Tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. ACL.

Maurizio Moraca, Gianluca Sabella, and Simone Morra. 2020. UninaStudents @ SardiStance: Stance detection in Italian tweets - Task A. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Finn Årup Nielsen. 2011. AFINN. *Richard Petersens Plads, Building*, 321.

Ashwin Rajadesingan and Huan Liu. 2014. Identifying users with opposing opinions in Twitter debates. In *Proceedings of the 7th Social Computing, Behavioral-Cultural Modeling and Prediction International Conference (SBP-BRiMS 2014)*. Springer.

Francisco Rangel and Paolo Rosso. 2018. On the implications of the general data protection regulation on the organisation of evaluation tasks. *Language and Law / Linguagem e Direito*, 5(2):95–117.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. HaSpeeDe 2@EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Mariona Taulé, M. Antònia Martí, Francisco M. Rangel Pardo, Paolo Rosso, Cristina Bosco, and Viviana Patti. 2017. Overview of the Task on Stance and Gender Detection in Tweets on Catalan Independence. In *Proceedings of the 2nd Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2017) co-located with 33th Conference of the Spanish Society for Natural Language Processing (SEPLN 2017)*. CEUR-WS.org.

Mariona Taulé, Francisco M. Rangel Pardo, M. Antònia Martí, and Paolo Rosso. 2018. Overview of the Task on Multimodal Stance Detection in Tweets on Catalan #1Oct Referendum. In *Proceedings of the 3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018) co-located with 34th Conference of the Spanish Society for Natural Language Processing (SEPLN 2018)*. CEUR-WS.org.

Jannis Vamvas and Rico Sennrich. 2020. X-Stance: A Multilingual Multi-Target Dataset for Stance Detection. In *Proceedings of the 5th Swiss Text Analytics Conference (SwissText 2020) & 16th Conference on Natural Language Processing (KONVENS 2020)*. CEUR-WS.org.

UNITOR @ Sardistance2020: Combining Transformer-based Architectures and Transfer Learning for Robust Stance Detection

Simone Giorgioni, Marcello Politi, Samir Salman, Danilo Croce and Roberto Basili
Department of Enterprise Engineering, University of Roma, Tor Vergata
Via del Politecnico 1, 00133 Roma, Italy
`{simone.giorgioni,marcello.politi,samir.salman}@alumni.uniroma2.eu`
`{croce,basili}@info.uniroma2.it`

Abstract

English. This paper describes the UNITOR system that participated to the Stance Detection in Italian tweets (Sardistance) task within the context of EVALITA 2020. UNITOR implements a transformer-based architecture whose accuracy is improved by adopting a Transfer Learning technique. In particular, this work investigates the possible contribution of three auxiliary tasks related to Stance Detection, i.e., Sentiment Detection, Hate Speech Detection and Irony Detection. Moreover, UNITOR relies on an additional dataset automatically downloaded and labeled through distant supervision. The UNITOR system ranked first in Task A within the competition. This confirms the effectiveness of Transformer-based architectures and the beneficial impact of the adopted strategies.

Italiano. *Questo lavoro descrive UNITOR, uno dei sistemi partecipanti allo Stance Detection in Italian tweet (SardiStance) task. UNITOR implementa un'architettura neurale basata su Transformer, la cui accuratezza viene migliorata applicando un metodo di Transfer Learning, che sfrutta le informazioni di tre task ausiliari, ovvero Sentiment Detection, Hate Speech Detection e Irony Detection. Inoltre, l'addestramento di UNITOR puó contare su un insieme di dati scaricati ed etichettati automaticamente applicando un semplice metodo di Distant Supervision. Il sistema si é classificato al primo posto nella competizione, confermando l'efficacia delle architetture basate su Transformer e il contributo delle strategie adottate.*

1 Introduction

Stance detection aims at detecting if the author of a text is in favor of a target topic, or against it (Krejzl et al., 2017). In this task, a text pair is generally considered: one text expresses the topic, while the other one reflects the author's judgments. In a possible variant to such a setting, the topic is implicit within an entire document collection over which the stance detection is applied.

In this work, we will consider this last setting, as defined in the in the Stance Detection in Italian Tweets (SardiStance) task (Cignarella et al., 2020) within the EVALITA 2020 (Basile et al., 2020). A set of texts (here tweets) is provided, almost all concerning the same topic, i.e., the Sardines Movement[1]. The goal is to recognize if each tweet is for or against (or neither) such target, only exploiting textual information. According to the task definition, this corresponds to the so-called Task A. This is quite challenging problem, since it requires at the same time to discover if a text refers to the target topic and the author's orientation, only relying on short messages written in a very conversational style.

We thus present the UNITOR system participating to the SardiStance task A. The system is based on a Transformer-based architecture for text classification (Devlin et al., 2019) that is directly pre-trained over a large-scale document collection written in Italian, namely UmBERTo. In a nutshell, the adopted architecture, which has been demonstrated achieving state-of-the-art results in many NLP tasks (Devlin et al., 2019), takes in input a message and associates it to one of the target classes indicating the stance. Moreover, due to the task complexity and the small size of the dataset, in order to improve the generalization capabilities of the neural network, we adopted a Transfer Learning approach (Pan and Yang, 2010). Our main assumption is that Stance Detection is tied to other tasks involving emotion and subjectivity

---

[1] `https://en.wikipedia.org/wiki/Sardines_movement`

analysis (such as Sentiment Analysis or Irony Detection) even though important differences do exist among them. As a simplified example, let us consider a message such as "*I like the Sardines Movement*": it clearly expresses a positive sentiment, also being in favour of the target topic. However, a message such as "*I like the EVALITA campaign.*" is positive as well but it does not express any support or opposition to the Sardines (and it should be associated to the `None` class). We thus speculate that an automatic system trained over an auxiliary task (e.g., Sentiment Classification) is beneficial, but the transfer process must be carefully designed in order to avoid catastrophic forgetting or interference problems (Mccloskey and Cohen, 1989).

In this work, we investigate the possible contribution of three auxiliary tasks involving the recognition of emotions according to different settings, i.e., Sentiment Detection and Classification, Hate Speech Detection and Irony Detection. We adopt three different classifiers (one for each auxiliary task) and use them to add additional information to the tweets provided in the SardiStance dataset. As an example, when considering the auxiliary task involving Hate Detection, the corresponding classifier will augment each input tweet by expressing if this expresses hate or not. After this step, the final classifier is expected to learn the association between messages and the stance categories, "being aware" (with some unavoidable noise) if the message expresses some sort of hate, irony and more generally, sentiment. Finally, we investigate the possibility of augmenting the training material by automatically downloading messages and labeling them through distant supervision (Go et al., 2009). We first selected few hashtags clearly in favour (or not) of the target topic to download and label a set of set of messages. Then, in order to add a set of neutral messages, we selected a set of news titles concerning the Sardines Movement.

The UNITOR system ranked first in the competition, suggesting that the combination of the Transformer-based learning with the adopted strategies of Transfer Learning and Data Augmentation is beneficial. In the rest of the paper, Sec. 2 describes UNITOR. In Sec. 3, the evaluations are reported while Sec. 4 derives the conclusions.

2 Transformer-based architectures and Transfer Learning for Stance Detection

The UNITOR system implements a Transformer-based architecture described in Section 2.1. The adopted auxiliary tasks are described in Section 2.2, while our Transfer leaning strategy is in Section 2.3. Finally, an automatic strategy for Data Augmentation is presented in Section 2.4.

2.1 UNITOR as a Transformer-based Architecture

The approach proposed in (Devlin et al., 2019), namely Bidirectional Encoder Representations from Transformers (BERT) provides a very effective model to pre-train a deep and complex neural network over large scale collections of non annotated texts and to apply it to a large variety of NLP tasks. The building block of BERT is the Transformer element (Vaswani et al., 2017), an attention-based mechanism that learns contextual relations between words in a text. BERT provides a sentence embedding (as well as the contextualized lexical embeddings of words in the sentence) through a pre-training stage aiming at the acquisition of an expressive and robust language and text model. The Transformer reads the entire input sequence of words at once and is optimized through two pre-training tasks. The first pre-training objective is the (*masked language modeling*) (Devlin et al., 2019). In addition, a *Next Sentence Prediction* task is used to jointly pre-train text embeddings able to soundly represent discourse level information. This last objective operates on text-pair representations and aims at modeling relational information, e.g. between the consecutive sentences in a text. On top of the produced embeddings, BERT applies a *fine-tuning* stage devoted to adapt the entire architecture to the targeted task.

The fine-tuning process of BERT for sentence classification (here adopted) operates on a single texts or text pairs, which can be given in input to BERT, in analogy with a next sentence prediction task. The special token `[CLS]` is used as first element of each input sequence and the embedding produced by BERT are used in input to a linear classifier customized for the target classification task. While the BERT architecture is pre-trained on large-scale corpora, its application to new tasks is generally obtained by customizing the final classifier to the targeted problem and fine-tuning all the network parameters for few epochs, to avoid catastrophic forgetting. In (Liu et al., 2019b) RoBERTa is proposed as a variant of BERT which modifies some key hyperparameters, including removing the next-sentence pre-training objective, and training on more data, with much larger mini-batches and learning rates. This allows RoBERTa to improve on the masked language modeling objective compared with BERT and leads to better downstream task performances.

UNITOR is based on a RoBERTa architecture pre-trained over Italian texts: we adopted UmBERTo[2] which is pre-trained over a subset of the OSCAR corpus, made of 11 billion tokens. These architectures achieved state-of-the-art results in a wide range of NLP tasks. However, they also rely on large scale annotated datasets composed of (possibly hundreds) thousands of examples. In order to improve the quality of this architecture in the SardiStance Task with a quite limited dataset, we adopted a simple Transfer Learning strategy by relying on the following three auxiliary tasks.

2.2 Supporting UNITOR through Auxiliary tasks

In this work, we speculate that the complexity of the Stance detection task can be simplified whenever the system to be trained is already aware if input messages express some sort of Sentiment, Irony or Hate. In order to expose UNITOR to such information, we trained specific classifiers over dedicated corpora made available in the previous editions of EVALITA, as it follows:

Sentiment Detection and Classification. This task consists in the automatic detection of subjectivity (and the eventual positive or negative polarity) in texts (Pang and Lee, 2008). Even though the Stance Detection is clearly different from a traditional task of Sentiment Analysis, we speculate that they are nevertheless related. As an example, we can suppose that the presence of stance is more probable in messages expressing subjectivity. We thus considered the setting proposed in SENTIPOLC 2016 (Barbieri et al., 2016) where a dataset of $8,000$ tweets is made available. For each message, the presence of subjectivity is made explicit and, eventually, the positive and negative polarity. The labeling provided in the dataset was slightly modified and mapped to a classification problem over three classes: all objective tweets were labeled with the special tag `<neutrale>`, the subjective and positive messages with `<positivo>` while the negative ones with `<negativo>`[3].

Irony Detection. We speculate that a robust detection of stance requires the recognition of irony, which can even reverse the output of the classification task. For example a false stance can be expressed through a ironic message, such as "*Le Sardine sono il futuro passato dell'Italia*"[4]. The objective of Irony Detection is to detect whether a given message is ironic or not. We used the dataset provided IronITA 2018 (Cignarella et al., 2018), where a dataset of $4,800$ labeled messages is made available. We adopted the original binary classification task, mapping ironic messages to the `<ironico>` and `<non ironico>` labels.

Hate Speech Detection. Being against a topic can be often expressed through messages expressing also hate. We thus introduce also the Hate Speech Detection task, which involves the automatic recognition of hateful contents. We considered the setting proposed in HaSpeeDe 2018 (Bosco et al., 2018), where a dataset of $3,000$ messages is made available. We adopted the original binary classification task: we mapped messages expressing hate with the `<odio>` label and `<non odio>` in the other case.

2.3 Transferring auxiliary tasks in the Transformer-based learning

In order to transfer the information from each auxiliary task into UNITOR, we first trained a specific UmBERTo-based sentence classifier on each of the datasets described in the previous section. In each case, the standard parameters proposed in (Devlin et al., 2019) are used to fine-tune the model[5]. After these three training steps, the entire SardiStance dataset is processed by each of the three classifiers and the resulting labels are used to "augment" the input messages. In particular, these labels generated a sort of new sentence, which is paired with the corresponding message. The following example shows how a tweet[6] against the movement is used in input to UNITOR:

"`[CLS]` *negativo ironico odio* `[SEP]` *#elezioniregionali Le Sardine aiuteranno a salvare il Paese! #mafammilpiacere Sono proprio dei bei perdigiorno falliti!* `[SEP]`"

Consistently with (Devlin et al., 2019), the first

---

[2] `https://huggingface.co/Musixmatch/umberto-commoncrawl-cased-v1`

[3] We discarded the few available messages with mixed polarity, to simplify the final classification task.

[4] In English: "*Sardines are the future past of Italy*"

[5] The number of epochs was tuned over a development set made of 10% of the corresponding dataset and the best epoch was selected by maximizing the classification accuracy.

[6] In English: "*#regionalelections The Sardines will help to save the country! #please They're just a bunch of losers!*"

pseudo-token `[CLS]` is added to generate the embedding used in input in the final linear classifier. Then, the pseudo-sentence "*negativo ironico odio*" suggests that the message expresses negative polarity and hate through the adoption of irony. Finally, between the `[SEP]` pseudo-tokens, the original message is reported. This particular schema resembles the classification of text pairs used in relational learning tasks, such as in Textual Entailment (Devlin et al., 2019). The output of the auxiliary classifiers defines a sort of hypothesis, i.e., *the authors aims at expressing a negative sentiment through an ironic message which also expresses hate*, while the original message is the direct consequence, i.e., the "implied" message[7]. The UNITOR model is thus an UmBERTo-based classifier trained over text pairs, where the first element encodes the information derived from the auxiliary tasks and the second one is the original message. Even though the quality of this labeling process can introduce noise (due to incorrectly classified messages) this augmented input is expected to simplify the final training process, by explicitly providing information about sentiment, hate and irony.

2.4 Distant Supervision for Stance Detection

In order to balance the limited amount of available data (especially considering the complexity of the task) we augmented the training material by labeling additional messages via Distant Supervision (Go et al., 2009). We speculate that a tweet containing an hashtag such as *#vivalesardine* (in English: *#ILikeSarine*) is in favour to Sardines instead of a tweet containing for example *#sardinefritte* (in English: *#friedSardine*) is against to our target. Hence, we downloaded from the TWITA corpus (Basile and Nissim, 2013) $3,200$ tweets and labeled them via Distant Supervision. In particular, the following subset are derived: $1,500$ tweets against the movement since containing *#gatticonsalvini* and 1,000 tweets in favour, since containing *#nessunotocchilesardine, #iostoconlesardine, #unmaredisardine, #vivalesardine* or *#forzasardine*. Finally, to enlarge the subset of messages without stance, 700 neutral statements were downloaded, which are actually titles from news, derived by querying "*sardine*" in Google news. In the experimental evaluations discussed in the next section, this dataset of "silver" data is simply added to the training material. To avoid over-fitting, we removed 90% of the occurrences of the hashtags used as query in the new data.

3 Results and Discussion

UNITOR participated to Task A - Textual Stance Detection (Cignarella et al., 2020) where the available dataset is composed by 2,132 tweets concerning the Sardines Movement: 1,028 tweets are against the movement (label `Against`), 589 tweets in favour of it (label `Favour`) and 515 tweets do not express any stance about the target topic (label `None`).

As discussed in Section 2, UNITOR is based on the UmBERTo pre-trained model, which relies on the RoBERTa architecture. For parameter tuning, we adopted a 10-cross fold validation, so that the training material is divided in 10 folds, each split according to 90%-10% proportion. The model is trained using a standard Cross-entropy Loss and an ADAM optimizer initialized with a learning rate set to $2 \cdot 10^{-5}$ and linearly decreased during the training process. We trained the model for 5 epochs, using a batch size of 32 elements. At test time, an Ensemble of such classifiers is used: each message is in fact classified using all 10 models trained in the different folds and the label suggested by the highest number of classifiers is selected. In the Task A, we submitted two constrained runs, i.e., system considering only tweets from the competition, and two unconstrained ones, where additional tweets were acquired and labeled by applying the approach presented in Section 2.2. All models are implemented using Pytorch[8] and experiments were run over Google Colab[9].

Results are reported in Table 1 in terms of Precision, Recall and F1 scores obtained by the different models with respect to each label. The final rank considers the average F1 (F1-avg) between the `Favour` and `Against` classes.

First of all, the high complexity of this task is confirmed by the results obtained by the strong Baseline method (the last row). It is a Support Vector Machine trained over a simple Bag-of-Word model (Cignarella et al., 2020) and achieves an average F1 of 57.84%, being competitive with many systems participating to the task and ranking 13^{th} over 22 submissions. One important re-

---

[7] We investigate different ways to encode this information, even using complex sentences, but negligible differences in the tuning process were measured, so we applied the simplest schema.

[8] `https://pytorch.org/`

[9] `http://colab.research.google.com/`

Rk	System	F1				Rec			Prec		
		avg	Against	Favor	None	Against	Favor	None	Against	Favor	None
1	**UNITOR_u_1**	**68.53%**	78.66%	**58.40%**	39.10%	**76.01%**	57.65%	45.35%	81.50%	**59.16%**	34.36%
2	UNITOR_c_1	68.01%	78.81%	57.21%	39.79%	74.66%	63.78%	43.60%	83.43%	51.87%	**36.59%**
3	UNITOR_c_2	67.93%	**79.39%**	56.47%	36.72%	**77.09%**	61.22%	37.79%	81.83%	52.40%	35.71%
4	Opponent_c_1	66.21%	75.80%	56.63%	**42.13%**	68.60%	**64.29%**	**52.91%**	**84.69%**	50.60%	35.00%
5	UNITOR_u_2	66.06%	76.89%	55.22%	37.02%	72.64%	56.63%	44.77%	81.67%	53.88%	31.56%
6	UmBERTo	65.69%	77.41%	53.97%	35.93%	74.12%	57.14%	40.11%	81.00%	51.14%	32.54%
13	Baseline	57.84%	71.58%	44.09%	27.64%	68.06%	49.49%	29.65%	75.49%	39.75%	25.89%

Table 1: Results obtained by UNITOR at the SardiStance task. In bold best results for each measure. In the system name "c" and "u" refer to constrained and unconstrained runs.

sult is obtained by the straight application of the UmBERTo model over the original messages (next to last row in Table 1). In fact, this Transformer-based architecture, empowered with the Ensemble technique, achieves an average F1 of 65.69%: a system which directly applies an Ensemble of UmBERTo-based models would have ranked 6th in the competition.

We thus trained UmBERTo by adopting the Transfer Learning approach presented in Section 2.3 in the constrained setting. The adoption of all the three auxiliary tasks led to the constrained submission called UNITOR_c_2. Moreover, we considered the training of UmBERTo by considering one auxiliary task at a time. When considering only the Hate Speech Detection task, better results were obtained over the development set, with respect to the adoption of the other tasks taken individually, i.e., Sentiment Detection and Irony Detection[10]. Such a variant, called UNITOR_c_1, considers tweets enriched only with information derived by the hate classifier and it generally shows higher precision with respect to the `Against` class. This suggests that a tweet expressing hate is more likely in opposition to the Sardines Movement. Both constrained models ranked 3rd and 2nd in the competition, respectively. These results are impressive as they both outperformed of about 2% of absolute F1 the standard UmBERTo. Moreover, they confirm the beneficial impact of Hate Speech Detection as an auxiliary task. Finally, we augmented the training dataset by using the additional data presented in Section 2.2. We extended the training material used to train UNITOR_c_2 in order to obtain the unconstrained submission called UNITOR_u_2. It is worth noticing that all three auxiliary tasks were used in this submission. This led to a performance drop, i.e. a 66.06% of average F1, which is lower with respect to the best opponent system, which achieved a 66.21% of F1. It seems that the noise added both from the auxiliary tasks and the additional data, negatively impacted the overall quality. On the contrary, when only the Hate Speech Detection task is considered (i.e., UNITOR_u_1) additional data are positively capitalized by the model, achieving the best average F1 score in the competition, i.e. 68.53%. These results suggest that the combination of the Transformer-based learning with the adopted strategies of Transfer Learning and Data Augmentation is highly beneficial, when only Hate is considered.

From an error analysis, it seems that a significant number of incorrect classifications occurred in longer and complex messages, where the topic of the stance is not clearly explicit nor captured by the UmBERTo model, such as in "*#carfagna: "io per i liberali che non si affidano a Salvini" e "dalle sardine buone idee". Auto-scacco in due mosse. Con la Polverini poi...*"[11]. This message is considered to be `Against` while the system assigns the label `None`. Here, it is very challenging to understand the connection between the "*good ideas of the sardines*" and the very colloquial expression "*Auto-scacco*" which can be translated as "*She messed herself*". The same appears in the tweet "*Ho finalmente capito chi mi ricordava Mattia Santori, quello delle sardine: Lodo Guenzi. (e infatti in quanto a democristianitá stiamo lá)*"[12] which again labeled `Against` but classified as `None`. Clearly the system is not able to link the movement to its leader nor to the negative opinion about belonging to the Christian Democrat Party. Another example is the tweet "*Dopo*

---

[10] The results of this tuning stage were not reported here for lack of space.

[11] In English: "*#carfagna: "come with me liberals who do not rely on Salvini" and "from Sardines movement good ideas." She messed herself up with two moves. Not to mention Polverini...*"

[12] In English: "*I finally understood who reminded me of Mattia Santori, the one with the Sardines movement: Lodo Guenzi. (in fact as far as Christian Democrats are concerned they are pretty the same).)*"

avere ascoltato @luigidimaio mi viene in mente una sola parola:grazie. Fiducia nelle sue scelte e immenso rispetto per i grandi risultati ottenuti. Ora un nuovo inizio, con un nuovo entusiamo. Andiamo versogli #statigenerali con serietà e maturità. Forza@mov5stelle!"[13]. Here the system incorrectly assigns the `Favour` label because the tweet is in favour of a different movement.

4 Conclusion

In this work we present the results obtained by the UNITOR system, which participated to the SardiStance task. UNITOR ranked first in Task A, both for constrained and unconstrained runs. These results confirm the beneficial impact of Transformer based architecture for text classification also in the Stance Detection task. Moreover, we demonstrate the beneficial impact of Hate Speech Detection as an auxiliary task in a Transfer Learning setting. Finally, we empirically demonstrate that the adoption of Distance Supervision is useful to reduce data sparseness. Future work will apply the above approaches to task B within SardiStance. Moreover, we will investigate multitask learning approaches (Liu et al., 2019a) to capitalize information from auxiliary tasks in a more principled way.

References

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the evalita 2016 sentiment polarity classification task. In *Proceedings of EVALITA 2016, Napoli, Italy, December 5-7, 2016*, volume 1749 of *CEUR Workshop Proceedings*.

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on italian tweets. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, Atlanta.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*. CEUR-WS.org.

Cristina Bosco, Felice Dell'Orletta, Fabio Poletto, M. Sanguinetti, and M. Tesconi. 2018. Overview of the evalita 2018 hate speech detection task. In *EVALITA@CLiC-it*.

Alessandra Teresa Cignarella, Simona Frenda, Valerio Basile, Cristina Bosco, Viviana Patti, Paolo Rosso, et al. 2018. Overview of the EVALITA 2018 task on irony detection in Italian tweets (IronITA). In *Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2018)*, volume 2263, pages 1–6.

Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. SardiStance@EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL 2019*, pages 4171–4186, Minneapolis, Minnesota, June.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. Technical report.

Peter Krejzl, Barbora Hourová, and Josef Steinberger. 2017. Stance detection in online discussions.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of ACL*, pages 4487–4496, Florence, Italy, July.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Michael Mccloskey and Neil J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. *The Psychology of Learning and Motivation*, 24:104–169.

S.J. Pan and Q. Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

---

[13] In English: "*After listening to @luigidimaio only one expression came to my mind: thank you. I have trust in his choices and a huge respect for the great results obtained. Now it's a new start, with new enthusiasm. Let's move towards the #statigenerali with seriousness and maturity.Forza@mov5stars*"

ghostwriter19 @ SardiStance: Generating new Tweets to Classify SardiStance EVALITA 2020 Political Tweets

Mauro Bennici
You Are My Guide
Torino
`mauro@youaremyguide.com`

Abstract[1]

English. Understanding the events and the dominant thought is of great help to convey the desired message to our potential audience, be it marketing or political propaganda.

Succeeding while the event is still ongoing is of vital importance to prepare alerts that require immediate action.

A micro message platform like Twitter is the ideal place to be able to read a large amount of data linked to a theme and self-categorized by its users using hashtags and mentions.

In this research, I will show how a simple translator can be used to bring styles, vocabulary, grammar, and other characteristics to a common factor that leads each of us to be unique in the way we express ourselves.

Italiano. Comprendere gli eventi e il pensiero dominante è di grande aiuto per veicolare alla nostra potenziale audience il messaggio desiderato sia esso di marketing o di propaganda politica.

Riuscirci mentre l'evento è ancora in corso è di vitale importanza per predisporre alert che richiedono un intervento immediato.

Una piattaforma di micro messaggi come Twitter è il luogo ideale per poter leggere una grande quantità di dati legata ad un tema, e spesso auto categorizzati dai suoi stessi utenti per mezzo di hashtag e menzioni.

In questa ricerca mostrerò come un semplice traduttore può essere usato per portare a fattor comune stili, lessico, grammatica e altre caratteristiche che portano ognuno di noi ad essere unico nel modo di esprimersi.

1 Introduction

Each of us has a unique way of writing. However, the fewer options we have to experience ourselves to express our concept, the more the necessary synthesis leads to the loss of precious information to accurately assess our real intentions.

Furthermore, the more the subject is debated, the more changes in style and tone occur. The conversation becomes full of irony or aggressive. Extrapolating a single line is dangerous without context. The same sentence can have different interpretations depending on the moment in which it is pronounced, the audience it is intended for, the place where you are, in the historical period in which it was composed.

My hypothesis is that we can translate all these different styles into a single "language style" that fully expresses the real intentions of the writer. The challenge is to understand when a user has expressed a comment in favor, against, or neutral towards the Sardines' Italian political movement.

The research was carried out for the SardiStance (Cignarella et al., 2020) task in the EVALITA 2020 (Basile et al., 2020). Two models were created for the Task 1, but they also performed well on the Task 2.

---

2 Description of the system

The two tasks are similar. In Task A, it is necessary to classify the stance of a tweet based only on the text of the tweet. Task A is divided into two subtasks:

- Constrained. It is allowed to use additional resources such as a Lexicon but no other resources (such as labeled tweets) to help the training process.
- Unconstrained. Where each resource used must be reported in the final report.

In Task B, you can use the context information provided by the post author. Additional information refers:

- to post statistics (favors, retweets, reply, source)
- to the author's information (number of posts, number of followers, emoji in the bio)
- to the author's circle of relationships (friends, replies, retweets, and quotes)

The research focuses on Task A Constrained.

Considering the constraints of Task A, it is not possible to access any additional information other than the text of the tweet, I concentrated on understanding how to clean it up.

The Training dataset contains:

- the tweet ID
- the user ID
- the text
- the label

The labels options are:

- Against
- Favor
- Neutral / None

To be sure to do not use any data except the text, the user id, useful for Task B, was discarded.

In order to validate my hypotheses, I used the AlBERTo model, created from tweets, (Polignano at al., 2019) and an auto training system such as Ktrain[2], a framework that wrap TensorFlow[3], to classify the tweets. To avoid manual error and involuntary optimization, I used the autofit option.

First, I wrote a series of algorithms to make the texts to be compared homogeneous.
The first one was to break up the composed hashtags into sentences and words.
For example, using capital letters as a separator:

- #IoStoConLeSardine has become "io sto con le sardine" ["I'm with sardines"].
- #NessunoTocchiLeSardine has become "nessuno tocchi le sardine"["nobody touches the sardines"].

As a second step, I made sure to remove repeated vowels in a sentence, such as:

- "Svegliaaaa" to get the word "Sveglia" [Wake up!].

I also replaced the word sardines with "PartitoPoliticoS" ["PoliticalPartyS"] to prevent the entity from being mistaken for the fish that is its symbol. I did not remove any stop words because it is useful to create the translation system.

At this point, I made a copy of the dataset to translate it. I used the spaCy[4] language functions of POS tagging, Dependency Parse, and Entity Recognition to have all the essential components of my translator.

The translator is a simple text representation. It is a matter of rewriting the sentence following the scheme:

- subject adjectives
- subjects
- verb in the infinitive form
- adjectives objects
- objects
- exclamations / other words

At this stage, the words are not modified to make the sentence grammatically correct. Words are exchanged places, only the verb are modified to the

---

[2] https://github.com/amaiya/ktrain

[3] https://www.tensorflow.org/

[4] https://spacy.io/api/annotation

infinitive form. The entities of type person [PER] take precedence over others.

The translator concentrates its attention on the aspect inside the sentences to be sure to do not remove valid sentiment polarity words (Barbieri et al, 2016). And to avoid to lose them in a round-trip translation activity on translation services (Marivate & Sefara, 2020). The attempt to represent the text in a more recognizable and identifiable form for an algorithm passes from the fact that it can still recognize the entities described and the polarity expressed for each of them. For this purpose, the translator makes several attempts to fit words into their suggested position.

Finally, I trained two models with the Ktrain framework. The model 1, which use the translated tweets, was submitted as ghostwriter19_Task_A_1_c. The model 2, trained with the only cleaned tweets, was submitted as ghostwriter19_Task_A_2_c.

2.1 First results

The model will be evaluated with the F1-score. The main score is the average of the F1-score of the Favor tweets and the F1-score of the Against tweets.

When comparing the two models, the first result is that the translated tweets performed worse, albeit by a few percentage points (table 1).

Model	**F1-Score**
ghostwriter19_Task_A_1_c	0.5613
ghostwriter19_Task_A_2_c	0.6004
Estimated Baseline	0.5386

Table 1: First results

Analyzing the results of both the models in detail (table 2 and 3), we have that:

ghostwriter19_Task_A_1_c	**F1-Score**
Against	0.69
Favor	0.43
Neutral	0.42

Table 2: F1-score details of model 1

ghostwriter19_Task_A_2_c	**F1-Score**
Against	0.70
Favor	0.50
Neutral	0.32

Table 3: F1-score details of model 2

The problem is evident. Model 1 has a more challenging time distinguishing the favor tweets from neutral ones. The good news is that both the models overcame the estimated baseline.

2.2 Hashtags and Mentions

Thinking that on Twitter the hashtags are also used for classification purposes, the operation that replaces them was modified. Now the hashtags are added at the end of the new tweets. Also, the mentions are considered and processed as hashtags (table 4).

Model	**F1-Score**
ghostwriter19_Task_A_1_c	0.5822
ghostwriter19_Task_A_2_c	0.6004
Estimated Baseline	0.5386

Table 4: Model 1 with hashtags and mentions in the translated tweets

Analyzing the results in detail (table 5), we can see that:

ghostwriter19_Task_A_1_c	**F1-Score**
Against	0.71
Favor	0.45
Neutral	0.41

Table 5: F1-score details of model 1 with hashtags and mentions in the translated tweets

The model gained two percentage points for both Against and Favor, compared with a one-point loss in Neutral. Unfortunately, it still remains two points below the model 2, with the only cleaned tweets.

2.3 Passive verbs

Analyzing the new texts generated, I noticed that essential information was lost by putting all the verbs in the infinitive. If the verb was in the passive form, the subject and object of the sentence were reversed. At the same time, I noticed that very long tweets contained more than one sentence.

I modified the translator to consider passive and active verbs, swapping the sentence's subject and object if necessary. The hashtags inserted at the end of the tweet only left at the end of the new tweet generated (table 6).

Model	F1-Score
ghostwriter19_Task_A_1_c	0.6306
ghostwriter19_Task_A_2_c	0.6004
Estimated Baseline	0.5386

Table 6: Model 1 with hashtags and mentions in the translated tweets, plus active / passive verbs

Analyzing the results in detail (table 7), we can see that:

ghostwriter19_Task_A_1_c	F1-Score
Against	0.76
Favor	0.50
Neutral	0.40

Table 7: F1-score details of model 1 with hashtags and mentions in the translated tweets, plus active / passive verbs

The model gained five percentage points for Against and Favor tweets, compared with a one-point more loss for Neutral ones. Now the translation model is the best model.

3 Results

Model 1 was ultimately 3 percentage points better than Model 2 with the Training dataset. The best performance of the model was also confirmed with Test datasets, with 2.5 percentage points of advantage.

3.1 Results for Task A

The final results with the Test dataset are:

Model	F1-score
ghostwriter19_Task_A_1_c	0.6257
ghostwriter19_Task_A_2_c	0.6004
Baseline	0.5784

Table 8: Test dataset results for Task A

The model 1 is about 7.5% better than the baseline (table 8).
I remember that both models were trained with the autofit option, so without any particular study, to validate whether a "translation" of the original text could bring apparent advantages.

3.2 Results for Task B

Although no context information was used, I still proposed the predictions for Task A to Task B.
The final results with the Test dataset are:

Model	F1-score
ghostwriter19_Task_A_1_c	0.6257
ghostwriter19_Task_A_2_c	0.6004
Baseline	0.6284

Table 9: Test dataset results for Task B

Even if model 1 was not able to reach the proposed baseline, the difference between the two systems is 0.4% (table 9). The detailed results of the models are showed in the tables 10 and 11.

3.3 Detailed results for Task A

model	f-avg	prec_a	prec_f	prec_n	recall_a	recall_f	recall_n	f_a	f_f	f_n
1_c	0.6257	0.8106	0.4709	0.3226	0.6981	0.5357	0.4651	0.7502	0.5012	0.3810
2_c	0.6004	0.8094	0.4772	0.2921	0.6523	0.4796	0.5349	0.7224	0.4784	0.3778
baseline	0.5784	0.7549	0.3975	0.2589	0.6806	0.4949	0.2965	0.7158	0.4409	0.2764

Table 10: TASK A detailed results of the proposed models compared to the baseline model.

3.4 Detailed results for Task B

model	f-avg	prec_a	prec_f	prec_n	recall_a	recall_f	recall_n	f_a	f_f	f_n
1_c	0.6257	0.8106	0.4709	0.3226	0.6981	0.5357	0.4651	0.7502	0.5012	0.3810
2_c	0.6004	0.8094	0.4772	0.2921	0.6523	0.4796	0.5349	0.7224	0.4784	0.3778
baseline	0.6284	0.7845	0.4506	0.3054	0.7507	0.5357	0.2965	0.7672	0.4895	0.3009

Table 11: TASK B detailed results of the proposed models compared to the baseline model.

4 Conclusion

In a preliminary way, the final results demonstrate that it is possible to obtain an improvement of the predictions by reducing the differences of expression to a predetermined structure.

The system is, however, right now, more efficient in terms of training times and final scores than ensemble systems of Bi-LSTM, which were used successfully up to 2 years ago (Bennici & Portocarrero, 2018).

The next step is also to optimize the model's training to ascertain that the performance gain is maintained and in what percentage. At the same time, the translator can be improved by switching to a sequence-to-sequence system for a meaningful and efficient text representation that will include, among other things, the change of every words forms accordingly with the grammar and the original intention of the writers (Lewis et al., 2019).

References

Barbieri, F., Basile, V., Croce, D., Nissim, M., Novielli, N., & Patti, V. (2016). Overview of the Evalita 2016 SENTIment POLarity Classification Task. In *Proceedings of Third Italian Conference on Computational Linguistics (CLiC-it 2016) & Fifth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2016)*, Naples, Italy. CEUR-WS.org.

Basile, V., Croce, D., Di Maro, M., & Passaro, L. (2020). EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020),* CEUR-WS.org.

Bennici, M., & Portocarrero, X. S. (2018). Ensemble for aspect-based sentiment analysis. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*, Turin, Italy. CEUR-WS.org.

Cignarella, A. T., Lai, M., Bosco, C., Patti, V., & Rosso, P. (2020). SardiStance@EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets. In *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., . . . Zettlemoyer, L. (2019, October 29). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. https://arxiv.org/abs/1910.13461

Marivate, V., & Sefara, T. (2020). Improving Short Text Classification Through Global Augmentation Methods. *Lecture Notes in Computer Science Machine Learning and Knowledge Extraction,* 385-399. doi:10.1007/978-3-030-57321-8_21

Polignano, M., Basile, P., de Gemmis, M., Semeraro, G., & Basile, V. (2019). Alberto: Italian bert language understanding model for nlp challenging tasks based on tweets. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*. CEUR-WS.org.

QMUL-SDS @ SardiStance2020: Leveraging Network Interactions to Boost Performance on Stance Detection using Knowledge Graphs

Rabab Alkhalifa[1,2] and **Arkaitz Zubiaga**[1]

[1]Queen Mary University of London, United Kingdom
[2]Imam Abdulrahman bin Faisal University, Saudi Arabia

Abstract

This paper presents our submission to the SardiStance 2020 shared task, describing the architecture used for Task A and Task B. While our submission for Task A did not exceed the baseline, retraining our model using all the training tweets, showed promising results leading to (f-avg 0.601) using bidirectional LSTM with BERT multilingual embedding for Task A. For our submission for Task B, we ranked 6th (f-avg 0.709). With further investigation, our best experimented settings increased performance from (f-avg 0.573) to (f-avg 0.733) with same architecture and parameter settings and after only incorporating social interaction features- highlighting the impact of social interaction on the model's performance.

1 Introduction

Framed as a classification task, the stance detection consists in determining if a textual utterance expresses a supportive, opposing or neutral viewpoint with respect to a target or topic (Küçük and Can, 2020). Research in stance detection has largely been limited to analysis of single utterances in social media. Furthering this research, the SardiStance 2020 shared task (Cignarella et al., 2020) focuses on incorporating contextual knowledge around utterances, including metadata from author profiles and network interactions. The task included two subtasks, one solely focused on the textual content of social media posts for automatically determining their stance, whereas the other allowed incorporating additional features available through profiles and interactions. This paper describes and analyses our participation in the SardiStance 2020 shared task, which was held as part of the EVALITA (Basile et al., 2020) campaign and focused on detecting stance expressed in tweets associated with the Sardines movement.

2 Related Work

In social media, **classical features** can be extracted by using *stylistic signals* from text such as bag of n-grams, char-grams, part-of-speech labels, and lemmas (Sobhani et al., 2019), *structural signals* such as hashtags, mentions, uppercase characters, punctuation marks, and the length of the tweet (Wojatzki et al., 2018; Sun et al., 2016), and *pragmatic signals* related to author's profile (Graells-Garrido et al., 2020). With modern deep learning models, there is shift towards **contextualised representations** using word vector representation algorithms, either by having personalised language models trained on task specific language or as a pre-trained language model offered after training using complex architecture and billions of documents. Using **deep learning layers** as automated feature engineering methods can be implemented to train the model afterwards. In (Augenstein et al., 2016), they utilized Bidirectional Conditional Encoding using LSTM achieving state-of-the-art results on stance detection task. Recently, there is a resurgence of research in incorporating **network homophily** (Lai et al., 2017) to represent social interactions within a network. Moreover, **Knowledge graphs** (Xu et al., 2019) can in turn represent these complex network relationships (e.g. authors friendships) as simple embedded vectors sampled considering the nodes and weighted edges within the network complexity structure.

3 Definition of the Tasks

The stance detection task has been defined in previous work as consisting in determining the

---

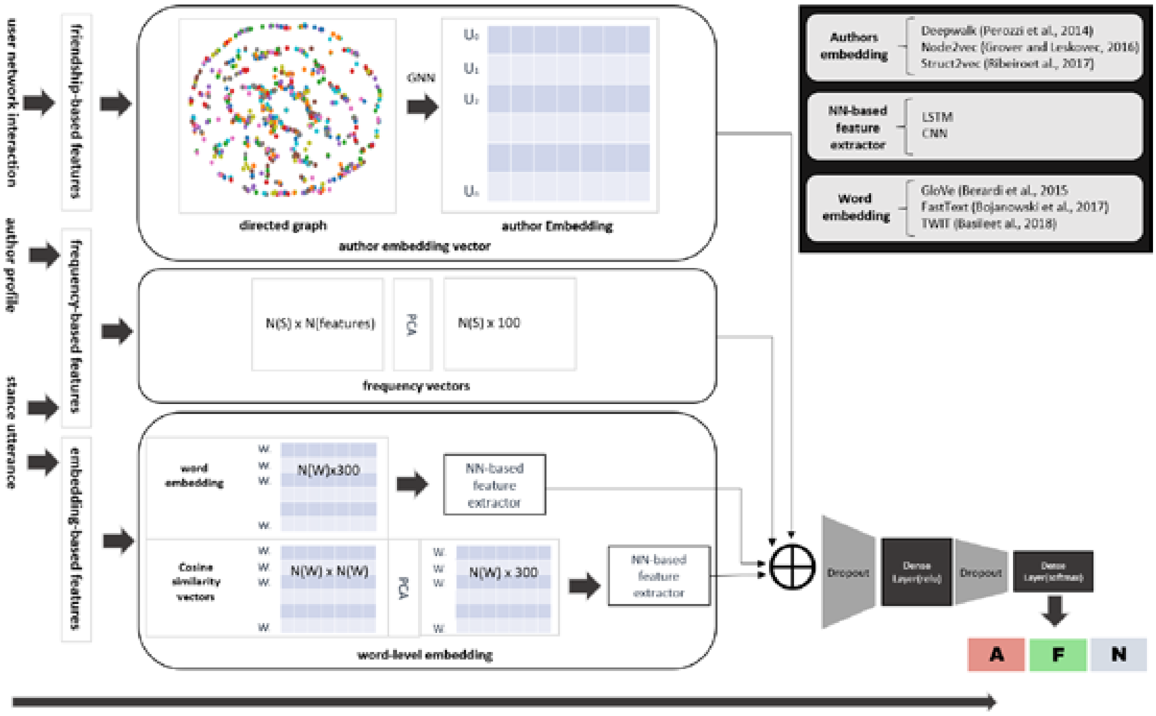


Figure 1: Our framework for investigating different combinations of features. For a network interaction graph, we generate user embeddings, using variations of graph neural network (GNN) embedding methods, namely deep-walk, struct2node and node2vec, and then concatenate author's vector with its corresponding utterance features for each stance. We also extract two types of text embedding representations for each utterance, embedding-based features, namely word embedding vectors and cosine similarity vectors, using different models including variations of CNN and bidirectional models. Further, the results of these two feature extraction methods are concatenated for the final classification step. We also consider the standard methods that extract frequency-based representations from author profiles and stance utterances including unigrams and Tfidf vectors. All these four features where combined and fed into the drop out and dense layers, to finally generate the final label using a softmax activation function. Though, we deactivate some of these four sources of features and alter the frequency-based vector by excluding some features, changing the embedding source and reducing the dimensionality for highly dimensional vectors (e.g. frequency-based features and cosine similarity vectors) using PCA.

viewpoint of an utterance with respect to a target topic (Küçük and Can, 2020), while others define it as that consisting in determining an author's viewpoint with respect to the veracity of a rumour, usually referred to as rumour stance classification (Zubiaga et al., 2018). SardiStance focuses on the former, and is split into two subtasks: Textual Stance Detection (Task A) and Contextual Stance Detection (Task B) (Cignarella et al., 2020). Baselines are provided for Task A using SVM+unigrams as (f-avg. 0.578), and for Task B as (f-avg. 0.628) (Lai et al., 2020).

4 Experimental Settings

Frequency-based features: These represent frequency vectors including unigram, punctuation and hashtags provided by (Cignarella et al., 2020). Further, we include TFiDF vectors.

Embedding-based features: ***word embedding*** Italian Wikipedia Embedding (Berardi et al., 2015) trained using GloVe [1], Fasttext with (Bojanowski et al., 2017) [2] trained using skip-gram model and with 300 dimensions, and TWITA embedding (Basile et al., 2018). For TWITA, two versions of the same tweets were generated. One preprocessing words where each vector has 100 dimensions, provided by (Cignarella et al., 2020)[3] and referred to as TWITA100. The other

---

[1] `https://github.com/MartinoMensio/it_vectors_wiki_spacy`

[2] `https://fasttext.cc/docs/en/pretrained-vectors.html`

[3] `https://github.com/mirkolai/`

one trained by us without any preprocessing and each vector has 300 dimensions, referred to as TWITA300. We also experimented with multilingual BERT in Task A [4] (Devlin et al., 2019).

Cosine similarity vectors which was introduced previously in (Eger and Mehler, 2016) to encode the word meaning within the embedding space. In our work, we used TWITA300 to train the similarity vectors of all the words in the training set.

Network-based features: ***Encoding users graph.*** To represent user interactions as nodes and edges, we used a counting scalar value and added one if each of the following relationships exists: friendships, retweets, quotes and replies, e.g. if all of them exist then the edge weight between two accounts is four. We calculated all the accounts provided and generate a directed complex graph conditioned by the existence of friendship, resulting in 669,745 nodes, 2,871,791 edges with an average in-degree of 4.2879 and average out-degree of 4.2879.

Generating GNN Embeddings. Taking as input the encoded network relationships, GNN embeddings use different sampling techniques to represent every node as a vector. To extract these vectors, we experiment with different graph neural network models, namely struct2vec (Ribeiro et al., 2017), deepwalk (Perozzi et al., 2014) and node2vec (Grover and Leskovec, 2016).

NeuralNetwork-based features As illustrated in Figure 1, we have different deep learning models to extract features separately for both word embedding and similarity vectors matrices. In our work, we experiment with Convectional Neural Network (CNN) models and Long short-term memory (LSTM) models. Variations of **CNN** models where applied to NLP downstream tasks as feature extraction methods for text classification. In our work, we used two variations of CNN. In one model, we used a CNN as a one-head *1D-CNN* with kernel size of 5 allowing the model to extract features with 5-grams vectors using 32 filters. Followed by a max pooling layer with pool size of 2 then flattened layer. In another model, we used a CNN as a multi-headed *2D-CNN* with 1, 2, 3, 5 grams filter sizes, initialising the kernel weights with a Rectified Linear Unit (ReLU) activation function and normal distribution weights. Followed by a max pooling layer with different pooling sizes taken as one columns pooling filter with the maximum text length excluding few grams sizes. For the **LSTM**, we used two variants. One is a simple *bidirectional LSTM* of 64 units followed by concatenations of max pooling and average pooling layers, and *attention bidirectional LSTM* proposed by (Yang et al., 2016) using 64 units followed by 128 units then attention layers[5].

Feature Reduction. We experiment with different reduction length: 50, 100 and 150. Then. we set our PCA reduction to 100 as it showed best performance on evolution set.

Sentence Cleaning. We set the cleaning function to match the preprocessing function by (Cignarella et al., 2020) to generate TWITA100.

We used four final layers to receive the features and concatenate them (see Figure 1). In all of the experiments, our dropout layer set to 0.2, followed by a dense layer with rule activation function and another dropout layer of 0.2. Finally, a probability vector of the three classes is generated. To determine the correct class, we choose the one class with the highest probability.

5 Results

In this section, we discuss the results of our systems submitted to the two tasks.

For Task A, we used attention Bidirectional LSTM model performance compared to using different word embedding models, also we analysed impact of the preprocessing of the runs. Since there are too many parameters to compare with, we compared the performance of the embedding models. Our submitted models, BERT and TWITA300 illustrated in Table 1 with $*$ showed most promising results using different settings. With only %80 training data, similarity vectors generalised better than all other embedding models. While, when all data are trained, the best model is the multilingual BERT embedding with no pre-processing (f-avg 0.601), followed by similarity vectors using cleaned text (f-avg 589).

For Task B, we used different feature extraction, frequency vectors, word embedding and social interaction embedding models, and monitor their performance while activating the pre-processing step in all experiments. With a diverse range of parameters, we experimented with a total of 3845 random runs. Then, we selected the best mod-

---

`evalita-sardistance/`

[4] `https://tfhub.dev/tensorflow/bert_multi_cased_L-12_H-768_A-12/2`

[5] `https://www.kaggle.com/mlwhiz/attention-pytorch-and-keras`

Task A				
	Eval.		Tst. f-avg	
Not-preprocessed				
Emd#	%	f-avg	*T%80*	*T%100*
BERT*	0.480	**0.532**	0.533*	**0.601**
SVs	**0.518**	**0.548**	**0.589**	0.532
TWITA300	**0.482**	0.526	**0.578**	0.551
TWITA100	0.480	0.521	0.494	0.551
Fasttext	0.485	0.521	0.479	0.482
GloVe	0.445	0.308	0.401	0.401
Preprocessed				
SVs	**0.515**	**0.556**	0.524	0.566
TWITA100	**0.513**	**0.543**	0.560*	0.566
FastText	0.485	0.489	0.532	0.528
TWITA300	0.447	0.490	0.541	0.506
GloVe	0.445	0.308	0.401	0.401
BERT	0.475	0.445	0.512	0.213
Baseline			*0.578*	*0.578*

Table 1: Results for Task A. We evaluate all the embeddings using Attention Bidirectional LSTM. Our submissions are the ones represented with *. *Bold fonts show results above baseline*

els considering macro f-score for the two classes under consideration (AGAINST and FAVOR) (f-avg). Results are shown in Table 2. By comparing our runs by adding social interaction features, our models with different settings showed a clear improvement on our models. In 1#M, we utilise Conv2D (see NeuralNetwork-based features) for embedding vectors with TfiDF unigram and tweet length, where the model achieved an increase on performance of (f-avg 0.16) when social interaction vectors incorporated into the model. All other models showed the same improvement with an increase of (f-avg 0.115, 0.118, 0.081, 0.021) for 3#M, 5#M, 7#M and 9#M, respectively.

6 Discussion and main findings

The pipeline depicted in Figure 1 was designed to investigate the impact of multiple features on stance detection using variations of feature extraction methods, which have been experimented in previous work but we adapted them to the Italian language in our settings. The training set contains 2132 instances with no evaluation set. In our work, we create a stratified split of 80-20 to evaluate the model, which leads to a training data with 1705 samples. Further, our investigation attempted to randomise different settings, with the aim of submitting the top two with highest f-avg score on the remaining set (Eval. 426) for both tasks. Consequently, we found that this methodology did not generalise well with the testing results. However, our main findings remain consistent across different settings when compared with our results using the stratified split (T%80) and when the model was retrained using all the data (T%100). While our submission evaluated both tasks separately, we discuss all conclusions jointly in this section.

Having different random settings over all frequency-based features (14, in our case) would be a bad strategy to evaluate the methods and come up with the best approach. To verify if we need to include all of these, we run an experiment by including only one feature from (unigram, Tfidf_unigram, chargrams, network_reply_community, userinfobio). The selection of these features where based on selecting the best runs using only one feature from our randomised parameters. Using all the training set and CONV2D with (fasttext,TWEC300) and reduced SVs with deepwalk user's social interaction vector, (userinfobio,chargrams) achieved (f-avg 0.703 and 0.704), respectively. This is also higher than using AttLSTM for the same settings which achieved (f-avg 0.638 and 0.610). In general, we achieve better performance with CONV2D than AttnLSTM for the same settings on the test data. In another experiment, we reduced all the 14 frequency-based parameters achieving (f-avg 0.714) which performs worse than our best 3#M (see 2). Our main conclusion is that the number of features available is not necessarily correlated with the model's performance boost.

In another experiment, we attempted to compare the performance of TWEC100 with TWEC300 (see Section 4). From Table 1, we observed that lower dimensionality and pre-processing may cause the model to under perform by around (f-avg 0.050), at least. Though, this impact was not significant with T%100. However, matching the processing between the embedding vocabulary and the annotated set yields better performance. For example, TWITA100 was more persistent on performance between T%80 and T%100. This highlights the importance of pre-processing and reducing the differences between the embedding vocabularies and labelled sentences. In general, our embedding experiment for Task A show high sensitivity on model performance with pre-processing settings.

Inspired by previous work on encoding word meanings, we experimented with SVs embedding. Interestingly, these vectors showed high f-avg,

Task B					
	Eval.		**Tst. f-avg**		
#M	*%*	**f-avg**	*T%80*	*T%100*	**Settings.**
1	0.590	0.651	**0.683**	**0.733**	Conv2D(FastText) + Conv2D(PCA(SVs)) + PCA(unigram + Tfidf_unigram + length) + DeepWalk
2	0.511	0.521	0.605	0.573	Conv2D(FastText) + Conv2D(PCA(SVs)) + PCA(unigram + Tfidf_unigram + length)
3	0.595	0.640	**0.662**	**0.719**	Conv2D(FastText)+ Conv2D(PCA(SVs)) + Conv2D(PCA(Tfidf_unigram + chargrams)) + DeepWalk
4	0.525	0.507	0.608	0.604	Conv2D(FastText)+Conv2D(PCA(SVs))+PCA(Tfidf_unigram + chargrams)
5	0.600	0.645	**0.710**	**0.718**	Conv2D(FastText) + Conv2D(PCA(SVs)) + PCA(unigram + length)+ DeepWalk
6	0.487	0.495	**0.661**	0.600	Conv2D(FastText + Conv2D(PCA(SVs)) + PCA(unigram + length)
7	**0.600**	**0.671**	**0.709***	**0.696**	Conv2D(TWITA300) + Conv2D(PCA(SVs)) + PCA(length + network_quote_community + network_reply_community + network_retweet_community + network_friend_community + userinfobio + tweetinfocreateat) + DeepWalk
9	0.574	0.532	**0.629**	0.615	Conv2D(TWITA300) + Conv2D(PCA(SVs)) + PCA(length + network_quote_community + network_reply_community + network_retweet_community + network_friend_community + userinfobio + tweetinfocreateat)
9	**0.602**	**0.691**	**0.677***	**0.681**	AttLSTM(FastText) + AttLSTM(PCA(SVs)) + PCA(puntuactionmarks + length + network_quote_community + network_retweet_community + network_friend_community + userinfobio) + Node2Vec
10	0.459	0.488	0.456	**0.660**	AttLSTM(FastText) + AttLSTM(PCA(SVs)) + PCA(puntuactionmarks + length + network_quote_community + network_retweet_community + network_friend_community + userinfobio)
Baseline			*0.628*	*0.628*	

Table 2: Top performing settings over all sampled runs using our architecture for Task B. Our submissions are the ones represented with *. *Bold fonts show highest/above baseline results*

better than BERT and TWITA300 with T%80 although it showed a significant drop when the model was trained with T%100. This finding opens an investigation towards the ability of SVs to perform better under different settings. For that, we removed PCA(SVs) and run same settings of #M1, and our model achieved (f-avg 0.678), showing a significant impact of SVs on model's performance. Further, we investigate the robustness of deepwalk modelling over node2vec and struct2vec for the same best settings of #M1, resulting on (f-avg 0.641 and 0.604) for node2vec and struct2vec, respectively. Also, in terms of accuracy, the deepwalk model produces an improved accuracy of (% 0.725) compared to node2vec (% 0.665) and struct2vec (% 0.658). This indicates that deepwalk is more reliable on this testing set than other models.

7 Conclusion

In this work, we described a state-of-the-art stance detection system leveraging different features including author profiling, word meaning context and social interactions. Using different random runs, our best model achieved (f-avg 0.733) leveraging deepwalk-based knowledge graphs embeddings, FastText and similarity feature vectors extracted by two multi-headed convolutional neural networks from auther's utterance. This motivates our future, aiming to reduce the model complexity and automate the feature selection process.

8 Acknowledgments

This research utilised Queen Mary's Apocrita HPC facility, supported by QMUL Research-IT.

References

Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885, Austin, Texas, November.

Valerio Basile, Mirko Lai, and Manuela Sanguinetti. 2018. Long-term social media data collection at the university of turin. In *Fifth Italian Conference on*

Computational Linguistics (CLiC-it 2018), pages 1–6. CEUR-WS.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*. CEUR-WS.org.

Giacomo Berardi, Andrea Esuli, and Diego Marcheggiani. 2015. Word embeddings go to italy: A comparison of models and training datasets. In *IIR*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. SardiStance@EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Steffen Eger and Alexander Mehler. 2016. On the linearity of semantic change: Investigating meaning variation via dynamic graph models. In *Proceedings of ACL (Volume 2: Short Papers)*, pages 52–58, Berlin, Germany, August.

Eduardo Graells-Garrido, Ricardo Baeza-Yates, and Mounia Lalmas. 2020. Every colour you are: Stance prediction and turnaround in controversial issues. In *12th ACM Conference on Web Science*, WebSci '20, page 174–183, New York, NY, USA. Association for Computing Machinery.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of ACM SIGKDD*, pages 855–864.

Dilek Küçük and Fazli Can. 2020. Stance detection: A survey. *ACM Computing Surveys (CSUR)*, 53(1):1–37.

Mirko Lai, Marcella Tambuscio, Viviana Patti, Giancarlo Ruffo, and Paolo Rosso. 2017. Extracting graph topological information and users' opinion. In *Lecture Notes in Computer Science*, volume 10456 LNCS, pages 112–118. Springer Verlag.

Mirko Lai, Alessandra Teresa Cignarella, Delia Irazú Hernández Farías, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. Multilingual stance detection in social media political debates. *Computer Speech & Language*, 63:101075.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of ACM SIGKDD*, pages 701–710.

Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *Proceedings of ACM SIGKDD*, pages 385–394.

Parinaz Sobhani, Diana Inkpen, and Xiaodan Zhu. 2019. Exploring deep neural networks for multitarget stance detection. *Computational Intelligence*, 35(1):82–97, feb.

Qingying Sun, Zhongqing Wang, Qiaoming Zhu, and Guodong Zhou. 2016. Exploring various linguistic features for stance detection. In *Natural Language Understanding and Intelligent Applications*, pages 840–847, Cham. Springer International Publishing.

Michael Wojatzki, Torsten Zesch, Saif Mohammad, and Svetlana Kiritchenko. 2018. Agree or Disagree: Predicting Judgments on Nuanced Assertions. In *Proceedings of *SEM*, pages 214–224, Stroudsburg, PA, USA.

Zhenhui Xu, Qiang Li, Wei Chen, Yingbao Cui, Zhen Qiu, and Tengjiao Wang. 2019. Opinion-aware knowledge embedding for stance detection. In Jie Shao, Man Lung Yiu, Masashi Toyoda, Dongxiang Zhang, Wei Wang, and Bin Cui, editors, *Web and Big Data*, pages 337–348, Cham.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of NAACL-HLT*, pages 1480–1489.

Arkaitz Zubiaga, Elena Kochkina, Maria Liakata, Rob Procter, Michal Lukasik, Kalina Bontcheva, Trevor Cohn, and Isabelle Augenstein. 2018. Discourse-aware rumour stance classification in social media using sequential classifiers. *Information Processing & Management*, 54(2):273–290.

DeepReading @ SardiStance: Combining Textual, Social and Emotional Features

María S. Espinosa
NLP & IR Group
UNED, Spain
mespinosa@lsi.uned.es

Rodrigo Agerri
HiTZ Center - Ixa
University of the Basque Country UPV/EHU
rodrigo.agerri@ehu.eus

Alvaro Rodrigo
NLP & IR Group
UNED, Spain
alvarory@lsi.uned.es

Roberto Centeno
NLP & IR Group
UNED, Spain
rcenteno@lsi.uned.es

Abstract

In this paper we describe our participation to the SardiStance shared task held at EVALITA 2020. We developed a set of classifiers that combined text features, such as the best performing systems based on large pre-trained language models, together with user profile features, such as psychological traits and social media user interactions. The classification algorithms chosen for our models were various monolingual and multilingual Transformer models for text only classification, and XGBoost for the non-textual features. The combination of the textual and contextual models was performed by a weighted voting ensemble learning system. Our approach obtained the best score for Task B, on Contextual Stance Detection.

1 Introduction

One of the most important research topics in the field of Natural Language Processing (NLP) is automatic information extraction from textual data. The recent rise of social media has completely changed the way in which people communicate their ideas and has thus led to the emergence of new research problems regarding the automatic analysis of online contents, such as sentiment analysis, emotion recognition, or fake news detection. Stance detection (usually considered as a subproblem of sentiment analysis) is part of the aforementioned family of research problems (Küçük and Can, 2020). While there are various formulation of the stance detection task, for SardiStance 2020 the aim is to detect the stance (AGAINST, FAVOR or NEUTRAL) conveyed by a given tweet with respect to a specific, previously given topic (Mohammad et al., 2016), namely, about the Sardines movement in Italy.

Thus, we address the problem of automatic stance detection in tweets written in Italian language for the SardiStance 2020 shared task (Cignarella et al., 2020), organized within EVALITA 2020 (Basile et al., 2020). In this paper we include the participation of three teams within the framework of the DeepReading project [1]: (1) Ixa Group, (2) UNED group, and (3) DeepReading Group. While Ixa focused on developing text classifiers based on textual information only (Task A), UNED was more interested in exploring how to use contextual information available (Task B). Likewise, DeepReading is the product of combining both Ixa and UNED systems into one.

In this sense, the main idea behind our model is to exploit textual information, based on fine-tuning large pre-trained language models for text classification, together with contextual information using several feature categories, such as psychological traits of the user, social media data, and network based features. As a result of our joint effort, we submitted 4 and 5 runs, respectively, to tasks A and B. The official results show that our systems obtained the 3rd position among the constrained runs submitted to Task A, which considered only textual information for prediction, and 1st position from 13 participants for Task B, which considered textual and contextual information.

---

[1] http://ixa2.si.ehu.es/deepreading/

2 Systems Description

In this section we first describe the text classification systems developed for Task A and then the contextual features used to train XGBoost classifiers for Task B. We also include a description of the strategies used to combine the classifiers from both tasks, which resulted in the winner system for Task B.

2.1 Task A: Textual Stance Detection

The main objective of our participation in Task A was to benchmark the performance, on the stance detection task for Italian, of large pre-trained language models based on the transformer architecture (Vaswani et al., 2017). This would help us to identify the best performing models which will be leveraged to generate features for Task B (Contextual Stance Detection).

As for many other Natural Language Processing (NLP) tasks, current best performing systems for text classification are based on large pre-trained language models which allow to build rich representations of text based on contextual word embeddings. Deep learning methods in NLP represent words as continuous vectors on a low dimensional space, called word embeddings. The first approaches generated static word embeddings (Mikolov et al., 2013; Bojanowski et al., 2017), namely, they provided a unique vector-based representation for a given word independently of the context in which the word occurs. This means that polysemy cannot be represented.

In order to address this problem, contextual word embeddings were proposed. The idea is to be able to generate word representations according to the context in which the word occurs. Currently there are many approaches to generate such contextual word representations, but we will focus on publicly available multilingual and monolingual pre-trained models for Italian.

There are several multilingual versions of these models. Thus, the multilingual version of BERT (Devlin et al., 2019) was trained for the top 100 languages with the largest Wikipedias. More recently, XLM-RoBERTa (Conneau et al., 2019) distributes a multilingual model which contains 104 languages trained on 2.5 TB of Common Crawl data. Italian is included in both multilingual models.

These multilingual models perform very well in tasks involving high-resourced languages such as English or Spanish, but their performance drops when applied to languages not so well represented in the language model (Agerri et al., 2020). Although this is still an open issue, a number of reasons can be found in the literature. First, each language has to share the quota of substrings and parameters with the rest of the languages represented in the pre-trained multilingual model. As the quota of substrings partially depends on corpus size, this means that larger languages such as English or Spanish are better represented than other languages such as Italian. Moreover, multilingual models also seem to behave better for structurally similar languages (Karthikeyan et al., 2020).

We have benchmarked four monolingual pre-trained language models for Italian: AlBERTo, GilBERTo, UmBERTo and Italian BERT XXL with the aim of comparing them with respect to the multilingual pre-trained models previosly mentioned, namely, mBERT and XLM-RoBERTa.

AlBERTo is a BERT *base* pre-trained lowercased model containing a vocabulary of 128k terms from 200M of Italian tweets (Polignano et al., 2019).

The Italian BERT XXL models [2] are also based on the BERT *base* architecture. The training data contains the Italian Wikipedia, various parts of the OPUS corpus and the OSCAR corpus for Italian (Ortiz Suárez et al., 2019), for a total of 81GB of Italian text.

GilBERTo[3] is based on the RoBERTa *base* (Liu et al., 2019) architecture, an improved, optimized version of BERT which discards the next sentence prediction task. The model was trained using the Italian Oscar (Ortiz Suárez et al., 2019), which contains 71GB of text. The vocabulary used consisted of 32k BPE subwords tokenized by the SentencePiece tokenizer[4].

UmBERTo[5] also leverages the RoBERTa *base* architecture, the OSCAR corpus for Italian and the SentencePiece tokenizer, but it adds Whole Word Masking to the training process. The idea is to mask an entire word, instead of subwords, if at least one of all (sub-)tokens generated by SentencePiece was originally selected as mask.

---

[2] https://github.com/dbmdz/berts

[3] https://github.com/idb-ita/GilBERTo

[4] https://github.com/google/sentencepiece

[5] https://github.com/musixmatchresearch/umberto

2.2 Task B: Contextual Stance Detection

In this task, we use several sets of features with the purpose of trying to model user's behaviour when writing a tweet. We obtain such features from both the text and the social network. Our hypothesis is that the stance of a user regarding a particular tweet is highly correlated with the way of writing of the own user extracted in terms of psychological and emotional features. On the other hand, we focus on exploring how the concept of "homopohily", namely, the tendency of individuals to associate and bond with similar individuals, previously studied in DellaPosta et al. (2015). In order to test this hypothesis, we have tested different models that are explained below.

In this task, we use several sets of features with the purpose of trying to model user's behaviour when writing a tweet. We obtain such features from text and the network.

The complete set of features extracted from the data is depicted in Table 1. The set of features used in the model can be divided into five main types: psychological, emotional, Twitter-based, network-based, and language model features.

Category	Feature name	Description
Psychological features	pers_pred	personality prediction
	self_pred	self-revealing prediction
	info_pred	information-seeking prediction
	action_pred	action-seeking prediction
	fact_pred	fact-oriented prediction
Emotion freatures	arousal	mean arousal value
	valence	mean valence value
	russell	emotion value on Russell's model
Twitter features	statuses_count	number of tweets posted by user
	friends_count	number of following users
	followers_count	number of follower users
	created_at	account creation date
Network features	d_favor	mean distance to users in favor
	d_against	mean distance to users against
	d_none	mean distance to neutral users
Language model features	p_favor	prob. of tweet being in favor
	p_against	prob. of tweet being against
	p_none	prob. of tweet being neutral

Table 1: Complete set of features extracted from the data.

Psychological features. These features were extracted using a third-party API developed by Symanto[6]. Each tweet was sent to the API in order to retrieve the personality traits and communication styles obtained from the analysis of the tweet contents.

The personality traits value would be either "emotional" or "rational" depending on the analysis of the user's text. The value returned by the API when the communication styles are requested is a collection of traits, such as *self-revealing*, which means sharing one's own experience and opinion; *fact-oriented*, which implies focusing on factual information, objective observations or statements; *information-seeking*, that is, posing questions; and *action-seeking* or aiming to trigger someone's action by giving recommendation, requests or advice.

Emotional features. In order to retrieve the emotion values from the tweets, we used Russell's circumplex model of affect (Russell, 1980). Russell argues that emotions can be conceptualized in a two-dimensional continuous space where the axes correspond to the degree of arousal and valence (or pleasure). These two dimensions form a Cartesian space that can be configured in a circular order in which the different combinations of valence and arousal correspond to one of four discrete emotion regions: tired, tense, excited, and pleased.

The values for the degree of arousal and valence of the tweets were obtained using an adaptation to Italian language of the Affective Norms for English Words (ANEW) (Bradley and Lang, 1999). This database was developed from translations of the 1,034 English words present in the ANEW dictionary and from words taken from Italian semantic norms (Montefinese et al., 2014).

Twitter features. Exploring how the users behave in the social network could offer some insights on the stance tendency of the users. The collection of Twitter data of each user contained four features: the number of statuses published by the user, the number of users followed by the user, the number of users following the user, and the creation date of the Twitter account of the user.

Network features. Using the `FRIEND.csv` data provided, we built a network consisting of 669817 nodes (or users) and 2847197 edges (or relationships) in order to represent the *following* network of the users. From that network, we extracted a sub-graph containing the users of known stance from the training data and the users involved in testing in order to calculate the mean distances of each user to the rest of known stance users using the following formula:

$$d_T(n) = \frac{\sum_{i=1}^{|T|} \frac{1}{d^2_{n \to i}}}{|T|}$$

where $|T|$ is the total number of users of a determined stance (AGAINST, FAVOR, NONE) and

---

[6] https://symanto-research.github.io/symanto-docs/

Team	Model	Rank	$\mathbf{F1}_{avg}$	$\mathbf{F1}_{Against}$	$\mathbf{F1}_{Favour}$	$\mathbf{F1}_{None}$
DeepReading	Italian BERT XXL	3	66.21	75.80	56.63	42.13
Ixa	UmBERTo	4	64.73	76.16	53.30	38.88
Ixa	GilBERTo	6	61.71	75.43	48.00	36.75
DeepReading	XML-RoBERTa	8	60.04	69.66	50.42	39.16
-	baseline	12-13	57.84	71.58	44.09	27.64

Table 2: Official Results for Task A.

$d^2_{n \to i}$ corresponds to the square distance in users from node n to node i. From this calculation we obtained 3 values per user: mean distance to users against ($d_{against}$), mean distance to users in favor (d_{favor}), and mean distance to neutral users (d_{none})

Language model features. In order to incorporate the language model results into the rest of the features of the system we choose the best performing, at the development phase, of the models described in Section 2.1, which was UmBERTo. Since this kind of language models use a great amount of features for learning and training, the strategy used in order to incorporate the language model without having a great imbalance in the number of features representing each category, consisted in extracting the probabilities assigned by the model to each class for each tweet. In this way, the language model would be present in 3 of the 18 features of the model, and it would therefore have a balanced size with regards to the rest of features of the model.

3 Results

3.1 Task A

As we use the base version of every transformer model we can fine-tune them in a basic GPU of 12GB RAM. Hyperparameter tuning (batch size, maximum sequence length, learning rate and number of epochs) was performed on the development set. For mBERT, AlBERTo, Italian BERT XXL and UmBERTo the best configuration was: maximum sequence length 256, batch 32, learning rate 5e-5, and 5 epochs. For GilBERTo we used the same values except the number of epochs, which was increased to 10. Finally, the best performing hyperparameters for XLM-RoBERTa was the following: maximum sequence length 256, batch 16, learning rate 2e-5, and 10 epochs.

While the monolingual models clearly outperformed both mBERT and XLM-RoBERTa on the development data, we decided to submit the three best monolingual runs and the best multilingual one. Table 2 reports the official results obtained by each of the models and their position with respect to the ranking of *constrained* runs for Task A released by the task organizers. Our submission based on Italian BERT XXL was clearly the best of our four runs, although its performance was around 1.5 scores in F1 lower than the winner system for Task A. Furthermore, the ranking obtained in the test does not correspond with the results obtained during the development phase, where UmBERTo outperformed the other monolingual models by more than 3 points in F1 score.

3.2 Task B

We presented a total of five models to Task B, which consisted of different combinations of the features listed in Table 1.

Models 1, 2, and 3. During the training and development phases of the models, several configurations were tested on models 1, 2, and 3, including training with different classifiers, such as Random Forest Classifier, Decision Tree Classifier and XGBoost Classifier. The best performing classifier was XGBoost configured for multi-class classification and taking into account class weights in order to deal with the imbalance present in the data. XGBoost is an efficient and scalable implementation of gradient boosting framework by (Friedman, 2001). With regards to the set of features, the first approach to the task considered only psychological, emotion, and Twitter features. For the second model, network features were added to the feature set. Finally, model 3 considered the probabilities of each class (AGAINST, FAVOR, NONE) predicted by the UmBERTo language model as three additional features for training.

Models 4 and 5. These two models were constructed using voting based ensemble learning. The voting system for model 4 considered predictions of models 1, 2, and 3 as well as predictions by the best performing language models on

Team	Model	Rank	$\mathbf{F1}_{avg}$	$\mathbf{F1}_{Against}$	$\mathbf{F1}_{Favour}$	$\mathbf{F1}_{None}$
Ixa	Model 5	1	74.45	85.62	63.29	42.14
DeepReading	Model 3	3	72.30	83.68	60.93	33.64
DeepReading	Model 4	4	72.22	83.00	61.43	42.51
UNED	Model 2	7	68.88	81.75	56.00	24.55
-	baseline	10-11	62.84	76.72	48.95	30.09
UNED	Model 1	13	53.13	73.99	32.26	20.00

Table 3: Ranking results of model 1 to 5 in task B of the competition.

the development data: UmBERTo, GilBERTo, and Italian BERT XXL, described in Section 2.1. The most common predicted value among the 6 systems was chosen as the final prediction of model 4. In case of having two or more values with the same counts, the final value is randomly selected. On the other hand, model 5 used a weighted voting ensemble learning in which each of the systems considered had as weight the F1 value obtained on the development data. Therefore, the model considered the weighted predictions of each system in order to choose the final prediction.

Table 3 shows the official results obtained by each model and their position with respect to the ranking for Task B on Contextual Stance Detection. As it can be noted, model 5 ranked first in this task, obtaining an average F1 of 0.7445. Models 3 and 4 also had promising results in the official test set, ranking third and fourth, respectively, and just 0.0079 below the system which obtained the second best result. Model 2 had a slightly worse performance, ranking seventh from a total of 13, but still 0.0604 above the baseline. Finally, model 1 had the lowest performance, ranking last for the task.

4 Discussion

Figure 1 shows the confusion matrices obtained from the released gold test data for each of the five runs submitted to task B. As it can be noticed, the performance of each model is increasingly better from the first to the fifth, as new features are added to them. The biggest increase, especially with respect to false positives in the AGAINST class, takes place from model 1 to model 2, that is, with the inclusion of network features into the model. This indicates that considering contextual information for stance detection tasks, such as the stance of those who are part of the friendship network of the user, can help determine their stance more accurately.

Furthermore, we can see that predictions from model 3 also experimented a great increase in true positives of each of the classes. This increase is related to the inclusion of the language model into the features of model 2, which demonstrates the importance of textual data in stance detection tasks.

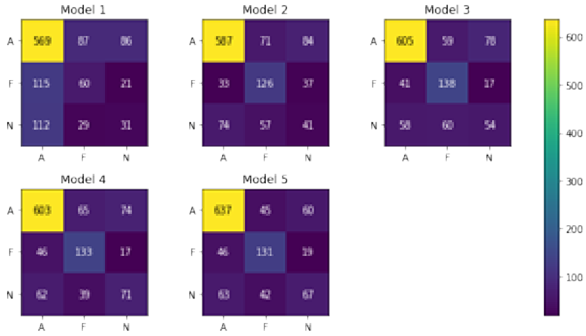


Figure 1: Confusion matrices for models 1 to 5 on test data.

Finally, models 4 and 5 shows the adequacy of combining several complementary systems in order to improve results. Since each single model can detect the stance for different instances, a proper combination of them could outperform single models.

5 Conclusions and Future Work

In this paper we have shown the benefits of exploiting information from different and heterogeneous sources. For our participation to the SardiStance 2020 shared task we have experimented with classifiers trained with the textual content of the tweets as well as with features based on social networks. This combination of features has allowed us to obtain the best overall results in the task.

As future work, we plan to further explore the contribution of network information. Besides, we want to develop new divergent models and study how to combine them.

Acknowledgments

This work has been partially funded by the Spanish Ministry of Science, Innovation and Universities (DeepReading RTI2018-096846-B-C21, MCIU/AEI/FEDER, UE), and DeepText (KK-2020/00088), funded by the Basque Government. Rodrigo Agerri is additionally funded by the RYC-2017-23647 fellowship and acknowledges the donation of a Titan V GPU by the NVIDIA Corporation. Maria S. Espinosa is also funded by the European Social Fund through the Youth Employment Initiative (YEI 2019).

References

[Agerri et al.2020] Rodrigo Agerri, Iñaki San Vicente, Jon Ander Campos, Ander Barrena, Xabier Saralegi, Aitor Soroa, and Eneko Agirre. 2020. Give your text representation models some love: the case for basque. In *LREC 2020*, pages 4781–4788.

[Basile et al.2020] Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *EVALITA 2020*. CEUR-WS.org.

[Bojanowski et al.2017] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL*, 5:135–146.

[Bradley and Lang1999] Margaret M Bradley and Peter J Lang. 1999. Affective norms for english words (anew): Instruction manual and affective ratings. Technical Report 1, Technical report C-1, the center for research in psychophysiology, University of Florida.

[Cignarella et al.2020] Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. SardiStance@EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

[Conneau et al.2019] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv:1911.02116*.

[Devlin et al.2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019*, pages 4171–4186.

[Friedman2001] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

[Karthikeyan et al.2020] K Karthikeyan, Zihan Wang, Stephen Mayhew, and Dan Roth. 2020. Cross-lingual ability of multilingual bert: An empirical study. In *International Conference on Learning Representations (ICLR)*.

[Küçük and Can2020] Dilek Küçük and Fazli Can. 2020. Stance detection: A survey. *ACM Computing Surveys (CSUR)*, 53(1):1–37.

[Liu et al.2019] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

[Mikolov et al.2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

[Mohammad et al.2016] Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. SemEval-2016 task 6: Detecting stance in tweets. In *SemEval-2016)*, pages 31–41.

[Montefinese et al.2014] Maria Montefinese, Ettore Ambrosini, Beth Fairfield, and Nicola Mammarella. 2014. The adaptation of the affective norms for english words (anew) for italian. *Behavior research methods*, 46(3):887–903.

[Ortiz Suárez et al.2019] Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019, pages 9–16.

[Polignano et al.2019] Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019. AlBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*, volume 2481. CEUR.

[Russell1980] James A Russell. 1980. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161.

[Vaswani et al.2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

TextWiller @ SardiStance, HaSpeede2: Text or Con-text? A Smart Use of Social Network Data in Predicting Polarization

Federico Ferraccioli[a], **Andrea Sciandra**[b], **Mattia Da Pont**[c], **Paolo Girardi**[a],
Dario Solari[d], **Domenico Madonna**[a], **Livio Finos**[a]
a. Università degli Studi di Padova
b. Università degli Studi di Modena e Reggio Emilia
c. WMRI
d. BeeViva
`ferraccioli@stat.unipd.it`, `andrea.sciandra@unimore.it`, `mattia.dapont@wmr.it`,
`paolo.girardi@unipd.it`, `dario.solari@gmail.com`, `domenico.madonna@studenti.unipd.it`,
`livio.finos@unipd.it`

Abstract

In this contribution we describe the system (i.e. a statistical model) used to participate in Evalita conference 2020, SardiStance (Tasks A and B) and Haspeede2 (Tasks A and B). We first developed a classifier by extracting features from the texts and the social network of users. Then, we fit the data through an extreme gradient boosting, with cross-validation tuning of the hyper-parameters. A key factor for a good performance in SardiStance Task B was the features extraction by using Multidimensional Scaling of the distance matrix (minimum path, undirected graph) applied on each network. The second system exploits the same features above, but it trains and performs predictions in two-steps. The performances proved to be lower than those of the single-step model.

1 Introduction

In this paper we describe and show the results of the approach we developed to participate in the SardiStance task (Cignarella et al., 2020) for the polarity detection (i.e. Task A and B, both with constrained data) within the EVALITA campaign (Basile et al., 2020). The goal of this task was a Stance Detection in Italian tweets about the Sardines movement. The Task A is a three-class classification task where the system has to predict whether a tweet is in *Favour*, *Against* or *Neutral/none* towards the given target, exploiting only textual information, i.e. the text of the tweet. The Task B is the same as the first one, except a wider range of contextual information are available, that is: the number of retweets, the number of favours, the type of posting source (e.g. iOS or Android), and date of posting. Furthermore, the networks of the users based on Friends, Quote, Reply and Retweet were provided. We developed two systems (i.e. models) extracting features from the text (both for Task A and B) and from the social network of the users (only for Task B) and then exploited extreme gradient boosting (Chen et al., 2020) to train the model on the data. A cross-validation hyper-parameter tuning was used to define the optimal set of parameters.

We use a very similar strategy for HaSpeede2 (Sanguinetti et al., 2020) where the goal is the prediction of Hate Speech (i.e. Task A) and Stereotype (i.e. Task B). In this case, however, the sample contains documents from three different topics. We believe that these may be characterized by different vocabularies and kind of speech. We take this in account in the prediction model as explained in 3.3.

2 Features extraction and E.D.A.

2.1 Text-based Features extraction

The text preprocessing was done in R (R Core Team, 2019) software with the package TextWiller (Solari et al., 2019) (function *normalizzaTesti* with default parameters). We describe the preocess used to define the features for both for SardiStance and HaSpeede2.

The first set of features is defined by the

---

columns of the DocumentTermMatrix which is a matrix having documents on the rows and a column for each term. The cells contain the number of given words in the document. We defined the matrix on the basis of the normalized texts and removing terms (i.e. columns) with a sparsity larger than .9. These procedures generated a 317 terms vocabulary for SardiStance and 170 terms for HaSpeede2.

In Figure 1 we plot the term frequencies of the ”In favour” and ”Against” stances. The terms close to the bisector are the ones with a similar frequency in the two classes (such as ”caro”, ”alto”, ”acqua”), so probably these terms don’t carry much useful information to our cause. More often we found interesting terms far from the bisector, like ”bolognanonsilega”, ”antifascismo”, ”abuso” or ”branco” and we expected these terms to carry more weight in the classification model.

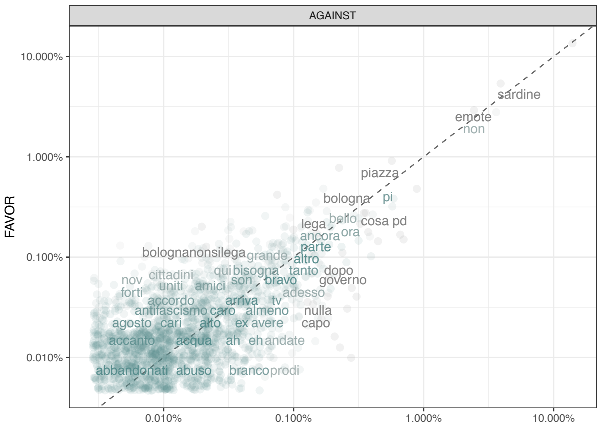


Figure 1: Scatterplot of ”Favour” and ”Against” term frequencies.

Further text features considered were: the number of characters and the number of words, the counts of ”?” and ”!” for each document. Moreover, a sentiment value was computed for each document by *sentiment* function of the R package TextWiller (Solari et al., 2019).

Figure 2 shows the association between True Stances and Sentiment. This variable will be used as a feature in Task A and B models.

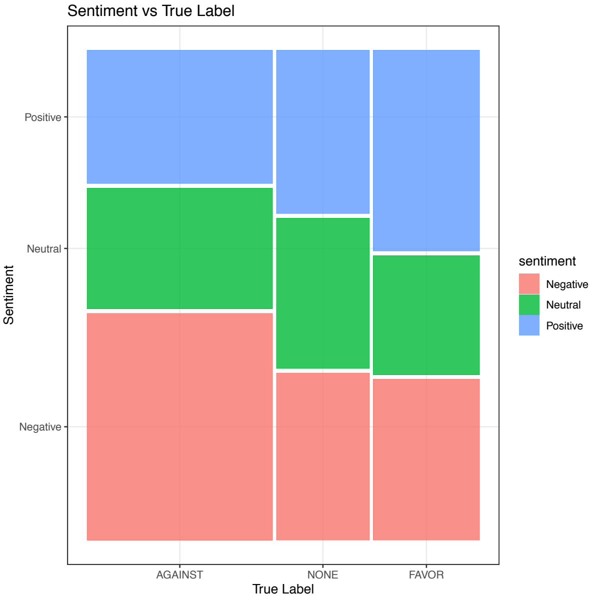


Figure 2: The Mosaic plot of True stances and Sentiment shows a clear association between the two variables.

Previous analyses, such as sentiment attribution through a lexicon, refer to a bag-of-words (BoW) approach. One of the most notable disadvantages of BoW is that it generally fails to capture words semantics by ignoring words order. A common solution to this problem involves the use of Word Embedding (WE). WE techniques are based on neural networks and generate dense vectors for word representation, by defining a context window, i.e. a string of words before and after a focal word, that will be used to train a word embedding model. In WE, words are represented as coordinates on a latent multidimensional space derived from an underlying deep learning model that considers the contiguous words. So, for both tasks we also used a WE technique to produce context-based features. In particular, we used the *word2vec* model (Mikolov et al., 2013), a widely used natural language processing technique to extract word associations from a large corpus of text. *word2vec* is a neural network prediction model containing continuous bag-of-words (CBoW) model and Skip-gram (SG) model. The CBoW model predicts a target word from its context words, while the SG model predicts the context words given a target word. Since WE needs a huge corpus of textual data for training and given the limited amount of tweets, we augmented the data with the corpus PAISÀ (Lyding et al., 2013), a large collection of Italian web texts. We trained the model with embedded dimension set to 50 and a 5 words context window. The results for each word are then combined via averaging to obtain the final features.

2.2 Network-based Features extraction

A key point to explain the good performance in the SardiStance Task B (i.e. second best score, F-avg = 0.7309) is the efficient extraction of features from the four Networks available, that is: Friends, Retweet, Reply, and Quote. For each network, a distance matrix among subjects was computed. The distance used is the shortest path, forcing the graph to be undirected. The Distance Matrix was then projected into a euclidean space trough a Multidimensional Scaling (MDS). Since we expected the users to be strongly polarized in clusters within the network, we also expected the largest dimension to discriminate among the stances. Therefore, we retained the first and second dimension for each of the four networks. This expectation was confirmed by Exploratory Data Analysis. As an example, in Figure 3 we show the scatter plot of the first two dimensions for the Friend Network. The First Dimension clearly discriminates the three stances (in particular *Favour* vs *Against*).

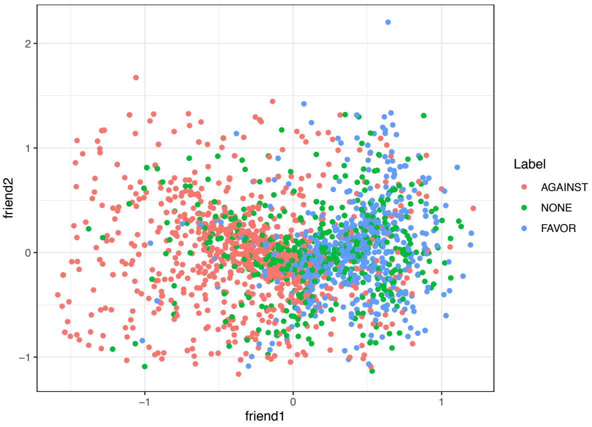


Figure 3: Scatter plot of the First and Second dimension extracted by the MDS from the distance matrix of the Friend Network (minimum path distance). There is a clear separation between the stances *Favour* and *Against* along the first axis.

3 Developed Systems

Due to the – relatively – small sample size of the train set (composed from 2,132 tweets in Italian, the BenderRule), we decided not to use any neural network. Instead, we preferred a Gradient Boost approach (Friedman, 1999). Since this method has been developed within the statistical learning community, we used the word "model" as a synonymous of "system". We adopted the R implementation of the XGBoost (eXtreme Gradient Boosting) (Chen et al., 2020). A cross-validation parameter tuning was used to define the optimal set of parameters.

3.1 System One

As features for Task A, we used information taken from the text, that is, words/emoticons, special characters, scores of word embedding (50 dimensions), sentiment, length of the message and number of words.

For Task B we used the same features used for Task A together with the first and the second dimension extracted from the MDS computed for each network (as explained in 2.2).

3.2 System Two

Since System Two uses the same features of System One for Task A and B, the focus here is on the employed metric: the average between $F1_{Against}$ and $F1_{Favour}$. With the aim to cast the model into the metric, we fitted two separated models (i.e. one for *Favour* and one for *Against*) in the first step and then we combine the two predictions in a second step. To be more precise, the two models used in the first step predict if a document is in Favour or not (first model) and if is Against or not (second model). The two prediction are combined in a final score by a simple subtraction: *(Predicted1==Favour) - (Predicted2==Against)* which makes a -1,0,1 final score.

3.3 System for HaSpeeDe2

The corpus of documents for HaSpeeDe2 is a sample of tweets from three different topics, namely Immigrants, Muslims and Roma communities. Since the vocabulary may change among topic, we want our models to account for this specificity. We leverage on this with models that use the estimated topic. The topic is estimated by a xgboost model (trained by cross-validation). Table 1 and Table 2 report the confusion matrix and performances indices of the trained model (cross-validated).

	Reference		
Prediction	Immigrants	Rom	Terrorism
Immigrants	408	24	55
Rom	24	780	16
Terrorism	41	8	192

Table 1: Confusion matrix for the xgboost model.

Index	Immigrants	Rom	Terrorism
Sensitivity	0.86	0.96	0.73
Specificity	0.93	0.95	0.96
F1	0.85	0.96	0.76

Table 2: Sensitivity, Specificity and F1 for each topic for the xgboost model.

System One is based on an xgboost with binomial response (for both tasks). The fitting is done separately, after splitting of the sample based on the topic classification provided by the model described above in this subsection. The model is trained with the same cross-validated strategy used to train System One for the SardiStance Task.

System Two is based on an xgboost with binomial response (for both tasks). The estimate is computed on the whole sample (i.e. without splitting of System One), but the topic classification is used as feature.

For both systems the basic set of features are the same used in the SardiStance - Task A.

4 Results and discussion

4.1 Results for HaSpeeDe2

The results of the two systems are disappointing. The final ranks are always at the very bottom of the rankings. This may be partially due to a suboptimal parameters optimization (we discovered a mistake in the parameter setting), but this is certainly not the only reason. We will take this result as an opportunity to revise the approach.

4.2 Results for SardiStance

System Two performed poorly in the final score for both Tasks. Our intuition was that the benefit of a separate optimization of $F_{Against}$ and F_{Favour} was overcome by the gain in doing a joint training (i.e. System One). We will address further efforts to better understand this result.

The results for System One are given in Table 3 for Task A and Table 4 for Task B, respectively.

The rank of System One in Task A is 13, that is just below the benchmark. The System was weak in the correct estimation of Against stance ($F1_{Against} = 0.776$), while it estimated fairly well Favour stance ($F1_{Favour} = 0.3791$).

The best performance of System One is on Task B ($F1_{Against} = 0.8505$, $F1_{Favour} = 0.6114$) where it scored 2^{nd} position.

	Reference		
Prediction	AGAINST	NONE	FAVOUR
AGAINST	613	118	108
NONE	32	22	12
FAVOUR	97	32	76

Table 3: Confusion Matrix for Task A (System One). $F1_{Against} = 0.776$, $F1_{Favour} = 0.3791$, Final: $(F1_{Against} + F1_{Favour})/2 = 0.5773$

	Reference		
Prediction	AGAINST	NONE	FAVOUR
AGAINST	623	71	29
NONE	54	44	27
FAVOUR	65	57	140

Table 4: Confusion Matrix for Task B (System One). $F1_{Against} = 0.8505$, $F1_{Favour} = 0.6114$, Final: $(F1_{Against} + F1_{Favour})/2 = 0.7309$

To support the intuition that network-based features play a crucial role in this model, we explore the Importance of the Features. Results are given in Table 4.2 (Top 10).

	Feature	Importance
1	NW_Retweet1	0.13
2	NW_Friend1	0.12
3	NW_Quote2	0.04
4	Created_at	0.02
5	WE24	0.02
6	Statuses_count	0.02
7	NW_retweet2	0.02
8	WE14	0.02
9	We10	0.01
10	WE25	0.01

Table 5: Top 10 Features' Importance. Legend: NW = MDS dimension of the network; WE = Word-Embedding dimension.

The top three far more important features were dimensions extracted by the MDS approach explained in section 2.2.

5 Conclusion

For SardiStance, the System One proposed here performed well in the Task B, while it has a much poorer result in Task A. It exploits a simple method to handle the network-based information, while further refinement should be made on the exploitation of text-based information. In this way we want to stress the importance of data mashup, as the system we deployed showed better results for Task B which contains, in addition to texts, information of a different nature derived from network structures.

It is to be expected that more networks should carry similar information. A future direction of research should be the joint analysis of the Networks. There is a sparkling community working on multilayer Networks (De Domenico et al., 2013) (Durante et al., 2017) that may inspire more effective use of this joint information.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*. CEUR-WS.org.

Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, Rory Mitchell, Ignacio Cano, Tianyi Zhou, Mu Li, Junyuan Xie, Min Lin, Yifeng Geng, and Yutian Li, 2020. *xgboost: Extreme Gradient Boosting*. R package version 1.0.0.2.

Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. SardiStance@EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Manlio De Domenico, Albert Solé-Ribalta, Emanuele Cozzo, Mikko Kivelä, Yamir Moreno, Mason A. Porter, Sergio Gómez, and Alex Arenas. 2013. Mathematical Formulation of Multilayer Networks. *Physical Review X*, 3(4):041022, October.

Daniele Durante, David B. Dunson, and Joshua T. Vogelstein. 2017. Nonparametric bayes modeling of populations of networks. *Journal of the American Statistical Association*, 112(520):1516–1530.

Jerome H. Friedman. 1999. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38:367–378.

Verena Lyding, Egon Stemle, Claudia Borghetti, Marco Brunello, Sara Castagnoli, Felice Dell'Orletta, Henrik Dittmann, Alessandro Lenci, and Vito Pirrelli. 2013. PAISÀ corpus of italian web text. Eurac Research CLARIN Centre.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

R Core Team, 2019. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Manuela Sanguinetti, Gloria Comandini, Elisa Di Nuovo, Simona Frenda, Marco Stranisci, Cristina Bosco, Tommaso Caselli, Viviana Patti, and Irene Russo. 2020. Overview of the evalita 2020 second hate speech detection task (haspeede 2). In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Dario Solari, Andrea Sciandra, and Livio Finos. 2019. Textwiller: Collection of functions for text mining, specially devoted to the italian language. *Journal of Open Source Software*, 4(41):1256.

UninaStudents @ SardiStance: Stance Detection in Italian Tweets - Task A

Maurizio Moraca, Gianluca Sabella, Simone Morra
Università degli Studi di Napoli Federico II
`[mau.moraca,gia.sabella,simone.morra2]@studenti.unina.it`

Abstract

English. This document describes a classification system for the SardiStance task at EVALITA 2020. The task consists in classifying the stance of the author of a series of tweets towards a specific discussion topic. The resulting system was specifically developed by the authors as final project for the Natural Language Processing class of the Master in Computer Science at University of Naples Federico II. The proposed system is based on an SVM classifier with a radial basis function as kernel making use of features like 2 char-grams, unigram hashtag and Afinn weight computed on automatic translated tweets. The results are promising in that the system performances are on average higher than that of the baseline proposed by the task organizers.

Italiano. *Questo documento descrive un sistema di classificazione per il task SardiStance di EVALITA 2020. Il task consiste nel classificare la posizione dell'autore di una serie di tweets nei confronti di uno specifico topic di discussione. Il sistema risultante è stato specificamente sviluppato dagli autori come progetto finale per il corso di Elaborazione del Linguaggio Naturale nell'ambito del corso di laurea magistrale in Informatica presso l'università degli studi di Napoli Federico II. Il sistema qui proposto si basa su un classificatore SVM con una funzione radiale di base come kernel facendo uso di features come 2 char-grams, unigram hashtag e l'Afinn weight calcolato sui tweet tradotti in automatico. I risultati sono promettenti in quanto le performance sono in media superiori rispetto a quelle della baseline proposta dagli organizzatori del task.*

1 Introduction

This work reports on the application of our system for the resolution of the EVALITA 2020's SardiStance task (Basile et al., 2020; Cignarella et al., 2020). Stance detection is a classification task aiming at determining the position (stance) of the author of a given text concerning the topic (target) treated in the text itself. In other words, the challenge deals with automatically guessing if the author of the text is in favour, against or is in a neutral position towards the topic subject of a given post. The utility of such an automatic system can be found in political analysis, marketing and opinion mining. Automatic determination of Stance is a new approach to opinion mining paradigm which finds better application in social and political applications. It is quite different form in which sentiment analysis in many views, but the main difference is the drastic reduction to a three class decision system (in favour, against, neutral) given its main fields of application. The challenge poses many challenges, as the real target might not be expressly cited in the text or could bear a not so clear expression of the author's opinion like in the following example (Lai et al., 2020):

Target: Donald Trump
Tweet: Jeb Bush is the only sane candidate in this republican lineup.

Although one could erroneously think that

---

this task is similar to sentiment analysis, the following example illustrates how, in some cases, stance detection results are opposed to those reached by sentiment analysis (Lai et al., 2020):

Target: Climate change is a real concern
Tweet: @RegimeChangeBC @ndnstyl It's sad to be the last generation that could change but does nothing. #Auspol
This tweet presents a negative polarity, although the author claims to be in favour of the target. Classification systems for stance detection, then, attempt the individuation of the author position on the target taking into account of features obtained by the text that are almost similar to those used in hate speech detection, irony detection, mood detection, but with some further effort devoted to the specificity of the task.

SardiStance is the first Italian Initiative focused on the automatic classification of stance in tweets. It includes two different tasks: A) Stance Detection at a textual level, where tasl participants are asked to resolve the guess basing only on the tweet textual content, and B) Stance Detection with the addition of contextual information about the tweet, such as the number of retweets, the number of favours or the date of posting; contextual information about the author, location, user's biography); we proposed runs only for task A). As required by the task proposal, task A requires a three-class classification process where the system has to predict whether the items in the set are in FAVOUR, AGAINST or NEUTRAL exploiting the text of the tweet.

2 Description of the System

The system is based on a SVM classifier with a radial basis function (rbf) kernel. Most of the features selected were inspired by (Lai et al., 2020) and correspond to the following ones:

- n-grams, bag of n consecutive words in binary representation (presence/absence) where n corresponds to 1, 2 or 3.
- char-grams, bag of n consecutive characters in binary representation (presence/absence) where n corresponds to 2, 3, 4 or 5.
- unigram hashtag, bag of hashtags in binary representation (presence/absence).
- unigram emoji, bag of emojis in binary representation (presence/absence)
- unigram mentions, bag of mentions in binary representation (presence/absence).
- num uppercase words, number of uppercase words in a tweet.
- punctuation marks, frequency of each punctuation mark (*. , ; ! ?*) and their total frequency.
- Afinn weight[1] (Nielsen, 2011), based on a sentiment analysis lexicon made up of 3500 English words manually annotated with a polarity value within the range [-5, +5]. The value of this feature is computed for each tweet as the sum of the polarities associated to the words constituting the tweet translated to English via Google Translate.
- Hu&Liu weight[2], based on a sentiment analysis lexicon composed of two separated lists of English words, where the first one contains 2,006 words with a positive connotation, and the second one contains 4,783 words with a negative connotation. In this work, a value of +1 is given to words which overlap with the positive ones in the lexicon and a value of -1 to the ones overlapping with the negative list. The total polarity of each tweet is computed as the sum of the weights given to the words in a tweet.
- NRC vector[3] (Bravo-Marquez et al., 2019), based on a lexicon consisting in a list of English words, each of which is associated to the most representative emotion. The emotion which are comprised are anger, fear, expectancy, trust, surprise, sadness, joy, and disgust. Furthermore, to each sample, a score indicating the emotion intensity is also associated. This score has a value within the range [0, 1].
- DPL vector[4] (Castellucci et al., 2016), based on a lexicon of 75,021 pairs of

---

[1] https://github.com/fnielsen/afinn/tree/master/afinn/data

[2] https://github.com/woodrad/Twitter-Sentiment-Mining/tree/master/Hu%20and%20Liu%20Sentiment%20Lexicon

[3] http://saifmohammad.com/WebPages/AffectIntensity.htm

[4] http://sag.art.uniroma2.it/demo-software/distributional-polarity-lexicon/

lemma::pos_tag associated to scores indicating the level of positivity, negativity, and neutrality of the lemma, as it follows

(1) buono::a 0.76691014 0.12262548 0.11046442

For each tweet of the dataset, each word was lemmatised and, for each resulting lemma, a morpho-syntactic category was associated. For this kind of analysis LinguA (Dell'Orletta, 2009; Attardi and Dell'Orletta, 2009; Attardi et al., 2009) was used. The DPL vector feature consists of a triplet of scores representing positivity, negativity, and neutrality levels in the tweet. To obtain this value, the scores of each pair lemma::pos_tag in a tweet were summed.

In order to select the best features combination, a wrapper-based feature selection algorithm was used to test all the possible features combinations. The best one resulting from the collected performance on the validation set was chosen, that is the one combining 2 char-grams, unigram hashtag and Afinn weight. The evaluation metrics are discussed in the next section (Section 3). Since a SVM classifier with an RBF kernel was used, it was important to tune the C and γ parameters.

To set the complexity of a generic SVM model, C is used: this parameter controls the acceptable distance of the decision boundary in the n-dimensional features space from the support vectors. A higher C complexity value increases the model's complexity, thus reducing the acceptable distance but also increasing the risk of overfitting; a lower C value leads to more general models that may have reduced discrimination capability. The γ parameter is specific for the RBF kernel. This parameter controls the influence single points have in the features space and controls the *smoothness* of the model, with lower values of γ leading to smoother models and vice-versa. SVMs are very sensitive to parameters tuning so specific optimisation strategies must be adopted. In this case, a grid search was performed using the following ranges of values:

- C [0.1, 0.2, …, 1.0, 10, 100, 1000]
- Gamma [0.001, 0.0009, 0.0008, …, 0.0001]

The best settings obtained on the validation set data correspond to $C = 10$ e $\gamma = 0.001$.

3 Results

In this section the performances of our system obtained during the test phase on the validation and test set are described. The validation set was obtained extracting a sample of tweets from the training set via the Stratified Sampling algorithm selecting the 20% of the training set. The evaluation metrics used are the mean value of the F1 score for the classes Against and Favour, Precision, Recall and F1 score for each class, and Accuracy. In table 3, the results obtained from the validation set are shown. From these results, the mean F1 score is obtained, corresponding to 0.5200. In table 3, the results obtained from the test set are presented.

	Precision	**Recall**	**F1 Score**
Against	0.5500	0.8300	0.6600
Favor	0.4400	0.3200	0.3100
None	0.3800	0.1300	0.0900

Table 1: Validation Set Performance

	Precision	**Recall**	**F1 Score**
Against	0.7300	0.8491	0.7850
Favor	0.4348	0.3571	0.3922
None	0.3488	0.1744	0.2326

Table 2: Test Set Performance

Team	**F1-score**		
	Against	**Favour**	**None**
UNITOR_1	0.7866	**0.5840**	0.3910
UNITOR_2	0.7881	0.5721	**0.3979**
UNITOR_3	**0.7939**	0.5647	0.3672
UNITOR_4	0.7689	0.5522	0.3702
UninaStudents	**0.7850**	0.3922	0.2326
Baseline	0.7158	0.4409	0.2764

Table 3: Results compared with the baseline and the winning system

In table 3, on the other hand, the results are compared with the baseline proposed by the task organizers and the winning systems whose runs were submitted by the UNITOR team (Giorgioni et al., 2020) for task A. Specifically, the baseline used a SVM classifier based on token uni-gram features, whereas UNITOR used UmBERTo[5], adding sentiment, hate and irony tags to the dataset sentences and using additional data to train their systems. As it may be noted, the *against* class result for our system is higher than the baseline and not so different from the first two runs of UNITOR. Further investigations are, conversely, needed as far as the other two classes are concerned.

4 Discussion

Our results are conditioned by the use of a training set originally in English and translated into Italian for our purposes, and, in particular, for the derivation of the Afinn weight features. As expected, the translation, made via Google translate is, in some cases poor and approximate, and can give rise to a significant level of ambiguity, however we decided to afford this risk, translating directly the tweets, instead of the lexicon, as we thought that in this last case the ambiguity could have been even greater, we just hoped that automatic translation is by far more uncertain because of polysemy, lack of flexive morphological information, and similar problems, as automatic translation skills are trained to solve at least at a first level of aproximation. In this view the use of an imperfect translation, however, is able to capture part of the semantic context in the texts, allowing us not to recur to lemmatization and further processes on the lexicon before translation. We choose to use a classic approach based on an SVM classifier in order to make our results explainable, given the scholar context in which this experience is grown. This possibility would have been impossible if we had used Deep Neural Networks, whose processes are not "readable" from an external point of view. Furthermore, the size of the data-set distributed for this challenge does not consent an affordable training with these systems. In this view, a comparison of results obtained in other stance detection challenges, similar to that proposed here in Evalita (Mohammad et al., 2016; Taulé et al., 2017; Lai et al., 2017), give strength to our choice concerning the use of SVM that often outperform DNNs. As Master students, we approached these NLP topics for the first time. Therefore, we are aware that our results are not at the state of the art in the field. However, a comparison with average performances in similar tasks for languages different from English indicates performances that are not significantly different.

Acknowledgements

We thank our teachers Francesco Cutugno and Maria Di Maro for letting us approach with NLP and EVALITA 2020 (Basile et al., 2020) and for supporting us in our work. We also thank them for giving us the opportunity to take part to the competition and for encouraging us to do our best.

References

Attardi, G. and Dell'Orletta, F. (2009). Reverse revision and linear tree combination for dependency parsing. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 261–264.

Attardi, G., Dell'Orletta, F., Simi, M., and Turian, J. (2009). Accurate dependency parsing with a stacked multilayer perceptron. *Proceedings of EVALITA*, 9:1–8.

Basile, V., Croce, D., Di Maro, M., and Passaro, L. C. (2020). Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Basile, V., Croce, D., Di Maro, M., and Passaro, L. C., editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Bravo-Marquez, F., Frank, E., Pfahringer, B., and Mohammad, S. M. (2019). Affectivetweets: a weka package for analyzing affect in tweets. *Journal of Machine Learning Research*, 20(92):1–6.

Castellucci, G., Croce, D., and Basili, R. (2016). A language independent method for generating large scale polarity lexicons. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 38–45.

Cignarella, A. T., Lai, M., Bosco, C., Patti, V., and Rosso, P. (2020). SardiStance@EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets. In Basile, V., Croce, D., Di Maro, M., and Passaro, L. C., editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

---

[5] https://huggingface.co/Musixmatch/umberto-commoncrawl-cased-v1

Dell'Orletta, F. (2009). Ensemble system for part-of-speech tagging. *Proceedings of EVALITA*, 9:1–8.

Giorgioni, S., Politi, M., Salman, S., Croce, D., and Basili, R. (2020). UNITOR@Sardistance2020: Combining Transformer-based architectures and Transfer Learning for robust Stance Detection. In Basile, V., Croce, D., Di Maro, M., and Passaro, L. C., editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Lai, M., Cignarella, A. T., Farías, D. I. H., Bosco, C., Patti, V., and Rosso, P. (2020). Multilingual stance detection in social media political debates. *Computer Speech & Language*, page 101075.

Lai, M., Cignarella, Alessandra Teresa, H. F. D. I., et al. (2017). itacos at ibereval2017: Detecting stance in catalan and spanish tweets. In *IberEval 2017*, volume 1881, pages 185–192. CEUR-WS. org.

Mohammad, S., Kiritchenko, S., Sobhani, P., Zhu, X., and Cherry, C. (2016). Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41.

Nielsen, F. Å. (2011). A new ANEW: evaluation of a word list for sentiment analysis in microblogs. In Rowe, M., Stankovic, M., Dadzie, A.-S., and Hardey, M., editors, *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages*, volume 718 of *CEUR Workshop Proceedings*, pages 93–98.

Taulé, M., Martí, M. A., Rangel, F. M., Rosso, P., Bosco, C., Patti, V., et al. (2017). Overview of the task on stance and gender detection in tweets on catalan independence at ibereval 2017. In *2nd Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2017*, volume 1881, pages 157–177. CEUR-WS.

SSN_NLP@SardiStance : Stance Detection from Italian Tweets using RNN and Transformers

Kayalvizhi S
SSN College Of Engineering
kayalvizhis@ssn.edu.in

Thenmozhi D
SSN College Of Engineering
theni_d@ssn.edu.in

Aravindan Chandrabose
SSN College Of Engineering
aravindanc@ssn.edu.in

Abstract

Stance detection refers to the detection of one's opinion about the target from their statements. The aim of sardistance task is to classify the Italian tweets into classes of favor, against or no feeling towards the target. The task has two sub-tasks : in Task A, the classification has to be done by considering only the textual meaning whereas in Task B the tweets must be classified by considering the contextual information along with the textual meaning. We have presented our solution to detect the stance utilizing only the textual meaning (Task A) using encoder-decoder model and transformers. Among these two approaches, simple transformers have performed better than the encoder-decoder model with an average F1-score of 0.4707.

1 Introduction

Stance is the opinion of a person against or in favor of the target. In the sardistance task, the stance detection refers to the detection of stance from the Italian tweets collected from Sardines movement. The tweets imply the authors' standpoint towards the target. The aim of this task is to detect the stance of the author with the help of textual and contextual information about the tweets. The task has two sub-tasks in which the stance is detected using only textual information in one sub-task while the other sub-task makes use of contextual meaning along with the textual meaning.

2 Related Work

Many approaches have been done to detect stance from the English text. Stance text are vectorized and then detected using Multi-layer Perceptron (MLP) (Riedel et al., 2017). Different methodologies like Support Vector Machine, Long Short Term Memory (LSTM) and Bi-directional LSTM (Augenstein et al., 2016) have also been used to detect stance. Recurrent Neural Network (RNN) (Yoon et al., 2019) and altering recurrent networks with different short connections pooling and attention layers have also been experimented in (Borges et al., 2019) to detect stance. Bi-directional Encoder Representation of Transformers (BERT) (Devlin et al., 2018) and Named Entity Recognition (NER) model (Küçük and Can, 2019) have also been used to detect stance. A large dataset has been collected from twitter and all the existing approaches have been discussed in (Conforti et al., 2020).

For other languages, a multilingual data set (Vamvas and Sennrich, 2020) have been taken, language is identified and then multi-lingual BERT model have been used to detect stance. Stance have been detected in Russian Language (Lozhnikov et al., 2018) by vectorizing using Tf-IDF and then classifying using different classifiers like Bagging, AdaBoost Boosting, Stochastic Gradient Descent classifier and Logistic Regression. Stance from different languages (Lai et al., 2020) like English, Italian, French, Spanish have been detected using different features extraction.

3 Task Description

The sardistance task (Cignarella et al., 2020) of Evalita (Basile et al., 2020) has two sub-tasks namely Task A - textual stance detection and Task B - contextual stance detection.

Both tasks are classification tasks that have three classes namely favor, against and none. In the first task, the system has to predict the class by using only the textual information from the tweets whereas in the second task it has to predict the label with the help of some additional information

---

like

Details of post : the number of re-tweets, replies, quotes

Details of user : the number of tweets, user bio's, user's number of friends and followers

Details of their social network : friends, replies, re-tweets, quotes' relation.

In both the tasks, there can be two submissions like constrained where we have to use only the dataset provided and unconstrained where we can use some additional data if required. Each team can submit two runs for both constrained and unconstrained runs.

3.1 Data set description

For Task A, the train.csv file was provided with three columns namely tweet_id,user_id and text label. For Task B, files namely tweet.csv, user.csv, friend.csv, quote.csv, reply.csv and re-tweet.csv are given to explain the contextual details about the post, user and social network. For both the tasks, the training set had about 2,132 instances and the test set had about 1,110 instances. In the training set, there are 1,028 instances in the against class, 587 favor instances and 515 neutral instances which is explained in Table 1. In the testing set, there are 742 against instances, 196 favor instances and 687 none instances.

4 Methodology

The stances were detected using an encoder-decoder model which is a recurrent neural network with different recurrent units and using transformers.

4.1 Data pre-processing

The data is pre-processed by removing the hash tags, '@' symbols, Unicode characters and punctuation.

4.2 Recurrent Neural Network

In this approach, the stance were detected using a encoder-decoder model (Luong et al., 2017) using Gated Recurrent unit(GRU) as its recurrent unit and Scaled Luong (Luong et al., 2015) as its attention mechanism. The model has two encoder-decoder layers along with the embedding layer that vectorizes the input and a loss layer that calculates the loss function. Recurrent Neural Network has been made use to detect the stance since it captures the contextual long-short term dependencies.

4.2.1 Encoder-Decoder Model

The encoder-decoder model is a Neural Machine Translation (NMT) model with sequential data model with Recurrent Neural Network (RNN). The Seq-to-Seq model differs in terms of type of recurrent unit, residual layers, depth, directionality and attention mechanism. The types of the recurrent unit are Long Short Term Memory(LSTM), Gated Recurrent Unit (GRU) and Google Neural Machine Translations. The depth is altered by changing the number of layers and the directionality is either uni-directionality or bi-directionality.The two types of attention mechanism are scaled luong (sl) and normed bahdanau (nb). The given training set is divided into development set and training set and the performance is measured using the development set which is shown in Table 2. The model was trained for about "10,000 steps", 6 epoch_step with "128 units", batch size of "128", dropout of "0.2" and learning rate of "0.1".

4.3 Transformers

In this approach, the stances were detected using simple transformers. Simple transformers are the wrapper of transformers. Transformers are mechanism that utilizes the attention mechanisms without using recurrent units. Bi-directional Encoder Representation of Transformers (BERT) is used to detect stance with the multilingual model and base model for the development set whose performance is given in Table 3. Multilingual Bert model (Devlin et al., 2018) of hugging face Pytorch transformers (Wolf et al., 2019) has been used to detect stance in our approach which was submitted as Run-1.

5 Results

Table 2 shows the different models evaluated based on the development set. From the table, the model with two layers of gated recurrent unit and scaled luong attention mechanism seems to perform better.

Table 4 shows the performance of various teams in this task of detecting stance. Twelve teams have participated in which one team have submitted both constrained and unconstrained runs which is denoted by the suffix "_u" in the table. Remaining all runs are constrained runs which are done only using the data set provided.

Data Distribution	**against**	**favor**	**none**	**Total**
Training set	1028	587	515	2132
Testing set	742	196	172	1110
Total instances	1770	783	687	3242

Table 1: Data distribution

Model name	**Accuracy**
2l_nb_gru	37.0
2l_sl_gru	**38.0**
3l_nb_gnmt	33.7
3l_sl_gnmt	33.7
4l_nb_gru	36.4
4l_sl_gru	35.7
3l_sl_gnmt_residual	37.5
3l_nb_gnmt_residual	37.5

Table 2: Performance of various models

Model	**mcc**	**loss function**
Bert- Multilingual	**0.167**	**1.098**
Bert - Base	0.141	1.150

Table 3: Performance of BERT models

The performance metrics used are class-wise prediction of precision, recall, F1-score and average F1-score. The ranking is done using an average F1-score which is shown in 4. The best performance in constrained run is 0.6801 whereas our approach of transformers (SSN_NLP run 1) has an average F1 score of 0.4707 and encoder-decoder model (SSN_NLP run 2) has an average score of 0.4473.

6 Conclusion

Italian tweets about the Sardines movement have been utilized to detect the opinion of the author towards the target. Different approaches have been made to detect the stance in the tweets by many other teams. We detected the stance using encoder-decoder model and simple transformers of multilingual Bert model in which transformers performed better than the encoder-decoder model with a F1-average score of 0.4707. The performance can further be improved by utilizing the additional dataset to train the model better to detect the stance in the tweets.

Acknowledgments

We would like to express our gratefulness towards DST-SERB funding agent and HPC laboratory of SSN College Of Engineering for providing space and resources required for this experiment.

References

Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. *arXiv preprint arXiv:1606.05464*.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*. CEUR-WS.org.

Luís Borges, Bruno Martins, and Pável Calado. 2019. Combining similarity features and deep representation learning for stance detection in the context of checking fake news. *Journal of Data and Information Quality (JDIQ)*, 11(3):1–26.

Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. SardiStance@EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Costanza Conforti, Jakob Berndt, Mohammad Taher Pilehvar, Chryssi Giannitsarou, Flavio Toxvaerd, and Nigel Collier. 2020. Will-they-won't-they: A very large dataset for stance detection on twitter. *arXiv preprint arXiv:2005.00388*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Dilek Küçük and Fazli Can. 2019. A tweet dataset annotated for named entity recognition and stance detection. *arXiv preprint arXiv:1901.04787*.

Team	F-average
SSN_NLP run 1 (transformers)	0.4707
SSN_NLP run 2 (encoder-decoder model)	0.4473
Team A - 1_u	0.6853
Team A - 1_c	0.6801
Team A - 2_c	0.6793
Team B - 1	0.6621
Team A - 2_u	0.6606
Team C - 1	0.6473
Team D - 1	0.6257
Team C - 2	0.6171
Team E	0.6067
Team B - 1	0.6004
Team D - 2	0.5886
Team F	0.5784
Team G - 1	0.5773
Team H	0.5749
Team I - 1	0.5595
Team I - 1	0.5329
Team J	0.4989
Team G - 2	0.4705
Team K	0.3637

Table 4: Results

Mirko Lai, Alessandra Teresa Cignarella, Delia Irazú Hernández Farías, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. Multilingual stance detection in social media political debates. *Computer Speech & Language*, page 101075.

Nikita Lozhnikov, Leon Derczynski, and Manuel Mazzara. 2018. Stance prediction for russian: data and analysis. In *International Conference in Software Engineering for Defence Applications*, pages 176–186. Springer.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Minh-Thang Luong, Eugene Brevdo, and Rui Zhao. 2017. Neural machine translation (seq2seq) tutorial. *https://github.com/tensorflow/nmt*.

Benjamin Riedel, Isabelle Augenstein, Georgios P Spithourakis, and Sebastian Riedel. 2017. A simple but tough-to-beat baseline for the fake news challenge stance detection task. *arXiv preprint arXiv:1707.03264*.

Jannis Vamvas and Rico Sennrich. 2020. X-stance: A multilingual multi-target dataset for stance detection. *arXiv preprint arXiv:2003.08385*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rmi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Seunghyun Yoon, Kunwoo Park, Joongbo Shin, Hongjun Lim, Seungpil Won, Meeyoung Cha, and Kyomin Jung. 2019. Detecting incongruity between news headline and body text via a deep hierarchical encoder. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 791–800.

SSNCSE-NLP @ EVALITA2020: Textual and Contextual Stance Detection from Tweets Using Machine Learning Approach

B. Bharathi, J. Bhuvana, Nitin Nikamanth Appiah Balaji
Department of CSE,
Sri Sivasubramaniya Nadar College of Enginnering
Chennai, India
(bharathib, bhuvanaj)@ssn.edu.in
nitinnikamanth17099@cse.ssn.edu.in

Abstract

Opinions expressed via online social media platforms can be used to analyse the stand taken by the public about any event or topic. Recognizing the stand taken is the stance detection, in this paper an automatic stance detection approach is proposed that uses both deep learning based feature extraction and hand crafted feature extraction. BERT is used as a feature extraction scheme along with stylistic, structural, contextual and community based features extracted from tweets to build a machine learning based model. This work has used multilayer perceptron to detect the stances as favour, against and neutral tweets. The dataset used is provided by SardiStance task with tweets in Italian about Sardines movement. Several variants of models were built with different feature combinations and are compared against the baseline model provided by the task organisers. The models with BERT and the same combined with other contextual features proven to be the best performing models that outperform the baseline model performance.

1 Introduction

In today's era everything is in the digital form, people started spending more time online to stay connected. We get to learn about the events across the universe via online social media platforms namely, Facebook, Twitter, Instagram and so on. Sharing everyone's opinion becomes the norm of today's digital world either towards or against or neutral upon a particular topic or event. Expressing one's stand on any matter is refereed to as stance. Recognizing the stance, the stance detection is an interesting part of Natural Language processing that gains lots of traction nowadays. Demand of automatic detection of stance is found in variety of applications such as rumour detection, political standpoint of public, predictions over election results, advertising, opinion survey and so on.

This paper proposes a method that can be used for textual and contextual stance detection for the task hosted by sardistance@evalita2020. The overview of the sardistance@evalita2020 shared task is given in Cignarella et al. (2020). The proceedings of the task EVALITA can be found in Basile et al. (2020). BERT is used to perform the classification of stance from the tweets. Two models have been constructed, where the first one will classify the stance of a tweet into 3 categories as favour, against and neutral, the second model is built to classify the tweets into same number of classes as above by considering the additional contextual information namely number of retweets, number of followers, replies and quote's relations.

2 Survey of Existing Stance Detections

As per the authors in Küçük and Can (2020) stance detection is related to so many NLP problems namely, emotion recognition, irony detection, sentiment analysis, rumour classification etc. In specific the stance detection is closely related to sentimental analysis of the text, which is concerned about feelings such as tenderness, sadness, or nostalgia etc., whereas the stance detection needs a specific target on which the text is opined about. Stance detection is similar to perspective identification as well.

Stance detection can be done using learning based approaches via training and testing stages along with necessary pre-processing. These methods are categorized into machine learning based,

---

deep learning based and ensemble based approaches. Conventional machine learning approaches require the features to be extracted from the text after the pre-processing operations like normalization, tokenization etc. The deep learning approaches use the pre-trained models for classification of text using word embeddings like word2vec, GloVe, ELMo, CoVe, etc., as features (Sun et al., 2019). Bidirectional Encoder Representations from Transformers, BERT is one of the recent pre-trained models designed by Google (Devlin et al., 2018), which is a bidirectional transformer.

In Lai et al. (2020), stance detection was done in multiple languages using Stylistic, Structural, Affective and Contextual features and are fed to Linear Regression and SVM classifiers. The authors reported that the machine learning classifiers are more efficient to classify the stance in multilingual dataset than the deep learning counterparts.

In Aldayel (2019), a stance detection was made using the features such as on-topic content, network interactions, user's preferences, online network connection say Connection Networks. Extracted features are given to the standard machine learning classifier Support Vector Machine (SVM) with linear kernel to classify the stance of tweets into Atheism, Climate change is a real concern, Hillary Clinton, Feminist movement and Legalization of abortion (LA) classes. The authors observed that the textual features combined with the network features helped in detecting the stance more accurately.

A fine tuned BERT model was used for same side stance classification in Ollinger (2020). The authors have used both base and Large models for binary classification and reported that the Large model has outperformed the other one. They also have observed that longer input sequences are predicted well when compared with the smaller ones with a precision of 0.85.

Bi-directional Recurrent Neural Networks (RNNs) (Borges et al., 2019) along with other features were used for the fake news identification. Sentence encoder for the headlines and document encoder for the content of the news were used along with the common features extracted by combining the headlines and the body of the news. The four stances detected are Agree, Disagree, Unrelated and Discusses. The authors have reported that the pre-training the sentence encoder has enhance the model performance.

After pre-processing steps like stemming, stop word removal , normalization and Hashtag Pre-processing, the data are fed to five different models such as 1-D CNN-based sentence classification, Target-Specific Attention Neural Network [TAN], Recurrent Neural Network with Long Short Term Memory(LSTM), SVM-based SEN Model, Two-step SVM for reproducibility. Apart from the above the authors Ghosh et al. (2019) have also used pre-trained BERT (Large-Uncased) model for stance detection. Experiments were conducted using SemEval microblog dataset and text dataset about health-related articles and applied voting scheme for final predictions. The authors observed that the pre-processing enhanced the performance and also reported that the contextual feature will help to improve the stance detection further.

To detect the stance of tweets as one of favour, against and none a new CNN named CCNN-ASA, the Condensed CNN by Attention over Self- Attention has been designed by Mayfield (2019). Self-attention based convolution module to improve the representation of each and every word and attention-based condensation module for text condensation are embedded. They have experimented on SemEval-2016 challenge for supervised stance detection in Twitter with three usual stances The works reported in Zhou et al. (2019) ,Sen et al. (2018) , Wei and Mao (2019), Popat et al. (2019) are few of the other stance detection articles.

3 Proposed System

3.1 Dataset Description

The dataset hosted by SardiStance has tweets in Italian language about Sardines movement. The total tweets are about 3,242 instances out of which, training set has 2,132 and testing will have 1,110. The three stances are Against, Favor and Neutral about the Sardines movement with 1,028, 589, 515 instances respectively.

3.2 Model Construction

The models are built in Python and used GPU system with NVIDIA GTX1080 for running the experiments. The features are extracted from the Italian tweets about Sardines movement to construct the model and the same is evaluated for performance using the tweets meant for testing.

Feature engineering in our work includes both via the explicit features and also using a deep learning model that does the same. We have used the pre-trained deep learning model BERT to collect the features that provides a sequence of vectors of maximum size 512 which represents the features extracted. Along with that both structural and stylistic features are also extracted from the training instances of the Italian tweets.

Stylistic features considered in our proposed work are as follows: unigram is the representation in binary of unigrams; Char-grams is the representation in binary with 2 to 5 char n-grams; Structural features extracted from the Italian tweets are num-hashtag which will use the count of most frequently occurred hashtags of the tweet; punctuation marks considers 6 punctuation marks such as *!?.,;* and their frequencies as numerical values; Length feature will extract the number of characters, the number of words, the average length of the words in each tweet;

Community based features are also used as discriminating features in our work that exhibits the relationship among the tweets, comments such as network quote community, network reply community, network retweet community, network friend community. These features are vectors of numerical attributes that represent the number of retweets, retweets with comments, number of friends, number of followers, count of lists, created at information and number of emojis in the twitter bio.

For the textual stance detection, features such as BERT, unigram, unigram-hashtag, char-grams, num-hashtag, punctuation marks and length are extracted from the training instances. These features are given to Multilayer Perceptron (MLP) with 128 hidden layers with 512 nodes each. The training uses K-fold cross validation to fine tune the model parameters with K = 5 folds.

For the contextual stance detection, along with the features mentioned for the textual SD, additional features of the tweet such as network quote community, network reply community, network retweet community, network friend community, user info bio, tweet info retweet, tweet info create at were also extracted from the training instances and all are fed to MLP classifier with 512 nodes in each of 128 hidden layers. The second model also undergoes 5 fold cross validation to avoid overfitting and selection bias problems.

14 different models with individual textual and contextual features have been built for stance detection. Along with that, to explore the combined feature space each of the above mentioned features have combined in two and three to built models. Totally 89 models were built to investigate the performance each of the feature is combined with other one and used for training the MLP classifier. And 147 variants of classifiers were constructed by combining three features together.

Both the classifiers are iterated for 1000 times with *relu* as its activation function in their hidden layers and *adam* as the optimization function which is a variant of stochastic gradient descent.

4 Results and Discussion

Models are built after 5-fold cross validation and with different combinations of both deep learning based BERT and hand crafted structural, contextual features together to investigate the performance of stance detection system. The few of the best cross validation results are shown in Table 1. The validation results show that the BERT works well either it is used alone for feature extraction or when combined with other features. In particular, when we analyze the validation results we found that the community based features contribute more towards the stance detection either independently or when combined with other textual features.

The models constructed for textual and contextual stance detection are tested with the instances of the test set. Two runs were submitted for each of the two tasks namely the textual stance and contextual stance detection under the name SSNCSE-NLP. Performance measures precision (P), recall (R), and F-score (F) for the three stances such as tweet towards the Sardines movement, against the movement and neutral ones are computed.

A baseline model was built by the task organizers of Sardistance using the conventional machine learning algorithm SVM with the help of uni-gram feature and has been used to compare the performance of our models.

Best results obtained are reported in Table 2, with macro average of F1 measure along with the scores for F1 for against tweets, for favour and for neutral tweets classification. The baseline that was used by the task organisers was the SVM with linear kernel obtained the F1 average as 0.5784. The Run 1 which has been built on the model using features extracted by the pre-trained BERT has shown a F1 score average of 0.6067 that is around 3%

Models with listed features	**F1 score**
BERT	0.5763
Unigram	0.5509
chargrams	0.5734
network quote community	0.5419
bert + unigram	0.5897
bert + unigramhashtag	0.5583
bert + chargrams	0.5721
bert + numhashtag	0.5773
bert + puntuactionmarks	0.5501
bert + length	0.5226
bert + network quote community	0.6212
bert + network reply community	0.5993
bert + network retweet community	0.6086
bert + network friend community	0.6482
bert + user info bio	0.5748
bert + tweet info retweet	0.6086
bert + tweet info create at	0.5431
unigram + chargrams	0.5834
unigram + network quote community	0.5965
bert + unigram+ length	0.5813
bert + unigram+ network reply community	0.6048
bert + chargrams + network quote community	0.5853
bert + chargrams+ user info bio	0.5834
bert + network quote community + network friend community	0.6436

Table 1: Results after 5-fold cross validation

Task A - Textual Stance Detection										
Run	**f-avg**	**prec_a**	**prec_f**	**prec_n**	**recall_a**	**recall_f**	**recall_n**	**f_a**	**f_f**	**f_n**
Baseline	0.5784	0.7549	0.3975	0.2589	0.6806	0.4949	0.2965	0.7158	0.4409	0.2764
1*	0.6067	0.7506	0.4245	0.2679	0.7951	0.4592	0.1744	0.7723	0.4412	0.2113
2*	0.5749	0.7798	0.3664	0.3196	0.6873	0.4898	0.3605	0.7307	0.4192	0.3388
Task B - Contextual Stance Detection										
Baseline	0.6284	0.7845	0.4506	0.3054	0.7507	0.5357	0.2965	0.7672	0.4895	0.3009
1*	0.6582	0.8321	0.4715	0.3508	0.7547	0.5918	0.3895	0.7915	0.5249	0.3691
2*	0.6556	0.8419	0.4574	0.3660	0.7466	0.6020	0.4128	0.7914	0.5198	0.3880

Table 2: Detection Results of SardiStance tasks using test data (* - Run 1 & 2 of proposed system)

more than the baseline model as shown in Table 2. The Run 2 has obtained a performance near to the baseline that has used the char n-gram as the feature extracted.

Our model for Run 1 has outperformed the baseline model in terms of precision of favour and neutral tweets also shown a 11% increase in recall of against tweets over the baseline model. This can be interpreted that the most of the testing instances are identified as relevant tweet against the Sardines movement.

For the second task on contextual stance detection, our models for Run 1 and 2 have performed better than the baseline model for the same, whose F1 average is given as 0.6284. The Run 1 for this task has used BERT, numhashtag, network_friend_community features whereas the run 2 has been built on BERT, network_quote_community, network_friend_community features.

This can be inferred that the additional information about the Sardine tweets such as the community based contextual features have contributed towards the classification of the tweets. Metadata about the tweets have served in discriminating the stance better than the textual information of the tweets themselves.

5 Conclusion

In this paper, we presented the suitable models for stance detection in Italian tweets about Sardine movement. The three stances considered for this work are in favour of the movement, against and neutral. Multilayer perceptron is the classifier used for classification of stance of tweets. The deep learning pre-trained model BERT has been used to extract the features from the tweets along with several stylistic, contextual and community based features namely, The features are extracted Unigram , Char-grams , num-hashtag , Length, network quote community, network reply community, network retweet community, network friend community, user info bio, tweet info retweet, tweet info create at are few of the attributes that are extracted to detect the stance. The Models are trained using the dataset provided by SardiStance task for textual and contextual stance detections. Three of models have outperformed when compared against the baseline model that have used the SVM for stance detection. A maximum of 5% increase is found in precision of in favour tweets over the baseline model for the same. In order to explore all feature spaces, in this work the structural, stylistic, contextual features are combined in different permutations and validated for their performance. The best performing models are found to be using BERT and char n-gram for textual stance and combinations such as BERT along with numhashtag, network friend community and BERT with network quote community, network friend community features for contextual stance detection.

We have observed that most contributing features along with the textual features are community based features of the tweets, those meta data serve well in discriminating the stance better. More analysis on these features and their combination can help in improving the performance of automatic stance detection system. Since the tweet exhibits the nature of stance a person takes on any event or topic also lead tot he violation of that person's privacy, which also needs to look at.

Acknowledgments

We would like to thank the SSN management for supporting the work by sponsoring the GPU systems for the research work.

References

Abeer Aldayel and Walid Magdy. 2019. Your stance is exposed! analysing possible factors for stance detection on social media. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–20.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Luís Borges, Bruno Martins, and Pável Calado. 2019. Combining similarity features and deep representation learning for stance detection in the context of checking fake news. *Journal of Data and Information Quality (JDIQ)*, 11(3):1–26.

Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. SardiStance@EVALITA2020: Overview of the Task on Stance Detection in Italian Tweets. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and*

Speech Tools for Italian (EVALITA 2020). CEUR-WS.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Shalmoli Ghosh, Prajwal Singhania, Siddharth Singh, Koustav Rudra, and Saptarshi Ghosh. 2019. Stance detection in web and social media: a comparative study. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 75–87. Springer.

Dilek Küçük and Fazli Can. 2020. Stance detection: A survey. *ACM Computing Surveys (CSUR)*, 53(1):1–37.

Mirko Lai, Alessandra Teresa Cignarella, Delia Irazú Hernández Farías, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. Multilingual stance detection in social media political debates. *Computer Speech & Language*, page 101075.

Elijah Mayfield and Alan W Black. 2019. Stance classification, outcome prediction, and impact assessment: Nlp tasks for studying group decision-making. In *Proceedings of the Third Workshop on Natural Language Processing and Computational Social Science*, pages 65–77.

Stefan Ollinger, Lorik Dumani, Premtim Sahitaj, Ralph Bergmann, and Ralf Schenkel. 2020. Same side stance classification task: Facilitating argument stance classification by fine-tuning a bert model. *arXiv preprint arXiv:2004.11163*.

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2019. Stancy: Stance classification based on consistency cues. *arXiv preprint arXiv:1910.06048*.

Anirban Sen, Manjira Sinha, Sandya Mannarswamy, and Shourya Roy. 2018. Stance classification of multi-perspective consumer health information. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, pages 273–281.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Penghui Wei and Wenji Mao. 2019. Modeling transferable topics for cross-target stance detection. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1173–1176.

Shengping Zhou, Junjie Lin, Lianzhi Tan, and Xin Liu. 2019. Condensed convolution neural network by attention over self-attention for stance detection in twitter. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

TRACK
“CREATIVITY AND STYLE”

CHANGE-IT: Style Transfer

CHANGE-IT @ EVALITA 2020: Change Headlines, Adapt News, GEnerate

Lorenzo De Mattei
University of Pisa
CLCG, University of Groningen
ItaliaNLP Lab, ILC-CNR
Pisa, Italy
`lorenzo.demattei@di.unipi.it`

Michele Cafagna
Aptus.AI, Pisa, Italy
University of Malta, Malta
`michele@aptus.ai`

Felice Dell'Orletta
ItaliaNLP Lab, ILC-CNR
Pisa, Italy
`felice.dellorletta@ilc.cnr.it`

Malvina Nissim
CLCG, University of Groningen
The Netherlands
`m.nissim@rug.nl`

Albert Gatt
University of Malta
Malta
`albert.gatt@um.edu.mt`

Abstract

We propose a generation task for Italian – more specifically, a style transfer task for headlines of Italian newspapers. This is the first shared task on generation included in the EVALITA evaluation framework. Indeed, one of the reasons to have this task is to stimulate more research on generation within the Italian community. With this aim in mind, we release to the participating teams not only training data, but also a baseline sequence to sequence model that performs the task in order to help everyone get started, even when not accustomed to Natural Language Generation (NLG) approaches. Contextually, we explore the complex issue of automatic evaluation of generated text, which is receiving particular attention in the NLG community.

1 Task and Motivation

We propose a generation task for Italian in the context of the EVALITA 2020 campaign (Basile et al., 2020). More specifically, we design a *style transfer task for headlines of Italian newspapers*.

We believe it is the first time that a shared task on generation is offered in the context of EVALITA. Indeed, one of the reasons to have this task is to stimulate more research on generation within the Italian community. With this goal in mind, we release to the potential participating teams not only training data, but also a baseline sequence to sequence model that performs the task in order to help everyone get started, even when not accustomed to generation models, yet. This baseline model casts the style transfer problem as an extreme summarisation task, just showing how versatile the problem is in terms of possible approaches. Contextually, this task will help to further explore the complex issue of evaluation of generated text, which is receiving particular attention in the Natural Language Generation international community (Gatt and Krahmer, 2018; van der Lee et al., 2019).

Task The task is cast as a "headline translation" problem, and it is as follows. Given a collection of headlines from two Italian newspapers at opposite ends of the political spectrum, call them G and R, change all G-headlines to headlines into style R, and all R-headlines to headlines in style G.

In the context of this task we need to take care of two crucial aspects: data and evaluation. Details on data are provided in Section 2, and on evaluation in Section 3.

2 Data

We have collected news coming from two of the most important Italian newspapers situated at opposite ends of the political spectrum, namely *la Repubblica* (left) and *Il Giornale* (right), totalling approximately 152,000 article-headline pairs, with the two newspapers equally represented. Although the task only concerns headline change, the teams will receive both the headlines as well as their respective full articles.

Leveraging on an alignment procedure described below (see Cafagna et al. (2019) for fur-

---

cosine score	newspaper	alignment
0.96 (strict)	rep	Estroverso o nevrotico? Lo dice la foto scelta per il profilo social *en:[Extrovert or neurotic? The photo chosen for the social profile says so]*
	gio	L'immagine del profilo usata nei social network rivela la nostra personalità *en:[The profile picture used in social networks reveals our personality]*
0.5 (strict)	rep	Egitto, governo si dimette a sorpresa *en:[Egypt, government resigns surprisingly]*
	gio	Egitto, il governo si dimette *en:[Egypt, government resigns]*
0.185 (loose)	rep	Elezioni presidenziali Francia, la Chiesa non si schiera né per Macron né per Le Pen *en:[Presidential elections France, the Church does not take sides either for Macron or for Le Pen]*
	gio	Il primo voto con l'incubo Isis ma il terrorismo esce sconfitto *en:[The first vote with the Isis nightmare but terrorism comes out defeated]*

Table 1: Example of alignments between *La Repubblica* and *Il Giornale*, extracted with different similarity scores. The second and the third examples would fall into the strict and the loose sets, respectively, according to the thresholds used to split the alignments. The first two headline pairs are well aligned, while the third pair has a very loose alignment.

ther details), we account for potential topic biases in the two newspapers, and we split the data set into strongly, weakly and not-aligned news. This information is useful in the creation of the datasets that we need to train our three evaluation classifiers (see Section 3). Additionally, it could help to better disentangle newspaper-specific style.

Alignment We compute the tf-idf vectors of all the articles of both newspapers and create subsets of relevant news filtering by date, i.e. considering only news which were published in approximately the same, short, temporal range for the two sources. On the tf-idf vectors we then compute cosine similarities for all news in the resulting subset, rank them, and retain only the alignments that are above a certain threshold. The threshold is chosen taking into consideration a trade-off between number of documents and quality of alignment. We choose two different thresholds: one is stricter (≥ 0.5) and we use it to select best alignments (*strict alignments*); the other one is looser (≥ 0.185, and < 0.5) — we define these latter as *weak alignments*. We consider the rest as basically not aligned.

Data splits We split the dataset into *strongly aligned news*, which are selected using the stricter threshold ($\sim$20K aligned pairs, set `A*` in Figure 1a), and *weakly aligned and non-aligned news* ($\sim$100K article-headline pairs equally distributed among the two newspapers, set `R` in Figure 1a).

The strictly aligned data is further split as shown in Figure 1a; this yields a total of four sets over the whole dataset (`A1`, `A2`, `A3`, and `R`). `A2` is left aside and used as test set for the final style transfer task. The remaining three sets are used for training the evaluation classifiers and the system for the target task. These are shown in Figure 1b. Note that all sets also always contain the headlines' respective full articles, though these are not necessarily used.

Format The data is distributed in the form of *one CSV file* with the following fields:

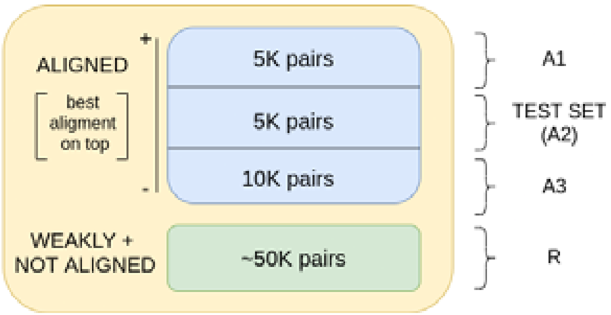
```
id, headline, article, label [R,G]
```

3 Evaluation

Human evaluation is generally viewed as the most desirable method to assess generated text (Novikova et al., 2018; van der Lee et al., 2019). However, human evaluation is not always a viable option, due to resources, but also due to the fact that humans might not be capable of reliably assessing the task at hand. Related to the current challenge, De Mattei et al. (2020a) have shown that people find it difficult to identify subtle stylistic differences between texts.

Automatic, reliable metrics should therefore also be sought (Novikova et al., 2017). For our task, we propose a fully automatic strategy based on a series of classifiers to assess style strength and content preservation. For style, we train a single classifier (*main*). For content, we train two classifiers that perform two 'sanity checks': one ensures that the two headlines (original and transformed) are still compatible (*HH classifier*); the other ensures that the headline is still compatible with the original article (*AH classifier*). See also Figure 1b.

In what follows we describe these classifiers in



(a) Overall data splits

EVALUATION		
train & test	main	R+A3+A1
	HH	A1 + random pairs
	AH	R+A3+A1
TASK		
train		R+A3
test		A2

(b) Training/test sets

Figure 1: Data splits and their use in the different training sets

more detail. When discussing baseline results, we will show how the contribution of each classifier is crucial towards a comprehensive evaluation.

Main classifier The main classifier uses a pre-trained BERT (Devlin et al., 2019) encoder with a linear classifier on top fine-tuned with a batch size of 256 and sequences truncated at 32 tokens for 6 epochs with learning rate 1e-05. Given a headline, this classifier can distinguish the two sources with an f-score of approximately 80% (see Table 2). Since style transfer is deemed successful if the original style is lost in favour of the target style, we use this classifier to assess how many times a style transfer system manages to reverse the main classifier's decisions.

HH classifier This classifier checks compatibility between the original and the generated headline. We use the same architecture as for the main classifier with a slightly different configuration: max. sequence length of 64 tokens, batch size of 128 for 2 epochs (early-stopped), with learning rate 1e-05. Being trained on strictly aligned data as positive instances (`A1`), with a corresponding amount of random pairs as negative instances, it should learn whether two headlines describe the same content or not. Performance on gold data is .96 (Table 2).

AH classifier This classifier performs yet another content-related check. It takes a headline and its corresponding article, and tells whether the headline is appropriate for the article. The classifier is trained on article-headline pairs from both the strongly aligned and the weakly and non-aligned instances (`R+A3+A1`, Figure 1b). At test time, the generated headline is checked for compatibility against the source article. We use the same base model as for the main and HH classifiers with batch size of 8, same learning rate and 6 epochs. Performance on gold data is >.97 (Table 2).

		prec	**rec**	**f-score**
main	**rep**	0.77	0.83	0.80
	gio	0.84	0.78	0.81
HH	**match**	0.98	0.95	0.96
	no match	0.95	0.98	0.96
AH	**match**	0.96	0.99	0.98
	no match	0.99	0.96	0.97

Table 2: Performance of the evaluation classifiers on gold data.

Overall compliancy We calculate a compliancy score which assesses the proportion of times the following three outcomes are successful (i) the *HH classifier* predicts 'match'; (ii) the *AH classifier* predicts 'match'; (iii) the *main classifier*'s decision is *reversed*. As upperbound, we find the compatibility score for gold at 74.3% for transfer from *La Repubblica* to *Il Giornale* (*rep2gio*), and 78.1% for the opposite direction (*gio2rep*).

4 Baseline System

We developed a baseline system using a summarisation approach, where headlines are viewed as an extreme case of summarisation and generated from the article. We exploit article-headline generators trained on opposite sources to do the transfer, as done in (De Mattei et al., 2020b). The advantage of this approach is that in principle it doesn't require parallel data for training.

Specifically, we use two pointer-generator networks (See et al., 2017), which include a *pointing mechanism* able to copy words from the

Il Giornale → La Repubblica		
E in Sicilia è scattata l'allerta rossa *[en: And in Sicily it's now red alert]*	⟶	Migranti, la Protezione civile continua dimenticata *[en: Migrants, the Civil Protection Department goes on forgotten]*
Nozze gay, toghe contro i sindaci: "Le trascrizioni sono illegittime" *[en: Gay marriages, gowns against mayors: "Transcriptions are not valid"]*	⟶	Il Consiglio di Stato boccia le nozze gay all'estero *[en: The State Council rejects gay marriages abroad]*
La Repubblica → Il Giornale		
Castelnuovo, lo sdegno di cittadini e associazioni: "Attacco all'integrazione che funziona" *[en: Castelnuovo, the indignation of citizens and associations: "Attack to the integration that works"]*	⟶	I migranti non sono più rifugiati *[en: Migrants are not refugees anymore]*
Da Renzi a Di Maio, ecco il reddito dichiarato dai politici italiani. Fedeli il ministro con l'imponibile più alto *[en: From Renzi to Di Maio: here it's the income declared by the Italian politicians. Fedeli is the minister with the highest taxable income]*	⟶	Grillo e Giggino italiani conquistano l'elenco dei redditi italiani *[en: Grillo and Giggino Italians conquer the list of Italian incomes]*

Table 3: Examples of headlines generated by the baseline system.

source as well as pick them from a fixed vocabulary, thereby allowing better handling of out-of-vocabulary words.

One model is trained on the *la Repubblica* portion of the training set, the other on *Il Giornale*. In a style transfer setting we use these models as follows: Given a headline from *Il Giornale*, for example, the model trained on *la Repubblica* can be run over the corresponding article from *Il Giornale* to generate a headline in the style of *la Repubblica*, and vice versa.

The results of the baseline system, measured as performance of each classifier as well as the overall compliancy score, are reported in Table 4.

	HH	**AH**	**Main**	**compl.**
rep2gio	.649	.876	.799	.449
gio2rep	.639	.871	.435	.240
avg	**.644**	**.874**	.616	.345

Table 4: Baseline performance on test data.

5 Outlook

This shared task proposal was intended to stimulate research in NLG, with a specific focus on style transfer and automatic evaluation, in the Italian community. Over ten teams expressed their interest in participating in the shared task officially, but eventually there were no submitted runs. We do hope that the materials developed in the context of this challenge will nevertheless be of use to promote research in a field that is still under-researched in the Italian NLP landscape. All materials are available: `https://github.com/michelecafagna26/CHANGE-IT`.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Michele Cafagna, Lorenzo De Mattei, and Malvina Nissim. 2019. Embeddings shifts as proxies for different word use in italian newspapers. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019), Bari, Italy*.

Lorenzo De Mattei, Michele Cafagna, Felice Dell'Orletta, and Malvina Nissim. 2020a. Invisible to People but not to Machines: Evaluation of Style-aware Headline Generation in Absence of Reliable Human Judgment. In *Proceedings of the Twelfth International Conference on Language Resources and Evaluation (LREC 2020)*, Marseille, France, May. European Language Resources Association (ELRA).

Lorenzo De Mattei, Michele Cafagna, Felice Dell'Orletta, and Malvina Nissim. 2020b. Invisible to People but not to Machines: Evaluation of Style-aware Headline Generation in Absence of Reliable Human Judgment. In *Proceedings of the Twelfth International Conference on Language Resources and Evaluation (LREC 2020)*, Marseille, France, May. European Language Resources Association (ELRA).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark, September. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2018. RankME: Reliable human ratings for natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 72–78, New Orleans, Louisiana, June. Association for Computational Linguistics.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, Sander Wubben, and Emiel Krahmer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 355–368, Tokyo, Japan, October–November. Association for Computational Linguistics.

TAG-it: Topic, Age and Gender Prediction

TAG-it @ EVALITA2020: Overview of the Topic, Age, and Gender Prediction Task for Italian

Andrea Cimino
ItaliaNLP Lab, ILC-CNR
Pisa, Italy
andrea.cimino@ilc.cnr.it

Felice Dell'Orletta
ItaliaNLP Lab, ILC-CNR
Pisa, Italy
felice.dellorletta@ilc.cnr.it

Malvina Nissim
Faculty of Arts - CLCG
University of Groningen, The Netherlands
m.nissim@rug.nl

Abstract

The Topic, Age, and Gender (TAG-it) prediction task in Italian was organised in the context of EVALITA 2020, using forum posts as textual evidence for profiling their authors. The task was articulated in two separate subtasks: one where all three dimensions (topic, gender, age) were to be predicted at once; the other where training and test sets were drawn from different forum topics and gender or age had to be predicted separately. Teams tackled the problems both with classical machine learning methods as well as neural models. Using the training-data to fine-tuning a BERT-based monolingual model for Italian proved eventually as the most successful strategy in both subtasks. We observe that topic and gender are easier to predict than age. The higher results for gender obtained in this shared task with respect to a comparable challenge at EVALITA 2018 might be due to the larger evidence per author provided at this edition, as well as to the availability of pre-trained large models for fine-tuning, which have shown improvement on very many NLP tasks.

1 Introduction

Author profiling is the task of automatically discovering latent user attributes from text, among which gender, age, and personality (Rao et al., 2010; Burger et al., 2011; Schwartz et al., 2013; Bamman et al., 2014; Flekova et al., 2016; Basile et al., 2017).

Past work in Natural Language Processing has contributed to advancing this task especially through the creation of resources, also in languages other than English (Verhoeven et al., 2016; Rangel et al., 2017, e.g.,), for training supervised models. Across the years, especially thanks to the organisation of shared tasks in the context of the PAN Labs, it has become evident that models that exploit lexical information, mostly in the form of word and character n-grams, make successful predictions (Rangel et al., 2017; Basile et al., 2018; Daelemans et al., 2019).

However, cross-genre experiments (Rangel et al., 2016; Busger op Vollenbroek et al., 2016; Medvedeva et al., 2017; Dell'Orletta and Nissim, 2018) have revealed that most successful approaches, exactly because they are based on lexical clues, tend to model *what* rather than *how* people write, capturing topic instead of style. As a consequence, they lack portability to new genres and more in general just new datasets.

The present work aims at shedding some more light in this direction, and at the same time increase resources and visibility for author profiling in Italian. We propose a shared task in the context of EVALITA 2020 (Basile et al., 2020) that can be broadly conceived as stemming from a previous challenge on profiling in Italian, i.e., GxG, a cross-genre gender prediction task. The new task is TAG-it (Topic, Age, and Gender prediction in Italian). With TAG-it, we introduce three main modifications with respect to GxG. One is that age is added to gender in the author profiling task. Another one is that, in one of the tasks, we conflate author and text profiling, requiring systems to simultaneously predict author traits and topic. Lastly, we restrict the task to in-genre modelling,

---

pan.webis.de

but we explicitly control for topic through two specific subtasks.

2 Task

TAG-it (Topic, Age and Gender prediction for Italian) is a profiling task for Italian. This can be broadly seen as a follow-up of the GxG (Dell'Orletta and Nissim, 2018) task organised in the context of EVALITA 2018 (Caselli et al., 2018), though with some differences.

GxG was concerned with gender prediction only, and had two distinctive traits: (i) models were trained and tested cross-genre, and (ii) evidence per author was for some genres (Twitter and YouTube) extremely limited (one tweet or one comment). The combination of these two aspects yielded scores that were comparatively lower than those observed in other campaigns, and for other languages. A core reason for the cross-genre setting was to remove as much as possible genre-specific traits, but also topic-related features. The two would basically coincide in most n-gram-based models, which are standard for this task.

In TAG-it, the task is revised addressing these two aspects, for a better disentanglement of the dimensions. First, only a single genre is considered (forum posts). Second, longer texts are used, which should provide better evidence than single tweets, and are more coherent than just the concatenation of more tweets. Third, "topic control" is introduced in order to assess the impact on performance of the interaction of topic and author's traits, in a more direct way than in GxG (where it was done indirectly via cross-genre prediction).

Data was collected accordingly, including information regarding topic and two profiling dimensions: gender and age. The interesting aspect of this is that we mix text profiling and author profiling, with tasks and analysis that treat their modelling both at once as well as separately. In practice, we devise and propose two tasks.

Task 1: Predict all dimensions at once Given a collection of texts (forum posts) the gender and the age of the author must be predicted, together with the topic the posts are about. The task is cast as a multi-label classification task, with gender represented as F (female) or M (male), age as five different age bins, as it has been done in past profiling tasks involving age (Rangel et al., 2015, e.g.,), and topic as 14 class values.

Task 2: Predict age and gender with topic control For posts coming from a small selection of topics not represented in the training data, systems have to predict either gender (Task 2a) or age (Task 2b).

For both tasks, participants were also free to use external resources as they wish, provided the cross-topic settings would be preserved, and that everything used would be described in detail.

3 Data

3.1 Collection

In order to generate the data for the tasks, we exploited a corpus collected by Maslennikova et al. (2019). This corpus consists of 2.5 million posts scraped from the ForumFree platform. The posts are written by 7.023 different users in 162 different forums. Information about the authors' gender and age is available.

In order to have enough data for the topic classification task, we decided to aggregate data from several forums into a single topic. For example, data from the forums *500x* and *a1audiclub* where manually classified into the *AUTO-MOTO* topic, while the forums *bellicapelli* and *farmacieonlinesicure* in the *MEDICINE-AESTHETICS* topic. At the end of the aggregation process, we obtained 31 different topics. The selection of the topics that we use in TAG-it is shown in Table 1.

For age classification, we bin age into 5 age groups: (0,19), (20, 29), (30, 39), (40, 49) and (50-100). In addition, we performed a final selection of users in order to have sufficient evidence per author. More precisely, we selected only the users that wrote at least 500 tokens across their posts. The first 500 tokens of their posts were used as textual data while the other posts from the same users were discarded. At the end of this process, we obtained posts belonging to 2,458 unique users. Table 1 reports some corpus statistics, already arranged according to the experimental splits that we used in the different tasks (see Section 3.2).

3.2 Training and test sets

The data obtained from the process described in the previous subsection was used to generate the training and test data. The training data is the same for Task 1 and Task 2. It contains a variety of topics, and we aimed at a good label distribution for both gender and age, though the forum

---

`https://www.forumfree.it/?wiki=About`

Topic	M	F	0-19	20-29	30-39	40-49	50-100
Training data for all tasks							
ANIME	133	114	77	112	33	19	6
MEDICINE-AESTHETICS	16	13	0	2	13	9	5
AUTO-MOTO	221	5	5	41	42	67	71
SPORTS	285	15	19	102	74	62	43
SMOKE	79	0	0	9	25	22	23
METAL-DETECTING	77	1	5	11	15	28	19
CELEBRITIES	23	26	1	25	8	7	8
ENTERTAINMENT	28	4	5	16	8	0	3
TECHNOLOGY	5	1	3	1	0	1	1
NATURE	24	12	7	9	9	4	7
BIKES	25	2	2	2	3	7	13
Test data for Task 1							
ANIME	46	51	27	43	13	8	6
MEDICINE-AESTHETICS	7	9	1	4	6	3	2
AUTO-MOTO	73	3	1	13	21	18	23
SPORTS	92	11	7	37	23	18	18
SMOKE	29	1	0	6	9	8	7
METAL-DETECTING	25	1	0	2	6	8	10
CELEBRITIES	7	15	0	8	5	2	7
ENTERTAINMENT	9	0	1	6	2	0	0
TECHNOLOGY	9	0	1	5	3	0	0
NATURE	7	4	1	3	6	1	0
BIKES	11	1	0	4	1	3	4
Test data for Task 2a							
GAMES	274	24	47	128	41	44	38
ROLE-GAMES	70	44	29	61	10	4	10
Test data for Task 2b							
CLOCKS	386	1	3	41	83	168	92
GAMES	274	24	47	128	41	44	38
ROLE-GAMES	70	44	29	61	10	4	10

Table 1: Number of unique users (shown by gender and age) for each topic in the training and test sets of both tasks.

data is overall rather unbalanced for these two dimensions. In the selection of test data, we had to differentiate between the two task since for Task 1 test topics should correspond to those in training, while they should differ for Task 2.

For Task 1, each topic was split into 70% for training and 30% for test. For Task 2, we picked posts from topics not present in the training data, and more specifically used the forums CLOCKS, GAMES, and ROLE-GAMES for Task 2a, and only GAMES and ROLE-GAMES for Task 2b in order to ensure more balanced data. Table 2 shows the size of the datasets in terms of tokens.

The data was distributed as simil-XML. The format can be seen in Figure 1. The test data was released blind to the participants who were given a week to return their prediction to the organisers.

4 Evaluation

System evaluation was performed using both standard (accuracy, precision, recall, and f-score), as well as ad hoc measures.

DATASET	M	F	0-19	20-29	30-39	40-49	50-100
Training for all Tasks	533,195	114,723	74,349	199,902	132,518	132,130	109,019
Test Task1	180,646	70,407	24,259	77,869	53,955	40,196	54,774
Test Task2a	225,416	43,318	47,659	135,347	29,337	27,623	28,768
Test Task2b	438,759	43,834	50,583	158,704	76,986	117,721	78,599

Table 2: Number of tokens for gender and age contained in training and test data.

Team Name	Research Group	# Runs
UOBIT	Computer Science Department, Universidad de Oriente, Santiago de Cuba, Cuba	9
UO4to	Computer Science Department, Universidad de Oriente, Santiago de Cuba, Cuba	2
ItaliaNLP	Aptus.AI, Computer Science Department, ItaliaNLP Lab (ILC-CNR), Pisa, Italy	9

Table 3: Participants to the EVALITA 2020 TAG-it Task with number of runs.

```
<user id="2" topic="BIKES" age="40-49" gender="M">

<post>
perfetto direi veramente ingegnoso
</post>

<post>
Ma come hai carpito queste notizie certe?
Hai fermato le signore ad un posto di blocco
spacciandoti per agente di polizia?
</post>

<post>
A chent'annos Alessandro.
</post>

[...]

</user>
```

Figure 1: Sample of a training instance.

For Task 1, the performance of each system was evaluated according to two different measures, which yielded two different rankings. In the first ranking we use a partial scoring scheme (Metric 1), which assigns 1/3 to each dimension correctly predicted. Therefore, if no dimension is predicted correctly, the system is scored with 0, if one dimension is predicted correctly the score is 1/3, if two dimensions are correct the score is 2/3, and if all of age, gender, and topic are correctly assigned, then the score for the given instance is 1.

In the second ranking (Metric 2), 1 point is assigned if all the dimensions are predicted correctly simultaneously, 0 otherwise. This corresponds to the number of '1' points assigned in Metric 1.

For each ranking, the final score is the sum of the points achieved by the system across all the test instances, normalized by the total number of instances in the test set.

For Task 2, the standard micro-average f-score was be used as scoring function. For carrying out further analysis, we also report macro-f.

Baselines For all tasks, we introduced two baselines. One is a data-based majority baseline, which assign the most frequent label in the training data to all test instances. The other one is an SVM-based model (*SVM baseline* hereafter), as SVMs are known to perform well in profiling tasks (Basile et al., 2018; Daelemans et al., 2019).

This classifier is implemented using scikit-learn's `LinearSVC` (Pedregosa et al., 2011) with default parameters, using as features up to 5-grams of characters and up to 3-grams of words (frequency counts).

5 Participants

Following a call for interest, 24 teams registered for the task and thus obtained the training data. Eventually, three teams submitted their predictions, for a total of 20 runs. Three different runs were allowed per task. A summary of participants is provided in Table 3.

Overall, participants experimented with more classical machine learning approaches as well as with neural networks, with some of them employing language model based neural networks models such as multilingual BERT (Devlin et al., 2019) and UmBERTo. While the UO4to team (Artigas Herold and Castro Castro, 2020) proposed a classical feature engineered ensamble approach, UOBIT (Labadie et al., 2020) and Ital-

---

`https://github.com/musixmatchresearch/umberto` iaNLP (Occhipinti et al., 2020) experimented different deep learning techniques. UOBIT proposed a novel approach based on a combination of different learning components, aimed at capturing different level of information, while ItaliaNLP experimented with both SVM and Single and Multi task learning settings using a state-of-the-art language model specifically tailored for the Italian language.

Even if allowed, the use of external resources was not explored most probably due to great performances already provided by the latest deep learning language models w.r.t featured engineered models.

The following paragraphs provide a summary of each team's approach for ease of reference.

UOBIT tested a deep learning architecture with 4 components aimed at capturing different information from documents. More precisely, they extracted information from the layers of a finedtuned multilingual version of BERT (T), used information from a LSTM trained with FastText input vectors (RNN-W), they added raw features for stylistic feature extraction (STY) and finally they extracted information from a sentence encoder (RNN-S). The information from all the four components is finally concatenated and fed into a dense layer.

UO4to participated to Task 1 with two different ensemble classifiers, using Random Forest, Nearest Centroid and OneVsOneClassfier learning algorithms provided by the scikit-learn library (Pedregosa et al., 2011). They used n-grams of characters using term frequency or TF-IDF depending on the used configuration.

ItaliaNLP tested three different systems. The first one is based on three different SVM models (one for each dimension to be predicted), using character n-grams, word n-grams, Part-Of-Speech n-grams and *bleached* (van der Goot et al., 2018) tokens. The second one is based on three different BERT-based classifier using UmBERTo as a pre-trained language model, modelling each task separately. Finally, they tested a multi–task learning approach to jointly learn the three tasks, again using UmBERTo as a language model.

6 Results and Analysis

Tables 4 and 6 report the final results on the test sets of the EVALITA 2020 TAG-it Task 1 and Task 2 respectively, using the official evaluation metrics. For all tasks, the ItaliaNLP system achieves the best scores. Before delving into the specifics of each task, and into a deeper analysis of the results, we want to make a general observation regarding approaches. SVMs have longed proved to be successful at profiling, and this trend emerged also at the last edition of the PAN shared task on author profiling (Daelemans et al., 2019). In our tasks, we also observe that the SVM baseline that we have trained for comparison is competitive. However, the submitted model that achieves best results is neural.

Team Name-MODEL	**Metric 1**	**Metric 2**
Majority baseline	0.445	0.083
SVM baseline	0.674	0.248
UOBIT-(RNN-W T STY)	0.686	0.250
UOBIT-(RNN-S T STY)	0.674	0.243
UOBIT-(RNN-W RNN-S T STY)	0.699	0.251
UO4to-ENSAMBLE-1	0.416	0.092
UO4to-ENSAMBLE-2	0.444	0.092
ItaliaNLP-STL-SVM	0.663	0.253
ItaliaNLP-MTL-UmBERTo	0.718	0.309
ItaliaNLP-STL-UmBERTo	**0.735**	**0.331**

Table 4: Results according to TAG-it's Metric 1 and Metric 2 for Task 1.

Task 1 The best ItaliaNLP model achieves the scores of 0.735 for Metric 1 and 0.331 for Metric 2, which accounts for correctly predicted instances according to all dimensions at once. The other systems' performance is quite a bit lower. For Metric 1 UOBIT's best system still performs above all baselines, while UO4to only above majority baseline. Also according to Metric 2, UO4to performs above majority baseline but not better than the SVM.

For a deeper understanding of the results in Task 1, we look at the separate performance on the various dimensions, including both micro-F and macro-F scores, as label distribution is not balanced (Table 5).

What clearly emerges from the table is that classification of gender and topic is much easier than classification of age. This seems to suggest that textual cues are more indicative of these dimensions than age. Gap between best submitted (neural) model and SVM is way wider for topic and gender than for age.

	Micro-F			Macro-F		
Team Name-MODEL	**Topic**	**Gender**	**Age**	**Topic**	**Gender**	**Age**
Majority baseline	0.251	0.766	0.319	0.036	0.434	0.097
SVM baseline	0.808	0.832	0.382	0.565	0.683	0.319
UOBIT-(RNN-W T STY)	0.859	0.842	0.358	0.751	0.736	0.343
UOBIT-(RNN-S T STY)	0.835	0.856	0.331	0.724	0.797	0.303
UOBIT-(RNN-W RNN-S T STY)	0.869	0.869	0.360	0.791	0.811	0.337
UO4to-ENSAMBLE-1	0.333	0.523	0.392	0.172	0.517	0.341
UO4to-ENSAMBLE-2	0.470	0.521	0.341	0.394	0.515	0.302
ItaliaNLP-STL-SVM	0.774	0.810	0.404	0.502	0.619	0.347
ItaliaNLP-MTL-UmBERTo	0.873	0.873	0.406	0.716	0.716	0.358
ItaliaNLP-STL-UmBERTo	**0.898**	**0.891**	**0.416**	**0.804**	**0.834**	**0.377**

Table 5: Results according to micro and macro F-score for TAG-it's Task 1, for each separate dimension.

	Task 2a		Task 2b	
Team Name-MODEL	**Micro-F**	**Macro-F**	**Micro-F**	**Macro-F**
Majority baseline	0.835	0.455	0.288	0.089
SVM baseline	0.862	0.618	0.393	0.304
UOBIT-(RNN-W T STY)	0.852	0.692	0.278	0.272
UOBIT-(RNN-S T STY)	0.883	0.796	0.370	0.320
UOBIT-(RNN-W RNN-S T STY)	0.893	0.794	0.308	0.303
ItaliaNLP-STL-SVM	0.852	0.608	0.374	0.300
ItaliaNLP-MTL-UmBERTo	**0.925**	**0.846**	0.367	0.328
ItaliaNLP-STL-UmBERTo	0.905	0.816	**0.409**	**0.344**

Table 6: Results according to micro and macro F-score for TAG-it's Task 2a (gender) and Task 2b (age).

Task 2 As for Task 1, the best system is a neural model submitted by ItaliaNLP, both for Task 2a (gender) and Task 2b (age). All of the models perform above majority baseline, in spite of this task being potentially more complex since train and test data are drawn from different topics. As observed before, the gap between models and both baselines is higher for gender than for age. In addition to the previous observation that textual clues could be more indicative of gender than age, this lower performance could also be due to the fact that gender prediction is cast as a binary task while age is cast as a multiclass problem, turning a continuous scale into separate age bins.

In-depth Analysis Although official results are provided as micro-F score, we also report macro-F since classes are unbalanced and it is important to assess the systems' ability to discriminate well both classes. In gender prediction (Task 2a), comparing macro and micro F-scores, we observe that the gap between the two metrics is much lower for the neural models than for the SVMs (both our baseline as well as the system submitted by ItaliaNLP). This suggests that neural models are better able to detect correct cases of both classes, rather than majority class only.

We can also observe that in both tasks, results for age are not only globally lower than for gender, but also closer to one another across the submissions. We therefore zoom in on the age prediction task by comparing the confusion matrices of our SVM baseline and the best ItaliaNLP model, both in Task 1 (just the age prediction part) and in Task 2b. These are shown in Figure 2 and Figure 3 respectively.

What can be observed right away is that errors are not random, rather they are more condensed in classes closer to each other, underlining the ability of the systems. This is particularly true for the neural model (left in the Figures), where we

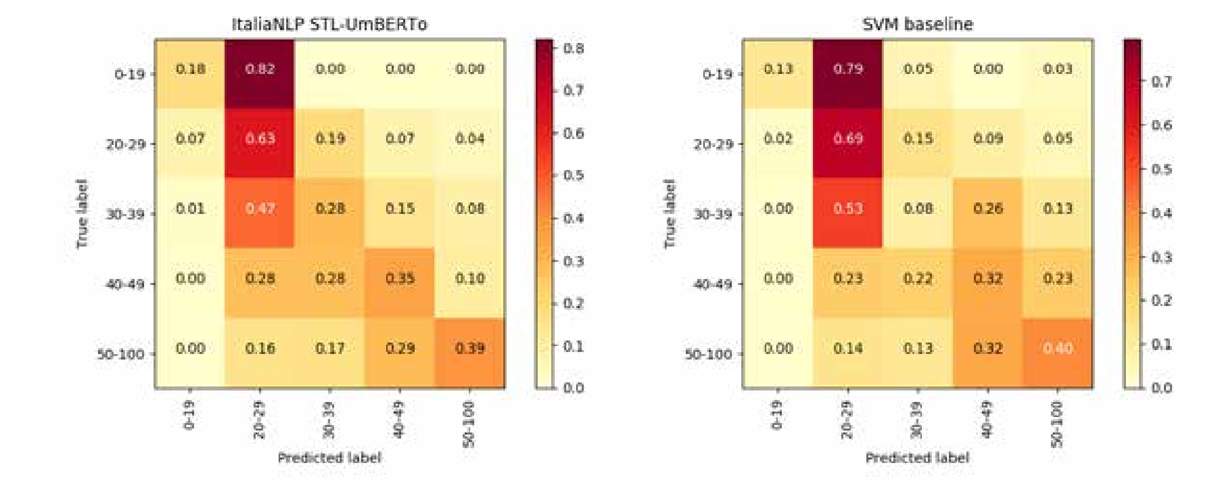


Figure 2: Normalized confusion matrices of the best ItaliaNLP system and the SVM baseline for Task 1 on the age dimension.

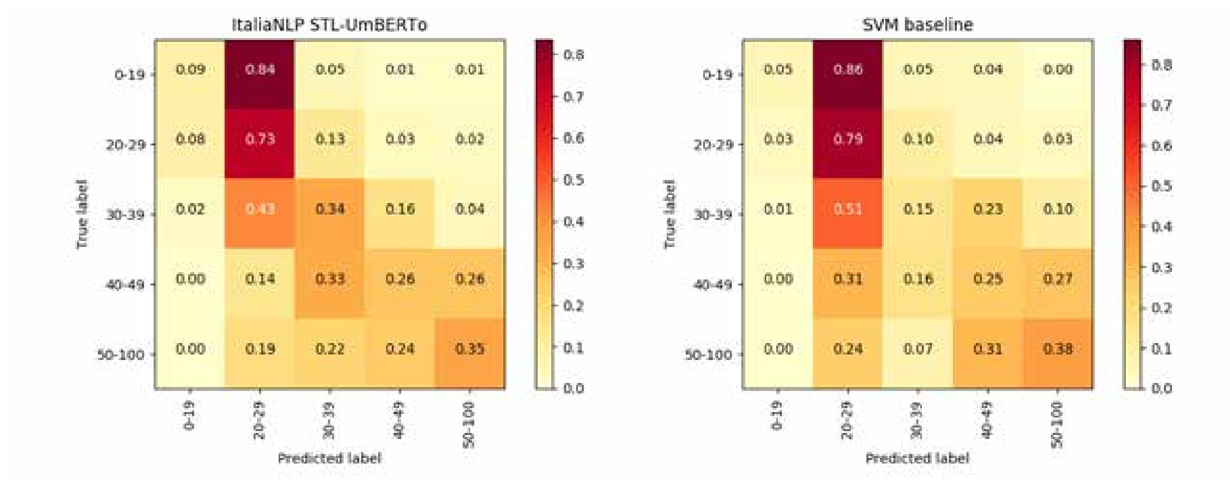


Figure 3: Normalized confusion matrices of the best ItalianNLP system and the SVM baseline for Task 2b.

can see the most confounded classes are the closest ones, thus generating a more uniform darker cluster along the diagonal.

Comparison to GxG As mentioned, TAG-it could be seen as a continuation of the GxG task at EVALITA 2018. In the latter, teams were asked to predict gender within and across five different genres. In TAG-it, in terms of profiling, we add age, which we cannot obviously compare to performances in GxG, and we use one genre only (forum posts), but implement a cross-topic setting.

We observe that results at TAG-it for gender prediction are higher than in GxG both within and cross-domain. We believe these are ascribable mainly to two relevant differences between the two tasks: (i) in this editions authors were represented by multiple texts, while in GxG, for some domains, evidence per author was minimal, and (ii) texts in TAG-it are probably less noisy, at least in comparison to some of the GxG genres (e.g., tweets and YouTube comments). Lastly, methods evolve fast, and since GxG was run in 2018, the use of Transformer-based models was not as spread as today. It would thus be interesting to assess the impact of fine-tuning large pre-trained models (as it's done in the best model at TAG-it) to gain further improvements in gender prediction.

One aspect that seems relevant in this respect is the appropriateness of the pre-trained model. Both ItaliaNLP and UOBIT used fine-tuned pre-trained models. However, while the latter used multilingual BERT as base, the former used the monolingual UmBERTo, obtaining higher results. This suggests, as it has been recently shown for a variety of tasks (Nozza et al., 2020), that monolingual models are a better choice for language-specific downstream tasks.

References

Maria Fernanda Artigas Herold and Daniel Castro Castro. 2020. TAG-it 2020: Ensemble of Machine Learning Methods. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

David Bamman, Jacob Eisenstein, and Tyler Schnoebelen. 2014. Gender identity and lexical variation in social media. *Journal of Sociolinguistics*, 18(2):135–160.

Angelo Basile, Gareth Dwyer, Maria Medvedeva, Josine Rawee, Hessel Haagsma, and Malvina Nissim. 2017. N-GrAM: New Groningen Author-profiling Model. In *Proceedings of the CLEF 2017 Evaluation Labs and Workshop - Working Notes Papers, 11-14 September, Dublin, Ireland.*

Angelo Basile, Gareth Dwyer, Maria Medvedeva, Josine Rawee, Hessel Haagsma, and Malvina Nissim. 2018. Simply the best: minimalist system trumps complex models in author profiling. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 143–156. Springer.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

John D. Burger, John Henderson, George Kim, and Guido Zarrella. 2011. Discriminating Gender on Twitter. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1301–1309, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Mart Busger op Vollenbroek, Talvany Carlotto, Tim Kreutz, Maria Medvedeva, Chris Pool, Johannes Bjerva, Hessel Haagsma, and Malvina Nissim. 2016. GronUP: Groningen user profiling notebook for PAN at CLEF. In *CLEF 2016 Evaluation Labs and Workshop: Working Notes Papers.*

Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso. 2018. Evalita 2018: Overview on the 6th evaluation campaign of natural language processing and speech tools for italian. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Walter Daelemans, Mike Kestemont, Enrique Manjavacas, Martin Potthast, Francisco Rangel, Paolo Rosso, Günther Specht, Efstathios Stamatatos, Benno Stein, Michael Tschuggnall, et al. 2019. Overview of pan 2019: Bots and gender profiling, celebrity profiling, cross-domain authorship attribution and style change detection. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 402–416. Springer.

Felice Dell'Orletta and Malvina Nissim. 2018. Overview of the EVALITA 2018 cross-genre gender prediction (GxG) task. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Lucie Flekova, Jordan Carpenter, Salvatore Giorgi, Lyle Ungar, and Daniel Preoţiuc-Pietro. 2016. Analyzing biases in human perception of user age and gender from text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 843–854, Berlin, Germany, August. Association for Computational Linguistics.

Roberto Labadie, Daniel Castro Castro, and Reynier Ortega Bueno. 2020. UOBIT@TAG-it: Exploring a multi-faceted representation for profiling age, topic and gender in Italian texts. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Aleksandra Maslennikova, Paolo Labruna, Andrea Cimino, and Felice Dell'Orletta. 2019. Quanti anni hai? age identification for italian. In *Proceedings of 6th Italian Conference on Computational Linguistics (CLiC-it), 13-15 November, 2019, Bari, Italy.*

Maria Medvedeva, Hessel Haagsma, and Malvina Nissim. 2017. An analysis of cross-genre and in-genre performance for author profiling in social media. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the 8th International Conference of the CLEF Association, CLEF 2017, Dublin, Ireland, September 11-14, 2017*, pages 211–223.

Debora Nozza, Federico Bianchi, and Dirk Hovy. 2020. What the [MASK]? Making sense of language-specific BERT models. arXiv:2003.02912.

Daniela Occhipinti, Andrea Tesei, Maria Iacono, Carlo Aliprandi, and Lorenzo De Mattei. 2020. ItaliaNLP @ TAG-IT: UmBERTo for Author Profiling at TAG-it 2020. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Francisco Rangel, Paolo Rosso, Martin Potthast, Benno Stein, and Walter Daelemans. 2015. Overview of the 3rd Author Profiling Task at PAN 2015. In *Working Notes Papers of the CLEF 2015 Evaluation Labs. CEUR Workshop Proceedings*.

Francisco Rangel, Paolo Rosso, Ben Verhoeven, Walter Daelemans, Martin Potthast, and Benno Stein. 2016. Overview of the 4th Author Profiling Task at PAN 2016: Cross-genre Evaluations. In *Working Notes Papers of the CLEF 2016 Evaluation Labs. CEUR Workshop Proceedings*.

Francisco Rangel, Paolo Rosso, Martin Potthast, and Benno Stein. 2017. Overview of the 5th author profiling task at PAN 2017: Gender and language variety identification in twitter. *Working Notes Papers of the CLEF*.

Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. 2010. Classifying latent user attributes in twitter. In *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, pages 37–44. ACM.

H Andrew Schwartz, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski, Stephanie M Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin EP Seligman, et al. 2013. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PloS One*, 8(9):e73791.

Rob van der Goot, Nikola Ljubešić, Ian Matroos, Malvina Nissim, and Barbara Plank. 2018. Bleaching text: Abstract features for cross-lingual gender prediction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 383–389.

Ben Verhoeven, Walter Daelemans, and Barbara Plank. 2016. Twisty: A multilingual twitter stylometry corpus for gender and personality profiling. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

UO_4to @ TAG-it 2020: Ensemble of Machine Learning Methods

Maria Fernanda Artigas Herold
Computer Science Department, Universidad de Oriente, Santiago de Cuba, Cuba
`nanda.ah@nauta.cu`

Daniel Castro Castro
Computer Science Department, Universidad de Oriente, Santiago de Cuba, Cuba
`danielcc@uo.edu.cu`

Abstract

This paper describes the proposal presented in the TAG-it author profiling task from EVALITA 2020 for sub-task 1. The main objective is to predict gender and age of some blog users by their posts, as well as topic they wrote about. Our proposal uses an ensemble of machine learning algorithms with three of the most used classifiers and language model of the n-grams of characters represented in a Bag of Word. To face this task we presented two different strategies aimed at finding the best possible results.

1 Introduction

With the growing development of technology and the frequent use of new forms of interactions and communications, Internet users spend more time sharing their ideas, thoughts, feelings and interests through social networks with diverse purposes, whether of personal businesses, self-expression, socialization, scientific, commercial, etc. In social media people often share their personal data, contact information, jobs, criteria and, in general, very useful information that can be used in research purposes about the behavior of people, development of marketing strategies and political campaigns, to serve various forensics applications, as well as strategies to determine certain demographic attributes of the person such as age, sex, characteristics of personality, geographic origins and even their occupation.

Precisely, one of the purposes of Natural Language Processing (NLP) research is to analyze the information obtained from users to create systems capable of extracting significant characteristics and improving the automatic understanding of written text.

Author Profiling (AP) is the main branch of NLP that studies the analysis of information to determine several demographic aspects of author such as age and gender given a set of documents presumably written by him, and recently some aspects such as the personality and occupation have also been included. The increased integration of social media in people's daily lives have made them a rich source of textual data for author profiling since data could be mined from the web, including emails and blogs, but there are still limitations in using social media as data source because data obtained may not always be reliable or accurate. Users used to provide false information about themselves that difficult the correct development of the task.

Document classification, also known as text tagging, is currently one of the most important subtask of Text Mining and NLP where the general idea is assign automatically one or more classes or categories in a set of predefined tags to a document using machine learning algorithms based on its content. Documents may be classified according to the subject, author or any other class that could be of interest in the research, as well as age and gender.

Recognized by the community, there is a theoretical evaluation framework, known as PAN[1], which encompasses authorship detection, author profiling, sentiment analysis, among others. On this platform, people can present and share their work, find out about the topics covered in previous works and participate in the tasks that are proposed each year for the community.

---

[1] https://pan.webis.de/

In 2019, at the PAN@CLEF evaluation forum (Rangel and Rosso, 2019), it was presented the Bots an Gender author profiling tasks, whose objective was determine if the author of a Twitter feed, in Spanish or English, had been written by a robot or a human, and in case of human, the gender should also be determined. To resolve this task, organizers proposed a set of baselines with models of n-grams of characters and words representation with a vocabulary reduction varying the parameters according to a few certain of configurations.

Another forum where the subject of author profiling has been worked on is MexA3T[2], another domain different from PAN for Spanish variants where generally works with the analysis of Mexican tweets. In 2019, it was proposed the MexA3T task for Author Profiling and Aggressiveness analysis focused on Mexican tweets (Aragón, 2019) as a follow-up of the task proposed in 2018 (Álvarez, 2018). The AP task comprises the detection of Place of Residence, Occupation and Gender of an user profile based on the set of tweets written by him. An user profile was distributed not only using the text of the tweets, but also images were incorporated on the profiles.

Several authors base their approaches on feature engineering and traditional machine learning classifiers. In previous works, methods have been proposed that work with comprising content-based (bag of words, word n-grams, term vectors, dictionary words), feature reduction (Castro, 2019) where the most used technique has been the selection of a subset of the most frequent features, stylistic-based features (frequencies, punctuation, POS, Twitter-specific elements, slang words) and approaches based on neural networks (CNN, LSTM) (Valdez, 2019).

2 TAG-it 2020

Despite the fact that Text Mining and NLP tasks focus a lot on the most used languages such as English and Spanish, others languages are also widely covered in several important forums. EVALITA[3] is a platform which promotes NLP tasks specifically for Italian language providing a shared framework where different systems and approaches can be evaluated in a consistent manner that has been working since 2007.

This year, TAG-it: Topic, Age and Gender Prediction for Italian from EVALITA (Cimino, 2020) propose three different sub-task of AP. The first one (subtask1) with the aim of predicting gender, age (in an age range, eg: 30-39) and the topic treated by the author given a collection of documents written by him/her in a blog, the three classes at once. The second one (subtask2a): for predicting gender only, and the third one (subtask2b): for predicting age.

For this task, a training corpus composed by texts written by users in a blog was offered, where each user has multiple posts. The information per user varies in length and quantity, in addition to the fact that the data is unbalanced for each class, which is not helpful for the training in classification task models.

2.1 Our method

According to the data corpus provided, our proposal is focused on classifying documents using a Bag of Word of n-grams characters representation, a feature reduction by a predefined number and an ensemble of machine learning algorithms: Random Forest, Support Vector Machine (SVM) and Centroid Nearest Neighbor classifiers, see Figure.1. We also consider Tf or a Tf-Idf as the weight of features.

We participate in the subtask1 where we present two different strategies. First we adjust the values of the parameters n for numbers of n-grams, k for feature reduction and the calculation of TF-IDF or not to the classification of each profile independently using a different configuration in each one according to the best results obtained in the individual classification. In the second proposal we adjust a general parameter and use the same configuration in the three profiles classification.

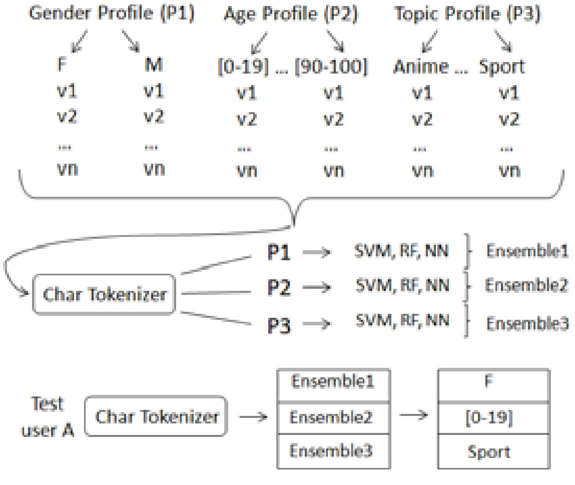


Figure.1 Ensemble architecture representation.

---

[2] https://sites.google.com/view/mex-a3t/

[3] http://www.evalita.it/

To represent the documents in a Bag of Word (BoW) model, we segment and preprocess the corpus and construct a vector of n-grams of characters ordered from highest to lowest by their respective frequency in the text per document. The parameters that we established for each configuration were: the n-grams of character representation, a size n from 1 to 5 characters and a number of 100, 500 and 1000 for feature reduction. Also for the weighing of the elements was considered the calculation of TF or TF-IDF, depending on the case, defined as follow:

$$tf_{ij} = \frac{n_{ij}}{\sum_k n_{ij}}$$

And TF-IDF value was defined as:

$$w_{ij} = tf_{ij} \times \log(\frac{N}{df_i})$$

Where tf_{ij} is the frequency of the token i in the document j, df_i is the number of documents that contain the token i and N is the total number of documents per user.

For machine learning algorithms we used the implementations that are arranged in Python sklearn library and among them we have RandomForestClassifier, NearestCentroid and OneVsOneClassifier for the three classifiers used in the ensemble.

To determine the definitive class to which a set of documents belongs with the ensemble of classifiers, we use a majority voting method, which consist of considering as the class of the document that which has been predicted by the largest number of classifiers.

For the validation process we use the StratifiedKFold from sklearn.model_selection module to perform a 5-Strified-K-Fold validation whit the training corpus which is divided into train and test respectively to be able to evaluate the effectiveness of the system. As an evaluation metrics we use F1 score for Topic an Age dimensions and for Gender we use Accuracy score from sklearn library in the first run. For second run we use the two different rankings proposed in the task to evaluate the participants: ranking 1 which evaluate the performance of each system using a partial scoring scheme, giving 1/3 of the points for each correctly predicted profile and 0 points if neither is correct; and ranking 2 which gives 1 point only if all classes are well predicted and 0 otherwise.

3 Experiments and Results

The test dataset provided by the tasks organizers was similar to train corpus (which was unbalanced especially for gender class, with a predominance of male users), and it was composed by posts of 411 different users with unknown age, gender and topic classes.

To obtain the best possible results with our method, we realized several experiments varying the values of the parameters in order to determine a good configuration per class. At the end of the experimentation process, we choose two different runs to be presented. The first one (Team2_1_1), see in Table.1, has a different configuration per class according to the best obtained result in the individual classification. Age class has been represented with a configuration of 2-grams of characters, a 1000 feature reduction and with TF-IDF as the weight of features. Gender class has been represented with a configuration of 4-grams of characters, a 1000 feature reduction and TF as the weight of features and Topic class has been represented with 4-grams of characters, a 1000 feature reduction and TF-IDF as the weight of features.

Using the Strified-K-Fold Cross-Validation we obtain as a result of the individual evaluation per class 0.3732, 0.8854 and 0.7051 for age, gender and topic respectively.

In the second run (Team2_1_2), see in Table.1, we have adjusted the parameters to be the same in the three classes and use a single configuration in all: 4-grams of characters, a 1000 feature reduction and TF for the weight of features.

Using the two metrics given in the TAG-it page we evaluate the second run and obtain 0.6801 and 0.2914 for Metric 1 and Metric 2 respectively as result.

Run	Metric 1	Metric 2
Team1_1_3	0,6991	0,2506
Team1_2_3	0,6739	0,2433
Team1_3_3	0,6991	0,2506
Team2_1_1	0,4160	0,0924
Team2_1_2	0,4436	0,0924
Team3_1_1	0,6626	0,2530
Team3_1_2	0,7177	0,3090
Team3_1_3	0,7347	0,3309

Table.1 Competition results for subtask 1.

The results obtained were not as good as expected compared with the results obtained in the validation process that we made, considering that n-gram of character representation obtained low scores for topic and age classification.

4 Conclusion

In this paper we described the proposal presented to participate in the TAG-it author profiling task from EVALITA 2020. Our proposal is based on an ensemble of machine learning algorithms with three well known classifiers and a Bag of Word of characters n-grams using a feature reduction by a predefined parameter and calculating TF or TF-IDF for features weight.

To resolve subtask 1 we proposed two different strategies where we first adjust the values of the parameters n for n-grams, k for feature reduction and Tf or TF-IDF for feature weight to the classification of each profile independently using a different configuration in each one, and in the second we just adjust a general parameter and use the same configuration in the three profiles classification at once.

Despite that the fact that in the evaluation process we carried out obtained better scores, the results of the task were not as good as expected, since low results were obtained for topic and gender dimension.

Reference

Francisco M. Rangel Pardo, Paolo Rosso: Overview of the 7th Author Profiling Task at PAN 2019: Bots and Gender Profiling in Twitter. *CLEF (Working Notes)* 2019

Mario Ezra Aragón, Miguel Ángel Álvarez Carmona, Manuel Montes-y-Gómez, Hugo Jair Escalante, Luis Villaseñor Pineda, Daniela Moctezuma: Overview of MEX-A3T at IberLEF 2019: Authorship and Aggressiveness Analysis in Mexican Spanish Tweets. *IberLEF@SEPLN 2019*: 478-494

Miguel Á. Álvarez-Carmona, Estefanía Guzmán-Falcón, Manuel Montes-y-Gómez, Hugo Jair Escalante, Luis Villaseñor-Pineda, Verónica Reyes-Meza, Antonio Rico-Sulayes: Overview of MEX-A3T at IberLEF 2018: Authorship and Aggressiveness Analysis in Mexican Spanish Tweets. *IberLEF@SEPLN 2018*

Valdez-Rodríguez, J.E., Calvo, H., Felipe-Riverón, E.M.: Author profiling from images using 3d convolutional neural networks. In: *In Proceedings of the First Workshop for Iberian Languages Evaluation Forum (IberLEF 2019),* CERUR WS Proceedings (2019)

Daniel Castro Castro, Maria Fernanda Artigas Herold, Reynier Ortega Bueno, Rafael Muñoz: Cerpamid-UA at MexA3T 2019: Transition Point Proposal.

Cimino A., Dell´Oreleta F., Nissim M. (2020). “TAG-it@EVALITA2020: Overview of the Topic, Age, and Gender prediction task for italian”. *In Proceedings of the Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian.*

UOBIT @ TAG-it: Exploring a Multi-faceted Representation for Profiling Age, Topic and Gender in Italian Texts.

Roberto Labadie Tamayo, Daniel Castro Castro and Reynier Ortega Bueno
Computer Science Departament, University of Oriente
Santiago de Cuba, Cuba
`roberto.labadie@estudiantes.uo.edu.cu,`
`{danielcc, reynier}@uo.edu.cu`

Abstract

English. This paper describes our system for participating in the TAG-it Author Profiling task at EVALITA 2020. The task aims to predict age and gender of blogs users from their posts, as the topic they wrote about. Our proposal combines learned representations by RNN at word and sentence levels, Transformer Neural Nets and hand-crafted stylistic features. All these representations are mixed and fed into a fully connected layer from a feed-forward neural network in order to make predictions for addressed subtasks. Experimental results show that our model achieves encouraging performance.

The growing integration of social media with people's daily live has made this medium a common environment for the deployment of technologies that allow the retrieval of useful information in the development of business activities, social outreach processes, forensic tasks, etc. That is because people frequently upload and share content in these media with various purposes such as socialization of points of view about some topic or promotion of personal business, etc. The analysis of textual information from such data, is one of the main reasons why researches become trending on the Natural Language Processing (NLP) field.

However, the fact that this information varies greatly in terms of its format, even when it comes from the same person, besides textual sequences are unstructured information, make challenging the process of analyzing it automatically. Author Profiling (AP) task aims at discovering different marks or patterns (linguistic or not) from texts, that allow a user to be characterized in terms of their age, gender, personality or any other demographic attribute.

Many forums, due to the applicability of AP, share tasks directed to mining features that in general way, predict that valuable information. Those tasks commonly make special focus on popular languages such as English and Spanish. Nevertheless, other languages are explored on important forums too, that is the case of EVALITA [1], this one, promoting analysis of NLP tasks in the Italian language. Among the challenges from its last campaign *EVALITA 2018* was the AP (in terms of gender) task *GxG* (Dell'Orletta and Nissim, 2018), exploring the gender-predicting issue.

The analysis of age, gender and the topic a text is related with, are tasks well explored and the most approaches employ data representation based on stylistic features, n-gram representations and/or words embedding combined with Machine Learning (ML) methods like Support Vector Machine (SVM) and Random Forest (Pizarro, 2019). Also some authors by using Deep Learning (DL) models like Convolutional Neural Networks (CNN) and Long-Short Term Memory (LSTM) combined with stylistic features (Aragón and López-Monroy, 2018) (Bayot and Gonçalves, 2018) have yield encouraging performances.

In this work we address precisely, the automatic detection of gender and age of the authors, besides the identification of the prevailing topic on textual information from blogs. Also, we describe our developed model for participating on *TAG-it: Topic, Age and Gender prediction for Italian*[2] (Cimino A., 2020) task at *EVALITA 2020* (Basile et al., 2020).

Having in account the proved ability of DL

---

[1] `http://www.evalita.it/`

[2] `https://sites.google.com/view/tag-it-2020`

models to learn abstract depictions that are omitted in hand-crafted features engine methods, our approach is mainly based on them, particularly on Bi-LSTM and Transformer Nets (Vaswani et al., 2017). We combine the feature representations learned by DL models, with hand-crafted ones based on Term Frequency-Inverse Document Frequency *(tf-idf)* and stylistic features.

This paper is organized as follow: in the next section a brief description about the different subtasks of *TAG-it* task. Next, we present our proposal. Specifically, we describe the data preprocessing as well as the DL methods and features used for depicting this data. Finally, the experimental setting, the experiments conducted and the results achieved.

1 TAG-it Tasks

Three sub-task have been proposed on TAG-it task.

- **subtask 1**: Toward to predict the gender, the age (as an age range, eg: 20-29) and the topic mentioned by the author given a collection of texts written by him/her from a blog, all this three dimensions at once.
- **subtask 2a**: For predicting gender.
- **subtask 2b**: For predicting age.

For these tasks a training corpus of texts written by blogs users, with possibly multiple posts per user, was provided. Each user information (i.e posts per user) varies in terms of its length and quantity, and the data for each subtask is unbalanced mainly for gender and topic prediction tasks, which place some complexity degree for the training stage of the models for these classification tasks.

2 Our Proposal

Deep Learning methods are capable to learn and project relationships between elements within textual information which are beyond the human abstract comprehension. Therefore the use of just hand-crafted representations may omit some important patterns on textual information analysis.

However, stylistic and linguistic features have proved to be good marks to determine some author characteristics. Within the used DL models on AP field, are the LSTM (Labadie-Tamayo et al., 2020) and the Transformers Neural Nets, which rely on two different paradigms. The first ones analyses the information sequentially, token by token whereas the second ones analyze all these tokens at once, relating every one with respect to each other. The opposite behavior of these two architectures implies learning different patterns which individually have proved to be an accurate way to synthesize the information.

We hypothesize that making an ensemble of these deep representations and fusing it with hand-crafted ones as we show on Figure. 1 could yield encouraging results on the proposed tasks.

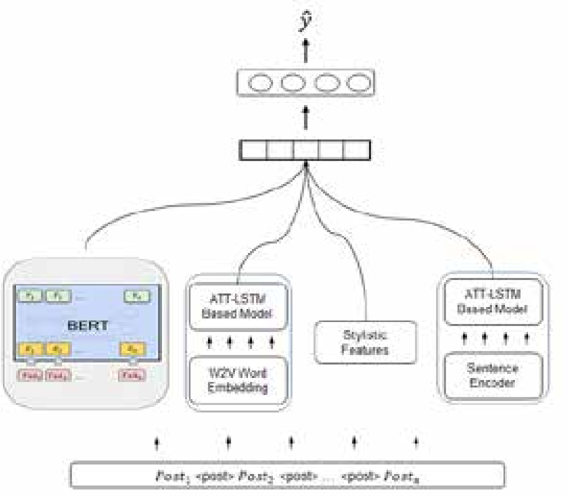


Figure 1: Representations Ensemble

The first representation *(Transformer Block)* based on Bidirectional Representation from Transformers (BERT) Architecture (Devlin et al., 2018). The second based on LSTM (Hochreiter and Schmidhuber, 1997) neural nets with self attention mechanism (Att-LSTM) by using words embedding *(Recurrent Word-Level Block)*. The third one, a condensed representation based on the combination of stylistic features and a vector with the *tf-idf* computation of some keys tokens from the text *(Stylistic Block)*. Finally *(Recurrent Sentence-Level Block)*, another representation based on Att-LSTM, but at this time, analyzing the sequence information at sentence level.

All these representations are concatenated and fed into a dense layer, by using Leaky Rectified Linear Unit (Leaky ReLU) activation function, to synthesize the extracted information on each block and its output vector goes to a softmax dense layer which have the same number of neurons as classes on the analyzed task, in order to make the predictions.

For dealing with the three classification tasks we used the same architecture, but trained separately for each of them, with different targets attending to the task.

2.1 Preprocessing

In the preprocessing stage we concatenate the posts corresponding to the same user, in order to treat them as only one super-document, but between each post we place a tag i.e ⟨ post ⟩ denoting the ending-beginning of them. Afterwards, the numbers and dates are recognized and replaced by a corresponding wildcard which encodes the meaning of these special tokens. Then, the text is tokenized and morphologically analyzed by means of FreeLing (Padró and Stanilovsky, 2012).

For computing the stylistic and *tf-idf* vectors as for feeding the deep models on prevailing topic detection task, we removed the stop words from the document and lemmatized the tokens to their canonical form.

2.2 Transformer Block. BERT

BERT (Bidirectional Encoder Representations from Transformers) is an architecture resulting of applying a bidirectional training to the attention model Transformer, designed for language modeling. The Transformer model has two mechanisms, the first one, known as the encoder, which is fed with the text and finds out an encoded representation for the sequence. The second one, the decoder, produces the predicted tokens for language modeling one at time, having in account the encoder's output and the previous predicted tokens on each time step.

The main advantage of this transformer models w.r.t. traditional sequential architectures like Gated Recurrent Unit (GRU) (Cho et al., 2014) is that instead of analyzing the textual information in one or another direction (e.g. right to left or left to right) it takes in account the entire information at once by using an attention mechanism, which relates each word on the text with its surrounding context.

Since the goal of BERT is to generate a language representation, only the encoder mechanism is necessary. It is structured with transformer blocks connected sequentially and each transformer block is composed by attention heads working in parallel. These transformer blocks give to their subsequent layer one representation for each element of the input text, but these representations correlates the entire input context.

The original BERT model is trained with two subtasks, one of them consisting on predict some masked words from a sentence and the other one consisting on predict if two sentences are consecutive in the given corpus text.

For the *TAG-it* tasks we employed a pre-trained BERT model on a multilingual corpus (multilingual_L-12_H-768_A-12)[3] (Turc et al., 2019), which is fed with the super-document sequence. From this model we just used the first two transformer blocks and as its output we keep the first and last vectors from the input sequence encoding, which are concatenated.

Also we applied fine tuning on BERT, adding an intermediate dense layer of 64 units by using Leaky ReLU activation function, and taking as target for training a multitask focus trying to make predictions for age, topic and gender tasks at once.

2.3 Recurrent Word-Level Block

The second representation block of our system is based on LSTM nets. This block takes as input a sequence of the preprocessed text information, which is fed into an embedding layer, set up with fixed weights from FastText (Grave et al., 2018) pretrained word embedding[4], obtaining from each word of the sequence a vectorial representation.

The textual sequence is provided with relevant or not information with respect to the task in analysis. In order to highlight the most important elements for encoding the message instead of making the network pays attention to all elements alike, the embedding layer output tokens are scored by its relative importance over the other elements on its context with Scaled Dot-Product Attention Mechanism (Vaswani et al., 2017). Then, the new scored sequence is fed into a Bidirectional-LSTM (BI-LSTM) (Schuster and Paliwal, 1997) layer with 64 neurons which perform two analysis over this sequence, in forward and backward directions, for detecting not just relations of an element with the previous ones, but also with the elements that appear after it. Afterwards, the hidden states from the Bi-LSTM layer are considered as a new sequence, which is fed into another LSTM with 64 neurons too, taking from its output just the last hidden state, which represents the Recur-

---

[3] `https://github.com/google-research/bert`

[4] `https://fasttext.cc/docs/en/crawl-vectors.html`

rent Word-Level Block encoding.

For training this block we applied dropout (Srivastava et al., 2014) to the neurons of the attention and LSTM layers in order to improve the generalizing capability of the model.

2.3.1 Scaled Dot-Product Attention

This attention function at first, maps for each sequence token three representations (the query and a key-value pair) for computing a compatibility index between every pair of elements. Afterwards, for each token t_i is evaluated its compatibility w.r.t every other sequence token t_j by relating its query vector q_i with all the keys k_j, then these compatibilities c_{ij} are normalized with a softmax function and used for scoring the value vectors v_j in front of that specific query. Finally, the attention based representation for t_i is computed as the weighted sum of these pondered values vectors. This computation is defined as follows:

$$Attention(Q, V, K) = softmax(\frac{Q \times K^T}{\sqrt{d_k}}) \times V \quad (1)$$

Where $Q, K \in \Re^{n \times d_k}$ and $V \in \Re^{n \times d_v}$ are matrices, which, on every row contain for query, key and value respectively the mappings of the sequence tokens, n corresponds to the length of the sequence and d_k, d_v to the dimension of mapping vectors for key and value respectively.

2.3.2 LSTM

LSTM networks are a special kind of RNNs, which are specialized on analyzing sequential data. These have a main cell unit (the recurrent unit) which explores the data sequence one element at each time step (left to right order). This network shares the information captured in previous steps, for computing the new hidden state at the current time step. Inside the main cell is contained a gate structure that informs to the network which information preserve or forget from the hidden sates of previous time steps for the current computation.

2.4 Stylistic Block. Stylistic Features

The Representation based on stylistic features is twofold; in one side we consider for characterizing a user attending to some classification task, a vector containing the *tf-idf* of a set of key tokens from the text and on the other side we construct a statistical style features vector which captures information from distinct lexical and syntactical linguistic layers.

For constructing the first one we used a feature selection approach which score every term employed by users corresponding to some category within a classification task and then are selected the more relevant ones.

For scoring the tokens we use IG (Sebastiani, 2002) standing for Information Gain, which takes into account the presence of a term in a category as well as its absence. The information gain of a term t in a class C is defined as:

$$IG(t, C) = \sum_{c \in \{C, \bar{C}\}} \sum_{x \in \{t, \bar{t}\}} P(x, c) \log_2 \frac{P(x, c)}{P(x)P(c)} \quad (2)$$

In this formula, probabilities are interpreted on an event space of documents (e.g. $P(\bar{t}, C)$ indicates the probability that, for a random document d, term t does not occur in d and d belongs to category C).

Once computed the IG for every term which belongs to documents of the class c_i, the $\frac{500}{l_c}$ tokens with highest IG are chosen for characterizing this class , where l_c is the number of the task classes. Finally a $500 - dimensional$ vector is constructed where its components are computed as the *tf-idf* of the representative terms from every class.

The second representation is computed independently of the addressed task as a $12 - dimensional$ vector where its components are real numbers corresponding to statistical values from lexical and syntactical linguistic layers (e.g sentence, paragraph, syntactic layers) such as:

- Paragraph layer: Standard deviation of the sentences' length written by the user.
- Text layer: Number of stop words used.
- Sentence layer: Average of words' length.
- Syntactic layer: Proportion of nouns over adjective.

These two representations are combined and fed into a 64-neurons dense layer to synthesize the information and later being fused it with the other blocks representations.

2.5 Recurrent Sentence-Level Block

This block shares the same structure with the *Recurrent Word-Level Block*, but instead to be fed with a sequence composed by word representations provided by a word embedding layer, it is fed with a sequence resulting of encoding each super-document's sentence by means of an encoder with a similar structure as the first analyzed *Transformer-Block* .

For this Recurrent Sentence-Level Block, we trained the sentence encoder with the same multi-task focus as in the *Transformer-Block* , but aiming to predict for each sentence from a document the annotated characteristics (i.e age and gender) of the user who it belongs to and the topic of its surrounding text. Then we encode all the sentences from the super-document composed by the user's posts, and we considered them as tokens from a sequence at sentence level. Afterwards, that sequence is fed into a model with the same structure Att-Bi-LSTM as the *Recurrent Word-Level Block* taking from this, as the user's profile encoding, the last hidden state from the second LSTM layer as in the Word-Level block.

3 Experiments and Results

The dataset used in this work was the one provided by the task organizers. This dataset is unbalanced, mainly for gender classification task, where the male class represents the 82.6% of the examples.
In order to prevent a biased training of the model we applied a class-weighting method, scoring the computed loss for every examples having in account the class which it belongs to (i.e for examples from male class we give to the computed loss a weight of 0.3 whereas for female examples we pondered the loss to 0.7) this makes that when parameters are updated by means of the gradients, the models pays more attention to the most weighted class, specifically to the under-represented class.
We pretrain the Transformer models from the *Transformer Block* and the sentence encoder of the *Recurrent Sentence-Level Block* independently of the entire model and then we fixed the learned weights.
For fine tuning these BERT models we employ Adam Optimizer, using categorical cross-entropy loss function for every output layer, since we applied multi-task learning over two epochs. The learning rate for this training was set up to a low value (lr=1e-5) since we wanted to keep the parameters learned from the original train with an enormous data as more as possible, while we made the model focus on our addressed tasks, also we set the decay = 2e-3 to the learning rate scheduler.

We evaluate and select the hyper-parameters as the representation and features that we used for our model by using a cross-validation method to obtain a more realistic an unbiased performance evaluation, making 5 splits for validation. On each cross validation step, the dataset was split in 20% for validation and 80% for training, keeping the distribution of examples relative to the split size.
The performance of the model on training stage was evaluated independently for each subtask by using different combinations of representations from Recurrent Word-Level Block (RNN-W), Recurrent Sentence-Level Block (RNN-S), Transformer Block (T) and Stylistic Block (STY).
For age and gender prediction we employed Micro-F1 metric whereas for topic prediction we used accuracy metric for the evaluation. In Table. 1 we summarize the results obtained in terms of the average of these metrics in cross-validation training.
As we can see, assembling the three deep representations with the stylistic one, yield a good performance in all cases through the cross-validation process. However, the stylistic representation had a soft negative influence on gender prediction task.

Table 1: Model Performance on training data.

Model	**Age**	**Gender**	**Topic**
	AVG-F1	AVG-F1	Acc
RNN(S+W)-STY-T	**0.378**	0.941	**0.935**
RNN(S+W)-T	0.203	**0.946**	0.885
RNNS-STY-T	0.348	0.940	0.931
RNNW-STY-T	0.339	0.919	0.903

Regarding the official results, we submitted 3 runs as **UOBIT** team, on each of them we employed the representations learned by the Transformer and Stylistic Blocks by tuning the use of the Recurrent Blocks' encode, as shown on Table. 2.

After the evaluation phase we try to remove the stylistic features based representation and we found out that this representation, possibly be-

Table 2: Model Performance on test data.

run	Model	Subtask 1		Subtask 2a	Subtask 2b
		Metric 1	Metric 2	Micro-F1	Micro-F1
run-1	RNN-W T STY	0.686	0.251	0.852	0.278
run-2	RNN-S T STY	0.674	0.243	0.883	0.370
run-3	RNN-W RNN-S T STY	**0.699**	**0.251**	0.893	0.308
	Unofficial				
-	RNN-W RNN-S T	0.680	0.248	**0.898**	**0.4680**
-	RNN-W RNN-S	0.667	0.243	0.893	0.369
-	T	0.436	0.067	0.835	0.283

cause of it introduces some noise, makes the model to have a worst performance, at least on those tasks related to the author attributes (i.e gender and age) corresponding to task 2a and task 2b. We think that noise introduced by these features mainly comes from the fact that they are computed based on key tokens from the text, these tokens may suggest to the model that texts with same topic belongs to the same class within gender or age classification task.

The performance of our system just by using the deep representations of the Recurrent and Transformer Blocks, yield a performance of 0.4606 under F1 metric on subtask 2b which improves the ones reached by the best team of 0.409, whereas this same combination improves our best official run on subtask 2a. These results are shown on Table. 2 under the row named Unofficial.

4 Conclusions

In this paper we described our system for participating in the TAG-it Author Profiling task at EVALITA 2020. Our proposal is based on an ensemble of RNN, Transformer Neural Nets and hand-crafted stylistic features. The system receives as input a user's profile textual information as an only one super document (sequence), this information is encoded in four different ways, the first one by a Transformer Block, specifically a fine tuned and reduced BERT model, the second one, by a Recurrent Block based on an Attention-Bi-LSTM model analyzing the information at word level, the third one by a feature representation based on the combination of *tf-idf* information and stylistic features extracted from the text. Finally the fourth one by the same recurrent structure as in the Recurrent Worf-Level Block, but analyzing the information at sentence level.

This four representations are mixed and fed into a dense layer for synthesize them and its output is received by another dense layer which classify this profile taking into account the classes from the addressed subtask.

The results shown that considering both the stylistic representation and the deep representations learned by Recurrent and Transformer models we obtain the best effectiveness based on the accuracy measure for the task related to the topic classification, but this behavior changed for age and gender classification, due to the relationship of syntactic structures of the text with the topic that the user's posts are related to. We think that excluding the stylistic features or at least those related to the frequency of tokens from the text, could be a way to increase the effectiveness of the ensemble, mainly on the age detection subtask. Also analyzing the content of the posts at character level, due to the informal text origin, would solve the problem of missidentification of some key words within te text. We would like to explore these ideas in future work.

References

Mario Ezra Aragón and A-Pastor López-Monroy. 2018. A straightforward multimodal approach for author profiling. In *Proceedings of the Ninth International Conference of the CLEF Association (CLEF 2018)*.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Roy Khristopher Bayot and Teresa Gonçalves. 2018. Multilingual author profiling using lstms: Notebook for pan at clef 2018. In *CLEF (Working Notes)*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Nissim M. Cimino A., Dell'Orletta F. 2020. Tag-it@evalita2020: Overview of the topic, age, and gender prediction task for italian.

Felice Dell'Orletta and Malvina Nissim. 2018. Overview of the evalita 2018 cross-genre gender prediction (gxg) task. *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:35.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Roberto Labadie-Tamayo, Daniel Castro-Castro, and Reynier Ortega-Bueno. 2020. Fusing Stylistic Features with Deep-learning Methods for Profiling Fake News Spreader—Notebook for PAN at CLEF 2020. In Linda Cappellato, Carsten Eickhoff, Nicola Ferro, and Aurélie Névéol, editors, *CLEF 2020 Labs and Workshops, Notebook Papers*. CEUR-WS.org, September.

Lluís Padró and Evgeny Stanilovsky. 2012. Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey, May. ELRA.

Juan Pizarro. 2019. Using n-grams to detect bots on twitter. In *CLEF (Working Notes)*.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.*, 45(11):2673–2681.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

ItaliaNLP @ TAG-IT: UmBERTo for Author Profiling at TAG-it 2020

Daniela Occhipinti⋆**, Andrea Tesei**⋆**, Maria Iacono**⋆**, Carlo Aliprandi**⋆ **and Lorenzo De Mattei**⋄ † ⋆
⋆ Aptus.AI / Pisa, Italy
⋄ Dipartimento di Informatica, Università di Pisa / Pisa, Italy
† Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC–CNR)
ItaliaNLP Lab / Pisa, Italy
{daniela,andrea,maria,carlo}@aptus.ai
lorenzo.demattei@di.unipi.it

Abstract

In this paper we describe the systems we used to participate in the task TAG-it of EVALITA 2020. The first system we developed uses linear Support Vector Machine as learning algorithm. The other two systems are based on the pretrained Italian Language Model UmBERTo: one of them has been developed following the Multi-Task Learning approach, while the other following the Single-Task Learning approach. These systems have been evaluated on TAG-it official test sets and ranked first in all the TAG-it subtasks, demonstrating the validity of the approaches we followed.

1 Introduction

Author Profiling (AP) is a known Natural Language Processing task consisting in the extraction or the prediction of information about the authors of some disputed documents. Such information can include the age and the gender of the authors. The AP problem is assuming more and more importance in several fields, such as security, forensics, marketing and sales, and so on. For example, in forensics, detecting the age and the gender of the author of a given document can be very helpful for determining whether a person should be considered as a suspect or not; from the marketing and sales' perspective, companies can understand what kind of people may or not like their products on the basis of the analysis performed on people's reviews or blog and social network posts (Rangel et al., 2015).

In the context of EVALITA 2020 (Basile et al., 2020), the periodic evaluation campaign of Natural Language Processing and speech tools for the Italian language, the task TAG-it (Cimino et al., 2020) is proposed. TAG-it is an AP task in which the goal is to provide a system capable of predicting the gender and the age of the authors of several blog posts and their topics. This task can be considered as a follow-up of the EVALITA 2018's GxG task (Dell'Orletta and Nissim, 2018) in which the goal was the prediction of the author's gender for Twitter posts, YouTube comments, Children Essays, Diaries and News; in GXG models were trained and tested *cross-genre*. These two aspects led to scores lower than ones observed in other campaigns and languages. In order to address this problem and get better performances, in TAG-it only blogs' genre is considered and longer texts are used, since they provide more evidence than tweets and Youtube comments, which are shorter than blog posts. Moreover, with respect to GxG, TAG-it adds the topic control with the aim of evaluating the interaction of topic and lexically rich models on performances in a more direct way than in GxG, in which this was indirectly done via cross-genre prediction. TAG-it is divided in two subtasks: the goal of the first one (Subtask 1) is to classify gender, age and topic at once, while the goal of the second one is to predict age (Subtask 2a) and gender (Subtask 2b) separately and with topic control.

De Mattei and Cimino (2018) and Cimino et al. (2018) demonstrated the validity of Multi-Task Learning approach to establish the state of the art for several Italian NLP task, in the context of GxG, Cimino et al. (2018) developed the best system for this task based on Bidirectional LSTM (Bi-LSTMs) trained using a Multi-Task Learning approach. For TAG-it we replicated the same approach: we developed a baseline system based on SVM, and two neural systems, the first one exploiting a Single-Task Learning approach, the second one a Multi-Task Learning approach. In-

---

stead of the Bi-LSTM model used by Cimino et al. (2018) for TAG-it we exploited a deeper neural pretrained language model: BERT (Devlin et al., 2019).

2 Description of the Systems

We implemented and tested three different systems. Our early experiments were led on a training set and a test set obtained by shuffling and splitting (80% training - 20% test) the training set provided by the organisers in order to analyse the classifiers' performances on a labeled dataset. At the end of our experiments, we trained our best classifiers on the whole training set and run them on the TAG-it test sets provided by the organisers.

For our experiments and runs, as a preprocessing phase, we filtered out all posts less than 20 characters in length and labeled each post of the dataset with the corresponding author's id, gender, age and topic. In Table 1 we report the distributions of the classes of the TAG-it dataset.

	Train	**Test1**	**Test2a**	**Test2b**
M	15070	315	344	730
F	3113	96	68	69
0-19	2232	39	76	79
20-29	5412	131	189	230
30-39	3569	95	51	134
40-49	3577	69	48	216
50-100	3393	77	48	140
ANIME	3925	97	0	0
AUTO-MOTO	3648	76	0	0
BIKES	468	12	0	0
CELEBRITIES	1063	22	0	0
ENTERTAINMENT	534	9	0	0
MEDICINE-AESTHETICS	370	16	0	0
METAL-DETECTING	1471	26	0	0
NATURE	481	11	0	0
SMOKE	1574	30	0	0
SPORTS	4593	103	0	0
TECHNOLOGY	56	9	0	0
GAMES	0	0	298	298
ROLE-GAMES	0	0	114	114
CLOCKS	0	0	0	387

Table 1: TAG-it datasets distributions

As a first step, our systems make their predictions by classifying the three dimensions post by post. Then they use a voting mechanism according to which the gender, the age and the topic of an author are represented by the most frequent values assigned by the classifiers to his/her posts.

The first system we implemented uses linear Support Vector Machine as learning algorithm and we used different features for predicting the core dimensions of the dataset, the second system is based on a Single-Task Learning BERT model and the third system is based on a Multi-Task Learning BERT model. In particular, we used UmBERTo[1], an Italian pretrained Language Model developed by Musixmatch.

In the following subsections we will describe these systems in detail.

2.1 Support Vector Machine Classifiers

As regards the system based on three linear SVM statistical models, we used the scikit-learn[2] Python library and we conducted several experiments by testing different configurations for feature extraction. In all the experiments we used the TF-IDF vectorizer, but we changed the tokenizer and the *n*-grams context window. In particular we tested five different kinds of features: character *n*-grams, word *n*-grams, lemma *n*-grams, Part-Of-Speech *n*-grams and bleached tokens. As regards the bleached tokens features, they were extracted after performing a bleach tokenization consisting in fading out lexicon in favour of an abstract token representation (van der Goot et al., 2018). The word *n*-grams, lemma *n*-grams and Part-Of-Speech *n*-grams features were extracted by using the linguistic pipeline for the Italian language provided by spaCy[3]. For the multi-class classification we applied the One-Vs-Rest method (Rennie and Rifkin, 2001). In Table 2 we report the performances in terms of micro-average f-score of the SVM models tested in our experiments.

These results led us to choose the best SVM classifiers for the official runs on the provided test set; analysing them, we can state that the best SVM classifiers tested in our experiments are the following:

- Topic Detection: One-Vs-Rest Linear SVM using features extracted through a TF-IDF Vectorizer considering character *n*-grams;
- Age Detection: One-Vs-Rest Linear SVM using features extracted through a TF-IDF Vectorizer considering lemma *n*-grams;
- Gender Detection: Linear SVM using features extracted through a TF-IDF Vectorizer considering word *n*-grams.

---

[1] `https://github.com/musixmatchresearch/umberto`

[2] `https://scikit-learn.org/stable/`

[3] `https://spacy.io`

	Gender	Age	Topic
word n-gram	**0.933**	0.3873	0.7882
char n-gram	0.9284	0.3739	**0.8333**
lemma n-gram	0.9265	**0.4189**	0.7928
pos n-gram	0.9223	0.3063	0.3873
bleached words	0.9223	0.3739	0.4775

Table 2: SVM classifiers' micro-average f1-scores on validation set

2.2 Single-Task BERT-based Classifiers

Our second system consists of three different BERT models and a classifier on top of each of them. More precisely, we used the UmBERTo language model, which was pretrained on a large Italian Corpus: OSCAR (Ortiz Suárez et al., 2020).

This language model have 12-layer, 768-hidden, 12-heads, 110M parameters. On top of the language model we added a ReLU classifier (Nair and Hinton, 2010). We applied dropout (Srivastava et al., 2014) to prevent overfitting. As loss function we used the sum of loss functions of the three classifiers. For each classifier, we used Cross Entropy as loss function.

In Table 3 we report the system's performances in terms of f1-score obtained on the validation set.

	f1-score
Gender	0.86
Age	0.35
Topic	0.66

Table 3: Single-Task Learning BERT-based system micro-average f1-scores on validation set

2.3 Multi-task BERT-based Classifier

Our last system is based on a unique UmBERTo model and three classifiers on top of it, each one responsible of predicting one of the three core dimensions of the dataset according to the Multi-Task Learning approach used in (Cimino et al., 2018). On top of the model we added three ReLU classifiers, we applied the dropout method and we used the sum of the Cross-Entropy loss functions of the three classifiers as loss function.

In Table 4 we report the system's performances in terms of f1-score obtained on the validation set.

	f1-score
Gender	0.86
Age	0.39
Topic	0.64

Table 4: Multi-Task Learning BERT-based system f1-scores on validation set

3 Results and Evaluation

We run all our three systems on the test sets provided by the task organisers. The performances of our systems are reported in Table 5.

For the Task 1 scoring, TAG-it considers two different rankings. The first ranking is obtained using a partial scoring scheme, giving 0 points if no correct predictions are provided for the three dimensions of the dataset, 1/3 points if one out of three correct answers is given, 2/3 points if two out of three correct answers are given and 1 point if all the answers given by the system are correct. The second ranking assigns 0 points if no correct predictions are provided for the three dimensions of the dataset and 1 point if all the answers given by the system are correct. In both cases, the final score is the sum of the points achieved by the system across all the documents normalized with respect to the number of documents in the test set. For the Task 2, the micro-average f-score is used as scoring function.

	STL-SVM	MTL-BERT	STL-BERT
Task 1 metric 1	0,6626	0,7178	**0,7348**
Task 1 metric 2	0,253	0,3090	**0,3309**
Task 2a	0,8519	**0,9247**	0,9053
Task 2b	0,3742	0,3667	**0,4093**

Table 5: Systems' performances evaluation with TAG-it metrics

Analysing the scores in Table 5, we can state that the best system in the TAG-it context is the one based on BERT using the Single-Task Learning (STL-BERT) approach, obtaining the best scores in Task 1 and Task 2b (age prediction). In Task 2a, consisting in gender prediction with topic control, the best system is the Multi-Task Learning BERT-based system (MTL-BERT). Hence, the systems based on deeper neural models outperform the systems based on traditional machine learning techniques, i.e. the SVM (STL-SVM).

Task 1: In order to compare classifiers' predictions on Task 1 with regard to each dimension and to understand the correlation between labels, we plotted and analysed some distributions.

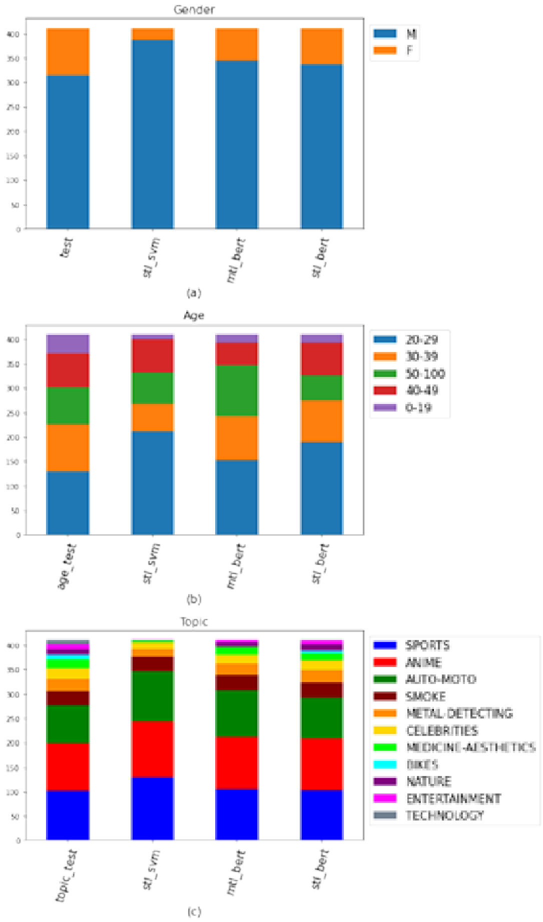


Figure 1: Task 1, Distributions of the dimensions' classes in test set and classifiers' predictions.

In Figure 1, we reported the distribution of the labels in the test set and in the classifiers' output. As regards the gender prediction (a), we can note that the STL-SVM classifier overestimates the `M` class, most likely because the `M` and `F` classes are very unbalanced in the training set. STL-BERT and MTL-BERT's distributions, on the contrary, are closer to the test set's one: in our setting the neural models appear less affected by the imbalance of a training set.

Observing the distributions of the Age classes in Figure 1 (b), we can observe that for all the three systems the distributions of the labels are not close to the distribution of the test set. The nearest distribution is the one of MTL-BERT's output.

Looking at the Topic classes distributions in Figure 1 (c), we can observe, once again, that the SVM-based system's one is the less close to the test set in that it has the tendency to overestimate the `SPORT`, `ANIME` and `AUTO-MOTO` classes and it does not recognise the `BIKES` and `TECHNOLOGY` classes as they are underrepresented in the training set (respectively the 2.574% and the 0.308% of training set). For the same reason, it has difficulties in recognising the classes `ENTERTAINMENT`, `MEDICINE-AESTHETICS` and `NATURE` (which are respectively the 2.937%, 2.035% and 2.645% of the training set). The two BERT-based systems, on the contrary, are less affected by this imbalance of the training set and their predictions reflect more the reality of the test set, even though, as STL-SVM, also MTL-BERT cannot recognise the `BIKES` and `TECHNOLOGY` classes.

In Figure 2 we report the distribution of the Age classes with respect to the Topic classes. Figure 2 (b) shows that in the STL-SVM's output the `0-19` age class is only related to the `ANIME` topic, the age `20-29` is related more or less with all the detected topics, the `30-39` class is mostly related to `SMOKE` and `MEDICINE-AESTHETICS`, the `40-49` class to the `METAL-DETECTING`, `AUTO-MOTO` and `SMOKE` topics and the `50-100` class mostly to `AUTO-MOTO`, `SPORTS` and `CELEBRITIES`. This distribution is quite far from the test set one and it seems that the relation between the class `0-19` and the topics is overestimated. In Figure 2 (c), which refers to MTL-BERT, we can note that authors classified as having age `20-29` are predicted to talk mostly about `ANIME`, `CELEBRITIES`, `NATURE` and `SPORTS` and are less related to `ENTERTAINMENT`, `MEDICINE-AESTHETICS` and `NATURE` topics than in STL-SVM's output; the relation between the `30-39` class and `ENTERTAINMENT` and `MEDICINE-AESTHETICS` categories on one hand, and `50-100` and `AUTO-MOTO`, `MEDICINE-AESTHETICS`, `METAL-DETECTING`, `NATURE` and `SMOKE` on the other is stronger than in STL-SVM's results. Also this distribution, though, is quite far from the test set's one, even if ages seem to be more distributed than in STL-SVM's output. As shown in Figure 2 (d), in STL-BERT's distribution, the age `0-19` seems mostly related to `TECHNOLOGY` and `ANIME`. The class `BIKES`, which has not been recognised by the other systems, is related to the classes `30-39`, `40-49` and, mostly, `50-100`. As regards the `20-29` class, its relations are quite similar to the ones found in the STL-SVM's

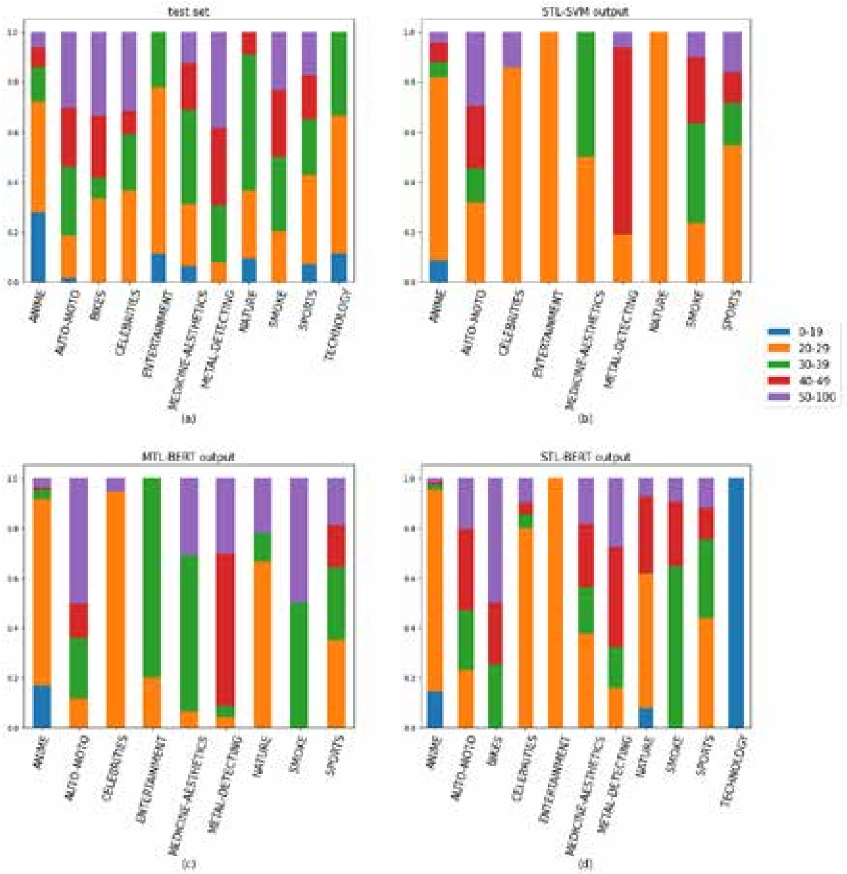


Figure 2: Task 1, Distributions of the Topic and Age dimensions in test set and classifiers' predictions.

results, except for the class `NATURE`, which is related also to the ages `0-19`, `40-49` and `50-100`. Also this distribution is quite far from the test's one. All the three distributions differ considerably from the test set because systems do not perform well enough in age prediction.

The distributions of the topics with respect to gender in the test set and the predictions are reported in Figure 3. As shown in the figure, all the three systems results relate the `F` class mostly to the `ANIME` topic, as it is also in the test set. In the STL-SVM's output, though, this relation seems to be overestimated. Moreover, in STL-SVM the `F` class, besides `ANIME`, is only related to a much lesser extent to `SMOKE`. The relation between `M` and `SMOKE` seems to be overestimated too with respect to the test set. As regards the `F` class in MLT-BERT and STL-BERT outputs, topics are more distributed than in STL-SVM, but the nearest to the test set's one is STL-BERT: MLT-BERT, in fact, seems to overestimate the relation between `F` and `BIKES` and `ENTERTAINMENT` and to underestimate the relation between `F` and `MEDICINE-AESTHETIC` and `SPORTS`. For what concerns the `M` class in MLT-BERT and STL-BERT distributions, we can state once again that the distribution which is closer to the test set one is given by STL-BERT: STL-SVM, MLT-BERT overestimates the relation between `M` and `SMOKE` and `NATURE`.

Task 2:

The results reported in Table 5 show that for Task 2a (gender prediction with topic control) the best classifier is MLT-BERT. In this subtask, BERT-based systems outperform in a significant way the system based on SVM.

As regards the Task 2b, consisting in the age prediction, the best metrics belong to the STL-BERT. In the age prediction the gap between all the systems' metrics is not very high. In this case, in which only the age dimension must be predicted, the best classifier is the one using a Single-Task Learning approach.

4 Conclusions

In this paper we reported the performances and the results of the systems we used to participate to the TAG-it task of EVALITA 2020. We com-

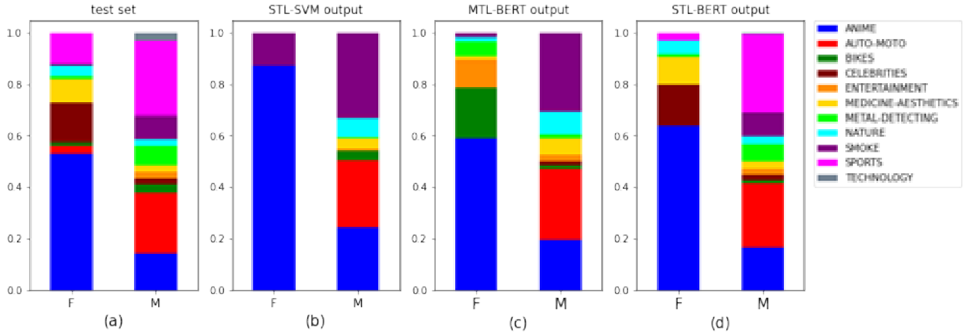


Figure 3: Task 1, Distributions of the Topic and Gender dimensions in test set and classifiers' predictions.

pared our systems' performances and noted that in the case in which the goal is to predict topic, age and gender dimensions at once, and in the case in which only the age must be predicted, the best classifier is the one developed using a Single-Task Learning approach and based on transformers. In the case in which the goal is the gender prediction only a Multi-task Learning approach combined with transformers have slightly better performances. These results prove that the proposed systems based on transformers, are more effective than traditional machine learning techniques in topic, age and gender classification achieving the state of the art for TAG-it shared task. Using deep pretrained language models on this task Multi-Task Learning does not provide any relevant boost of performances.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Andrea Cimino, Lorenzo De Mattei, and Felice Dell'Orletta. 2018. Multi-task learning in deep neural networks at EVALITA 2018. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Andrea Cimino, Dell'Orletta Felice, and Nissim Malvina. 2020. Tag-it@evalita2020: Overview of the topic, age, and gender prediction task for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Lorenzo De Mattei and Andrea Cimino. 2018. Multi-task learning in deep neural network for sentiment polarity and irony classification. In *Proceedings of the 2nd Workshop on Natural Language for Artificial Intelligence*, November.

Felice Dell'Orletta and Malvina Nissim. 2018. Overview of the evalita 2018 cross-genre gender prediction (gxg) task.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML*.

Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online, July. Association for Computational Linguistics.

Francisco Rangel, Fabio Celli, Paolo Rosso, Martin Potthast, Benno Stein, and Walter Daelemans. 2015. Overview of the 3rd author profiling task at pan 2015.

Jason D. M. Rennie and Ryan Rifkin. 2001. Improving multiclass text classification with the support vector machine.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Rob van der Goot, Nikola Ljubesic, Ian Matroos, Malvina Nissim, and Barbara Plank. 2018. Bleaching text: Abstract features for cross-lingual gender prediction.

TRACK
“SEMANTICS AND MULTIMODALITY”

DANKMEMES: Multimodal Artefacts Recognition

DANKMEMES @ EVALITA 2020: The Memeing of Life: Memes, Multimodality and Politics

Martina Miliani[1,2] and **Giulia Giorgi**[3] and **Ilir Rama**[3]
and **Guido Anselmi**[3] and **Gianluca E. Lebani**[4]

[1] University for Foreigners of Siena
[2] CoLing Lab, Department of Philology, Literature, and Linguistics, University of Pisa
[3] Department of Social and Political Sciences, University of Milan
[4] Department of Linguistics and Comparative Cultural Studies, Ca' Foscari University of Venice

martina.miliani@fileli.unipi.it, giulia.giorgi@unito.it
ilir.rama@unimi.it, guido.anselmi@unimi.it,
gianluca.lebani@unive.it

Abstract

DANKMEMES is a shared task proposed for the 2020 EVALITA campaign, focusing on the automatic classification of Internet memes. Providing a corpus of 2.361 memes on the 2019 Italian Government Crisis, DANKMEMES features three tasks: A) Meme Detection, B) Hate Speech Identification, and C) Event Clustering. Overall, 5 groups took part in the first task, 2 in the second and 1 in the third. The best system was proposed by the UniTor group and achieved a F_1 score of 0.8501 for task A, 0.8235 for task B and 0.2657 for task C. In this report, we describe how the task was set up, we report the system results and we discuss them.

1 Introduction

Internet memes are understood as "pieces of culture, typically jokes, which gain influence through online transmission" (Davison, 2012). Specifically, a meme is a multimodal artefact manipulated by users, who merges intertextual elements to convey an ironic message. Featuring a visual format that includes images, texts or a combination of them, memes combine references to current events or relatable situations and pop-cultural references to music, comics and movies (Ross and Rivers, 2017).

The pervasiveness of meme production and circulation across different platforms increases the necessity to handle massive quantities of visual data (Tanaka et al., 2014) by leveraging on automated approaches. Efforts in this direction focused on the generation of memes (Peirson V and Tolunay, 2018; Gonçalo Oliveira et al., 2016) and on automated sentiment analysis (French, 2017), while stressing the need for a multimodal approach able to contextually consider both visual and textual information (Sharma et al., 2020; Smitha et al., 2018).

As manual labelling becomes unfeasible on a large scale, scholars require tools able to classify the huge amount of memetic content continuously produced on the web. The main goal of our shared task is to evaluate a range of technologies that can be used to automatize the process of meme recognition and sorting with an acceptable degree of reliability.

2 Task Description

The DANKMEMES task, presented at the 2020 EVALITA campaign (Basile et al., 2020), encompasses three subtasks, aimed at: detecting memes (Task A), detecting the hate speech in memes (Task B) and clustering memes according to events (Task C). Participants could decide to take part in one or more of these tasks, with the only recommendation that Task 1 functions as the compulsory preliminary step for the other two tasks.

Task A: Meme Detection. The lack of consensus around what defines a meme (Shifman, 2013) led to different definitions, focusing on circulation (Davison, 2012; Dawkins, 2016), formal features (Milner, 2016), or content (Gal et al., 2016; Kno-

---

bel and Lankshear, 2007). For this dataset, manual coding focused both on formal aspects (such as layout, multimodality and manipulation) as well as content, e.g. ironic intent (Giorgi and Rama, 2019); the exponential increase in visual production, however, warrants an automated approach, which might be able to further tap into stable and generalizable aspects of memes, considering form, content and circulation. Given the dataset minus the variable strictly related to memetic status, participants must provide a binary classification, distinguishing memes (1) from non memes (0).

Task B: Hate Speech Identification. Hate speech became a relevant issue for social media platforms. Even though the automatic classification of posts may lead to censorship of non-offensive content (Gillespie, 2018), the use of machine learning techniques became more and more crucial, since manual filtering is a very time consuming task for the annotators (Zampieri et al., 2019b). Recent studies have also shown that multimodal analysis is fundamental in such a task (Sabat et al., 2019). In this direction, SemEval 2020 proposed the "Memotion Analysis" among its tasks, to classify sarcastic, humorous, and offensive meme (Sharma et al., 2020). This kind of analysis assumes a specific relevance when applied to political content. Memes about political topics are a powerful tool of political criticism (Plevriti, 2014). For these reasons, the proposed task aims at detecting memes with offensive content. Following Zampieri (2019a) definition, an offensive meme contains any form of profanity or a targeted offense, veiled or direct, such as insults, threats, profane language or swear words. Thus, the second task consists in a binary classification, where systems have to predict whether a meme is offensive (1) or not (0).

Task C: Event Clustering. Social media react to the real world, by commenting in real-time to mediatised events in a way that disrupts traditional usage patterns (Al Nashmi, 2018). The ability to understand which events are represented and how, then, becomes relevant in the context of an hyper-productive Internet.

The goal of the third subtask is to cluster a set of memes that may be or may be not related to the 2019 Italian government crisis into five event categories (see Table 1).

Participants' goal is to apply supervised techniques to cluster the memes, so that memes pinpointing to the same events are classified in the same cluster.

Label	Description
0	Residual category
1	Beginning of the government crisis
2	Conte's speech and beginning of consultations
3	Conte is called to form a new government
4	5SM holds a vote on the platform Rousseau

Table 1: Categories for Task C: Event Clustering.

3 Dataset

3.1 Composition of the dataset

The DANKMEMES dataset is comprised of 2,361 images (for each subtask a specific dataset was provided), automatically extracted from Instagram through a Python script aimed at the hashtag related to the Italian government crisis ("#crisidigoverno"). The corpus includes 367 offensive political memes unrelated to the government crisis, and aimed at augmenting and balancing the dataset for task 2.

3.2 Annotation of the dataset

For each image of the dataset we provide both the name of the .jpg image file, the date of publication and the engagement, i.e. the number of comments and likes of the post. The dataset also includes image embeddings. The vector representations are computed employing ResNet (He et al., 2016), a state-of-the-art model for image recognition based on Deep Residual Learning. Providing such image representations allows the participants to approach these multimodal tasks focusing primarily on its NLP aspects (Kiela and Bottou, 2014). The annotation process involved two Italian native speakers, who study memes at an academic level, and focused on detecting and labelling 7 relevant categories:

- **Macro status**: refers to meme layouts and their relation to diffused, conventionalised formats called macros. The category has 0 and 1 as labels, where the value 1 represents well-known memetic frames, characters and layouts (e.g. Pepe the Frog). The identification of macros relied both on external sources (e.g. the website "Know Your Meme") and the annotators' literacy on memes.

- **Picture manipulation**: entails the degree of visual modification of the images. Non-manipulated or low impact changes are labeled 0 (e.g. the addition of a text or a logo). Heavily manipulated, impactful changes (e.g. images edited to include political actors) are labeled 1.

- **Visual actors**: the political actors (i.e. politicians, parties' logos) portrayed visually, regardless whether edited into the picture or portrayed in the original image.

- **Text**: the textual content of the image has been extracted through optical character recognition (OCR) using Google's Tesseract-OCR Engine, and further manually corrected.

- **Meme**: binary feature, where 0 represents non meme images and 1 meme images. This is the target label for Task A.

- **Hate Speech**: binary feature only for memes. It differentiates memes with offensive language (1) from non offensive memes (0). This is the target label for Task B.

- **Event**: it is a feature only for meme images, categorizing them according to 4 events (described in 4), plus a residual category labeled as 0. This is the target label for Task C.

The final inter-annotator agreement (IAA) has been calculated by two of the authors on a subset of the dataset through Krippendorff's alpha (Krippendorff, 2018). Four features have been considered: Macro status ($\alpha = 0.755$), Picture manipulation ($\alpha = 0.930$), Hate Speech ($\alpha = 0.741$) and Meme ($\alpha = 0.884$). Other features were either objective (i.e. Visual and textual actors) or inferred from external data (i.e. events).

Participants were allowed to use external resources, lexicons or independently annotated data. Given that, although we provided ResNet image embeddings, participants could make use of any other image representations.

3.3 Training and Test Data

The initial dataset was split into three datasets, one for each task, structured as follows:

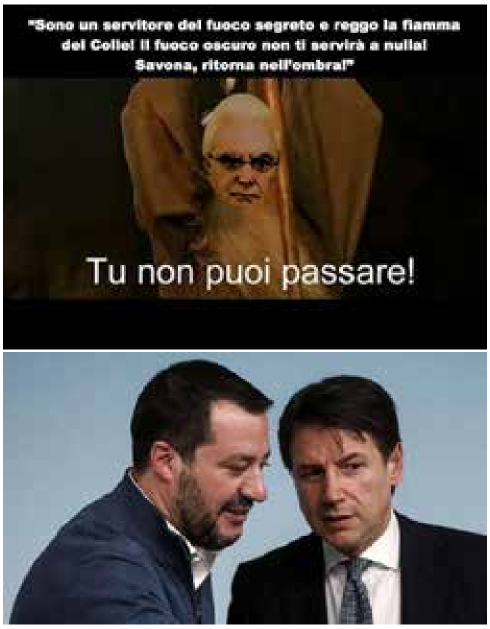

"Sono un servitore del fuoco segreto e reggo la fiamma del Colle! Il fuoco oscuro non ti servirà a nulla! Savona, ritorna nell'ombra!"
Tu non puoi passare!

Figure 1: Two examples from the dataset for Meme Detection: the image at the top is a meme, whereas the image at the bottom is not a meme.

Dataset for Meme Detection (Task A). The whole dataset counts 2,000 images, half memes and half not (see Figure 1 for an example). We split the dataset into training and test sets, in a proportion of 80-20% of items. Table 2 represents the format of the training dataset. The test dataset has been provided without gold labels, i.e. without the "Meme" attribute.

Dataset for Hate Speech Identification (Task B). The whole dataset counts 1,000 memes (see Figure 2 for an example). We split the dataset into training and test sets, in a proportion of 80-20% of items. Table 3 represents the format of the training dataset. The test dataset has been provided without the gold label "Hate Speech" for testing purposes.

Dataset for Event Clustering (Task C). The whole dataset counts 1,000 memes (see Figure 3 for an example). We split the dataset into training and test sets, in a proportion of 80-20% of items. Table 4 shows the format of the training set. The test set has been provided without gold labels (i.e. without the "Event" attribute) for testing purposes.

3.4 Data release

Both the training and the test sets were released on our website and protected with a password. As described in Section 3.3, the development data con-

File	Engagement	Date	Manip.	Visual	Text	Meme
1.jpg	21,053	22/08/19	1	Conte	aiuto	0
56.jpg	114	22/08/19	0	Salvini	alle solite	1

Table 2: An excerpt from the dataset for Task A, Meme Detection.

File	Engagement	Manip.	Visual	Text	Hate Speech
62.jpg	21,053	1	Conte	aiuto	0
114.jpg	12,572	1	Salvini	merdman	1

Table 3: An excerpt from the dataset for Task B, Hate Speech Identification.

File	Engagement	Date	Macro	Manip.	Visual	Text	Event
43.jpg	21,053	22/08/19	1	1	Conte	aiuto	1
23.jpg	114	22/08/19	1	0	Salvini	alle solite	0
114.jpg	12,572	25/08/19	0	1	Salvini	merdman	2

Table 4: An excerpt from the dataset for Task C, Event Clustering.

Team Name	Affiliation	Task
DMT	RN Podar School	A
Keila	Dipartimento di Matematica e Informatica di Perugia	A
UniTor	Università degli Studi di Roma ”Tor Vergata”	A,B,C
UPB	Univesity Politehnica of Bucharest	A,B
SNK	ETI3	A

Table 5: Participants along with their affiliations and the tasks they participated in.

sisted of three distinct datasets, one for each task. The participants could download a distinct folder for each task, which contained:

- A UTF-8 encoded comma separated “.csv” file with 800 items (1,600 for task A), containing the metadata described in Section 3.3;
- A folder containing the images in .jpg format;
- A .csv file containing the relative image embeddings.

As for the test data, we released three folders whose structure is similar to the ones of the training sets. Each folder for the train sets contains:

- A UTF-8 encoded comma separated “.csv” file with 200 items (400 for Task A), which features the same metadata of the corresponding training set minus the golden label (i.e. “Meme” for Task A, “Hate speech” for Task B and “Event” for Task C);
- A folder containing the images in .jpg format;
- A .csv file containing the relative image embeddings.

All material was released for non-commercial research purposes only under a Creative Common license (BY-NC-ND 4.0). Any use for statistical, propagandistic or advertising purposes of any kind is prohibited. It is not possible to modify, alter or enrich the data provided for the purposes of redistribution.

4 Evaluation Measures

For all tasks, the models have been evaluated with $Precision$, $Recall$, and F_1 scores defined as follows:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

where TP are true positives, and FN and FP are false negatives and false positives, respectively. We computed $Precision$, $Recall$, and F_1

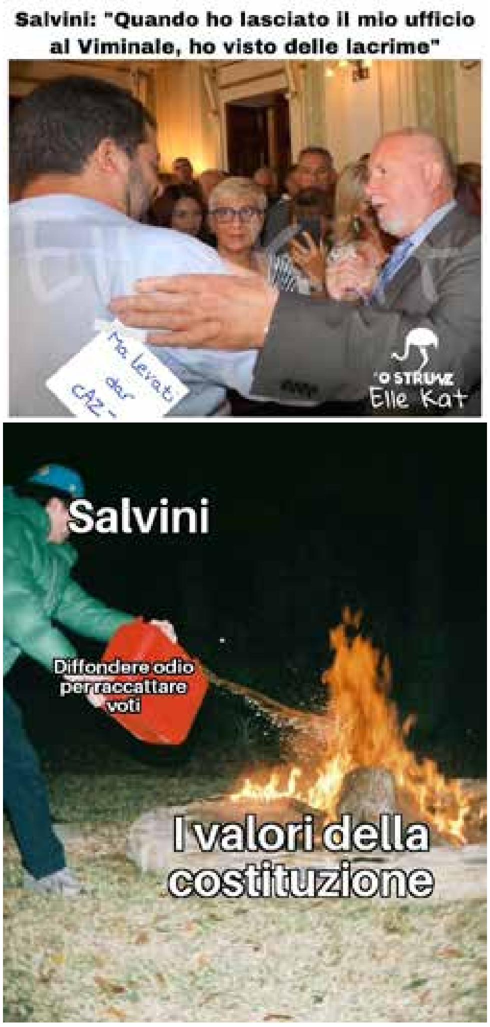

Salvini: "Quando ho lasciato il mio ufficio al Viminale, ho visto delle lacrime"
Ma levati dar CAZ-
'O STRUNZ
Elle Kat
Salvini
Diffondere odio per raccattare voti
I valori della costituzione

Figure 2: Two examples from the dataset for Hate Speech Identification: the meme at the top is classified as hate speech content, whereas the meme at the bottom is not.

for Task A and Task B considering only the positive class. For what concerns Task C, which is a multiclass classification task, we computed the performance for each class and then calculated the macro-average over all classes.

Different baselines were used for the different tasks:

Task A: Meme Detection. The baseline is given by the performance of a random classifier, which labels 50% of images as meme.

Task B: Hate Speech Identification. The baseline is given by the performance of a classifier labeling a meme as offensive when the meme text contains at least a swear word[1].

Task C: Event Clustering. The baseline is given by the performance of a classifier labeling every meme as belonging to the most numerous class (i.e. the residual one).

5 Participants and Results

In total, 16 teams registered for DANKMEMES, and five of them participated in at least one of the tasks: DankMemesTeam (DMT) (Setpal and Sarti, 2020), Keila, UPB (Vlad et al., 2020), SNK (Fiorucci, 2020), and UniTor (Breazzano et al., 2020).

All of the 5 teams participated in Task A, while 2 teams participated in Task B and 1 in Task C. Participants could submit up to two runs per task: all of the teams did so consistently across tasks, with the exception of one team submitting a single run in Task A. This amounts to 9 runs for Task A, 4 for Task B and 2 for Task C, as detailed in Table 5.

Task A: Meme Detection. Task A consisted in differentiating between a meme and a not-meme. Five teams presented a total of 9 runs, as detailed in Table 6. The best scores have been achieved by the UniTor team with an F_1-measure of 0.8501 (with a Precision score of 0.8522 and a Recall measure of 0.848). The SNK and UPB teams followed closely, but all teams consistently showed a drastic improvement over the baseline.

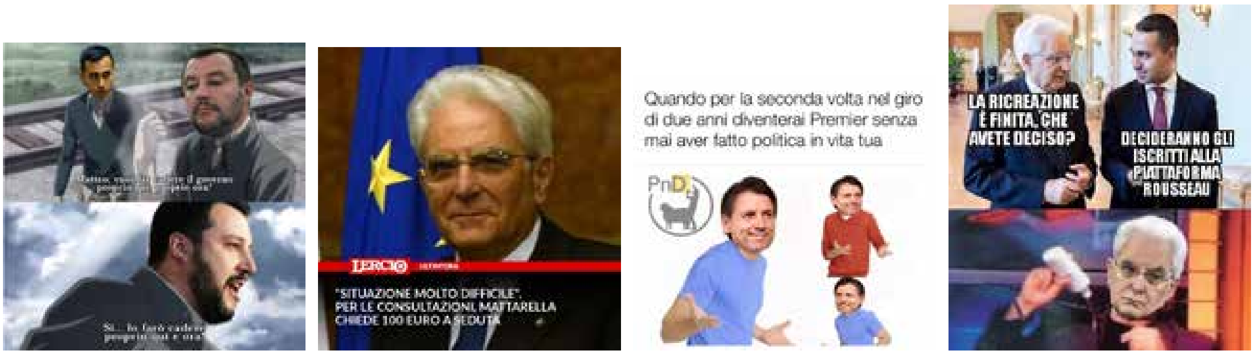
Team	Run	Recall	Precision	F_1
Unitor	2	0.8522	0.848	0.8501
SNK	1	0.8515	0.8431	0.8473
UPB	2	0.8543	0.8333	0.8437
Unitor	1	0.839	0.8431	0.8411
SNK	2	0.8317	0.848	0.8398
UPB	1	0.861	0.7892	0.8235
DMT	1	0.8249	0.7157	07664
Keila	1	0.8121	0.6569	0.7263
Keila	2	0.7389	0.652	0.6927
baseline	1	0.525	0.5147	0.5198

Table 6: Results of Task A.

Task B: Hate Speech Identification. Task B consisted in the identification of whether a meme

---

[1] The list of swear words was downloaded from: `https://www.freewebheaders.com/italian-bad-words-list-and-swear-words/` (last access: 2nd November 2020).


LERCIO
"SITUAZIONE MOLTO DIFFICILE", PER LE CONSULTAZIONI MATTARELLA CHIEDE 100 EURO A SEDUTA
Quando per la seconda volta nel giro di due anni diventerai Premier senza mai aver fatto politica in vita tua
PnD
LA RICREAZIONE È FINITA, CHE AVETE DECISO?
DECIDERANNO GLI ISCRITTI ALLA PIATTAFORMA ROUSSEAU

(a) (b) (c) (d)

Figure 3: Examples of memes from the dataset for Event Clustering task. Each meme refers to an event: (a) Beginning of the governement crisis; (b) Conte's speech and beginning of consultations; (c) Conte is called to form a new government; (d) 5SM holds a vote on the platform Rousseau.

is offensive or not. As detailed in Table 7, 2 teams participated in this task for a total of 4 runs (2 each). The best scores are achieved by the UniTor team for the F_1-measure at 0.823 and the Recall score of 0.8667, while the UPB team scored the best Precision measure at 0.8056. The scores improve over the baseline consistently across teams for what concerns the Recall score and the F_1-measure, while the Precision measure was not reached by any participant.

Team	Run	Recall	Precision	F_1
UniTor	2	0.7845	0.8667	0.8235
UniTor	1	0.7686	0.8857	0.823
UPB	1	0.8056	0.8286	0.8169
UPB	2	0.8333	0.7143	0.7692
baseline	1	0.8958	0.4095	0.5621

Table 7: Results of Task B.

Task C: Event Clustering. Task C consisted in clustering memes into 5 events using supervised classification. As seen in Table 8, a single team participated with 2 runs: the best score is therefore that of the UniTor team, with an F_1-score of 0.2657.

Team	Run	Recall	Precision	F_1
UniTor	1	0.2683	0.2851	0.2657
UniTor	2	0.2096	0.2548	0.2183
baseline	1	0.096	0.2	0.1297

Table 8: Results of Task C.

6 Discussion

We compare the participating systems according to the following main dimensions: classification framework, exploitation of available features, multimodality of the adopted approaches, exploitation of further annotated data, and use of external resources. Since this is the first task about memes within the EVALITA campaign, we could not compare the obtained results with those achieved in any previous edition. A task about memes, Memotion, has been organized under SemEval 2020 (Sharma et al., 2020). However, the Memotion subtasks (Sentiment Classification, Humor Classification, and Scales of Semantic Classes) are quite different from those presented in DANKMEMES, and the results are hardly comparable.

System architecture. All the submitted runs to DANKMEMES leverage on neural networks, including very simple but equally efficient architectures. Multi-Layer Perceptrons (MLP) have been adopted by UniTor and SNK, ranked first and second in the the Meme Detection task, respectively. UPB adopted a Vocabulary Graph Convolutional Network (VGCN) combined with BERT contextual embeddings for text analysis. This team employed this architectural design within a Multi-Task Learning (MTL) technique, based on two main neural network components: one for the text and the other for the image analysis. The outputs of these two elements were concatenated and used to feed a Dense layer. The system in DMT is composed of three 8-layer feed-forward networks, each taking as input a different image vector representation. Finally, Keila exploited Convolutional Neural Networks (CNN) in each of the submitted run.

External resources. All the presented models employed external resources to feed their neural architecture with image and text representations. The text contained in the images was encoded by using different flavours of word embeddings. Most of the participants exploited one of the available BERT contextual embeddings model for the Italian language (AlBERTo, UmBERTo, or GilBERTo). However, with its first run, SNK achieved the second position in the Meme Detection task using the pre-trained FastText embeddings for the Italian language. Similarly, Keila adopted pre-trained Word2Vec for the Italian language, though achieving lower results. As for the visual channel, the DANKMEMES datasets provided a state-of-the-art representation of images, obtained with the ResNet50 architecture. Most of the participants experimented the use of other image vector representations as well: DMT used three different image vector: AlexNet, ResNet, and DenseNet; UniTor and UPB examined several models, among which: EfficientNET, VGG-16, YOLOv4, ResNet50, and ResNet152. UniTor chose EfficientNet for their final models, while UPB based their ssystems on ResNet50 and ResNet152.

Multimodality. The exploitation of both images and text turned out to be fundamental for the task of Meme Detection. Since memes adhere to specific visual conventions, participants tried to exploit visual data at their best. The first run of UniTor only relied on an image classifier, whereas DMT exploited the information resulting from three different image classification models, then combined with word embeddings. Nevertheless, the best results were obtained by the combination of text and image information. In its second run, UniTor concatenated the image representation returned by their first model with pre-trained contextual word embeddings fine-tuned on DANKMEMES data. Similarly, SNK and UPB leveraged both textual and image data. Keila was the only participant who did not combine text and image information in any of the submitted runs. For what concerns the second task, the first UniTor run only relied on textual data and was slightly overcame only by their second run. As observed by the team, in the Hate Speech Identification task, textual data heavily impact the classification results. Finally, UPB combined both image and textual data for this task.

Data Augmentation. Several participants chose to adopt a data augmentation technique. UniTor successfully manipulated the provided images by horizontally mirroring them. On the contrary, DMT created nine versions of each image at first, editing brightness, rotation, and zoom, but then dropped them due to the overfitting caused by the unmodified metadata associated with each image. Keila augmented textual data by firstly translating the image texts in English and then back to Italian. Regarding the second task on Hate Speech Identification, UniTor trained for a few epochs the UmBERTo embeddings on a dataset made available within the Hate Speech Detection (HaSpeeDe) task (Bosco et al., 2018) before training it on the DANKMEMES dataset.

Exploited features. SNK encoded and concatenated in a single vector picture manipulation, visual, and engagement, along with the sentence and the image representation of each meme. Keila employed engagement and manipulation features as well. DMT normalized engagement and represented dates with the count of days from a selected reference date. Along with the other provided data, temporal features were exploited by UPB as well, through the computation of complementary sine and cosine distances, in order to preserve the cyclic characteristics of days and months. Finally, UniTor relied only on visual and textual information.

Event Clustering. The goal of this task was to assign each meme to the event it refers to. Only UniTor participated in this task, modeling it as a classification problem in two distinguished runs. The first model only exploited textual data representation provided by the Transformer architecture to feed the MLP classifier. Furthermore, UniTor submitted a second run. The team mapped the original classification problem, which counted five different labels (each corresponding to an event) over a binary classification one. After pairing a meme to each event, a pair was labeled as positive if the association was correct, negative otherwise. However, this run did not overpass the first one, the outcome of which doubled the provided baseline.

7 Final Remarks

The paper describes a task for the detection and analysis of memes in the Italian language.

DANKMEMES is the first task of this kind in the EVALITA campaign. Although memes are widespread on the Web, it is still hard to define them precisely. However, DANKMEMES highlighted the fundamental role of multimodality in memes detection, mainly the combined use of texts and images for their classification. Therefore, we could say that memes share peculiar linguistic features, other than conventional layouts. Future work will focus on the extension of the dataset, which showed some limitations, especially for its reduced size and for the unbalanced representation of some events. This is due to the difficulty of meme collection, especially when filtered in relation to a specific event (e.g., the 2019 Italian government crisis).

References

Eisa Al Nashmi. 2018. From selfies to media events: How instagram users interrupted their routines after the charlie hebdo shootings. *Digital Journalism*, 6(1):98–117.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Cristina Bosco, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. 2018. Overview of the evalita 2018 hate speech detection task. In *EVALITA 2018-Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*, pages 1–9.

Claudia Breazzano, Edoardo Rubino, Danilo Croce, and Roberto Basili. 2020. Unitor @ dankmemes: Combining convolutional models and transformer-based architectures for accurate meme management. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Patrick Davison. 2012. The language of internet memes. *The Social Media Reader*, pages 120–134.

Richard Dawkins. 2016. *The Selfish Gene*. Oxford University Press.

Stefano Fiorucci. 2020. Snk @ dankmemes: Leveraging pretrained embeddings for multimodal meme detection. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Jean French. 2017. Image-based memes as sentiment predictors. In *2017 International Conference on Information Society (i-Society)*.

Noam Gal, Limor Shifman, and Zohar Kampf. 2016. "it gets better": Internet memes and the construction of collective identity. *New media & society*, 18(8):1698–1714.

Tarleton Gillespie. 2018. *Custodians of the Internet: Platforms, content moderation, and the hidden decisions that shape social media*. Yale University Press.

Giulia Giorgi and Ilir Rama. 2019. "one does not simply meme". framing the 2019 italian government crisis through memes. In *La comunicazione politica nell'ecosistema dei media digitali Convegno dell'Associazione Italiana di Comunicazione Politica (ASSOCOMPOL)*.

Hugo Gonçalo Oliveira, Diogo Costa, and Alexandre Pinto. 2016. One does not simply produce funny memes! – explorations on the automatic generation of internet humor. In *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2016)*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Douwe Kiela and Léon Bottou. 2014. Learning image embeddings using convolutional neural networks for improved multi-modal semantics. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 36–45.

Michele Knobel and Colin Lankshear. 2007. Online memes, affinities, and cultural production. *A new literacies sampler*, 29:199–227.

Klaus Krippendorff. 2018. *Content analysis: An introduction to its methodology*. Sage publications.

Ryan M. Milner. 2016. *The World Made Meme: Public Conversations and Participatory Media*. MIT Press.

Abel L. Peirson V and E. Meltem Tolunay. 2018. Dank learning: Generating memes using deep neural networks. *CoRR*, abs/1806.04510.

Vasiliki Plevriti. 2014. Satirical user-generated memes as an effective source of political criticism, extending debate and enhancing civic engagement.

Andrew S. Ross and Damian J. Rivers. 2017. Digital cultures of political participation: Internet memes and the discursive felegitimization of the 2016 us presidential candidates. *Discourse, Context & Media*, 16:1–11.

Benet Oriol Sabat, Cristian Canton Ferrer, and Xavier Giro-i Nieto. 2019. Hate speech in pixels: Detection of offensive memes towards automatic moderation. *arXiv preprint arXiv:1910.02334*.

Jinen Setpal and Gabriele Sarti. 2020. Dankmemesteam @ dankmemes: Archimede: A new model architecture for meme detection. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Chhavi Sharma, Deepesh Bhageria, William Scott, Srinivas PYKL, Amitava Das, Tanmoy Chakraborty, Viswanath Pulabaigari, and Bjorn Gamback. 2020. Semeval-2020 task 8: Memotion analysis – the visuo-lingual metaphor! *arXiv preprint arXiv:2008.03781*.

Limor Shifman. 2013. Memes in a digital world: Reconciling with a conceptual troublemaker. *Journal of computer-mediated communication*, 18(3):362–377.

E. S. Smitha, Selvaraju Sendhilkumar, and G. S. Mahalaksmi. 2018. Meme classification using textual and visual features. In *Computational Vision and Bio Inspired Computing*, pages 1015–1031.

Emi Tanaka, Timothy Bailey, and Uri Keich. 2014. Improving meme via a two-tiered significance analysis. *Bioinformatics*, 30:1965–1973, 03.

George-Alexandru Vlad, George-Eduard Zaharia, Dumitru-Clementin Cercel, and Mihai Dascalu. 2020. Upb @ dankmemes: Italian memes analysis: Employing visual models and graph convolutional networks for meme identification and hate speech detection. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019a. Predicting the type and target of offensive posts in social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1415–1420.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019b. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86.

SNK @ DANKMEMES: Leveraging Pretrained Embeddings for Multimodal Meme Detection

Stefano Fiorucci
Machine learning engineer
ETI3
stefano.fiorucci@virgilio.it

Abstract

English. In this paper, we describe and present the results of meme detection system, specifically developed and submitted for our participation to the first subtask of DANKMEMES (EVALITA 2020). We built simple classifiers, consisting in feed forward neural networks. They leverage existing pretrained embeddings, both for text and image representation. Our best system (SNK1) achieves good results in meme detection (F1 = 0.8473), ranking 2nd in the competition, at a distance of 0.0028 from the first classified.

Italiano. *In questo articolo, descriviamo e presentiamo i risultati di un sistema di individuazione dei meme, ideato e sviluppato per partecipare al primo subtask di DANKMEMES (EVALITA 2020). Abbiamo realizzato dei semplici classificatori, costituiti da una rete neurale feed-forward: essi sfruttano embedding preesistenti, per la rappresentazione numerica di testo e immagini. Il nostro miglior sistema (SNK1) raggiunge buoni risultati nell'individuazione dei meme (F1 = 0.8473) e si è classificato secondo nella competizione, ad una distanza di 0.0028 dal primo classificato.*

1 System description

1.1 General approach and tools

DANKMEMES (Miliani et al., 2020) is a task for meme recognition and hate speech/event identification in memes and is part of the EVALITA 2020 evaluation campaign (Basile et al., 2020).

For our participation to the first subtask of DANKEMES, we built simple classification models for meme detection.

The main challenge is to effectively combine textual and image inputs. We tried to exploit the ability of pretrained embedding to represent the information present in text and images, paying a limited computational cost.

To quickly build various prototypes of neural networks, we used Uber Ludwig framework (Molino et al., 2019): a toolbox built on top of TensorFlow, which facilitates and speeds up the training and testing of various models.

We trained our models using Google Colaboratory, a hosted Jupyter notebook service, which provides free access to GPUs, with some resource and time limitations.

1.2 Features

1.2.1 DANKMEMES dataset

The dataset provided for the first subtask has the following features:

- **File**: the name of the .jpg image file.
- **Date**: when the image has first been posted on Instagram.
- **Picture manipulation**: entails the degree of visual modification of the images. Non-manipulated or low impact changes are labeled 0. Heavily manipulated, impactful changes are labeled 1.
- **Visual actors**: the political actors (i.e. politicians, parties' logos) portrayed visually, regardless whether edited into the picture or portrayed in the original image.
- **Engagement**: the number of comments and likes of the image.
- **Text**: the textual content of the image.

- **Meme**: binary feature, where 0 represents non meme images and 1 meme images. This is the target label.

The dataset also includes **image embeddings**.

1.2.2 Feature selection and preprocessing

We discarded Date feature, because it seems irrelevant for meme detection.

Picture manipulation and Meme are simple binary features and do not require preprocessing.

We chose to scale Engagement feature, using min-max normalization.

Visual actors feature was preprocessed using Ludwig approach for sets. We report an extract of the official framework documentation[1]:

“Set features are expected to be provided as a string of elements separated by whitespace.

The string values are transformed into a binary valued matrix of size n x l (where n is the size of the dataset and l is the minimum of the size of the biggest set and a max_size parameter) [...]

The way sets are mapped into integers consists in first using a tokenizer to map from strings to sequences of set items. Then a dictionary of all the different set item strings present in the column of the dataset is collected, then they are ranked by frequency and an increasing integer ID is assigned to them from the most frequent to the most rare (with 0 being assigned to PAD used for padding and 1 assigned to UNK item).”

1.2.3 Text representation

For text representation, we chose to use pretrained word embeddings for the Italian language.

Our first model used fastText word representations (Bojanowski et al., 2016): non-contextual word embeddings. fastText word embeddings rely on subword information (bag of character n-grams) and thus provide valid representations for rare, misspelled or out-of-vocabulary words. Particularly, we used word vectors for the Italian language officially distributed in 2018 (Grave et al., 2018). Word embeddings are trained on Common Crawl and Wikipedia, using CBOW with position-weights, in dimension 300, with character n-grams of length 5, a window of size 5 and 10 negatives. We calculated the sentence vectors starting from the word vectors and using get_sentence_vector method of fastText python wrapper: each word vector is divided by its L2 norm and then averaged. Obtained sentence vector has dimension 300.

Our second classifier used BERT word representations (Devlin et al., 2018): context-based word embeddings. BERT model uses word-piece tokenization: therefore it too provides embeddings for unseen words. In particular, we used GilBERTo[2], an Italian pretrained language model based on Facebook RoBERTa architecture and CamemBERT text tokenization approach; it was trained with the subword masking technique for 100k steps managing 71GB of Italian text with more than 11 billion words. As an interface for this language model, we used python library HuggingFace’s Transformers (Wolf et al., 2019). To obtain sentence vectors, we took the output from the [CLS] token, which is prepended to the sentence during the preprocessing phase and is typically used for classification tasks; undoubtedly, there are also other methods for extracting sentence embeddings from BERT models that may prove more effective. Obtained sentence vector has dimension 768.

1.2.4 Image representation

For image representation, we used the embeddings provided in DANKMEMES dataset. The vector representations are computed employing ResNet (He et al., 2016), a state-of-the-art model for image recognition based on Deep Residual Learning. Every image vector has dimension 2048.

1.3 System architecture

Figure 1 shows a block diagram of system architecture, which is very simple. Picture manipulation, Visual actors, Engagement, Image vector and Sentence vector (obtained from word embedding) were combined by concatenation. The resulting multimodal feature vector was fed as input into a feed-forward neural network with two hidden layers of 256 and 16 neurons respectively, with a ReLU activation function. The last single neuron predicts whether the image is a meme or not.

2 Experiments and results

2.1 Experimental settings

To train our neural networks, we chose cross-entropy loss as the objective function. As defined in the subtask, the metrics of interest are precision, recall and F1 score. In the following, all metrics

---

[1] `https://ludwig-ai.github.io/ludwig-docs/user_guide/#set-features-preprocessing`

[2] `https://github.com/idb-ita/GilBERTo`

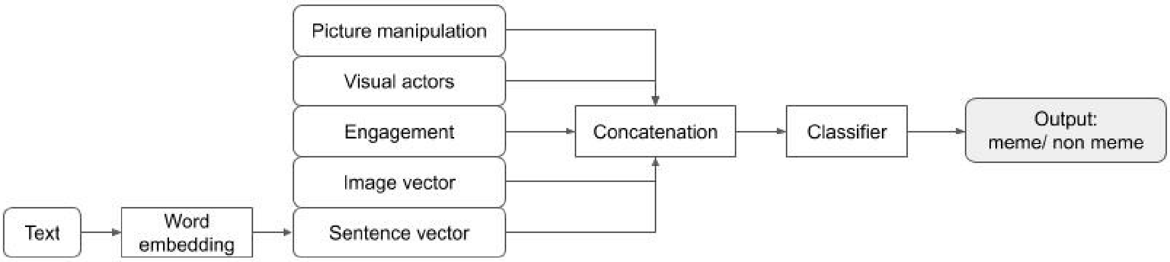


Figure 1: System architecture

reported were calculated using the officially provided evaluation script[3].

We used Adam optimizer with the following parameters: $\beta 1 = 0.9$, $\beta 2 = 0.999$, $\varepsilon = 10^{-8}$. We set an early stop of 5 epochs, in order to avoid overfitting.

Hyperparameter optimization was manually conducted and we tried various combinations of learning rate and batch size: our final models have learning rate of 10^{-5} and batch size of 10.

During our experiments, we studied the impact of a multimodal analysis, compared to using language or vision only.

We trained various models, including different combinations of basic features (Picture manipulation, Visual actors and Engagement), text representation (fastText or GilBERTo) and image representation (ResNet).

2.2 Results

Model	**Pr**	**Re**	**F1**
random baseline	0.525	0.5147	0.5198
Basic Features	**0.8732**	0.6078	0.7168
BF+fastText	0.8253	0.6716	0.7405
BF+GilBERTo	0.7685	0.7647	0.7666
BF+ResNet	0.8341	0.8382	0.8362
BF+fastText+ ResNet (**SNK1**)	0.8515	0.8431	**0.8473**
BF+GilBERTo+ ResNet (**SNK2**)	0.8317	**0.848**	0.8398

Table 1: Experimented models

We observe that basic features are quite informative: the model based only on them far outperforms the random baseline.

Models based on basic features and visual representations perform meme detection well. It should be noted that unimodal vision models perform significantly better than textual models. As Sabat et al. (2019) pointed out, an obvious reason is that the dimensionality of the image representation (2048) is much larger than the linguistic one (fastText: 300; GilBERTo: 768), so it has the capacity to encode more information. It would be interesting to conduct further experiments to investigate less obvious motivations and understand if the image representation actually conveys features of the visual scene, which are specific and distinctive of a meme.

As shown by Beskow et al. (2019), multimodal classifiers are considerably better than textual models and provide some improvement over unimodal vision models, which nevertheless provide solid performance in meme detection.

Team + Run	**Pr**	**Re**	**F1**
A2	0.8522	**0.848**	**0.8501**
SNK1	0.8515	0.8431	0.8473
B2	0.8543	0.8333	0.8437
A1	0.839	0.8431	0.8411
SNK2	0.8317	**0.848**	0.8398
B1	**0.861**	0.7892	0.8235
...			
baseline	0.525	0.5147	0.5198

Table 2: DANKMEMES subtask 1 results table

With reference to the competition, model SNK1 (Basic features + fastText + ResNet) ranked 2nd, at a short distance from the first classified. Model SNK2 (Basic features + GilBERTo + ResNet) ranked 5th.

---

[3] `https://github.com/gianlucalebani/dankmemes2020`

3 Discussion and conclusion

In this paper, we have presented simple multimodal systems for meme detection, based on a neural network classifier; they leverage existing pretrained embeddings to represent both text and image. Our systems achieve good performance, providing improvements over unimodal classifiers. In the first subtask of DANKMEMES (EVALITA 2020), our models ranked 2nd and 5th.

Based on our experiments, it is observed that pre-trained embeddings can be used effectively and with little effort to represent information conveyed by visual and textual components. While we haven't explicitly included irony or other distinctive aspects derived from text or image among the features, it is understood that the vectors generated by the embeddings express them implicitly.

Starting from the simple model used, it could be interesting to conduct in-depth analyzes to understand which of the basic features are most important. Furthermore, we could build saliency maps (Simonyan et al., 2013) to understand which areas of the images are most relevant for the meme detection task.

The proposed model could be improved. With more time and computational resources, a broader experimentation campaign could be conducted, using Bayesian hyperparameter optimization; we could try different numbers of neurons in hidden layers and other neural network architectures. To improve the classifier without much effort, we could also make an ensemble of our best performing models.

In our classifier, we used BERT powerful language model to get text vectors. We could do BERT fine tuning, in order to obtain better textual embedding, aimed at meme detection task.

Finally, to overcome the limits of this simple model, we could look for a more explicit way to encode the irony present in the text, drawing inspiration from IronITA (Cignarella et al., 2018).

References

Valerio Basile, Danilo Croce, Maria Di Maro and Lucia C. Passaro 2020. *EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian.* Valerio Basile, Danilo Croce, Maria Di Maro and Lucia C. Passaro (eds.). Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)

David Beskow, Sumeet Kumar and Kathleen Carley 2019. *The Evolution of Political Memes: Detecting and Characterizing Internet Memes with Multimodal Deep Learning* Information Processing & Management, volume 57

Piotr Bojanowski, Edouard Grave, Armand Joulin and Tomas Mikolov 2016. *Enriching Word Vectors with Subword Information* arXiv:1607.04606

Alessandra Teresa Cignarella, Simona Frenda, Valerio Basile, Cristina Bosco, Viviana Patti and Paolo Rosso 2018. *Overview of the EVALITA 2018 Task on Irony Detection in Italian Tweets (IronITA)* Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2018)

Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova 2018. *BERT: Pre-training of Deep Bidirectional Transformers for Language* arXiv:1810.04805

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin and Tomas Mikolov 2018. *Learning Word Vectors for 157 Languages* Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)

Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun 2016. *Deep Residual Learning for Image Recognition* 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Martina Miliani, Giulia Giorgi, Ilir Rama, Guido Anselmi and Gianluca E. Lebani 2020. *DANKMEMES @ EVALITA2020: The memeing of life: memes, multimodality and politics.* Valerio Basile, Danilo Croce, Maria Di Maro and Lucia C. Passaro (eds.). Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)

Piero Molino, Yaroslav Dudin and Sai Sumanth 2019. *Ludwig: a type-based declarative deep learning toolbox* arXiv:1909.07930

Benet Oriol Sabat, Cristian Canton Ferrer and Xavier Giro-i-Nieto 2019. *Hate Speech in Pixels: Detection of Offensive Memes towards Automatic Moderation* arXiv:1910.02334

Karen Simonyan, Andrea Vedaldi and Andrew Zisserman 2013. *Deep inside convolutional networks: Visualising image classification models and saliency maps* arXiv:1312.6034

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest and Alexander M. Rush 2019. *HuggingFace's Transformers: State-of-the-art Natural Language Processing* arXiv:1910.03771

UPB @ DANKMEMES: Italian Memes Analysis - Employing Visual Models and Graph Convolutional Networks for Meme Identification and Hate Speech Detection

George-Alexandru Vlad*, **George-Eduard Zaharia***,
Dumitru-Clementin Cercel, Mihai Dascalu
University Politehnica of Bucharest, Faculty of Automatic Control and Computers
{george.vlad0108, george.zaharia0806}@stud.acs.upb.ro
{dumitru.cercel, mihai.dascalu}@upb.ro

Abstract

Certain events or political situations determine users from the online environment to express themselves by using different modalities. One of them is represented by Internet memes, which combine text with a representative image to entail a wide range of emotions, from humor to sarcasm and even hate. In this paper, we describe our approach for the DANKMEMES competition from EVALITA 2020 consisting of a multimodal multi-task learning architecture based on two main components. The first one is a Graph Convolutional Network combined with an Italian BERT for text encoding, while the second is varied between different image-based architectures (i.e., ResNet50, ResNet152, and VGG-16) for image representation. Our solution achieves good performance on the first two tasks of the current competition, ranking 3^{rd} for both Task 1 (.8437 macro-F1 score) and Task 2 (.8169 macro-F1 score), while exceeding by high margins the official baselines.

1 Introduction

During the past two decades, the Internet evolved massively and the social web became a hub where people share their opinions, cooperate to solve issues, or simply discuss on various topics. There are many ways in which users can express themselves: plain text, videos, or images. The latter option became widely used due to its convenience; however, images are frequently accompanied by a short text description to better convey information. As the Internet and the online social interactions evolved, certain image templates emerged and gained global popularity, contributing to a *de facto* standardization of joint text-image usage, and thus leading to the creation of memes. Memes can be humorous, satirical, offensive, or hateful, therefore encapsulating a wide range of emotions and beliefs. Properly identifying memes from non-memes, and then analyzing them to detect the users' intentions is becoming a stringent task in online marketing campaigns by targeting the automated identification of opinions pertaining to certain groups of users.

The DANKMEMES competition [22] from EVALITA 2020 [19] challenged participants to approach the previously mentioned issues by creating systems that identify and analyze Internet memes in Italian. The competition consists of three tasks, out of which we tackled two. Task 1 - *Meme Detection* considers the identification of memes from a collection of images, such that a clear distinction can be made between memes and ordinary images. Afterwards, Task 2 - *Hate Speech Identification* targets the classification of images in terms of their purpose, by analyzing content and identifying whether images are hateful or not.

2 Related Work

2.1 Multimodal Fake News Detection

Singhal et al. [16] employed the usage of multimodal techniques for fake news detection. The authors introduced SpotFake, an architecture divided into three sub-parts: one for identifying textual features using Bidirectional Encoder Representations from Transformers (BERT) [10], a second for visual analysis based on VGG-19 [5], while the third combines the previously mentioned elements into a single feature vector.

Similarly, Shah and Priyanshi [23] performed

---

*These authors contributed equally.

multimodal fake news detection by using two separate channels, visual and textual, both of them aiming to extract relevant features. Moreover, they included a Cultural Algorithm that introduces another dimension by employing situational knowledge, i.e. information about the depicted event as seen by a specific individual. Another approach regarding fake news detection was introduced by Khattar et al. [12] who created MVAE, a multimodal autoencoder including encoders (both visual and textual), decoders, and a detection module for classifying the inputs.

2.2 Multimodal Hate Speech Identification

Kiela et al. [20] created a new dataset specifically designed for identifying hateful speech in memes. At the same time, the authors also introduced a series of baselines for further comparison, including ResNet-152 [7] and VilBERT [13] for the visual channel, and BERT for the textual counterpart.

Furthermore, Sabat et al. [15] tackled the problem of hate speech identification in memes by also employing a multimodal system. However, they used an Optical Character Recognition system for extracting the textual component from the inputs, alongside visual features from a VGG-16 component and the text encoded with BERT.

3 Method

Our approach for both tasks consists of a multitask learning technique [1] and our architecture consists of two main neural network components, one for the text input, while the other for the image input. Thus, we combined the outputs of these two components and used the learned features for determining the required class, either for Task 1 or Task 2.

3.1 Corpus

The dataset for the meme detection task is split into two parts, train and test. The training dataset contains 1,600 image entries, together with a CSV file containing other useful metadata, such as: the engagement (i.e., number of comments and likes), date, and manipulation (i.e., binary coding denoting the low/high level of image modifications), alongside a transcript of the text present in the image. We kept 85% of the entries for training, while 15% are used for validation; the same class distribution is kept in both partition. The test dataset for the first task contains 400 entries with a corresponding CSV file of a similar structure. The second task offers a dataset containing 800 entries which was partitioned in a similar manner.

3.2 Image Component

Several image-based neural networks were considered for the first component of our final architecture. First, we used VGG-16 which consists of five stacks of Convolutional Neural Networks [4] accompanied by max-pooling layers. Pretrained weights on the ImageNet dataset [3] were afterwards fine-tuned. Second, we also experimented with ResNet in two variants, ResNet50 and ResNet152. ResNet introduced the concept of skip connections as a solution to the vanishing gradient problem; as such, the networks could be further scaled in terms of depth, enabling more abstract high-level features to be extracted from the input images. Similar VGG-16 architecture, pretrained weights on ImageNet were fine-tuned for ResNet152, whereas pretrained weights on VGGFace2 [9] were used for ResNet50.

3.3 Text Component

A Graph Convolutional Network (GCN) [18] for representing long-term dependencies between tokens was selected, alongside a pretrained version of BERT for Italian (ItalianBERT)[1] to model the contextual information at sample level. The underlying implementation of the textual feature extractor follows the architectural design of Vocabulary Graph Convolutional Network with BERT (VGCN-BERT) [21].

The proposed architecture (VGCN-ItalianBERT) uses a tight coupling between the graph convolutional layers and the ItalianBERT embeddings, enabling the model to better adjust the GCN extracted features through ItalianBERT's attention mechanism. The input to the VGCN layer is represented by a vector $X_{d,v}$, where d is the dimension of the ItalianBERT embedding and v is the number of tokens in the dataset vocabulary. A symmetric adjacency matrix $A_{v,v}$ is built to preserve the prior global relationship between tokens, where v is the vocabulary dimension. The edge weight between two nodes i, j, denoted as $A_{i,j}$, is initialized with the normalized point-wise mutual information (NPMI) value [2] between the two vocabulary tokens i, j. The mechanism of

---

[1] `https://github.com/dbmdz/berts#italian-bert`

the VGCN layer is formally summarized by the following equations:

$$H_{v,h} = Dropout(\widetilde{A}_{v,v} W_{v,h}) \tag{1}$$

$$H_{d,h} = ReLU(X_{d,v} H_{v,h}) \tag{2}$$

$$H_{d,g} = H_{d,h} W_{h,g} \tag{3}$$

where terms $W_{v,h}$ and $W_{h,g}$ represent the weights of the two GCN internal layers, with *v* the vocabulary dimension, *h* and *g* the output feature dimensions. In Equation 1, we add the global context by multiplying the normalized adjacency matrix $\widetilde{A}$ with the weight matrix of the first GCN layer. We use the normalized adjacency matrix $\widetilde{A} = D^{-1/2} A D^{-1/2}$ to ensure numerical stability. A convolution between the input vector $X_{d,v}$ and the result from the previous operation (Equation 2) is performed to combine the global information with the ItalianBERT embeddings. Lastly, Equation 3 projects the features to the dimensions required to fill in the reserved VGCN-ItalianBERT embedding slots.

Visual text features describing the actors of a meme are added as the pair sentence to ItalianBERT's input. We cap the second sentence containing the visual text features to *K* tokens, overflowing tokens being dropped. Considering *L* the maximum number of input tokens, the remainder of $L - K$ tokens are being split between the text tokens associated with a meme and *G* VGCN reserved slots. Those slots are kept empty to be internally filled with VGCN embeddings during training. Alongside ordinary inputs required by ItalianBERT (i.e. *input ids*, *input masks* and *segment ids*), we build a *gcn ids* vector similarly to *input ids*, by mapping each unique input token to the corresponding index in the task vocabulary V_{task}; V_{task} represents the set of tokens available in the task text corpus and in the ItalianBERT's vocabulary. The second additional input is represented by a binary mask vector having the value of 1 for the VGCN reserved tokens, and 0 otherwise. During training, all ItalianBERT layers with the exception of the last 4 encoder blocks were frozen.

3.4 Multimodal Architecture

The final solution consists of a multimodal architecture with two main components, each specialized on processing one informational channel, namely text or image-based. The dates are segmented and encoded by using complementary sine and cosine functions to preserve the cyclic characteristics of days (in a month) and months. Equation 4 describes the time cyclical encoding procedure, where *n* represents the day value subtracted by 1 and divided by the number of days in the corresponding month. The same operations are applied for the months encoding over the month index, but the denominator is 12 in this case. Additional metadata (i.e., manipulation and engagement) was also encoded and used in the final prediction. Values representing the year and engagement were normalized to ensure the model's stability during training.

$$\begin{gathered} \theta = 2 * \pi * n \\ time_{sin} = sin(\theta); time_{cos} = cos(\theta) \end{gathered} \tag{4}$$

The two feature vectors from the image and text components were fused together by concatenation into a single vector and passed through two fully connected layers, followed by a dropout layer of 0.5. The output of the dropout layer is then concatenated together with the other extracted features like time, engagement, manipulation, and fed to the output layer. Softmax activation function is used over the last fully connected layer to compute the distribution probability over the task classes. L2 regularization kernel is used on the two hidden layers before fusion to account for large activations and to keep our output layer sensible to the metadata encoded features.

In addition, an *ensemble*-based architecture using our ResNet50 + VGCN-ItalianBERT model was also considered. First, the training dataset was split into 5 sets, while preserving the class distribution of each fold. The aforementioned model was trained 5 times using 4/5 sets for training, and the remainder set for validation. A weighted voting procedure is performed at prediction time, in which the weights are represented by the average confidence score of the voters in the class receiving the highest probability after softmax. Thus, we advocate for higher confidence scores over the number of voters in choosing the predicted class.

3.5 Experimental Setup

Preprocessing steps were performed to feed the datasets to our architecture. The texts were tokenized using the ItalianBERT tokenizer, and then the *input ids*, *input masks*, *segment ids*, *gcn ids* and *gcn masks* were computed. Images were resized to a uniform dimension (i.e., 448 x 448) and were serialized alongside the text components in a *tfrecords* file specific for Tensorflow [6]. An Adam Weight decay optimizer [8] with a learning rate of *1e-5* and a weight decay rate of 0.01 were used in all conducted experiments. Furthermore, the warm up proportion was set to 0.1.

The maximum input length was limited to $L = 100$ tokens and the *Visual* text features to $K = 20$ tokens as the textual channel of memes is represented by short sentences. Following the experimental setup described in [21], we reserve $G = 16$ slots to be filled with the resulted VGCN-ItalianBERT embeddings. Moreover, only NPMI values larger than 0.3 are kept in the adjacency matrix A, corresponding to a higher semantic correlation between words; all the other values below this threshold are set to 0.

We empirically found 1e-5 to be a good learning rate value, which is on par with the results of [21]. Lastly, we choose to train all the models for 9 epochs with a batch size of 8 examples.

3.6 Results

Table 1 contains the results obtained by our models for the first two tasks of the DANKMEMES competition. The components that were frozen during the training process are varied for the three main conducted experiments (i.e. combining ItalianBERT with VGCN and ResNet50, ResNet152 and VGG-16, respectively) to identify proper adjustments for the weights of the pretrained models. The best results among the four evaluated sets (i.e. validation, test for Task 1 and validation, test for Task 2) are obtained by either freezing only the VGCN-ItalianBERT component or by freezing both textual and image components. The necessity of freezing the text branch of the architecture underlines the fact that the pretrained weights for the ItalianBERT model already properly capture specific traits of Italian and prove to be a viable option, even when analyzing short texts such as memes. Furthermore, the last convolutional block of the image component needs to be unfrozen because training an architecture on potential meme images is a more specific task when compared to analyzing Italian text.

The best results are obtained using variations of the ResNet50 + VGCN-ItalianBERT model, with an .9041 macro-F1 score for the custom validation dataset used for Task 1, and .8745 and .8169 macro-F1 scores on the validation and test datasets for Task 2. However, the best result for the Task 1 test set is yielded by the ResNet152 + VGCN-ItalianBERT architecture, with an .8700 macro-F1 score.

ItalianBERT, ResNet50, and ResNet50 + ItalianBERT are used as baseline models to explore the improvements made by adding VGCN to the textual architecture while maintaining the same experimental setup. As expected, the model using only the textual channel (i.e. ItalianBERT baseline model) is performing considerably worse than the joint architecture ResNet50 + ItalianBERT, thus arguing for the importance of considering images in disambiguating the textual input. The ResNet50 + VGCN-ItalianBERT model performs consistently better than its baseline counterpart (i.e., ResNet50 + ItalianBERT), by obtaining improvements of 2.92% and 3.35% macro-F1 score on the validation sets for Task 1 and Task 2, respectively.

3.7 Error Analysis

Although the models performed arguably well on both task, the identified misclassifications represent a good starting point for further analysis and improvement. Figure 1 depicts a series of misclassified entries from both tasks.

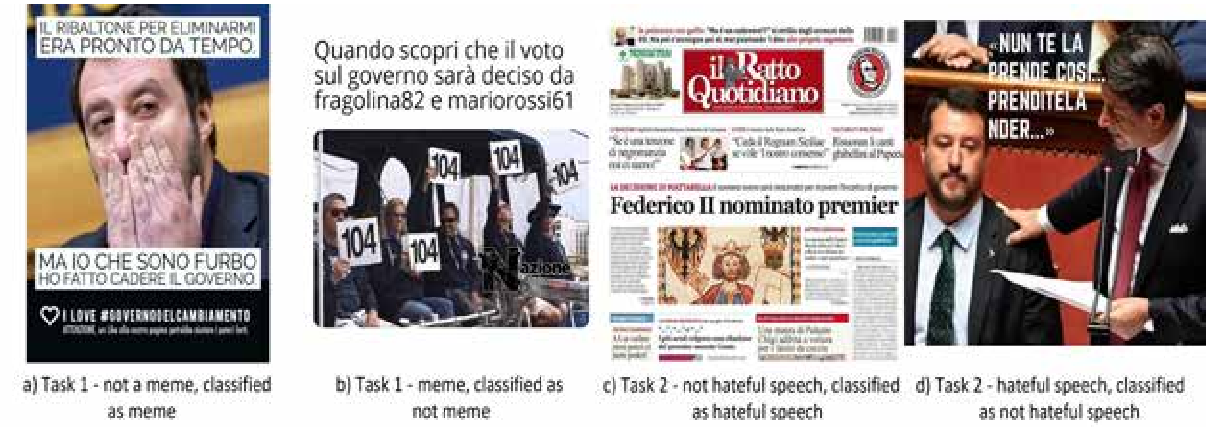
The short texts encountered in memes require in several situations prior information on the sociopolitical context, therefore making the detection of memes an exceedingly difficult task. In general, a few well known and highly popular image templates are reused, by changing or partially adjusting the text to expressively convey an idea or a view on a certain subject. However, the used templates in the current competition are extensively customized and tailored specifically to the political context of Italy. In addition, the subjectivity of the annotators also plays a decisive role, considering that the concept of the hateful speech tag for the second task is not well defined for all situations and can be interpreted differently.

4 Conclusion and Future Work

This paper introduces our multimodal architecture for the first two tasks of the DANKMEMES competition from EVALITA 2020. Several

Table 1: Macro-F1 scores on the validation and test datasets, for both Task 1 and Task 2. Submitted models are shown in italics.

Neural Architecture	Frozen Component		Task 1		Task 2	
	Image	Text	Dev	Test	Dev	Test
ItalianBERT	-	-	0.7618	0.7546	0.8083	0.7996
ResNet50	-	-	0.8203	0.7899	0.5661	0.5598
ResNet50 + ItalianBERT	-	✓	0.8749	0.8499	0.8331	0.7949
ResNet50 + VGCN-ItalianBERT	-	-	0.8666	0.8348	0.8413	0.8150
ResNet50 + VGCN-ItalianBERT	-	✓	**0.9041**	0.8235	0.8666	**0.8169**
ResNet50 + VGCN-ItalianBERT	✓	-	0.8874	0.8375	0.8493	0.7584
ResNet50 + VGCN-ItalianBERT	✓	✓	0.8833	0.8499	**0.8745**	0.7992
ResNet152 + VGCN-ItalianBERT	-	-	0.8458	0.8424	0.8331	0.7998
ResNet152 + VGCN-ItalianBERT	-	✓	0.8791	**0.8700**	0.8666	0.7994
ResNet152 + VGCN-ItalianBERT	✓	-	0.8246	0.8474	0.8310	0.8093
ResNet152 + VGCN-ItalianBERT	✓	✓	0.8915	0.8273	0.8489	0.7490
VGG-16 + VGCN-ItalianBERT	-	-	0.8124	0.7923	0.6906	0.5478
VGG-16 + VGCN-ItalianBERT	-	✓	0.8083	0.7620	0.5566	0.5469
VGG-16 + VGCN-ItalianBERT	✓	-	0.7485	0.7447	0.6414	0.5263
VGG-16 + VGCN-ItalianBERT	✓	✓	0.7621	0.7248	0.6003	0.5388
Ensemble Architecture	-	-	0.8916	0.8437	0.7874	0.7692
Competition Baselines	-	-	-	0.5198	-	0.5621


IL RIBALTONE PER ELIMINARMI ERA PRONTO DA TEMPO.
MA IO CHE SONO FURBO HO FATTO CADERE IL GOVERNO
I LOVE #GOVERNODELCAMBIAMENTO
Quando scopri che il voto sul governo sarà deciso da fragolina82 e mariorossi61
104
il Ratto Quotidiano
Federico II nominato premier
«NUN TE LA PRENDE COSÌ... PRENDITELA NDER...»

a) Task 1 - not a meme, classified as meme

b) Task 1 - meme, classified as not meme

c) Task 2 - not hateful speech, classified as hateful speech

d) Task 2 - hateful speech, classified as not hateful speech

Figure 1: Examples of misclassified samples for both tasks.

joint text - Vocabulary Graph Convolutional Network alongside an Italian BERT model - and image-based architectures - ResNet50, ResNet152, VGG-16 - were experimented. The consideration of meme meta-information, such as cyclic temporal characteristics and post engagement, boosted even further our F1-scores when compared to the competition baseline.

In terms of future work, we intend to experiment with other visual architectures, including VGG-19 [5] and EfficientNet [17], and also with multilingual neural networks, such as mBERT [14] and XLM-RoBERTa [11], that will empower transfer learning across meme datasets in different languages.

References

[1] Rich Caruana. “Multitask learning”. In: *Machine learning* 28.1 (1997), pp. 41–75.

[2] Gerlof Bouma. “Normalized (pointwise) mutual information in collocation extraction”. In: *Proceedings of GSCL* (2009), pp. 31–40.

[3] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE Conference on Computer Vision and*

Pattern Recognition. Ieee. 2009, pp. 248–255.

[4] Yoon Kim. “Convolutional neural networks for sentence classification”. In: *arXiv preprint arXiv:1408.5882* (2014).

[5] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[6] Martin Abadi et al. “Tensorflow: Large-scale machine learning on heterogeneous distributed systems”. In: *arXiv preprint arXiv:1603.04467* (2016).

[7] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[8] Ilya Loshchilov and Frank Hutter. “Decoupled weight decay regularization”. In: *arXiv preprint arXiv:1711.05101* (2017).

[9] Qiong Cao et al. “Vggface2: A dataset for recognising faces across pose and age”. In: *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE. 2018, pp. 67–74.

[10] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[11] Alexis Conneau et al. “Unsupervised cross-lingual representation learning at scale”. In: *arXiv preprint arXiv:1911.02116* (2019).

[12] Dhruv Khattar et al. “Mvae: Multimodal variational autoencoder for fake news detection”. In: *The World Wide Web Conference*. 2019, pp. 2915–2921.

[13] Jiasen Lu et al. “Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 13–23.

[14] Telmo Pires, Eva Schlinger, and Dan Garrette. “How multilingual is Multilingual BERT?” In: *arXiv preprint arXiv:1906.01502* (2019).

[15] Benet Oriol Sabat, Cristian Canton Ferrer, and Xavier Giro-i-Nieto. “Hate Speech in Pixels: Detection of Offensive Memes towards Automatic Moderation”. In: *arXiv preprint arXiv:1910.02334* (2019).

[16] Shivangi Singhal et al. “SpotFake: A Multi-modal Framework for Fake News Detection”. In: *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*. IEEE. 2019, pp. 39–47.

[17] Mingxing Tan and Quoc V Le. “Efficientnet: Rethinking model scaling for convolutional neural networks”. In: *arXiv preprint arXiv:1905.11946* (2019).

[18] Liang Yao, Chengsheng Mao, and Yuan Luo. “Graph convolutional networks for text classification”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 2019, pp. 7370–7377.

[19] Valerio Basile et al. “EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian”. In: *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*. Ed. by Valerio Basile et al. Online: CEUR.org, 2020.

[20] Douwe Kiela et al. “The Hateful Memes Challenge: Detecting Hate Speech in Multimodal Memes”. In: *arXiv preprint arXiv:2005.04790* (2020).

[21] Zhibin Lu, Pan Du, and Jian-Yun Nie. “VGCN-BERT: Augmenting BERT with Graph Embedding for Text Classification”. In: *European Conference on Information Retrieval*. Springer. 2020, pp. 369–382.

[22] Martina Miliani et al. “DANKMEMES @ EVALITA2020: The memeing of life: memes, multimodality and politics”. In: *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*. Ed. by Valerio Basile et al. Online: CEUR.org, 2020.

[23] Priyanshi Shah and Ziad Kobti. “Multimodal fake news detection using a Cultural Algorithm with situational and normative knowledge”. In: *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE. 2020, pp. 1–7.

ArchiMeDe @ DANKMEMES: A New Model Architecture for Meme Detection

Jinen Setpal
RN Podar School
Mumbai, India
`jinens8@gmail.com`
`jinen.setpal@rnpodarschool.com`

Gabriele Sarti
Department of Mathematics and Geosciences
University of Trieste & SISSA
Trieste, Italy
`gsarti@sissa.it`

Abstract

English. We introduce ArchiMeDe, a multimodal neural network-based architecture used to solve the DANKMEMES meme detections subtask at the 2020 EVALITA campaign. The system incorporates information from visual and textual sources through a multimodal neural ensemble to predict if input images and their respective metadata are memes or not. Each pre-trained neural network in the ensemble is first fine-tuned individually on the training dataset to perform domain adaptation. Learned text and visual representations are then concatenated to obtain a single multimodal embedding, and the final prediction is performed through majority voting by all networks in the ensemble.

Italiano. *Presentiamo ArchiMeDe, un'architettura multimodale basata su reti neurali per la risoluzione del subtask di "meme detection" per DANKMEMES a EVALITA 2020. Il sistema unisce informazione visiva e testuale attraverso un insieme multimodale di reti neurali per prevedere se immagini e rispettivi metadati corrispondano a meme o meno. Ogni rete neurale pre-allenata all'interno dell'insieme è inizialmente adattata al dominio specifico del dataset di training. In seguito, le rappresentazioni di ogni rete per immagini e testo vengono concatenate in un unico embedding multimodale, e la previsione finale è effettuata tramite un voto di maggioranza effettuato da tutte le reti nell'insieme.*

1 Introduction

In recent years, the democratization of data collection procedures through web scraping and crowdsourcing has led to the broad availability of public datasets spanning modalities like language and vision. Contemporary state-of-the-art machine learning models can leverage those resources to achieve highly accurate and often superhuman performances using millions or even billions of parameters (Brown et al., 2020), but are heavily reliant on an abundance of computational resources to work properly. Consequently, such architectures' training is often inaccessible to smaller research centers – let alone individual users. To counter this tendency, the availability of pre-trained open-source models has dramatically reduced the computational threshold required to obtain state-of-the-art results in multiple languages and vision tasks (Devlin et al., 2019; He et al., 2016). Pre-trained systems are often leveraged in a two-step framework: first, they undergo an unsupervised or semi-supervised pre-training to learn general knowledge representations, then they are fine-tuned in a supervised way to adapt their parameters in the context of downstream tasks. This transfer learning approach stems from the computer vision literature (He et al., 2019) but has been recently adopted for natural language processing tasks with positive results (Howard and Ruder, 2018; Devlin et al., 2019; Liu et al., 2019).

In this paper, we present ArchiMeDe, a multimodal system leveraging pre-trained language and vision models to compete in the DANKMEMES (Miliani et al., 2020) shared task at the EVALITA 2020 campaign (Basile et al., 2020). Following recent transfer learning approaches, our system leverages pre-trained visual and word embeddings in a multimodal setup, obtaining strong results on the meme detection subtask. Specifically, we participated in the first sub-

---

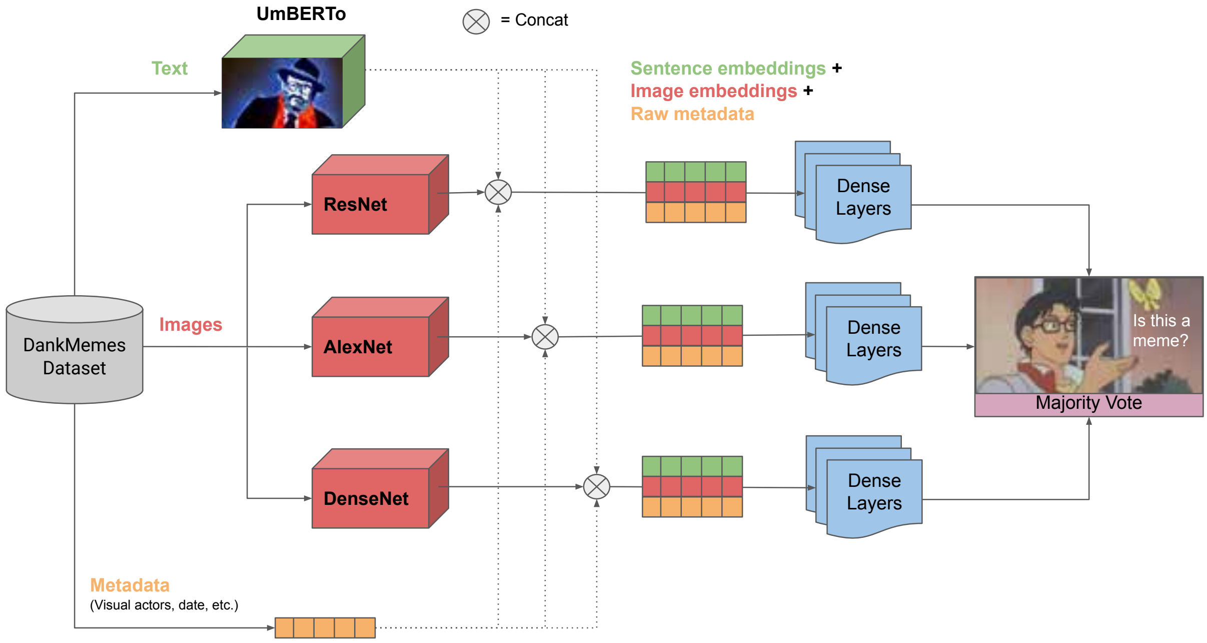


Figure 1: The ArchiMeDe system architecture. Sentence embeddings produced by the UmBERTo NLM are concatenated to metadata and image embeddings produced by three popular pre-trained vision modals. The three resulting multimodal embeddings are fed separately to feedforward networks, and the final outcome is selected through majority voting.

task of DANKMEMES, aimed at discriminating memes from standard images containing actors from the Italian political scene. Task organizers extracted a total of 1600 training images from the Instagram platform, and data available from each dataset entry – text, actors and user engagement, among others – were leveraged to train an ensemble of multimodal models performing meme detection through majority-vote. The following sections present our approach in detail, first showing our preliminary evaluation of multiple modeling approaches and then focusing on the final system's main modules and the features we leverage from the dataset. Finally, results are presented, and we conclude by discussing the problems we faced with some inconsistencies in the data. Our code is made available at `https://github.com/jinensetpal/ArchiMeDe`

2 System Description

ArchiMeDe is composed of a multimodal learning ensemble, with the final output being the result of a majority vote. Figure 1 visualizes our approach. First, the transcript associated with each image is fed to an UmBERTo (Francia et al., 2020) neural language model (NLM) pre-trained on the Italian language to produce sentence embeddings. Then, we leverage three popular pre-trained vision architectures, namely ResNet (He et al., 2016), DenseNet (Huang et al., 2017a) and AlexNet (Krizhevsky et al., 2017), to produce three independent image embeddings for each input image. These embeddings can be considered as different views over an image that may provide us with complementary information about its content. Then, each image embedding is concatenated with the sentence embedding and the raw image metadata and fed as input to an 8-layer feed-forward neural network to predict an image's meme status. The feed-forward network also includes a single dropout layer to prevent overfitting and improve generalization. Lastly, the three predictions are weighted through majority voting to obtain the final prediction of the ensemble. Other simpler strategies using a single vision model to produce image embeddings were initially envisaged as potential candidates for our submission but were finally dismissed in light of the promising performances of the ArchiMeDe ensembling approach. We discuss those perspectives in Section 4.

The remaining part of this section contains an in-depth description of our ensemble's components, focusing on the input features that were used and how those were preprocessed to best suit learning. Moreover, we also include transfer learning specifications with some details about their impact on the overall system accuracy.

2.1 Metadata

Engagement User engagement per post is expressed as a numeric integer value. We scale and standardize engagement values to obtain a distribution centered in 0 with $\sigma = 1$. This procedure is a standard practice to avoid passing extreme absolute values as inputs for the neural network.

Date We decided to leverage temporal information in our system, building upon the intuition that memes often rely on a small set of templates that undergo a significant variation in popularity through time. Temporal information may thus provide our system with additional cues about an image's meme status in a specific time-frame. In the training dataset, dates for each post has been presented in the yyyy-mm-dd format. This date was compared with the predetermined date, 1st January 2015, to derive a numeric value representing the number of days from the date of reference. Min-max scaling is then applied to the numeric values, further deriving float numeric values between in the range [0,1], subsequently fed into each training model.

Manipulation The manipulation field provides boolean information about whether an image has been manipulated before being added to the dataset. We found this information noisy and a weak predictor of meme status; therefore, it was dropped as input.

Visual Actors Each entry was additionally provided with a list of names of the visual actors present in the frame. In the specific case of the DANKMEMES shared task, visual actors can be especially useful to identify meme images. For example, we can hypothesize that politicians who maintain a strong public presence by making claims that produce a high level of public engagement are more likely to be the subject of meme images. Moreover, some combinations of actors may be particularly likely for memes e.g. politicians belonging to parties at the political compass's antipodes. In order to produce a unified representation of visual actors for our system, we perform a one-hot encoding of all the actors occurring in the training set: if a specific politician is present in an image, the corresponding entry is true; conversely, if no such actor is present, the binary field is set to false. Actors that were not present in the training set are disregarded during evaluation: while this step is required given the context, we assume that this may significantly impact the outcome in images for which new actors were introduced.

2.2 Textual input

The analysis of textual content in meme images is critical to the success of the overall system. Indeed, ironical or satyrical comments may deeply affect the users' interpretation of an image that would otherwise be classified as normal. We note that this problem cannot be approached similarly to standard textual analytic frameworks since memes are elucidated in short, concise phrases and do not necessarily comply with standard grammatical rules. They also tend to contain slang and vernacular expressions, which, albeit conveying the intended meaning to the reader, greatly increase the need for high model capacity and adhoc training data. For this reason, we selected UmBERTo (Francia et al., 2020), a RoBERTa-based (Liu et al., 2019) neural language model pre-trained on Italian texts extracted from the OSCAR corpus (Ortiz Suárez et al., 2020), for producing text representations.[1] In a recent study by Miaschi et al. (2020), the model was highlighted as one of the top Italian NLMs for encoding linguistic information about social media excerpts taken from the TWITTIRÒ and PoSTWITA Twitter corpora (Cignarella et al., 2019; Sanguinetti et al., 2018). UmBERTo has a high model capability with 125M trainable parameters and was trained on online crawled data, making it suitable for processing meme language.

SentenceTransformers We use the Sentence-Transformers framework (Reimers and Gurevych, 2019) to produce sentence embeddings by averaging all word embeddings produced by the original UmBERTo model since Miaschi and Dell'Orletta (2020) showed that those are usually much more informative than the default [CLS] sentence embedding. We fine-tune representations over the available meme textual data and use them as components of our end-to-end system.

---

[1] `umberto-commoncrawl-cased-v1` in the HuggingFace's model hub (Wolf et al., 2019)

2.3 Visual input

While we have so far discussed only using metadata to predict our results, it is essential to address the core of a meme: the image itself. We can internally distinguish a meme from a standard image through the aforementioned broken sentence structure, meme templates, and quick and messy edits, among other aspects. As previously mentioned, memes can be very difficult to individuate when they look like standard images but gain meme status through real-world knowledge grounding.

Due to the inherently large variance in meme images' styles and contents, it is impractical to expect a single framework to effectively describe each distinguishable feature and utilize it to classify an entry. Hence, we split the representational burden across multiple pre-trained model architectures. Each of them uses a fundamentally different approach to extract image embeddings, making the resulting ensemble predictions more flexible in general settings. The three networks we used for producing image embeddings are:

ResNet Residual Networks, or ResNets (He et al., 2016), learn residual functions in relation to layer inputs. If $\mathcal{H}(x)$ is the standard underlying target mapping, ResNet layers are instead trained to fit another mapping $\mathcal{F}(x) = \mathcal{H}(x) - x$. The original mapping is thus recast into $\mathcal{F}(x)+x$. This approach makes the optimization process easier, allowing for deeper architectures. The default vector representation provided by task organizers is produced by a ResNet-50, with fifty blocks of residual layers. We use those image embeddings of size 2048 without further adjustments.

AlexNet AlexNet (Krizhevsky et al., 2017) is a vision architecture built with 5 layers of convolution and 3 fully-connected layers. AlexNet specializes in identifying depth; the network architecture effectively classifies objects such as keyboards and a large subset of animals. This fact makes AlexNet embeddings good predictors for features such as depth that are generally problematic in memes due to image subsections (e.g. text boxes). We use an embedding size of 4096 in the context of our experiments.

DenseNet Pre-trained models such as ResNet and AlexNet use a large number of hidden layers. While the increase in depth allows for better feature abstraction, it often leads to vanishing-gradient problems during training. DenseNet (Huang et al., 2017b) introduces dense blocks where the feature-maps of all preceding layers are used as inputs to the layer, and its feature-maps are used as inputs into all subsequent layers. This approach encourages feature reuse and may lead to more generalizable image embeddings. Each DenseNet image embedding has a size of 1000 weights.

	Run #	Precision	Recall	F1
Baseline		0.525	0.5147	0.5198
UniTor	1	0.839	0.8431	0.8411
	2	0.8522	**0.848**	**0.8501**
SNK	1	0.8515	0.8431	0.8473
	2	0.8317	**0.848**	0.8398
UPB	1	**0.861**	0.7892	0.8235
	2	0.8543	0.8333	0.8437
ArchiMeDe	1	0.8249	0.7157	0.7664
Keila	1	0.8121	0.6569	0.7263
	2	0.7389	0.652	0.6927

Table 1: System ranking for the DANKMEMES meme detection subtask. Top scores are in **bold**, our system is underlined.

The aim of using multiple vector embeddings was to cumulatively cover a significant portion of possible meme combinations and templates. As a result, is Section 4 we show how the ensemble of systems using different image embeddings leads to significant increases in validation accuracy.

3 Results

Table 1 presents the system ranking for the meme detection subtask. Our system placed 7th in terms of F1 score,[2] impeded primarily by inconsistent recall performances but significantly better than the random baseline (+0.2466 F1).

Results suggest that ArchiMeDe has developed inductive biases for specific image features that strongly influence the classification outcome. By inspecting validation folds over training data, we observe that most false negatives produced by the system involve distinct facial characteristics of scene actors. Inversely, ArchiMeDe effectively classifies images containing text bubbles and evident manual edits. Another notable failure case we identified is due to face-swapping. This failure is especially relevant since face-swapping is com-

---

[2] The F1 score is the harmonic mean between precision and recall, commonly used to evaluate classification systems.

Encoder	**Precision**	**Recall**	**F1**
AlexNet	.83/.77	.75/.85	.79/.81
DenseNet	.87/.83	.82/.87	.84/.85
ResNet	.83/.79	**.87/.86**	.85/.83
ResNeSt	.80/.84	.84/.76	.82/.79
ArchiMeDe	**.87/.85**	.84/.87	**.86/.86**

Table 2: Performances of ArchiMeDe variants with single image encoders over a validation split of the DANKMEMES training set. Scores are presented for non-meme/meme classes.

monly used to add an ironic component to meme images, but it is hardly detectable due to missing real-world context.

4 Other Embedding Approaches

As a complementary perspective on our experiments' nature, in this section, we present other approaches tested in the context of meme detection and that were finally disregarded in favor of the ArchiMeDe approach presented in the previous section.

CNN without Metadata Preliminary runs on the DANKMEMES dataset relied solely on the use of standard convolutional neural networks. The target architecture was fed the image itself without associated metadata to ensure that the standalone impact of the architecture was shown. The system performed poorly, performing only slightly better than the baseline scores. Additional measures to optimize this network were not taken since we assumed that this naive approach would not lead to substantial gains in performances over the baseline.

Single Pre-trained Image Encoder Before working with an ensemble, we estimated the performances of its components in performing meme detection. Besides the three models that we finally included in ArchiMeDe, we also tested ResNeSt (Zhang et al., 2020), which was finally dropped due to the similarity of its predictions to those of ResNet-50. Table 2 presents the performances of the individual image encoders and the final ensemble over a validation split containing 320 examples equally distributed over (meme, non-meme) classes. Results show how the DenseNet model appears to be better in terms of precision, while ResNet is worse but compensates with a higher recall. We found that misclassified observations were different across models, suggesting that each model could capture different properties of the input. The only exception was the ResNeSt model, which produced errors very close to the ResNet ones and was henceforth dropped for further experiments.

Multimodal Ensemble Following the complementary viewpoints of different encoders, we decided to evaluate the performances of an ensemble. Table 2 shows that our ArchiMeDe ensemble outperforms single systems in terms of both precision and recall when considering both classes, compensating the weaknesses of individual systems. The resulting majority-vote ensemble was optimized and used as the final system for our submission. Multiple experimental iterations showed that an increase in depth, followed by a reduction in layers' width, led to increased accuracy scores. Each model was trained with a batch size of 64 sets, 100 epochs fitted with test accuracy callbacks, and an early stopping strategy with a five epochs' patience value. Each model utilized the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.001 and was trained using a binary cross-entropy loss over the two categories.

4.1 Data Augmentation

Given the relatively small size of the available training dataset and since popular classification models are often trained using thousands if not millions of images, we tested some data augmentation strategies to improve our system's generalization performances. We applied random changes for each image to augment data, modifying it with random brightness, rotation, and zoom in a reasonable margin to keep it distinguishable. 9 augmented images were produced for every initial image entry. As a result, the training dataset is increased from 1280 to 12800 images.

Every augmented image is associated with the same metadata as the original, varying only in the visual embedding itself. The result we aimed for was an increase in generalization performances, as the model fits better to the general rule of recognizing memes. However, our results showed the opposite behavior: the system would easily overfit individual observation when data augmentation was used. We think this was partly due to augmentations not pertinent to the general meme template and partly because of the significant increase in the number of entries having the same associated metadata.

An extensive set of augmentation strategies was tested over the dataset, modifying factors, ranges, and augmentation count. No iteration significantly and consistently improved the system's performance, and thus the augmentation process was determined noisy, relatively inconclusive, and therefore dropped from the training procedure.

5 Discussion and Conclusion

In this paper, we presented ArchiMeDe, our multimodal system used for participating in the DANKMEMES task at EVALITA 2020. The results produced by the system are promising, even if the systems do not encode inductive biases that are specific neither for multimodal artifact recognition nor to meme detection in particular. The entry is not far behind in terms of precision from the best-performing systems, and several paths display considerable potential for improving its performances. The paper effectively highlights the crucial impact of transfer learning on the success of this system. Notably, ArchiMeDe can be easily trained with standard consumer-level GPUs.

A direction that can be explored to improve the current system would be to modify the recall threshold, obtaining a better precision-recall balance for predictions. Another possibility involves introducing an aggregator network on top of the ensemble instead of using majority vote: in this way, the network can learn whether the predictions of a single subnetwork are reliable, regardless of it being part of the majority. The ensemble could also include more varied models with differing architecture to further accentuate differences in feature representations. Above all, we believe that leveraging additional data (not necessarily in Italian) could significantly improve the system's performance at the cost of increased time and computational costs.

Memes today are one of the most formidable modes of portraying one's idea while building a strong interpersonal connection between creators and users. The informality of memes, combined with their ease of making and distribution, has greatly accentuated their growth in the last few years. To be able to interpret memes effectively is a task far deeper than what can be intuitively thought. As humans continue to unravel their minds and derive ingenious computational methods, we realize the importance of slang and how it relates directly to the core human principle of community belonging. A piece of our culture, memes are the best represented and documented cultural artifacts we have today, and to effectively interpret them would mean to cross a significant milestone for the field NLP, with lasting impacts on our society as a whole.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

T. Brown, B. Mann, Nick Ryder, Melanie Subbiah, J. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, G. Krüger, Tom Henighan, R. Child, Aditya Ramesh, D. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, E. Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, J. Clark, Christopher Berner, Sam McCandlish, A. Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Alessandra Teresa Cignarella, Cristina Bosco, and Paolo Rosso. 2019. Presenting TWITTIRÒ-UD: An italian twitter treebank in universal dependencies. In *Proceedings of the Fifth International Conference on Dependency Linguistics (Depling, SyntaxFest 2019)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Simone Francia, Loreto Parisi, and Magnani Paolo. 2020. UmBERTo: an italian language model trained with whole word maskings.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Kaiming He, Ross B. Girshick, and P. Dollár. 2019. Rethinking ImageNet pre-training. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4917–4926.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia, July. Association for Computational Linguistics.

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. 2017a. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269.

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. 2017b. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, May.

Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *ArXiv*, abs/1907.11692.

Alessio Miaschi and Felice Dell'Orletta. 2020. Contextual and non-contextual word embeddings: an in-depth linguistic investigation. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 110–119, Online, July. Association for Computational Linguistics.

Alessio Miaschi, Gabriele Sarti, Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi. 2020. Italian transformers under the linguistic lens. In *Proceedings of the Seventh Italian Conference on Computational Linguistics (CLiC-it)*.

Martina Miliani, Giulia Giorgi, Ilir Rama, Guido Anselmi, and Gianluca E. Lebani. 2020. DANKMEMES @ EVALITA2020: The memeing of life: memes, multimodality and politics. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online, July. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November. Association for Computational Linguistics.

Manuela Sanguinetti, Cristina Bosco, Alberto Lavelli, Alessandro Mazzei, and Fabio Tamburini. 2018. PoSTWITA-UD: an Italian Twitter Treebank in universal dependencies. In *Proceedings of the Eleventh Language Resources and Evaluation Conference (LREC 2018)*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Zhi-Li Zhang, Haibin Lin, Yu e Sun, Tong He, Jonas Mueller, R. Manmatha, M. Li, and Alex Smola. 2020. Resnest: Split-attention networks. *ArXiv*, abs/2004.08955.

UNITOR @ DANKMEMES: Combining Convolutional Models and Transformer-based architectures for accurate MEME management

Claudia Breazzano and **Edoardo Rubino** and **Danilo Croce** and **Roberto Basili**
University of Roma, Tor Vergata
Via del Politecnico 1, Rome, 00133, Italy
`claudiabreazzano@outlook.it, edoardo.ru94@libero.it`
`{croce,basili}@info.uniroma2.it`

Abstract

This paper describes the UNITOR system that participated to the "multimoDal Artefacts recogNition Knowledge for MEMES" (DANKMEMES) task within the context of EVALITA 2020. UNITOR implements a neural model which combines a Deep Convolutional Neural Network to encode visual information of input images and a Transformer-based architecture to encode the meaning of the attached texts. UNITOR ranked first in all subtasks, clearly confirming the robustness of the investigated neural architectures and suggesting the beneficial impact of the proposed combination strategy.

1 Introduction

In Social networks, the ways to express opinions evolved from simply writing a post to publishing more complex contents, e.g., the composition of images and texts. These multi-modal objects, if adhering to some specific social conventions and visual specifications, are called MEMEs. In particular, a MEME is a multi-modal artifact, manipulated by users, who combines intertextual elements to convey a message. Characterized by a visual format that includes images, text, or a combination of them, MEMEs combine references to current events or related situations and pop-cultural references to music, comics and films (Ross and Rivers, 2017). In this context, the multimoDal Artefacts recogNition Knowledge for MEMES (DANKMEMES) task is the first EVALITA (Basile et al., 2020) task for MEMEs recognition and hate speech/event identification in MEMEs (Miliani et al., 2020). This task is divided into three subtasks: in MEME Detection, system is required to determine whether an image is a MEME, according to the definition of (Shifman, 2013); in Hate Speech Identification the aim is to recognize if a MEME expresses an offensive message; finally, in Event Clustering the aim is to cluster MEMEs according to their referring topics.

In this work, we present the UNITOR system participating in all three subtasks. Since MEMEs convey their content through the multimodal combination of an image and a text, UNITOR implements a neural network which combines state-of-the-art architectures for Computer Vision and Natural Language Processing. In particular, Deep Convolutional Neural Networks, such as (He et al., 2016; Tan and Le, 2019) are used to encode visual information into dense embeddings and Transformer-based architectures, such as (Devlin et al., 2019; Liu et al., 2019) encode the meaning of the added overlaid captions. UNITOR then stacks a multi-layered network in order to effectively combine the evidences captured by both encoders, in the final classification.

The UNITOR system ranked first in each subtask, clearly confirming the robustness of the investigated neural architectures and suggesting the beneficial contribution of the proposed combination strategy. In the rest of the paper, in Section 2 the UNITOR system is described while Section 3 reports the experimental results.

2 UNITOR Description

CNNs for Image classification. Recent years demonstrated that Convolutional Neural Networks (CNNs) are able to achieve state-of-the-art results in image processing (Jiao and Zhao, 2019), by implementing deep and complex stackings of Convolutional layers, which capture different aspects of input images at different levels of the networks.

Among the investigated architectures, we first considered ResNET (He et al., 2016): this network is the first introducing Residual Learning to define very deep and effective CNNs. Several ResNET architectures are defined by stacking 50, 101, 152 up to 1001 layers of convolu-

---

tion layers and skip connectors: as a result, deeper networks achieved significant improvements of previous state-of-art in a wide plethora of image processing tasks. Moreover, we investigated the recently proposed EfficientNet (Tan and Le, 2019): unlike ResNET, this is not a real architecture, but it provides an automatic methodology to improve the performance of an existing CNN (such as ResNET) by tuning its depth, width and resolution dimensions. The adoption of this methodology led to the definition of 8 CNNs (namely EfficientNET-B0, EfficientNET-B1 up to EfficientNET-B7), each characterized by an increasing depth and width. They achieve impressive results by efficiently balancing the number of the parameters of the network. The tuning process of (Tan and Le, 2019) demonstrated that a network such as EfficientNet-B3 achieves higher accuracy than ResNeXt101 (Xie et al., 2016) in using 18x fewer neural operations. Regardless of the adopted networks, these are already trained in a classification task involving the recognition of thousands of object types in several millions of images, i.e. in the ImageNet dataset (Deng et al., 2009). This pre-training step enables the network to recognize many "basic entities" (such as people or animals) before being applied to a new task, e.g., MEME Detection. The customization to a new task is obtained just by replacing the last classification layer with a new one (sized based on the number of targeted classes) and by fine-tuning the entire architecture. It is worth noticing that, once the architecture is fine-tuned on the new down-stream task, it can be also used as an *Image Encoder*: the embeddings generated on the layer previous the classification one can be used as low-dimensional representations of input images. Most importantly, these embeddings are correlated with the down-stream task, as they are expected to lay in linearly separable sub-spaces (Goodfellow et al., 2016), where the final classifier is applied. In UNITOR these vectors are used to combine visual information with other evidences: in practice, they will be used in combination with the embeddings produced from the Transformer-based architectures (applied to texts) before being used in input to the final classifier.

Transformer-based Architectures for text classification. A MEME is a combination of visual information and the overlaid caption. In this work, we thus also investigated classifiers based on the text made available via OCR to the participants by the DANKMEME organizers. In particular, we adopt the approach proposed in (Devlin et al., 2019), namely Bidirectional Encoder Representations from Transformers (BERT). It provides an effective way to pre-train a neural network over large-scale collections of raw texts, and apply it to a large variety of supervised NLP tasks, here text classification. The building block of BERT is the Transformer element (Vaswani et al., 2017), an attention-based mechanism that learns contextual relations between words in a text. The pre-training stage is based on two auxiliary tasks, whose aim is the acquisition of an expressive and robust language and text model: the *Masked Language* model acquires a meaningful and context-sensitive representation of words, while the *Next Sentence Prediction* task captures discourse level information. In particular, this last task operates on text-pairs to capture relational information between them, e.g. between the consecutive sentences in a text. The straightforward application of BERT has shown better results than previous state-of-the-art models on a wide spectrum of natural language processing tasks. In (Liu et al., 2019) RoBERTa is proposed as a variant of BERT which modifies some key hyperparameters, including removing the next-sentence pre-training objective, and training on more data, with much larger mini-batches and learning rates. This allows RoBERTa to improve on the masked language modelling objective compared with BERT and leads to better down-stream task performances. We adopt here the fine-tuning process for sequence classification, where sequences correspond to texts extracted from images. The special token `[CLS]` is added as a first element of each input sentence, so that BERT associates it a specific embedding. This dense vector represents the entire sentence and is used in input to a linear classifier customized for the target classification task: in MEME Detection and Hate Speech Identification, two classes are considered, while in Event Clustering five classes reflect the target topics. During training, all the network parameters are fine-tuned. BERT and RoBERTa are pre-trained over text in English, and they are able to capture language models specific for this language. In order to apply these architectures in Italian, we investigate several alternative models, pre-trained using document collections in Italian or in multiple languages. Among these models, AlBERTo (Polignano et al., 2019) is a BERT-based model pre-trained over the Twita corpus (Basile and Nissim, 2013) (made of millions of Italian tweets) while GilBERTo[1] and UmBERTo[2] are RoBERTa-based models pre-trained over the OSCAR corpus and the Italian version of Wikipedia, respectively. Among the multi-lingual models, we investigate multilingual BERT (mBERT) (Pires et al., 2019) and XLM-RoBERTa (Conneau et al., 2020) which extends the corresponding pre-training over texts in more than 100 languages.

Regardless of the adopted Transformer-based architecture, we also investigated the adoption of additional annotated material to support the training of complex networks over very short texts extracted from MEMEs. In particular, in Hate Speech Identification, we used an external dataset which addressed the same task, but within a different source. We thus adopted a dataset made available within the Hate Speech Detection (HaSpeeDe) task (Bosco et al., 2018) which involves the automatic recognition of hateful contents in Twitter (HaSpeeDe-TW) and Facebook posts (HaSpeeDe-FB). Each investigated architecture is trained for few epochs only over on the HaSpeeDe dataset before the real training is applied to the DANKMEMES material. In this way, the neural model, which is not specifically pre-trained to detect hate speech, is expected to improve its "expertise" in handling such a phenomenon (even though using material derived from a different source) before being specialized on the final DANKMEMES task[3].

We trained UmBERTo both on HaSpeeDe-TW and on HaSpeeDe-FB and on the merging of these, too. Initial experiments suggested that a higher accuracy can be achieved only considering the material from Facebook (HaSpeeDe-FB). We suppose this is mainly due to the fact that messages from HaSpeeDe-FB and DANKMEMES share similar political topics. As for a CNN, once the Transformer-based architecture is fine-tuned on the new task, it can be used as text encoder, by removing the final linear classifier and selecting the embedding associated to the `[CLS]` token. These vectors will be used in UNITOR in combination with the embeddings derived from the CNN architecture, as described hereafter.

Combining visual and semantic evidences. UNITOR adopts an approach similar to the Feature Concatenation Model (FCM) already seen in (Oriol et al., 2019; Gomez et al., 2020) to combine visual and textual information. For each subtask, the specific CNN achieving best results on the development set is selected, among the investigated ones. The same happens for the Transformer-based architectures. When the "best" architectures are selected and fine-tuned for visual and textual analysis, these are used to encode the entire dataset. It allows training a new classifier which accounts on the evidences from both aspects. In UNITOR these encodings are concatenated, so that the final classifier is a Multi-layered Perceptron[4]. Only this final classifier is fine-tuned, as the remaining parameters are supposed to be already optimized for the task. Future work will consider the fine-tuning of all the parameters of this combined network, here ignored for the (too) high computational cost required from this more elegant approach. It must be said that other information is available in the competition: for example, each MEME was supported with its publication date or the list of politicians appearing in the picture. We investigated the manual definition of feature vectors to be added in the concatenation described above. Unfortunately, these vectors did not provide any significant impact during our experiments, so we only relied on visual and textual information. We suppose this additional information it is too sparse (given the dataset size) to provide any valuable evidence.

Modelling Event Clustering as a Classification task. While Event Clustering may suggest a straightforward application of unsupervised algorithms, we adopted a supervised setting, by imposing the hypothesis that train and test datasets share the same topics. We modelled this subtask as a classification problem, where each MEME is to be assigned to one of the five classes reflecting the underlying topic. UNITOR implements two different approaches. In a first model, the same setting adopted in the other subtasks is used: a CNN and a Transformer-based are optimized on the Task 3 and used as encoder to train the final

---

[1] `https://huggingface.co/idb-ita/gilberto-uncased-from-camembert`

[2] `https://huggingface.co/Musixmatch/umberto-wikipedia-uncased-v1`

[3] An alternative approach consists in adding the messages from HaSpeeDe to the training set: this approach led to lower results, not reported here due to lack of space.

[4] We investigated also more complex combinations, such as the weighed sum, or point-wise product of embeddings, but lower results were obtained.

MLP classifier. Unfortunately, most of the texts are really short to be valuable in the final classification. We thus adopted a second model which is inspired by the capability of BERT-based models to effectively operate over text pairs, achieving state-of-the-art results in tasks such as in Textual Entailment and in Natural Language Inference tasks (Devlin et al., 2019). In this second setting, each input MEME generates five pairs (one for each topic) which are in the form ⟨*topic definition*, *text*⟩. Let us consider the example *"ma come chi sono? presidé só io senza fotoscioppe!"*, associated to the topic #2, defined[5] as *"L'inizio delle consultazioni con i partiti politici e il discorso al Senato di Conte"*. It generates new inputs in the form "`[CLS]` *ma come chi . . . fotoscioppe!* `[SEP]` *L'inizio delle . . . Senato di Conte.* `[SEP]`" which defines sentence pairs in BERT-like architectures. The same approach is applied with respect to each topic. In other words, the original classification problem over five classes is mapped to a binary classification one: each pair is a positive example when the text is associated to the correct topic, negative otherwise. In this way, we expected to detect a possible "semantic connection" between the extracted text and the paired (correct topic) description. At classification time, for each MEME, five new examples are derived (one per topic) and classified. The one generated by the topic receiving the highest softmax score is selected as output.

3 Experimental evaluation and results

UNITOR participated to all subtasks within DANKMEMES. For parameter tuning, we adopted a 10-cross fold validation, so that the training material is divided in 10 folds, each split according to 90%-10% proportion. The model is trained using a standard Cross-entropy Loss and an ADAM optimizer initialized with a learning rate set to $2 \cdot 10^{-5}$. We trained the model for 5 epochs, using a batch size of 32 elements. When combining the networks, the number of hidden layers in the MLP classifier is tuned between 1 and 3. At test time, for each task, an Ensemble of such classifiers is used: each image is in fact classified using all 10 models trained in the different folds and the label suggested by the highest number of classifiers is selected. UNITOR is implement using pytorch[6].

System	Precision	Recall	F1	Rank
UNITOR-R2	**0.8522**	**0.8480**	**0.8501**	1
SNK-R1	0.8515	0.8431	0.8473	2
UNITOR-R1	0.8390	0.8431	0.8411	4
Baseline	0.5250	0.5147	0.5198	-

Table 1: UNITOR Results in Task 1.

Task 1 - MEME Detection. For the subtask 1, the training dataset counts 1,600 examples, equally labelled as "`MEME`" and "`NotMEME`". Results of UNITOR is reported in Table 1, where results are evaluated in terms of Precision, Recall and F1-measure, calculated over the binary classification task (this last used to rank systems). The last row reports a baseline model which randomly assigns labels to images. MEMEs generally adhere to specific visual conventions, where the meaning of text is secondary: as a consequence, our first model (UNITOR-R1) only relies on an image classifier. In particular, it corresponds to the fine-tuning of EfficientNet-B3 over the official dataset. In order to improve the robustness of such a CNN, we adopted a simple data augmentation technique, by duplicating the training material and horizontally mirroring it. UNITOR-R1 ranked forth (over 10 submissions) in the competition. This clearly confirms the effectiveness of EfficientNet, combined with the adopted Ensemble technique. We also investigated larger variants of EfficientNet but they did not outperform the B3 variant: we suppose these larger architectures are more exposed to over-fitting, also considering the dataset size.

Moreover, we adopted a model that combines the output of EfficientNet-B3 with a Transformer-based architecture. Among all the investigated architecture, AlBERTo achieved the highest classification accuracy. Once tuned (in the same 10-cross fold evaluation schema) it is used to encode the entire dataset and the embeddings are concatenated to the ones from EfficientNet-B3. This enables the training of 10 MLPs (one per fold) whose Ensemble defines UNITOR-R2, which ranked first in the task, with a F1 of 0.8501. The overall results thus confirm also the beneficial (although limited) impact of textual information in this subtask.

Task2 - Hate Speech Identification. The training dataset available for the subtask 2 contains 800 training examples, labelled as "`Hate`" and "`NotHate`", while the test dataset counts 200 ex-

---

[5] In a simplified English: *"Are you seriously asking who I am? Mr President, it's me without Photoshop effects!"*

[6] `https://pytorch.org/`

amples. In Table 2 the results obtained by UNITOR are reported, according to the same metrics adopted in Task 1. Unlike the first subtask, Hate Speech is more related to the textual information. Even the baseline is given by the performance of a classifier labelling a MEME as offensive whenever it includes at least a swear word (resulting in a system with a high Precision and a very low Recall).

System	Precision	Recall	F1	Rank
UNITOR-R2	**0.7845**	0.8667	**0.8235**	1
UNITOR-R1	0.7686	**0.8857**	0.8230	2
UPB	0.8056	0.8286	0.8169	3
Baseline	0.8958	0.4095	0.5621	-

Table 2: UNITOR Results in Task 2.

In this task, we adopted UmBERTo (pre-trained over Wikipedia), fine-tuned for 3 epochs over the HaSpeeDe dataset and then for 3 epochs over the DANKMEMES dataset. Again, a 10-cross fold schema is adopted and the final ensemble of such UmBERTo models originated UNITOR-R1, which ranked 2 over 5 submissions. The improvements with respect to the first competitive system confirms the robustness of the adopted Transformer-based architecture combined with the adopted auxiliary training step. We thus combined this model with a CNN (here ResNET152) to exploit also visual information as for the previous subtask. This combination originated UNITOR-R2, which again provided the best results in the competition, even though a very little margin is obtained w.r.t. UNITOR-R1.

Task3 - Event Clustering. The training dataset available for the subtask 3 contains 800 training examples for the 5 targeted topics and a test dataset made of 200 examples. In Table 3 the performances of UNITOR are reported, as for the previous subtask. Since it is a multi-class classification task, each system is evaluated with respect to each of the 5 labels in a binary setting and then the macro-average is applied to Precision, Recall and F1. Here, the baseline is given by a classifier labelling every MEME as belonging to the most represented class (i.e. topic 0, containing miscellaneous examples). Its results, i.e. a F1 of 0.1297, suggest this is a very challenging task, where the dataset is quite limited, especially considering the overlap that exists among all political topics. In the first row, the run UNITOR-R1 is reported: it corresponds to a model that combines the embeddings from ResNET152 and those obtained by AlBERTo, both achieving best accuracy in our initial tuning within this subtask. UNITOR-R1 ranked first (among three submissions) in this competition with a F1 of 0.2657, which doubles the result obtained from the baseline. It must be said that the Transformer achieves significantly better results with respect to the CNN, suggesting that the visual information is negligible also in this subtask[7]. We thus evaluated a model which considers only text, by fine-tuning an AlBERTo model adopting the pair-based approach presented in Section 2, where each text is associated with the description of the topic. Unfortunately, this model, namely UNITOR-R2, under-performed the first submission, with a F1 of 0.2183.

System	Precision	Recall	F1	Rank
UNITOR-R1	**0.2683**	**0.2851**	**0.2657**	1
UNITOR-R2	0.2096	0.2548	0.2183	2
Baseline	0.0960	0.2000	0.1297	-

Table 3: UNITOR Results in Task 3.

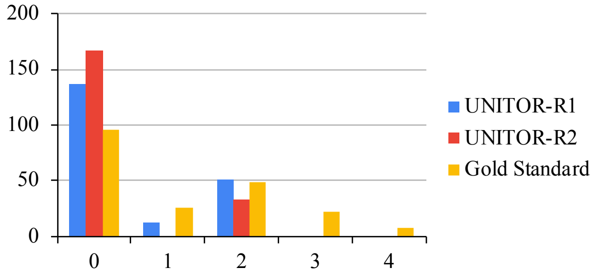


Figure 1: Distribution of labels and classifications in Task 3.

For an error analysis, we compared the assignments provided in the test set and the ones derived from UNITOR, as shown in Figure 1. First, it is clear that the dataset is highly unbalanced, with half of the examples assigned to the class with uncertain topics. Moreover, it can be seen that the combination of textual and visual information makes UNITOR-R1 more robust in detecting topic 2, and most importantly, topic 1, which is ignored from UNITOR-R2. Topics 3 and 4 are ignored by UNITOR but they are also under-represented in the training material. UNITOR-R2 seems more conservative with respect to the largest class (topic 0): it is clear that the repetition of the same topic over many examples introduced a bias. Future work will consider the adoption of more expressive and varied topic descriptions to be paired with texts: for examples, we will select headline news that can be retrieved using Retrieval Engines (e.g.,

---

[7] These results are not reported for lack of space.

by querying with the topic description) to have a more expressive representation of the topics.

4 Conclusions

This work presented the UNITOR system participating to DANKMEMES task at EVALITA 2020. UNITOR merges visual and textual evidences by combining state-of-the-art deep neural architectures and ranked first in all subtasks defined in the competition. These results confirm the beneficial impact of the adopted Convolutional and Transformer-based architecture in the automatic recognition of MEMEs as well as in Hate Speech Identification or Event Clustering. Future work will investigate multi-task learning approaches to combine the adopted architectures in a more principled way.

References

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on italian tweets. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, Atlanta.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Cristina Bosco, Felice Dell'Orletta, Fabio Poletto, Manuela Sanguinetti, and Maurizio Tesconi. 2018. Overview of the EVALITA 2018 hate speech detection task. In *Proceedings of EVALITA 2018, Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL 2020, Online, July 5-10, 2020*, pages 8440–8451.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL 2019*, pages 4171–4186, Minneapolis, Minnesota, June.

R. Gomez, J. Gibert, L. Gomez, and D. Karatzas. 2020. Exploring hate speech detection in multimodal publications. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1459–1467.

Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press, Cambridge, MA, USA.

K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

L. Jiao and J. Zhao. 2019. A survey on the new generation of deep learning in image processing. *IEEE Access*, 7:172231–172263.

Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Martina Miliani, Giulia Giorgi, Ilir Rama, Guido Anselmi, and Gianluca E. Lebani. 2020. Dankmemes @ evalita2020: The memeing of life: memes, multimodality and politics). In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Benet Oriol, Cristian Canton-Ferrer, and Xavier Giró i Nieto. 2019. Hate speech in pixels: Detection of offensive memes towards automatic moderation. In *NeurIPS 2019 Workshop on AI for Social Good*, Vancouver, Canada, 09/2019.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019. AlBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*, volume 2481. CEUR.

Andrew Ross and Damian J. Rivers. 2017. Digital cultures of political participation: Internet memes and the discursive delgitimization of the 2016 u.s. presidential candidates. *Discourse, Context and Media*, 16:1–11, 01.

Limor Shifman. 2013. Memes in a digital world: Reconciling with a conceptual troublemaker. *J. Comput. Mediat. Commun.*, 18:362–377.

Mingxing Tan and Quoc V. Le. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *arXiv e-prints*, page arXiv:1905.11946, May.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2016. Aggregated Residual Transformations for Deep Neural Networks. *arXiv e-prints*, page arXiv:1611.05431, November.

CONcreTEXT: Concreteness in Context

CONcreTEXT @ EVALITA2020: The Concreteness in Context Task

Lorenzo Gregori
University of Florence
lorenzo.gregori@unifi.it

Maria Montefinese
University of Padua
maria.montefinese@unipd.it

Daniele P. Radicioni
University of Turin
daniele.radicioni@unito.it

Andrea Amelio Ravelli
Istituto di Linguistica Computazionale
"Antonio Zampolli" (ILC–CNR) - ItaliaNLP Lab
andreaamelio.ravelli@ilc.cnr.it

Rossella Varvara
University of Florence
rossella.varvara@unifi.it

Abstract

Focus of the CONcreTEXT task is conceptual concreteness: systems were solicited to compute a value expressing to what extent target concepts are concrete (i.e., more or less perceptually salient) within a given context of occurrence. To these ends, we have developed a new dataset which was annotated with concreteness ratings and used as gold standard in the evaluation of systems. Four teams participated in this first edition of the task, with a total of 15 runs submitted.

Interestingly, these works extend information on conceptual concreteness available in existing (non contextual) norms derived from human judgments with new knowledge from recently developed neural architectures, in much the same multidisciplinary spirit whereby the CONcreTEXT task was organized.

1 Introduction

Concept concreteness – that is, how directly a concept is related to sensorial experience (Brysbaert et al., 2014a)– is a fundamental dimension of conceptual semantic representation that has attracted more and more interest and attention in psycholinguistics in the last decade. This dimension is usually assessed by participants ratings on a Likert scale: concrete concepts lie herein on one side of the scale and refer to something that exists in reality and can be experienced immediately through the senses; abstract concepts lie on the opposite side of the scale and are grounded in the internal sensory experience and linguistic information. While concrete concepts have direct sensory referents (Crutch and Warrington, 2005) and greater availability of contextual information (Connell et al., 2018; Kousta et al., 2011; Montefinese et al., 2020), abstract concepts tend to be more emotionally valenced (Kousta et al., 2011) and less imageable (Montefinese et al., 2020; Garbarini et al., 2020).

The CONcreTEXT task challenges participants to build NLP systems to automatically assign a concreteness value to words in context. It is aimed at investigating how the concreteness information affects sense selection: different from past research (Brysbaert et al., 2014b; Montefinese et al., 2014), we are interested in assessing the concreteness of concepts within the context of real sentences rather than in isolation. Additionally, the concreteness score is assumed to be a property of meanings rather than a property of word forms; thus, scoring the concreteness of a concept in context implicitly requires to individuate its underlying sense, by handling lexical phenomena such as polysemy and homonymy.

Ordinary experience suggests that concepts' concrete/abstract status can affect their semantic representation, and lexical access and processing: concrete meanings are acknowledged to be more quickly and easily delivered in human communication than abstract meanings (Bambini et al., 2014). Historically, it has been observed that concrete concepts are responded to more quickly than abstract concepts in lexical decision tasks (Bleasdale, 1987; Kroll and Merves, 1986), although more recent experiments have shown that abstract

---

concepts might have an advantage when other variables have been accounted for (Kousta et al., 2011). Concrete concepts are also easier to encode and retrieve than abstract concepts (Romani et al., 2008; Miller and Roodenrys, 2009), are easier to make associations with (de Groot, 1989), and are more thoroughly described in definition tasks (Sadoski et al., 1997). Moreover, it takes generally less time to comprehend a concrete sentence than an abstract one (Haberlandt and Graesser, 1985; Schwanenflugel and Shoben, 1983). Thus, it has been proposed that different organizational principles govern semantic representations of concrete and abstract concepts: concrete concepts are predominantly organized by featural similarity measures, and abstract concepts by associative relations, co-occurrence patterns and syntactic information (Vigliocco et al., 2009).

All surveyed features make aspects ingrained in the distinction between concreteness/abstractness a stimulating and challenging field also for computational linguistics. Among the earliest attempts at grasping concreteness, we find works that investigated on concreteness/abstractness information in its interplay with metaphor identification and figurative language more in general (Turney et al., 2011) (and, more recently (Mensa et al., 2018b)). Although concreteness information is acknowledged to be central to, e.g., word-sense induction and compositionality modeling (Hill et al., 2013), the contribution of concreteness/abstractness to semantic representations is not fully grasped and exploited in existing approaches and resources, with the notable exception of works aimed *i)* at learning multimodal embeddings, and how abstract and concrete representations can be acquired by multi-modal models (Hill and Korhonen, 2014); and *ii)* at exploring in how far concreteness information is represented in the distributional patterns in *corpora* (Hill et al., 2013). Moreover, some approaches exist that attempted to create lexical resources by also employing common-sense information (Mensa et al., 2018a; Colla et al., 2018).

Characterizing tokens within sentences with their concreteness requires integrating both word-specific and contextual information. In our view, the CONCRETEXT Task entails dealing with a relaxed form of word sense disambiguation; such aspects were faced by our participants by devising methods relying on both traditional knowledge-based approaches, and more recent language models and sequence-to-sequence models. Finally, like in many real-world cases, the provided trial data is rather scarce, in the order of hundred sentences for the Italian language, and as many for English. This aspect forced our participants to face something similar to a 'cold start' problem. We hope that this edition of the CONCRETEXT task will be the first appointment in a series for those who are interested in the issues posed by the contextual conceptual concreteness to research on natural language semantics.

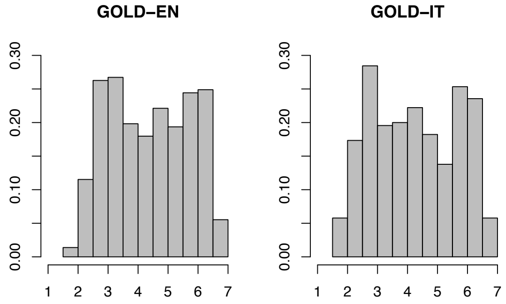
2 Task Definition

The task CONCRETEXT (so dubbed after CONcreteness in conTEXT) focuses on automatic concreteness (and conversely, abstractness) recognition. Given a sentence along with a target word, we asked participants to propose a system able to assess the concreteness of a concept expressed by a given word within a sentence, on a 7-point Likert-like scale where 1 stands for completely abstract (e.g., 'freedom') and 7 for completely concrete (e.g., 'car'). For example, in the sentence "In summer, wheat *fields* are coloured in yellow" the noun *field* refers to an entity that can smell, be touched, and pointed to. In this case, in a scale ranging from 1 to 7 its concreteness may be evaluated as 7, because it refers to an extremely concrete concept. In contrast, the same noun *field* in the sentence "Physics is Alice's research *field*" refers to a scientific subject, i.e., something that cannot be perceived through the five senses, but that can be explained through a linguistic description. In this sentence, the noun *field* may be evaluated 1 because it refers to an extremely abstract concept. Moreover, the task targets can be halfway between completely abstract and completely concrete, as in the case of "Magnetic *field* attracts iron", where the noun *field* refers to something more abstract compared to "wheat *fields*" but more concrete compared to "research *field*". As anticipated, the concreteness score being assigned to the word should be evaluated in context: the word should not be considered in isolation, but as part of a given sentence.

Participants were invited to exploit all possible strategies to solve the task, including (but not limited to) knowledge bases, external training data, word embeddings, etc.

Table 1: Basic statistics on the CONcreTEXT dataset used as gold standard.

	Italian	English
Unique Verb targets	52	44
Unique Noun targets	96	73
Num. Sentences	550	534
Num. Sentences Verb target	189	210
Num. Sentences Noun target	361	324
Avg. sent. length	14.43	14.33
Avg. sent. length (no punct)	13.03	12.87
Avg. full words per sent.	7.14	7.15
Num. Annotators	333	310
Human ratings (HR)	18,726	16,522
Min HR per sentence	30	30



(a) English dataset. (b) Italian dataset.

Figure 1: Distribution of human ratings for the English and Italian datasets.

3 Dataset

The dataset used for this task has been taken from the English-Italian parallel section of The Human Instruction Dataset (Chocron and Pareti, 2018), derived from WikiHow instructions.[1] All such documents had been anonymized beforehand, so that downloaded data present no privacy nor data sensitivity issues.

The dataset is composed of overall $1,096$ sentences, arranged as follows: 562 Italian sentences plus 534 English sentences. Each sentence contains a target term (either verb or noun) with its associated concreteness score (1–7 scale). Such score is derived from the average of at least 30 human judgments from native Italian and English speakers about the concreteness of a target word in a given sentence (see Table 1 for the dataset numbers).

The reliability of the collected data within each language (Italian, English) for the trial and test phases was evaluated separately by applying the split-half correlations corrected with the Spearman-Brown formula after randomly dividing the participants into two subgroups of equal size. All the reliability indexes were calculated on $10,000$ different randomizations of the participants. The mean correlations between the two groups are very high for both the trial and test phases, ranging from a minimum of $r = 0.87$ for English (at the test phase) to a maximum of $r = 0.98$ for Italian (at the trial phase), showing that the resulting ratings are highly reliable and can be used across the entire Italian – and English – speaking populations.

The dataset has been split into trial and test data, with a 20–80 ratio. Trial data has been released with the concreteness scores, while the test data has been provided at the beginning of the evaluation window without any score.[2]

4 Evaluation Measures and Baselines

We chose the Spearman correlation indices as our main evaluation measure; for the sake of completeness, we also report Pearson indices (substantially in accord with the previous metrics). We chose the former measure because the collected ratings are not normally distributed, which makes the Spearman correlation more suited to the data. In fact, by running the Shapiro–Wilk test we obtained a p-value < 0.001. The non normal distribution of data is also confirmed by the plot of the gold standard ratings, as illustrated in Figure 1.

Two baselines have been designed for this task.

Baseline One. The first baseline for the Italian language is derived as follows. The fastText word embeddings have been acquired beforehand by training the model on the Italian dump of the WikiHow instructions. We chose fastText for its support to the handling of OOV terms (Bojanowski et al., 2017), which is a crucial feature in the present setting. The cited norms by Montefinese et al. (2014) (referred to as 'the norms' hereafter) have been used herein. The average score of terms in each input sentence $S = \{t_1, t_2, \dots t_K\}$ has been

---

[1] The whole Human Instruction Dataset dataset is freely available on Kaggle, `https://www.kaggle.com/paolop/human-instructions-multilingual-wikihow`

[2] The dataset employed in the CONcreTEXT task is available at the URL `https://lablita.github.io/CONcreTEXT/`.

computed by scrolling through the content words of the sentence. Each term t is searched in the norms: if the term is found, the associated concreteness score $c(t)$ is returned; otherwise, if the term is not present in the norms, the ranking of the l ($l = 20,000$) elements most similar to t is generated through fastText. In this case, we scan the whole norms list and employ the concreteness score of the element in the norms closest to those in the fastText ranking. In either case we obtain a score for each and every term in the input sentence, so that the concreteness score of the target token $\hat{t}$ is computed as the averaged score of the terms in the input sentence:

$$c(\hat{t}) = \frac{1}{K} \cdot \sum_{i=1}^{K} c(t_i).$$

The first baseline for the English language is analogous to the Italian one, except for the fact that the English tokens from the norms are accessed in this case. The same strategy governs the handling of the fastText resource, that in this case has been trained on the English dump of the Human Instruction Dataset.

Baseline Two. The second baseline for the Italian language implements a simple lookup function. More specifically, input sentences have been translated into English through the Google Translate ajax API implementation, and then the concreteness scores associated to the terms in the norms by Brysbaert et al. (2014b) are retrieved (in the unlikely case the term is not found, it is dropped, thus not contributing to the final score). The concreteness score of the target term is thus assigned to the average concreteness of terms in the given input sentence. The baseline two for the English language employs the concreteness score —by also employing the norms by Brysbaert et al. (2014b)— associated to all terms in the input sentence, finally assigning to the target token the average concreteness score for the whole sentence.

5 Systems Descriptions

In this Section we briefly describe the systems that participated in the competition. As a first edition, the CONCRETEXT task recorded a good feedback from the community, with 4 teams, overall 7 participants and 15 submitted system runs. In the next Section we report the results obtained by all such systems, while anonymizing a withdrawn participant.

5.1 ANDI

The ANDI team (Rotaru, 2020) proposed a system based on multiple classes of concreteness score predictors. The first class of predictors has been derived from large datasets of behavioral norms, collected for a wide variety of psycholinguistic factors. Beside well known concreteness norms, ANDI takes into account also semantic diversity, age of acquisition, emotional and sensori-motor dimensions, as well as frequency and contextual diversity counts. The vocabulary resulting from the merging of these words collections comprises more than 70K words, and it is the base vocabulary used to extract all the predictors. The second class of predictors has been derived from context-independent distributional models, namely Skip-gram, GloVe, and NumberBatch embeddings, as well as from the concatenation of the three. The third class of predictors has been derived from features obtained through recent transformers models, i.e. context-dependent representations. The models exploited are: BERT, GPT-2, Bart, and ALBERT. The final rating has been computed through a ridge regression over the three classes.

5.2 CAPISCO

The CAPISCO Team (Bondielli et al., 2020) submitted 3 systems for both Italian and English.

NON-CAPISCO. The first system computes a variation of the Baseline Two; that is, the target concreteness is obtained by combining the concreteness value of the target term (taken in isolation), and the average concreteness of the whole sentence. Improvement from baseline comes from considering differently the weight of the concreteness of the target term and of the context.

CAPISCO-CENTROIDS. This system is based on the assumption that close semantic spaces are featured by similar concreteness scores. In this case the authors first build two centroids, one for concrete and one for abstract concepts based on the norms by Brysbaert et al. (2014b) and Della Rosa et al. (2010), by employing fastText pretrained embeddings. The concreteness score of a term is then computed by averaging the distance of the first 50 lexical substitutes of the target (identified through BERT) from the two polarized centroids. Introducing a list of target substitutes in a given context is thus the gist of this approach.

CAPISCO-TRANSFORMERS. In this variant, the CAPISCO team fine-tuned a pre-trained BERT model on the concreteness rating task, by complementing the CONCRETEXT training data with newly generated training data. The new data generation is twofold: for each original sentence, new sentences are generated by replacing the target term with the first lexical substitutes derived with BERT target masking approach. Then, more sentences are borrowed from Italian and English reference corpora.

5.3 KONKRETIKA

The KONKRETIKA team (Badryzlova, 2020) presented a system that first assigns a concreteness and an abstractness score to the target lemma, and then it adjusts these values based on the surrounding context. In the first step, the system computes semantic similarity between the target vectors and a "seed list" consisting of abstract and concrete words (extracted from the MRC Psycholinguistic Database). In the second step, the values where adjusted to the sentential context considering the mean concreteness index of the entire sentence. The team submitted 4 runs based on a heuristically selected coefficient.

6 Results

Four teams participated in the CONCRETEXT competition: ANDI, CAPISCO, KONKRETIKA, and a withdrawn team. ANDI and CAPISCO developed a system for both languages (English and Italian), while KONKRETIKA participated in the English track only, and the same did the withdrawn participant. Each team was allowed to submit the output of up to 4 system runs; the final ranking has been compiled based on the results of the best run.

In Tables 2 and 3 we present the score of each run for the English and Italian language, respectively. Although, as mentioned, the Spearman indices were adopted as our main evaluation metrics, we also report Pearson correlation indices and Euclidean distance, that may be useful to complete the assessment of the results. The final ranking is provided in Tables 4 and 5.

We can observe a substantial agreement between Spearman and Pearson indices: the averaged delta between such figures amounts to 0.012 and to 0.008 on the English and Italian dataset, respectively. Also the Euclidean distance seems to substantially confirm the results: for the results on English (Table 2) it is minimal for the output of the ANDI system, and it increases while Spearman correlation values decrease. The same trend is also confirmed on Italian results (Table 3).

Table 2: Results for each run on English test set.

System run	Spear	Pears	Eucl.D
ANDI	**0.833**	**0.834**	**15.409**
NON-CAPISCO	0.785	0.787	35.663
KONKRETIKA_3	0.663	0.668	28.613
KONKRETIKA_1	0.651	0.667	29.933
Baseline_2	0.554	0.567	38.451
KONKRETIKA_4	0.542	0.545	29.836
CAPISCO_CENTR	0.542	0.538	48.864
KONKRETIKA_2	0.541	0.545	30.322
CAPISCO_TRANS	0.504	0.501	29.927
Baseline_1	0.382	0.377	31.738
withdrawn_run3	-0.013	0.067	41.109
withdrawn_run1	-0.124	-0.123	44.068
withdrawn_run2	-0.127	-0.129	43.890

Table 3: Results for each run on Italian test set.

System run	Spear	Pears	Eucl.D
ANDI	**0.749**	**0.749**	**19.950**
CAPISCO_TRANS	0.625	0.617	24.367
CAPISCO_CENTR	0.615	0.609	28.608
NON-CAPISCO	0.557	0.557	31.588
Baseline_2	0.534	0.522	40.114
Baseline_1	0.346	0.368	31.046

Tables 6 and 7 report disaggregated Spearman correlations for verbs and nouns. This allows to highlight if and to what extent the participating systems obtained better results on either POS. ANDI obtained the best results on both verbs and nouns in both languages. This system (and NON-CAPISCO as well) obtained analogous results on verbs and nouns. On the whole, the rest of the systems obtained results clearly better on English verbs and slightly better on Italian nouns. In particular, KONKRETIKA (English only) is strongly biased on verbs: its performances on verbs are higher in all 4 runs. CAPISCO systems exhibit the most varied behavior.

7 Discussion

The obtained results confirm transformers as a good device to compute concreteness score for words in context. The virtues of transformers in grasping contextual information are largely

Table 4: Final ranking on English test set.

Team	Spear	Pears	Eucl.D
ANDI	**0.833**	**0.834**	**15.409**
CAPISCO	0.785	0.787	35.663
KONKRETIKA	0.663	0.668	28.613
withdrawn	-0.013	0.067	41.109

Table 5: Final ranking on Italian test set.

Team	Spear	Pears	Eucl.D
ANDI	**0.749**	**0.749**	**19.950**
CAPISCO	0.625	0.617	24.367

Table 6: Spearman rank differences between nouns and verbs on English test set.

	Spear.N	Spear.V	Diff
CAPISCO_TRANS	0.443	0.654	0.211
KONKRETIKA_4	0.502	0.701	0.199
KONKRETIKA_2	0.502	0.683	0.181
CAPISCO_CENTR	0.478	0.659	0.181
KONKRETIKA_3	0.629	0.762	0.133
KONKRETIKA_1	0.611	0.741	0.13
ANDI	0.836	0.857	0.021
NON-CAPISCO	0.779	0.782	0.003

Table 7: Spearman rank differences between nouns and verbs on Italian test set.

	Spear.N	Spear.V	Diff
NON-CAPISCO	0.579	0.507	0.072
CAPISCO_TRANS	0.607	0.667	0.060
CAPISCO_CENTR	0.625	0.591	0.034
ANDI	0.762	0.749	0.013

known, but in the present setting we observe that their output can be further improved by integrating behavioral information (this seems to be one major difference between the systems ANDI and CAPISCO-TRANSFORMERS).

The most important output of this challenge is definitely the great performance of the ANDI system, that proves to be robust and reliable for the considered task: the system obtains the best ranking in both languages, a low deviation from the gold standard and a substantial stability in processing both verbs and nouns. Moreover, the proposed system is ready to be applied in a multi-language environment, given that non-English sentences are automatically translated into English. The ANDI system exploits different kinds of available resources and works with local and contextual information. This shows that deriving the concreteness score of a word in context is a complex task, involving different semantic, cognitive and experiential levels.

The high correlation obtained by the NON-CAPISCO in the English task is somehow surprising, since this system makes use only of the mean concreteness of the sentence (computed from existing norms) as contextual information. This result is thus related to the availability of existing norms, but it shows that there is a link between the concreteness score of a target word in context and the concreteness scores of the words it occurs with. Further analysis are needed, but it suggests that concrete interpretations of a target word are associated with concrete context words. Of course, systems based exclusively on behavioral norms are strongly dependent on the coverage of the considered vocabulary. In fact, the NON-CAPISCO Italian performances (obtained exploiting a $\sim$ 1.2K vocabulary) are lower than all the other systems, while on the English track it ranks second (using a $\sim$ 70K vocabulary).

8 Conclusions

We presented the results of the CONCRETEXT task at EVALITA 2020 (Basile et al., 2020). The task challenges participants to build NLP systems to automatically assign a concreteness score to words in context, evaluating to what extent target concepts are concrete (i.e., more or less perceptually salient) within a given context of occurrence. A novel dataset was developed for this task as a multilingual comparable *corpus* composed of 550 Italian sentences and 534 English sentences, annotated with the concreteness/abstractness rating of target nouns and verbs. Three teams completed their participation to the task, obtaining the following ranking: ANDI (Rotaru, 2020), CAPISCO (Bondielli et al., 2020), and KONKRETIKA (Badryzlova, 2020).

Future work will address the following steps. First of all, we will improve our dataset by including further languages, also from different language families and under-resourced languages. Also the set of considered targets should be expanded, to ensure a broader coverage to the dataset, and more significant results (thanks to the larger experimental base) to its future users as well.

References

Yulia Badryzlova. 2020. KonKretiKa @ CONcreTEXT: Computing concreteness indexes with sigmoid transformation and adjustment for context. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Valentina Bambini, Donatella Resta, and Mirko Grimaldi. 2014. A dataset of metaphors from the italian literature: Exploring psycholinguistic variables and the role of context. *PloS one*, 9(9):e105634.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Fraser A Bleasdale. 1987. Concreteness-dependent associative priming: Separate lexical organization for concrete and abstract words. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 13(4):582.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information.

Alessandro Bondielli, Gianluca E. Lebani, Lucia C. Passaro, and Alessandro Lenci. 2020. Capisco @ CONcreTEXT: (Un)supervised Systems to Contextualize Concreteness with Norming Data. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Marc Brysbaert, Michaël Stevens, Simon De Deyne, Wouter Voorspoels, and Gert Storms. 2014a. Norms of age of acquisition and concreteness for 30,000 dutch words. *Acta psychologica*, 150:80–84.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014b. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods*, 46(3):904–911.

Paula Chocron and Paolo Pareti. 2018. Vocabulary alignment for collaborative agents: a study with real-world multilingual how-to instructions. In *IJCAI*, pages 159–165.

D. Colla, E. Mensa, A. Porporato, and D.P. Radicioni. 2018. Conceptual Abstractness: From Nouns to Verbs. In *Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018)*, volume 2253. CEUR.

Louise Connell, Dermot Lynott, and Briony Banks. 2018. Interoception: the forgotten modality in perceptual grounding of abstract and concrete concepts. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1752):20170143.

Sebastian J Crutch and Elizabeth K Warrington. 2005. Abstract and concrete concepts have structurally different representational frameworks. *Brain*, 128(3):615–627.

Annette M de Groot. 1989. Representational aspects of word imageability and word frequency as assessed through word association. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 15(5):824.

Pasquale A Della Rosa, Eleonora Catricalà, Gabriella Vigliocco, and Stefano F Cappa. 2010. Beyond the abstract—concrete dichotomy: Mode of acquisition, concreteness, imageability, familiarity, age of acquisition, context availability, and abstractness norms for a set of 417 italian words. *Behavior research methods*, 42(4):1042–1048.

Francesca Garbarini, Fabrizio Calzavarini, Matteo Diano, Monica Biggio, Carola Barbero, Daniele P Radicioni, Giuliano Geminiani, Katiuscia Sacco, and Diego Marconi. 2020. Imageability effect on the functional brain activity during a naming to definition task. *Neuropsychologia*, 137:107275.

Karl F Haberlandt and Arthur C Graesser. 1985. Component processes in text comprehension and some of their interactions. *Journal of Experimental Psychology: General*, 114(3):357.

Felix Hill and Anna Korhonen. 2014. Learning abstract concept embeddings from multi-modal data: Since you probably can't see what i mean. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 255–265.

Felix Hill, Douwe Kiela, and Anna Korhonen. 2013. Concreteness and corpora: A theoretical and practical study. In *Proceedings of the Fourth Annual Workshop on Cognitive Modeling and Computational Linguistics (CMCL)*, pages 75–83.

Stavroula-Thaleia Kousta, Gabriella Vigliocco, David P Vinson, Mark Andrews, and Elena Del Campo. 2011. The representation of abstract words: why emotion matters. *Journal of Experimental Psychology: General*, 140(1):14.

Judith F Kroll and Jill S Merves. 1986. Lexical access for concrete and abstract words. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 12(1):92.

Enrico Mensa, Aureliano Porporato, and Daniele P. Radicioni. 2018a. Annotating concept abstractness by common-sense knowledge. In Chiara Ghidini, Bernardo Magnini, Andrea Passerini, and Paolo

Traverso, editors, *AI*IA 2018 – Advances in Artificial Intelligence*, pages 415–428, Cham. Springer International Publishing.

Enrico Mensa, Aureliano Porporato, and Daniele P. Radicioni. 2018b. Grasping metaphors: Lexical semantics in metaphor analysis. In Aldo Gangemi, Anna Lisa Gentile, Andrea Giovanni Nuzzolese, Sebastian Rudolph, Maria Maleshkova, Heiko Paulheim, Jeff Z Pan, and Mehwish Alam, editors, *The Semantic Web: ESWC 2018 Satellite Events*, pages 192–195, Cham. Springer International Publishing.

Leonie M Miller and Steven Roodenrys. 2009. The interaction of word frequency and concreteness in immediate serial recall. *Memory & Cognition*, 37(6):850–865.

Maria Montefinese, Ettore Ambrosini, Beth Fairfield, and Nicola Mammarella. 2014. The adaptation of the affective norms for english words (anew) for italian. *Behavior research methods*, 46(3):887–903.

Maria Montefinese, Ettore Ambrosini, Antonino Visalli, and David Vinson. 2020. Catching the intangible: a role for emotion? *Behavioral and Brain Sciences*, 43.

Cristina Romani, Sheila Mcalpine, and Randi C Martin. 2008. Concreteness effects in different tasks: Implications for models of short-term memory. *Quarterly Journal of Experimental Psychology*, 61(2):292–323.

Armand Rotaru. 2020. ANDI @ CONcreTEXT: Predicting concreteness in context for English and Italian using distributional models and behavioural norms. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Mark Sadoski, William A Kealy, Ernest T Goetz, and Allan Paivio. 1997. Concreteness and imagery effects in the written composition of definitions. *Journal of Educational Psychology*, 89(3):518.

Paula J Schwanenflugel and Edward J Shoben. 1983. Differential context effects in the comprehension of abstract and concrete verbal materials. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 9(1):82.

Peter Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 680–690.

Gabriella Vigliocco, Lotte Meteyard, Mark Andrews, and Stavroula Kousta. 2009. Toward a theory of semantic representation. *Language and Cognition*, 1(2):219–247.

ANDI @ CONcreTEXT: Predicting concreteness in context for English and Italian using distributional models and behavioural norms

Armand Stefan Rotaru
Independent researcher
`armand.rotaru@gmail.com`

Abstract

In this paper we describe our participation in the CONcreTEXT task of EVALITA 2020, which involved predicting subjective ratings of concreteness for words presented in context. Our approach, which ranked first in both the English and Italian subtasks, relies on a combination of context-dependent and context-independent distributional models, together with behavioural norms. We show that good results can be obtained for Italian, by first automatically translating the Italian stimuli into English, and then using existing resources for both Italian and English.

1 Introduction

In our everyday life we rarely encounter words in isolation. Instead, we typically process words as part of sentences or phrases, and these linguistic contexts shape our understanding of individual words. However, for various reasons, the overwhelming majority of behavioural norms that have been collected so far focus only on single words or word pairs (Johns et al., 2020).

Thus, the EVALITA 2020 (Basile et al., 2020) CONcreTEXT Task (Gregori et al., 2020) represents a timely and valuable contribution to the study of context-dependent semantics. The task asks competitors to predict subjective ratings of concreteness for words presented within sentences. As mentioned by the organizers, being able to automatically compute contextual concreteness ratings would have a several practical applications, such as identifying the use of figurative language, detecting words that might be difficult to understand for language learners, and allowing tighter control of contextual variables in psycholinguistic experiments.

In this paper we describe our computational models, based on pre-trained distributional models and behavioural norms, which ranked first in both the English and Italian tracks of the competition[1]. We find that the best performance can be obtained by employing a combination of transformer models, developed in the last 2 years. Moreover, for Italian, it is possible to reach good levels of performance by relying on both the original stimuli and their English translation, which allows access to resources for both languages.

1.1 General description

In order to predict concreteness in context, we use information derived from three type of sources, namely behavioural norms and distributional models, both context-independent (i.e., a model outputs the same vector representation for a given word, regardless of the context in which the word is encountered), and context-dependent (i.e., a model outputs a potentially different representations for a given word, as a function of the context in which the word is presented).

Firstly, we employ behavioural norms collected for a wide variety of psycholinguistic factors. Of particular interest to us are norms for concreteness (Brysbaert et al., 2014), semantic diversity (Hoffman et al., 2013), age of acquisition (Kuperman et al., 2012), emotional dimensions (i.e., valence, arousal, and dominance; Mohammad, 2018), and sensorimotor dimensions (i.e., modality strengths for the tactile, auditory, olfactory, gustatory, visual, and interoceptive modalities; interaction strengths for the mouth/throat, hand/arm, foot/leg, head excluding mouth/throat, and torso effectors; Lynott et al., 2019), as well as frequency and contextual diversity counts (Van Heuven et al., 2014).

---

[1] https://github.com/armandrotaru/TeamAndi-CONcreTEXT

We focus on these specific factors since they are meaningfully related to word concreteness (see the previous references).

Secondly, we employ context-independent distributional models, namely Skip-gram (Mikolov et al., 2013), FastText (Bojanowski et al., 2017), GloVe (Pennington et al., 2014), and ConceptNet NumberBatch (Speer et al., 2017). Such models have been used in order to accurately predict a range of psycholinguistic variables, including concreteness ($\rho = .88$; Paetzold & Specia, 2016).

Thirdly, we employ context-dependent distributional models, namely BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2018), AlBERTo (Polignano et al., 2019), GPT-2 (Radford et al., 2019), Bart (Lewis et al., 2019), and ALBERT (Lan et al., 2020). Although they have become extremely popular after achieving human-level performance in various linguistic tasks (e.g., those in the GLUE benchmark; Wang et al., 2018), we are not aware of studies looking at whether such models can accurately predict (contextualized) subjective ratings. Nevertheless, since these models were specifically designed to process rich contextual information, they could be a valuable tool for predicting ratings of concreteness in context.

1.2 Predictors for English

We tested (combinations of) three groups of predictors. The first group was derived from large datasets of ratings for concreteness, semantic diversity, age of acquisition, emotional dimensions, and sensorimotor dimensions, as well as frequency and contextual diversity counts based on the SUBTLEX-UK and BNC corpora (see the references from the beginning of the previous section). In order to extend the coverage of the subjective ratings, we did not directly use them as predictors of concreteness in context. Instead, we relied on the Skip-gram, GloVe, and ConceptNet NumberBatch models, as a means of estimating the subjective ratings for more than 100,000 words, via linear regression. For the frequency and contextual diversity counts, we kept the original values, as they already have very good coverage. The intersection of the two datasets, which includes more than 70,000 words, served as the basis for our predictors of concreteness. More specifically, for each variable V (e.g., semantic diversity), we generated four predictors, namely $V(w)$, $V(c)$, $V(w) * V(c)$, and abs($V(w)$ - $V(c)$), where:

- $V(w)$ denotes the value of V corresponding to the word w (e.g., w = "offend"). If w is not present in our norms, we set $V(w)$ to the average value of V, computed over the entire norms;
- $V(c)$ denotes the value of V corresponding to the context c in which the word w is encountered (e.g., w = "offend", c = "Do not insult or ___ anyone ."). Computing this value involves calculating the average $V(c) = \frac{\sum_{i=1}^{N} V(c_i)}{N}$, where $V(c_i)$ is the value of V corresponding to the i-th context word, calculated as described previously, and N is the number of words that make up the context.

These predictors allowed us to include both the individual contributions of word w and its context c, as well as certain interactions between w and c.

The second group was derived from Skip-gram, GloVe, and ConceptNet NumberBatch embeddings, as well as from the concatenation of the three types of embeddings. The vocabulary of the four models is that described in the discussion above. Given the large number of dimensions involved (i.e., 300 + 300 + 300 + 900 = 1,800), we first extracted the top 20 principal components from each model (although comparable results can also be obtained by using a larger number of components). Then, for each variable V (e.g., PC_3 from the GloVe model) we generated four predictors, namely $V(w)$, $V(c)$, $V(w) * V(c)$, and abs($V(w)$ - $V(c)$), following the same procedure as in the previous discussion. In addition, based on (Frassinelli et al., 2017), for each distributional model we added four predictors based on a measure of neighbourhood density (i.e., the mean cosine similarity between a vector and its closest 20 vectors), using the same procedure as described above.

The third group was derived from the BERT, GPT-2, Bart, and ALBERT models. We used the standard (base) versions of each model (i.e., without task-specific fine-tuning), as described in the original papers, and obtained from the Hugging Face repository (https://huggingface.co/models).

Unlike for the previous two groups, the predictors consist only of a word's activations from the last hidden layer (i.e., for the GPT-2, Bart, and ALBERT models), or averaged from the last four hidden layers (i.e., for the BERT model).

Importantly, for each group of predictors we generated two sets of variables, based on two versions of the target words (i.e., the words rated by the participants). In the first set we used the uninflected form of the target words, taken from the TARGET column. In contrast, in the second set of we used the inflected form of the target words, taken from the words in the TEXT column located at the positions specified in the INDEX column. More details can be found in Table 1.

For predicting ratings of concreteness in context, we employed ridge regression, with large values of the parameter lambda (i.e., strong regularization), after standardized all the variables.

1.3 Predictors for Italian

Our approach was similar to that for English, but with certain significant changes, as follows:

- for the first group of predictors, we began by automatically translating the Italian stimuli (i.e., the TARGET and TEXT columns) into English, using the MarianMT translation model (Junczys-Dowmunt et al., 2018). Next, for the translated stimuli we derived the predictors using the exact same procedure as in the case of English;
- for the second group of predictors, we employed Italian versions of the FastText and ConceptNet NumberBatch models), together with their concatenation. We derived the predictors based on the top 30 principal components for each model, rather than the top 20 principal components, as in the case of English (although comparable results can also be obtained by using a larger number of components);
- for the third group of predictors, we again employed the English translations and relied on the same models as for English, and also the RoBERTa model. For the BERT model, we only used the activations from the last hidden layer. We also added the AlBERTo model, but with the Italian stimuli.

As in the case for English, we generated two sets of predictors, using either the uninflected or inflected forms of the target words, together with their corresponding English translations. More details can be found in Table 1.

Once more, we employed ridge regression, with large values of the parameter lambda (i.e., strong regularization), after standardizing all the variables.

2 Results and discussion

The results for English and Italian are shown in Figures 1 and 2, respectively, for various sets of predictors and regularization strengths. Results are averaged over 1,000 rounds of 5-fold cross-validation, using only the training dataset.

For English, the results indicate that context-dependent models (Fig. 1c-d) outperform behavioural norms (Fig. 1a) and context-independent models (Fig. 1b). For the latter, even though we introduced contextual variables by averaging a given variable (e.g., concreteness) over the words that make up the context, it appears that this simple average does not properly capture contextual information and/or interactions between single word and contextual information. The addition the behavioural norms and/or context-independent models has a negligible effect on performance (Fig. 1e). In this respect, the excellent results for context-dependent models are likely due to several factors, such as the highly non-linear integration of contextual information, the use of attention mechanisms, and that of more sophisticated learning objectives (e.g., next sentence prediction).

Interestingly, predictors based on inflected targets consistently outperform those based on uninflected targets, especially for the context-dependent models. This shows that morphological information can be quite valuable. Also, even for the largest sets of predictors, consisting of more than 3,200 variables per 80 data points, the degree of regularization appears to matter very little, indicating surprisingly small levels of overfitting.

In the case of Italian, the findings are somewhat different from those for English. Performance is roughly 10% lower than that for English. This is expected, given that perfect translation from Italian to English is impossible, and that the majority of predictors depend on this translation. The gaps in performance between predictors for inflected vs uninflected targets (Fig. 2c-d), and between the various classes of predictors (Fig. 2a-e), are also smaller. Moreover, the performance of context-dependent models can be increased to a small degree by adding behavioural norms and/or context-independent models (Fig. 2f).

Our best models, as described in Figures 1 and 2, ranked first in both the English track ($\rho = .83$), and the Italian track ($\rho = .75$). The two correlations are smaller than those for the best models in the two figures, but this is likely to be an effect of distributional differences between the training set and the test set.

3 Conclusion

Our results suggest that a variety of approaches can be quite successfully employed in order to predict concreteness in context. The most effective predictors are those derived from context-dependent models (e.g., BERT), but relatively good results can be obtained also by using context-independent models (e.g., Skip-gram) and behavioural norms (e.g., ratings of semantic diversity).

Such an approach works very well for English, but less so for Italian, where the range of available predictors (i.e., pre-trained distributional models and large behavioural norms) is limited. One surprisingly effective solution to this problem is to simply translate the Italian stimuli into English, by relying on a neural machine translation system (e.g., MarianMT), and then make use of existing predictors for English. As an alternative to translating stimuli, it would be interesting to test whether comparable results can be obtained using multilingual versions of context-dependent models, such as BERT.

Acknowledgements

We would like to thank the anonymous reviewers, for their comments and suggestions, as well as the organizers of the competition, for their support.

Table 1. Type and number of predictors obtained from behavioural norms and distributional models. The same number of predictors are derived for both the inflected and uninflected versions of the target word. As predictors for the context-dependent models, we use the activations associated with the target, when presented in context (i.e., we do not have separate predictors for the target, context, and their potential interactions). More details regarding each set of predictors can be found in Subsections 2.2 and 2.3, as well as in Figures 1 and 2.

<table>
<tr><th colspan="5"><b>Predictors for English</b></th></tr>
<tr><td>Source of predictors</td><td># preds. V(w)</td><td># preds. V(c)</td><td># preds. V(w) * V(c)</td><td># preds. abs(V(w) - V(c))</td></tr>
<tr><td>Behavioural norms (frequency, etc.)</td><td>20</td><td>20</td><td>20</td><td>20</td></tr>
<tr><td>Skip-gram (Google News – 100B)</td><td>21</td><td>21</td><td>21</td><td>21</td></tr>
<tr><td>GloVe (Common Crawl – 840B)</td><td>21</td><td>21</td><td>21</td><td>21</td></tr>
<tr><td>ConceptNet NumberBatch (ConceptNet + Skip-gram + GloVe)</td><td>21</td><td>21</td><td>21</td><td>21</td></tr>
<tr><td>Concatenation of Skip-gram, GloVe, and ConceptNet NumberBatch</td><td>21</td><td>21</td><td>21</td><td>21</td></tr>
<tr><td>ALBERT (last hidden layer)</td><td colspan="4">768</td></tr>
<tr><td>Bart (last hidden layer)</td><td colspan="4">768</td></tr>
<tr><td>BERT (last four hidden layers)</td><td colspan="4">768</td></tr>
<tr><td>GPT-2 (last hidden layer)</td><td colspan="4">768</td></tr>
<tr><th colspan="5"><b>Predictors for Italian</b></th></tr>
<tr><td>Source of predictors</td><td># preds. V(w)</td><td># preds. V(c)</td><td># preds. V(w) * V(c)</td><td># preds. abs(V(w) - V(c))</td></tr>
<tr><td>Behavioural norms (frequency, etc.)</td><td>20</td><td>20</td><td>20</td><td>20</td></tr>
<tr><td>FastText (Common Crawl + Wikipedia)</td><td>31</td><td>31</td><td>31</td><td>31</td></tr>
<tr><td>ConceptNet NumberBatch (ConceptNet + Skip-gram + GloVe)</td><td>31</td><td>31</td><td>31</td><td>31</td></tr>
<tr><td>Concatenation of FastText and ConceptNet NumberBatch</td><td>31</td><td>31</td><td>31</td><td>31</td></tr>
<tr><td>ALBERT (last hidden layer)</td><td colspan="4">768</td></tr>
<tr><td>AlBERTo (last hidden layer)</td><td colspan="4">768</td></tr>
<tr><td>Bart (last hidden layer)</td><td colspan="4">768</td></tr>
<tr><td>BERT (last hidden layer)</td><td colspan="4">768</td></tr>
<tr><td>GPT-2 (last hidden layer)</td><td colspan="4">768</td></tr>
<tr><td>RoBERTa (last hidden layer)</td><td colspan="4">768</td></tr>
</table>

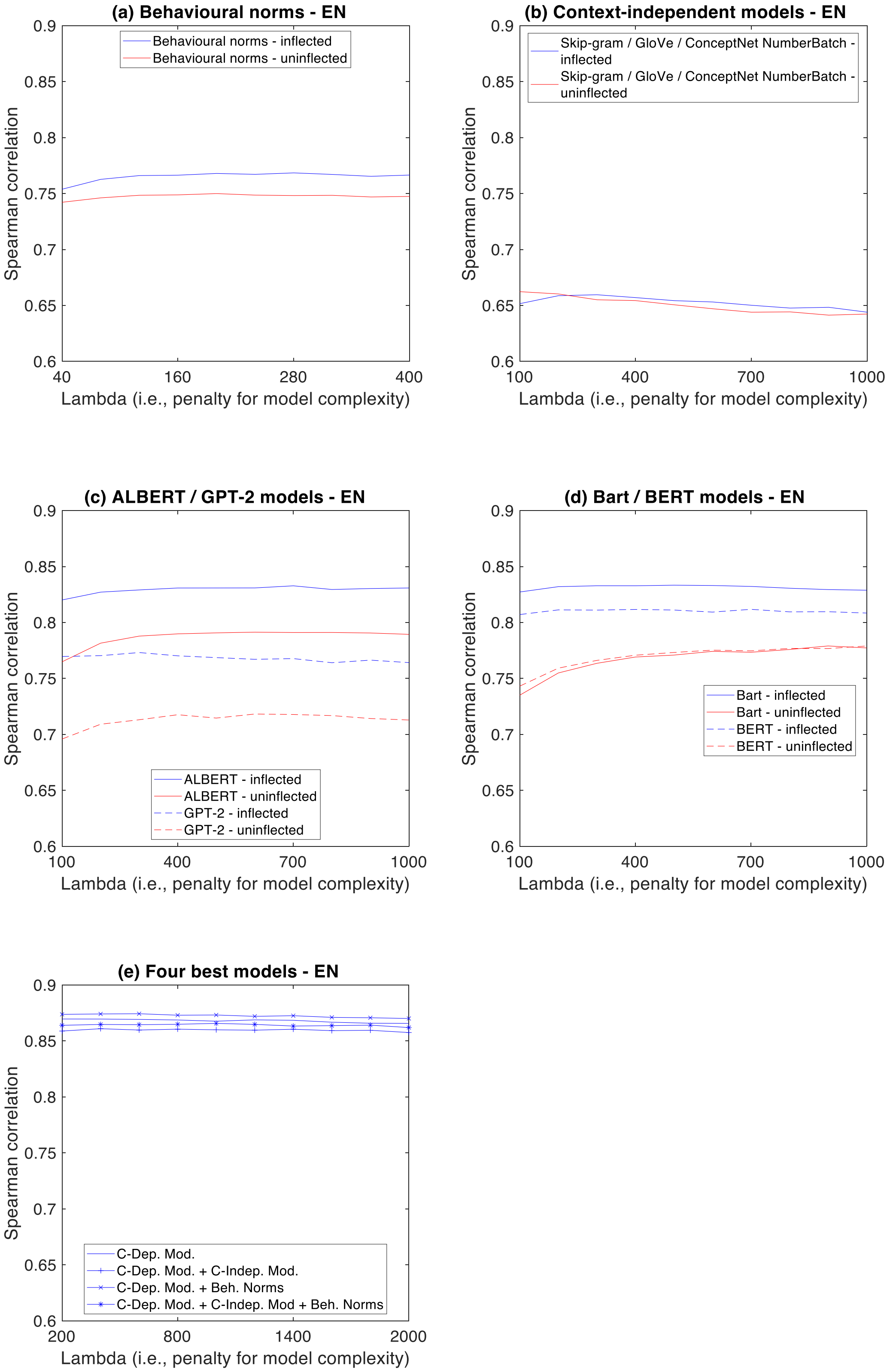


Figure 1: English: Spearman correlations between predicted and actual ratings, for various groups of predictors and regularization strengths (i.e., values of lambda). C-Dep. Mod.: the combination of the ALBERT, GPT-2, Bart, and BERT models; C-Indep. Mod.: the combination of the Skip-gram, GloVe, and ConceptNet NumberBatch models, their concatenation, and neighbourhood density measures; Beh. Norms: the predicted psycholinguistic ratings, together with frequency and contextual diversity counts. For the best four models, all predictors were derived from the inflected form of the target words. Our submission to the competition was based on C-Dep. Mod. + Beh. Norms (lambda = 500).

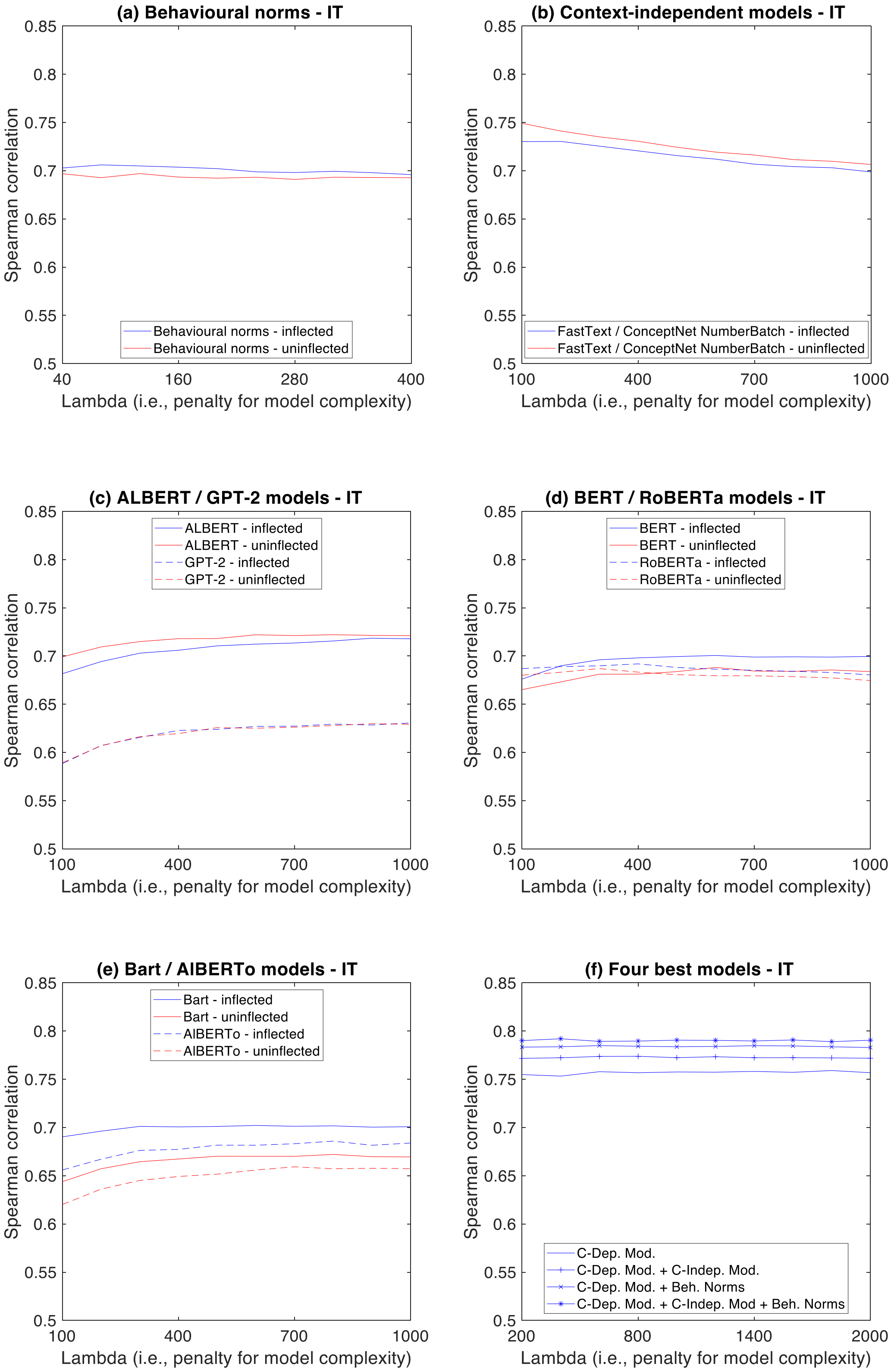


Figure 2. Italian: Spearman correlations between predicted and actual ratings, for various groups of predictors and regularization strengths (i.e., values of lambda). C-Dep. Mod.: the combination of the ALBERT, GPT-2, BERT, RoBERTa, Bart, and AlBERTo models; C-Indep. Mod.: the combination of the FastText and ConceptNet NumberBatch models, their concatenation, and neighbourhood density measures; Beh. Norms: the predicted psycholinguistic ratings, together with frequency and contextual diversity counts. For the best four models, all predictors were derived from the inflected form of the target words, except for the RoBERTa, FastText, and ConceptNet NumberBatch models (uninflected), and the behavioural norms (inflected and uninflected). Our submission to the competition was based on C-Dep. Mod. + C-Indep. Mod. + Beh. Norms (lambda = 500).

References

Basile, V., Croce, D., Di Maro, M., Passaro, L.C., 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian, in: Basile, V., Croce, D., Di Maro, M., Passaro, L.C. (Eds.), Proceedings of 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020). CEUR.org, Online.

Brysbaert, M., Warriner, A. B., & Kuperman, V. (2014). Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods*, *46*(3), 904-911.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the NAACL-HLT* (pp. 4171-4186). Stroudsburg, PA: ACL.

Frassinelli, D., Naumann, D., Utt, J., & im Walde, S. S. (2017). Contextual characteristics of concrete and abstract words. In C. Gardent & C. Retoré (Eds.), *Proceedings of the IWCS* (pp. 1-7). Stroudsburg, PA: ACL.

Gregori, L., Montefinese, M., Radicioni, D. P., Ravelli, A. A., & Varvara, R. (2020). CONcreTEXT @ Evalita2020: the Concreteness in Context Task. In V. Basile, D. Croce, M. Di Maro, & L. C. Passaro (Eds.), *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*. Online: CEUR.org.

Hoffman, P., Lambon Ralph, M. A., & Rogers, T. T. (2013). Semantic diversity: A measure of semantic ambiguity based on variability in the contextual usage of words. *Behavior Research Methods*, *45*(3), 718-730.

Johns, B. T., Jamieson, R. K., & Jones, M. N. (2020). The continued importance of theory: Lessons from big data approaches to language and cognition. In S. E. Woo, R. Proctor, & L. Tay (Eds.), *Big data methods for psychological research: New horizons and challenges* (pp. 277-295). Washington, DC: APA.

Junczys-Dowmunt, M., Grundkiewicz, R., Dwojak, T., Hoang, H., Heafield, K., Neckermann, T., Seide, F., Germann, U., Aji, A. F., Bogoychev, N., Martins, A., & Birch, A. (2018). Marian: Fast neural machine translation in C++. In F. Liu & T. Solorio (Eds.), *Proceedings of the ACL - System Demonstrations* (pp. 116-121). Stroudsburg, PA: ACL.

Kuperman, V., Stadthagen-Gonzalez, H., & Brysbaert, M. (2012). Age-of-acquisition ratings for 30,000 English words. *Behavior Research Methods*, *44*(4), 978-990.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). ALBERT: A lite BERT for self-supervised learning of language representations. In *Proceedings of the ICLR* (pp. 1-17).

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In D. Jurafsky, J. Chai, N. Schluter, & J. Tetreault (Eds.), *Proceedings of the ACL* (pp. 7871-7880). Stroudsburg, PA: ACL.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis. M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint:1907.11692*.

Lynott, D., Connell, L., Brysbaert, M., Brand, J., & Carney, J. (2019). The Lancaster Sensorimotor Norms: Multidimensional measures of perceptual and action strength for 40,000 English words. *Behavior Research Methods*, *52*, 1-21.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. In J. Bengio & Y. LeCun (Eds.), *Proceedings of the Workshop at the ICLR* (pp. 1-12).

Mohammad, S. (2018). Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 English words. In I. Gurevych & Y. Miyao (Eds.), *Proceedings of the ACL - Long Papers* (pp. 174-184). Stroudsburg, PA: ACL.

Paetzold, G., & Specia, L. (2016). Inferring psycholinguistic properties of words. In K. Knight, A. Nenkova, & O. Rambow (Eds.), *Proceedings of the NAACL-HLT* (pp. 435-440). Stroudsburg, PA: ACL.

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. In A. Moschitti, B. Pang, & W. Daelemans (Eds.), *Proceedings of the EMNLP* (pp. 1532-1543). Stroudsburg, PA: ACL.

Polignano, M., Basile, P., de Gemmis, M., Semeraro, G., & Basile, V. (2019). AlBERTo: Italian BERT language understanding model for NLP challenging tasks based on tweets. In R. Bernardi, R. Navigli, & G. Semeraro (Eds.), *Proceedings of CLiC-it*. Aachen, Germany: CEUR.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*.

Speer, R., Chin, J., & Havasi, C. (2017). ConceptNet 5.5: An open multilingual graph of general knowledge. In S. P. Singh & S. Markovitch (Eds.), *Proceedings of the AAAI* (pp. 4444-4451). Palo Alto, CA: AAAI Press.

Van Heuven, W. J., Mandera, P., Keuleers, E., & Brysbaert, M. (2014). SUBTLEX-UK: A new and improved word frequency database for British English. *The Quarterly Journal of Experimental Psychology*, *67*(6), 1176-1190.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. In T. Linzen, G. Chrupała, & A. Alishahi (Eds.), *Proceedings of the EMNLP Workshop BlackboxNLP* (pp. 353-355). Stroudsburg, PA: ACL.

CAPISCO @ CONcreTEXT 2020: (Un)supervised Systems to Contextualize Concreteness with Norming Data

Alessandro Bondielli[1,2] and **Gianluca E. Lebani**[3] and **Lucia C. Passaro**[2]
and **Alessandro Lenci**[2]

[1] Dipartimento di Ingegneria dell'Informazione, Università degli studi di Firenze
[2] CoLing Lab, Dipartimento di Filologia, Letteratura e Linguistica, Università di Pisa
[3] Dipartimento di Studi Linguistici e Culturali Comparati, Università Ca' Foscari Venezia

alessandro.bondielli@unifi.it
gianluca.lebani@unive.it
lucia.passaro@fileli.unipi.it
alessandro.lenci@unipi.it

Abstract

English. This paper describes several approaches to the automatic rating of the concreteness of concepts in context, to approach the EVALITA 2020 "CONcreTEXT" task. Our systems focus on the interplay between words and their surrounding context by (i) exploiting annotated resources, (ii) using BERT masking to find potential substitutes of the target in specific contexts and measuring their average similarity with concrete and abstract centroids, and (iii) automatically generating labelled datasets to fine tune transformer models for regression. All the approaches have been tested both on English and Italian data. Both the best systems for each language ranked second in the task.

1 Introduction

The characterization of the conceptual concreteness of a word in context is a task that requires a level of analysis that goes well beyond the identification of the properties of the referent (or denotation) of the target word. The overall linguistic context should be taken into consideration as well, along with its interaction with the target word. Even addressed in the most simplistic way, i.e. ignoring the context and focusing solely on the target word in isolation, it is a daunting task in which the machine is asked to draw inferences on a level of semantic representation that the speaker builds by integrating experiential and linguistic information (Vigliocco et al., 2009). Moreover, figurative uses of words (e.g., metaphors) determine important shifts in their concreteness values. For example, the word *head* in the sentence *Take your safety pins and attach one card to the head of your bed* can be considered as highly concrete, as it describes a physical object. Conversely, the same word in the sentence *The pope is also head of the world's smallest sovereign state, The Vatican* has a more abstract meaning, denoting the title of a person. Similarly the verb *fly* is more concrete in the sentence *The plane flies in the sky* than in the metaphorical sentence *Time flies*.

The context-sensitive nature of word concreteness is one of the key elements that make its identification very interesting and complex from a Natural Language Processing (NLP) perspective (Naumann et al., 2018). Unfortunately, to the best of our knowledge only a handful of scholars have addressed this topic. Notable mentions are Hill et al. (2013), and Hill and Korhonen (2014).

As it is common for other NLP and NLP-related tasks and topics, an invaluable source of knowledge that can be used both to train models and to gain some insights on the nuances of the problem itself can be found in the psycho-linguistic tradition, and especially in those normative studies built to analyze collections of human-elicited concreteness judgements (Brysbaert et al., 2013; Montefinese et al., 2013; Della Rosa et al., 2010). Most of these works, however, share the common limitation of ignoring the polysemic nature of words and the effect of context on their concreteness (Reijnierse et al., 2019). As an NLP task, the automatic estimation of the degree of concreteness carried by a given word in a given linguistic context can play a part in well-known and long-standing NLP issues such as word sense disam-

---

biguation (Agirre and Edmonds, 2007) and figurative language interpretation (Veale et al., 2016). All such tasks require a deep understanding of the linguistic context and are quite hard to model with traditional NLP models. Moreover, the fortune of language models specifically focused on modelling the meaning of words in context, such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), demonstrates how meaning construction is an appealing topic for the whole NLP community.

The CONcreTEXT task (Gregori et al., 2020) of EVALITA 2020 (Basile et al., 2020) focuses on modelling the concreteness of concepts in context. Given a sentence and a target word, the goal is to predict the word concreteness on a scale from 1 (fully abstract) to 7 (fully concrete). Results are evaluated by estimating their Spearman correlation with the (average of the) human-generated ratings. For the task, two trial datasets were made available, one for English and the other for Italian. Each trial dataset contains 100 sentences, two for each of the 50 target words.

In order to address this task, we propose three families of distributional semantic methods relying on several existing concreteness norms. Our general approach revolves around the idea that taking into account both the context and the target word, as well as words that play a similar role in the same context, may help us in overcoming limitations due to scarce training data, and may prove beneficial for predicting more accurate ratings.

The paper is organized as follows: Section 2 describes the proposed approach based on both supervised and unsupervised methods. Section 3 presents the results, which are discussed in Section 4. Finally, Section 5 draws some conclusions.

2 Methods

We propose three different "CAPISCO" (for *CA' Foscari and PISa COncretext project*) approaches for predicting the concreteness of a word in a given context of occurrence. Each method exploits the assumption that the concreteness of a word is influenced by its surrounding context. We explore both unsupervised and supervised techniques. In fact, two such approaches are unsupervised, and exploit either pre-trained word embeddings, or pre-trained transformer language models, while the third method is supervised:

NON-CAPISCO – the concreteness of the target word is modelled as a function of its concreteness value in isolation and of the average concreteness of its surrounding context.

CAPISCO-CENTROIDS – the concreteness of the target word is estimated as a function of the concreteness values of its closer synonyms according to a pre-trained transformer language model. Crucially, the concreteness ratings of the target synonyms are estimated by computing their distance from two reference points in the distributional space corresponding to the centroids of the highly concrete and highly abstract terms.

CAPISCO-TRANSFORMER – a supervised regressor is trained to predict concreteness ratings. Specifically, we fine-tune a transformer model to predict the target concreteness of the sentence, exploiting the available dataset augmented with new data automatically generated from several different norms of concreteness.

2.1 NON-CAPISCO

The NON-CAPISCO system is rather simple, both conceptually and implementation-wise. It is based on a minor change in the baseline proposed by the task organizers.

The task baseline is computed by averaging over the concreteness ratings of all the words in the sentence. Ratings are obtained from the norms by Montefinese et al. (2013) for Italian, and from those by Brysbaert et al. (2013) for English. Words missing from these resources are replaced by their closest neighbor among those for which human ratings are available. Closest neighbors are identified using fastText (Grave et al., 2018). On the trial dataset, our implementation of the baseline obtained a Spearman correlation score of 0.47 for Italian and 0.57 for English.

Crucially, this baseline takes into account the concreteness rating of the target word, but it has the same weight as all the other words in the sentence on the final prediction. On the other hand, we noticed that a simple method based solely on the concreteness score of the target word achieves a performance of 0.69 for Italian and 0.69 for English, much higher than that of the task baseline. This led us to surmise that, at least in the task dataset, the concreteness of the word in context is strongly affected by its value in isolation.

The NON-CAPISCO method gives more weight to the target word, by multiplying its concreteness rating for the mean concreteness of the whole sentence. On the trial dataset this combined score obtained a Spearman correlation of 0.73 for Italian and 0.73 for English.

2.2 CAPISCO-CENTROIDS

The CAPISCO-CENTROIDS approach is based on the assumption that semantically similar words are expected to be similarly rated for concreteness and that, conversely, words associated with highly different concreteness scores should be placed far away from each other in semantic space. This assumption is driven by the fact that concrete (or abstract) senses are typically found in co-occurrence with other concrete (or abstract) ones (Frassinelli et al., 2017). Thus, semantically similar words, i.e. that typically occur in the same context, are expected to have similar concreteness as well.

The first step of this method consisted in the building of two reference vectors: one representing the prototypical abstract concept; the other representing the prototypical concrete concept. To this end, we first identified highly concrete and highly abstract terms from two available resources: the Brysbaert et al. (2013) norms for English and the Della Rosa et al. (2010) norms for Italian. The latter has been preferred to more comprehensive alternatives, like the Montefinese et al. (2013) norms, due to its covering of a significant set of highly polarized words.

For each resource, the clusters of most concrete and abstract words were identified by fitting a mixture-of-Gaussian model on the human judgments, and choosing the most distant clusters. We used the expectation-maximization algorithm available in scikit-learn.[1] To set the number of clusters and type of covariance, we chose the pair that minimized the Bayesian information criterion. After identifying the groups of most polarized words in our reference norm, we used English and Italian pre-trained word embeddings from fastText (Grave et al., 2018) to identify their respective centroids in the vector space, by simply averaging the embeddings of highly concrete and highly abstract words. In the case of the English vector space, the dimensionality was left to the default value of 300. In the case of the Italian space, the dimensionality was further reduced to 100, as we saw an increase in performances, which instead was not the case for English.

However, predicting the concreteness of a target word solely based on its proximity with the centroids could be biased by its semantic relatedness with the words used for building the centroids. To smooth this bias, the final score for a given target word was calculated as the average of the similarities of its potential lexical substitutes. BERT was used to identify the substitutes of each target word in context. Operationally, we masked the target word in each sentence, and asked the model to predict the 50 most likely words that may fill the masked token, which is likely to include the target itself. After several experiments, we chose 50 words as they gave us the best overall results. We can argue that it is probably the best trade-off between number of neighbors and their actual similarity with the target word. We used the `bert-base-uncased` model for English, and the `bert-base-italian-xxl-uncased` model for Italian. Table 1 reports some potential substitutes of the target word in the sentence.

TARGET	MASKED SENT.	FILLERS
lawsuit	In a typical [MASK] , the defendant frequently brings a motion [...].	case trial proceeding
love	Give your friends [MASK] , positivity , and compliments .	attention kindness respect

Table 1: Prediction of fillers in context with BERT.

To avoid noise due to the fact that sometimes BERT predicts a token with a different syntactic role, all the fillers with a different Part-of-Speech (PoS) tag than that of the target word were filtered out. To this end, we PoS-tagged all the sentences produced by replacing the target word and kept only those with the same PoS sequence of the original sentence. This way, we obtained, for each target word, a list of lexical substitutes in a particular context. Each substitute was assigned a concreteness score based on its proximity to the two prototypical vectors. More specifically, we computed the concreteness of a word as the absolute value of the difference between its cosine with the concrete centroid and its cosine with the abstract one normalized on a 1-7 scale. Finally, each target word was assigned with a concreteness value obtained by averaging the concreteness of its substitutes.

2.3 CAPISCO-TRANSFORMER

The CAPISCO-TRANSFORMER system addresses the problem from a supervised perspective. The system is based on the BERT Transformer archi-

---

[1] `https://scikit-learn.org/`

tecture (Devlin et al., 2019). BERT and the other Transformer allow for transfer learning in NLP tasks, by means of unsupervised pre-training followed by supervised fine-tuning for downstream tasks. Such models have obtained state-of-the-art results in most NLP supervised and unsupervised tasks (Devlin et al., 2019). We used a BERT pre-trained model and fine-tuned it on the concreteness rating task. Given the very small size of the trial dataset provided for the task, we tried to improve generalization capabilities by dynamically generating additional training data to feed the model. To this end, we used two different approaches.

On the one hand, we generated potential substitutes of the target word with the same techniques used in Section 2.2. In this case, we generated three sentences containing as target word the three most likely lexical substitutes of the original one. Such new target words were assigned the same concreteness rating of the original one, modified by a small random value in the range [-0.2,0.2], to avoid repetition of target values for the training set derived from the gold data.

On the other hand, we extended the dataset with new sentences which were assigned the concreteness scores found in the concreteness norm. For English, we extracted from the BNC corpus (The British National Corpus, 2007) all the sentences containing words rated in the Brysbaert et al. (2013) norms. For Italian language, we extracted from La Repubblica corpus (Baroni et al., 2004) all the sentences containing words rated in the Montefinese et al. (2013) or in the Della Rosa et al. (2010) norms. As we are interested in mostly unambiguous target words with different concreteness ratings, we chose to select, for each considered norm, only words with a low standard deviation that are in a specific range of values for concreteness. Therefore, we obtained three sets of very concrete, very abstract and mildly concrete words. Thresholds were manually set for each resource in order to address their different distribution and scales in terms of concreteness ratings. Once sentences containing such target words were collected, we sampled three random sentences for each target and we assigned each sentence the concreteness rating of its target word in the norm. We obtained 8,813 training sentences for English and 3,467 for Italian. The Italian training set is smaller as the Italian resources contain fewer words.

The whole extended dataset is then used to fine-tune the BERT model to predict the concreteness rating assigned to the whole sentence by means of regression. Operationally, we use the implementation of BERT provided in the Huggingface library.[2] For the English model, initial weights are taken from `bert-base-uncased`, while for Italian we used initial weights from `bert-base-italian-xxl-uncased`. Both pre-trained models are available within the Transformer library. We trained each model for 2 epochs, with a batch size of 8 and the learning rate set to 2e-5, on a machine equipped with a Titan Xp GPU. At inference time, we simply feed the fine-tuned model with test sentences and ask it to directly predict the concreteness rating.

3 Results

We proposed three different approaches for the estimation of concreteness. The performances obtained for each model for the Italian language and for the English language are presented respectively in Tables 2 and 3. Given the absence of a training set, we decided to give more emphasis to the unsupervised method (NON-CAPISCO) based on the concreteness of target words and of the surrounding context. It is clear that the results of this method are highly influenced by the annotated resources exploited to infer the concreteness. The results revealed that while for English such approach was quite effective, for Italian it is not, probably due to the smaller dimension and quality of the resources taken into consideration. In fact, if we look at the ranking of our models in the two languages, the results are reversed. On the one hand, the best CAPISCO approach for English is the NON-CAPISCO system, in which concreteness ratings are obtained from Brysbaert et al. (2013). Such resource counts ratings for about 40 thousand of English lemmas that have been annotated for several variables. On the other hand, the Italian resources (Della Rosa et al., 2010; Montefinese et al., 2013) are orders of magnitude smaller than English ones thus causing a big drop in performances of the proposed approach. This issue will be discussed in detail in Section 4.

4 Discussion

In light of the reported results, several interesting observations can be made. For both languages, our best-performing model ranked sec-

---

[2] `https://huggingface.co`

RANK	SYSTEM	SPEARMAN
1	****	0.749
2	CAPISCO-TRANSFORMER-IT	0.625
3	CAPISCO-CENTROIDS-IT	0.615
4	NON-CAPISCO-IT	0.557
5	Baseline_2	0.534
6	Baseline_1	0.346

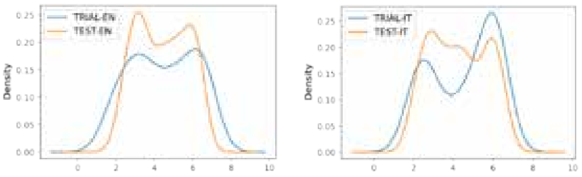
Table 2: CAPISCO performances for the Italian.

RANK	SYSTEM	SPEARMAN
1	****	0.833
2	NON-CAPISCO-EN	0.785
3	****	0.663
4	****	0.651
5	Baseline_2	0.554
6	****	0.542
7	CAPISCO-CENTROIDS-EN	0.542
8	****	0.541
9	CAPISCO-TRANSFORMER-EN	0.504
10	Baseline_1	0.383
11	****	-0.013
12	****	-0.124
13	****	-0.127

Table 3: CAPISCO performances for the English.

ond overall. However, we can notice how neither numerical results nor the ranking of the system are consistent across languages. For English, the best performing system is NON-CAPISCO. The system strongly outperforms both baselines and the other two methods. We must also note that both CAPISCO-CENTROIDS and CAPISCO-TRANSFORMER perform worse than one of the two baselines. On the other hand, for Italian, CAPISCO-TRANSFORMER performed best, closely followed by CAPISCO-CENTROIDS. Both outperform the NON-CAPISCO approach, and all three systems perform better than the baselines. This discrepancy may be due to several key aspects concerning both the resources used as well as some crucial differences among trial and test samples of the dataset.

We can identify several key differences among English and Italian resources that may justify such drastically different performances. While for English a comprehensive resource with 40,000 words is available, both resources for Italian are orders of magnitude smaller. In addition to this, especially for ratings contained in Montefinese et al. (2013), the distribution is unbalanced towards mid-range and high values of concreteness, while ratings for Brysbaert et al. (2013) are more evenly distributed across the spectrum. For the NON-CAPISCO system, this may lead to poor performances since for the system is more difficult to predict higher values for the Italian dataset. While predictions for the English model closely follow the distribution of ratings in the test set, predictions for Italian are unbalanced towards lower values.



(a) (b)

Figure 1: Distribution of ratings for trial and test sets, for (a) English and (b) Italian.

On the contrary, for the CAPISCO-CENTROIDS system, this has the opposite effect. In fact, given that it is more difficult to isolate extremely abstract and extremely concrete terms, centroids built from Italian resources are closer one another, and thus prediction based on the difference between distances to the centroids almost always fall in the middle of the range, while for English the same approach has the effect of yielding results that are mostly close to the lower-end of the spectrum. This, in turn, has the effect of seemingly improving performances for Italian, because too high and too low prediction balance each other, while errors for English are more pronounced.

Finally, for the CAPISCO-TRANSFORMER system, it may be possible that the fact that English norms contains more high frequency words, may hinder the generalization capabilities of the model. In fact, if such words are found in very different sentences, all such sentences are assigned very similar concreteness scores and the predictions are biased towards certain values for many different sentences. Therefore, the distribution of predictions follow the same tripartite distribution of the sampled words in terms of concreteness.

Finally, we must point out that the distribution of ratings in the trial and test set are rather different, as shown in Figure 1. This may have hindered our judgment on the quality of all proposed systems, both unsupervised and supervised.

5 Conclusions and Future works

The models proposed are based on both supervised and unsupervised approaches. The choice was motivated by the fact that the trial dataset proposed for the task is too small to effectively train supervised learning models on it. The key assumption that drove the development is that the concreteness of a word is influenced by its surrounding context, as claimed by the task organizers as well. The best CAPISCO systems for both Italian and English ranked second in the CONcreTEXT task despite the fact that results differ a lot in terms of absolute performances and used method. For Italian, the best CAPISCO system is based on Transformers and reaches a Spearman correlation of 0.625 with gold data. The best CAPISCO model for English, on the contrary, is unsupervised and reaches a Spearman correlation with gold data of 0.785.

In the future, we plan to perform some additional hyper-parameter tuning on the models. Moreover, we would like to test this approach in similar tasks (e.g. predicting abstractness). We are confident that by exploiting the dynamic selection of training data in addition to an annotated dataset such as the test dataset provided by the task organizers would improve the results of our systems, and in particular of the transformers-based one.

Acknowledgments

We gratefully acknowledge the support of the NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

Eneko Agirre and Philip Edmonds. 2007. *Word Sense Disambiguation: Algorithms and applications*. Springer Science & Business Media.

Marco Baroni, Silvia Bernardini, Federica Comastri, Lorenzo Piccioni, Alessandra Volpi, Guy Aston, and Marco Mazzoleni. 2004. Introducing the La Repubblica corpus: A large, annotated, TEI (XML)-compliant corpus of newspaper Italian. *Proc. of LREC 2004*.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proc. of EVALITA 2020*, Online. CEUR.org.

Marc Brysbaert, Amy Warriner, and Victor Kuperman. 2013. Concreteness ratings for 40 thousand generally known english word lemmas. *Behav. Res. Methods*, 46:904–911.

Pasquale Della Rosa, Eleonora Catricalà, Gabriella Vigliocco, and Stefano Cappa. 2010. Beyond the abstract-concrete dichotomy: Mode of acquisition, concreteness, imageability, familiarity, age of acquisition, context availability, and abstractness norms for a set of 417 italian words. *Behav. Res. Methods*, 42:1042–1048.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT 2019*, pages 4171–4186.

Diego Frassinelli, Daniela Naumann, J. Utt, and Sabine Schulte im Walde. 2017. Contextual characteristics of concrete and abstract words. In *Proc. of IWCS 2017*.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proc. of LREC 2018*.

Lorenzo Gregori, Maria Montefinese, Daniele P. Radicioni, Andrea Amelio Ravelli, and Rossella Varvara. 2020. CONcreTEXT @ EVALITA2020: the Concreteness in Context Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proc. of EVALITA 2020)*, Online. CEUR.org.

Felix Hill and Anna Korhonen. 2014. Learning abstract concept embeddings from multi-modal data: Since you probably can't see what I mean. In *Proc. of EMLP 2014*, pages 255–265.

Felix Hill, Douwe Kiela, and Anna Korhonen. 2013. Concreteness and corpora: A theoretical and practical study. In *Proc. of CMCL 2013*, pages 75–83.

Maria Montefinese, Ettore Ambrosini, Beth Fairfield, and Nicola Mammarella. 2013. The adaptation of the affective norms for english words (anew) for italian. *Behav. Res. Methods*, 46:887–903, 10.

Daniela Naumann, Diego Frassinelli, and Sabine Schulte im Walde. 2018. Quantitative semantic variation in the contexts of concrete and abstract words. In *Proc. of STARSEM 2018*, pages 76–85, New Orleans, Louisiana, June. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL 2018*, page 2227–2237.

W. Gudrun Reijnierse, Christian Burgers, Marianna Bolognesi, and Tina Krennmayr. 2019. How polysemy affects concreteness ratings: The case of metaphor. *Cognitive Science*, 43(8):e12779.

The British National Corpus. 2007. version 3 (BNC XML Edition).

Tony Veale, Ekaterina Shutova, and Beata Beigman Klebanov. 2016. *Metaphor: A Computational Perspective*. Morgan & Claypool.

Gabriella Vigliocco, Lotte Meteyard, Mark Andrews, and Stavroula Kousta. 2009. Toward a theory of semantic representation. *Language and Cognition*, 1(2):219–247.

KonKretiKa @ CONcreTEXT: Computing Concreteness Indexes with Sigmoid Transformation and Adjustment for Context

Yulia Badryzlova
HSE University
Moscow, Russia
`yuliya.badryzlova@gmail.com`

Abstract

The present paper is a technical report of KonKretiKa, a system for computation of concreteness indexes of words in context, submitted to the English track of the CONcreTEXT shared task. We treat concreteness as a bimodal problem and compute the concreteness indexes using paradigms of concrete and abstract seed words and distributional semantic similarity. We also conduct sigmoid transformation to achieve greater similarity to the psycholinguistically attested data, and apply dynamic adjustment of static indexes for sentential context. One of the modifications of the presented system ranked third in the task, with $r_s = .6634$ and $r = .6685$ against the gold standard.

1 Introduction

This paper is a description of the system with the working title KonKretiKa, which was submitted to the English track of CONcreTEXT, the shared task on evaluation of concreteness in context (Gregori et al., 2020) offered at EVALITA 2020, the 7th evaluation campaign of Natural Language Processing and speech tools for the Italian language (Basile et al., 2020).

KonKretiKa stems from our previous work on computation of such indexes for the purposes of metaphor identification.

Computationally obtained indexes of concreteness are extensively explored in experiments for automated metaphor identification. Application of concreteness indexes to metaphor identification relies on the assumptions made by the theories of embodied and grounded cognition (Barsalou, 2008), and primary and conceptual metaphor (Lakoff and Johnson, 1980). These theories claim that human thinking is intrinsically metaphoric, since the conceptual representations underlying knowledge are grounded in sensory and motor systems, and conceptual metaphor is the primary mechanism for transferring conventional mental imagery from sensorimotor domains to the domains of subjective experience.

An established method to compute the concreteness index of a word is to collect two sets of lexemes (‘seed lists’, or ‘paradigms’) consisting of abstract and concrete words – and to measure the lexical similarity between each word in the lexicon and each of the paradigm words.

Turney et al. (2011) use concreteness indexes to identify linguistic metaphor in the TroFi dataset (Birke and Sarkar, 2006). They compute the concreteness index of a word by comparing its distributional semantic embedding to the vector representations of 20 abstract and 20 concrete words. The paradigm words are automatically selected from the MRC Psycholinguistic Database Machine Usable Dictionary (Coltheart, 1981), a collection of 4,295 English words rated with degrees of abstractness by human subjects in psycholinguistic experiments.

Tsvetkov et al. (2013) also compute the concreteness indexes of English words by using a distributional semantic model and the MRC database. They train a logistic regression classifier on 1,225 most abstract and 1,225 most concrete words from MRC; the degree of concreteness of a word is the posterior probability produced by the classifier. The Tsvetkov et al. system for metaphor identification with concreteness indexes is based on cross-lingual model transfer, when the model is trained on English data, and then the classification features are translated into other languages by means of an electronic dictionary.

Concrete	albatross, balloon, bench, bridge, catfish, cauliflower, chicken, clown, corkscrew, crab, daisy, deer, eagle, egg, frog, garlic, goat, harpsichord, lion, mattress, mussel, nightgown, nightingale, owl, ox, pants, peach, piano, pig, potato, quilt, rabbit, saxophone, sheep, shrimp, skyscraper, sofa, stoat, tulip, turtle
Abstract	affirmation, animosity, demeanour, derivation, determination, detestation, devotion, enunciation, etiquette, fallacy, forethought, gratitude, harm, hatred, ignorance, illiteracy, impatience, independence, indolence, inefficiency, insufficiency, integrity, intellect, interposition, justification, malice, mediocrity, obedience, oblivion, optimism, prestige, pretence, reputation, resentment, tendency, unanimity, uneasiness, unhappiness, unreality, value

Table 1. The concrete and the abstract paradigm lists.

$$\forall\, v_i, \forall s_j \ \exists\, D_i = \{Sim(v_i, s_1),\ Sim(v_i, s_2), \dots,\ Sim(v_i, s_j),\ \dots, Sim(v_i, s_k)\}, \qquad (1)$$

where V is the set of words in the vocabulary,
S is the set of words in the seed list, k is the number of elements in S

$$NN = \{d'_{i_1},\ d'_{i_2}, \dots, d'_{i_{10}}\}, \qquad (2)$$

where $D_i{}'$ is a linearly ordered set of D_i (in ascending order)

$$I = Mean\{NN\} \qquad (3)$$

Equations 1-3. Computation of indexes with paradigm lists.

Badryzlova (2020) explores concreteness and abstractness indexes for linguistic metaphor identification in Russian and English. The paradigm words are selected in a semi-automatic fashion: the Russian paradigm is derived from the Open Semantics of the Russian Language, the semantically annotated dataset of the KartaSlov database (Kulagin, 2019); the English paradigm is selected from the MRC database (Coltheart, 1981). The indexes of concreteness and abstractness are computed for large sets of Russian and English words (about 18,000 and 17,000 lexemes, respectively). The metaphor identification in Russian is conducted on the RusMet corpus (Badryzlova, 2019; Badryzlova and Panicheva, 2018), and the English on the TroFi dataset. The author shows that the distributions of concreteness and abstractness indexes in the two languages follow the same pattern: in the lexicon, there is a distinct group of highly concrete words, which have very high concreteness and very low abstractness indexes; similarly, there is a group of distinctly abstract vocabulary, with low concreteness and high abstractness scores. Moreover, there is a general trend for abstractness indexes to increase as the corresponding concreteness indexes decrease. The author also observes statistical correlation between two Russian abstractness ratings, which may indicate that the category of abstractness is more semantically homogeneous than the category of concreteness.

The present work develops and extends the method of Badryzlova (2020) in two directions: (a) we apply sigmoid transformation to fit the curve comprised of the computed concreteness and abstractness indexes to the distribution of indexes in psycholinguistic data; and (b) we suggest a method for dynamic adjustment of the obtained indexes for sentential context, according to the requirements of the CONcreTEXT shared task (Gregori et al., 2020). The working title of the proposed system is KonKretiKa.

2 Description of the system

We demonstrate a method for evaluating concreteness on English data; however, it can be transferred to any other language provided that the following types of resources are available: (1) a lexicon with semantic (e.g. Fellbaum, 1998; Kulagin, 2019) or psycholinguistic (e.g. Brysbaert et al., 2014; Coltheart, 1981) annotation to select the paradigm words from; (2) a pre-trained distributional semantic model; and (3) a relatively large wordlist containing lexemes with different frequencies of occurrence (ipm) in order to ensure the maximum possible variation in concreteness across the lexicon.

When analyzing the distribution of psycholinguistic concreteness ratings, Brysbaert et al. (2014) observe that “concreteness and abstractness may be not the two extremes of a quantitative continuum […], but two qualitatively different

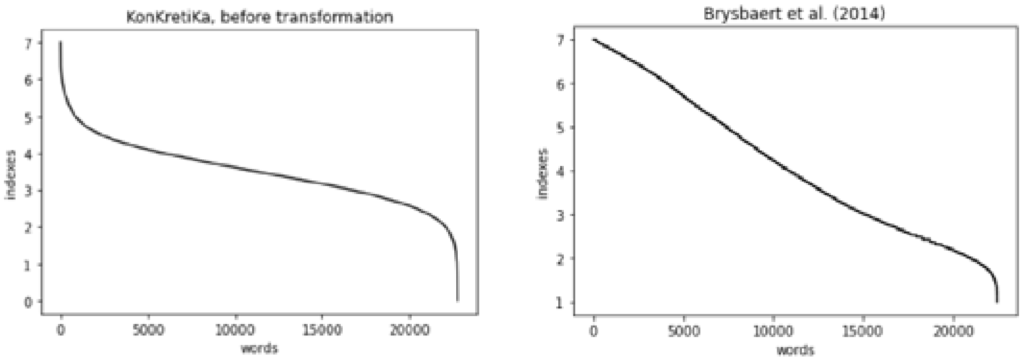


Figure 1. Distribution of computational (raw KonKretiKa) and psycholinguistic (Brysbaert et al.) indexes

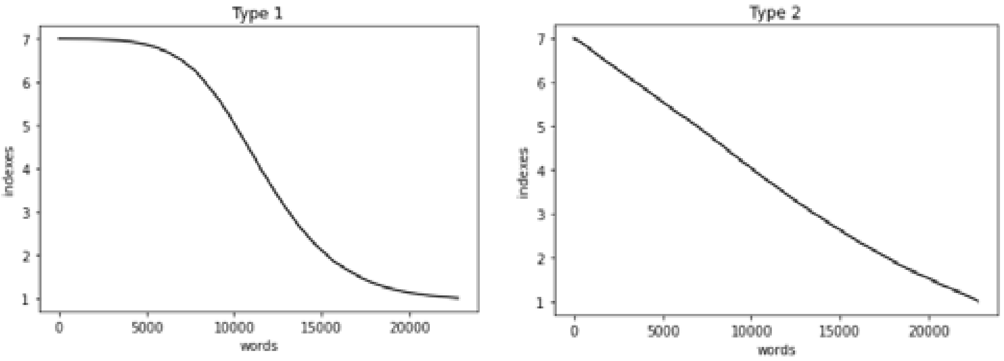


Figure 2. Sigmoid transformations

characteristics." of a word. Following this observation and the previous work in (Badryzlova, 2020), we treat concreteness as a bimodal property investing the word with two characteristics: the rate of concreteness and the rate of abstractness. Thus, we start by computing the standalone indexes of concreteness and of abstractness; then, the single aggregate index is computed as a function of these two indexes.

2.1 Computation of raw indexes with paradigm words and distributional semantic similarity

Computation of the standalone concreteness and abstractness indexes is based on paradigm lists of concrete and abstract words; we use the English concrete and abstract paradigms from Badryzlova (2020). These paradigms were compiled from the MRC Psycholinguistic Database: nouns from the top and from the end of the MRC concreteness rating were drawn to populate the concrete and the abstract paradigms, respectively. The paradigm lists are presented in Table 1.

The indexes of concreteness and abstractness were computed using a Continuous Skip-Gram model (Kutuzov et al., 2017) which had been pre-trained on the lemmatized Gigaword 5th Edition corpus (Parker et al., 2011).

As shown in Equations 1-3, to compute a concreteness or an abstractness index (*I*) of a word, we measured semantic similarity (cosine distance) *Sim* between the vectors of this word and each word in the paradigm (concrete or abstract, respectively), and took the mean of the ten nearest semantic neighbors (*NN*).

In total, we computed concreteness and abstractness indexes for approximately 23,000 English words (nouns, verbs, adjectives, and adverbs); this lexicon was taken from the Brysbaert et al. (2014) ranking, which allowed us to analyze the correlation between the computational and the large-scale psycholinguistic data at the subsequent stages of the present study (see Section 3).

The obtained computational sets of concreteness and abstractness indexes were normalized to the range [1, 7][1] in order to comply with the scale set by the CONcreTEXT shared task. In order to obtain an aggregate single-value index of a word, which would be representative of both its concreteness and abstractness, we subtracted the abstractness indexes from the concreteness indexes.

---

[1] Scikit-learn's MinMax Scaler (Pedregosa et al., 2011)

System	Transformation type	Contextual adjustment (*c*)	Result (r_s)	Result (r)
Leader-1			0.83313	0.83406
Leader-2			0.78541	0.78682
KonKretiKa-3	2	0.5	0.6634	0.6685
KonKretiKa-1	1	0.5	0.65102	0.66652
Baseline-2			0.55449	0.56742
KonKretiKa-4	2	0.8	0.54216	0.54465
KonKretiKa-2	1	0.8	0.54089	0.54479
Baseline-1			0.3825	0.37743

Table 2. Modifications of KonKretiKa and their results in the shared task.

2.2 Sigmoid transformation of raw indexes

Figure 1 shows distributions of our raw aggregate indexes and the indexes attested in psycholinguistic research (Brysbaert et al., 2014). It is noticeable that the curve of computational indexes has a much steeper slope, resulting in lower variance; consequently, the discriminative power of such indexes will also be lower.

The raw KonKretiKa curve has the shape of a sigmoid; in generic form, the sigmoid function is described by the equation:

$$S(x) = \frac{1}{1 + \exp(-ax + a * b)}$$

where a defines the slope of the function and b defines the inflection point. Consequently, we can transform the sigmoid by changing the a and b coefficients.

In the submissions to the CONcreTEXT shared task, we experimented with two transformations of the raw KonKretiKa curve (Figure 2). In the first transformation, we applied a heuristically chosen combination of a and b which was intended to increase the slope and the curvature while preserving the *S*-shape of the sigmoid. The second transformation was intended to attain maximum resemblance of its shape to the Brysbaert et al. curve. We used grid search with different combinations of coefficients a and b to maximize the correlation between the two curves. During this fitting, only the values of the indexes are adjusted, while their initial ranks remain intact – thus, there is no data leakage from the psycholinguistic ranking. [2]

2.3 Contextual adjustment

Since the CONcreTEXT shared task requires that the concreteness indexes of target words be dynamically adjusted to their sentential context, the following heuristic was applied in the submitted KonKretiKa models. We computed the mean concreteness of all content words in the sentence (with the target word excluded) and adjusted the concreteness value of the target word accordingly. The adjusted index A was computed as follows:

$$A_t = R_t - (M * c)$$

where t is the target word, R is the raw index from the KonKretiKa ranking, M is the mean concreteness of the sentence, and c is the adjustment coefficient. In the models submitted to the CONcreTEXT shared task, we applied two heuristically defined c coefficients: $c = 0.5$ and $c = 0.8$.

Thus, the four modifications of KonKretiKa submitted to the shared task were differentiated by the two parameters: the type of transformation and the contextual adjustment coefficient.

3 Results and discussion

The parameters of the four modifications and their results are presented in Table 2 (along with the Baselines and the Leaders). The results indicate that systems with the lower coefficient of sentential adjustment (0.5) perform better than systems with the higher adjustment coefficient (0.8) irrespective of the type of sigmoid transformation; yet, the system with Type 2 (fitted to the psycholinguistic data) transformation somewhat outperforms the system with Type 1 (*S*-shaped) transformation.

The best of our modifications, KonKretiKa-3, demonstrated Spearman correlation with the gold standard $r_s = .6634$ and Pearson correlation $r = .6685$, ranking our system third in the track, yet by a substantial margin behind the two winning system (with $r_s = .83313$ and $r = .83406$ and $r_s = .78541$ and $r = .78682$, respectively).

---

[2] The KonKretiKa ranking is available at: https://github.com/yubadryzlova/CONcreTEXT-2020

Dataset	Gold (dynamic)	BRY (static)
KKK (static)		r_s =.743 r =.751
KKK (dynamic)	r_s = .663 r = .669	
BRY (static)	r_s = .755 r = .761	

Table 3. Pairwise correlations: KKK – KonKretiKa, BRY – Brysbaert et al., Gold – CONcreTEXT gold standard.

3.1 Analysis of contextual adjustment

We carried out a post hoc analysis of the contextal adjustment coefficient (*c*) by using grid search to maximize the correlation between KonKretiKa (Type 2 transformation) and the gold standard. Moreover, we altered the scope of the context words for which the mean sentential concreteness (*M*) was computed – by taking 2-3 nearest semantic neighbors (either of any part of speech, or only nouns, or only verbs); this was done in order to reduce the possible noise from the words that are not semantically related to the target in the sentence. The change of the contextual scope did not lead to a substantial difference in the result. As for the contextual adjustment coefficient, the grid search showed that $c = 0.32$ – which is lower than the most efficient coefficient from our earlier submissions ($c = 0.5$ in KonKretiKa-3) – results in a slight increase of correlations: $r_s = .678$ and $r = .688$.

A closer analysis of the test sentences suggests that contribution of contextual adjustment presumably may be increased by considering a broader context of a sentence – for instance, spanning over 1-3 adjacent sentences from the left and the right contexts; this option constitutes a possible direction for future work.

3.1 Comparison of computational and psycholinguistic data

Pairwise correlations between the computational (KonKretiKa, KKK) and the psycholinguistic rankings (Brysbaert et al., BRY and the gold standard) are shown in Table 3. It can be seen that KKK better correlates with the BRY data than with the gold standard ($r_s = .743$, $r = .751$ vs. $r_s = .663$, $r = .669$, respectively). Presumably, such difference in the two correlations is due to the much larger size of the BRY lexicon. The correlation between the two psycholinguistic datasets (BRY vs. Gold) is $r_s = .755$, $r = .761$, which is close to the correlation between KKK and BRY.

word	BRY	KKK	Diff
handmaiden (N)	6.45	1.54	4.91
tire (V)	7	2.18	4.82
bedrock (N)	6.18	1.55	4.63
alarm (N)	6.19	1.58	4.61
text (N)	6.89	2.31	4.58
nonreactive (ADJ)	2.25	6.82	-4.57
temptingly (ADV)	1.72	6.26	-4.55
hail (V)	5.96	1.5	4.47
stance (N)	5.53	1.11	4.42
nudge (N)	6.19	1.8	4.39
chasm (N)	5.84	1.45	4.39

Table 4. Top residuals: Brysbaert et al. (BRY) vs. KonKretiKa (KKK).

We undertook closer pairwise comparative analysis between two pairs of rankings:

1. Static KonKretiKa indexes (the indexes after Type 2 sigmoid transformation, without contextual adjustment) vs. the Brysbaert et al. ranking (which is also static): approximately 23,000 words – nouns, verbs, adjectives, and adverbs (the two wordlists are identical).
2. Indexes of the target words from the CONcreTEXT test data as presented in the dynamic version of KonKretiKa (the sigmoid-transformed Type 2 indexes with contextual adjustment coefficient $c = 0.32$) vs. the Gold standard (where the target words are also ranked dynamically in context): 436 words – verbs and nouns.

The top residuals between the KonKretiKa and the Brysbaert et al. indexes are presented in Table 4. Analysis of these discrepancies suggests that most of them stem from polysemy and the differences between its representation in distributional semantic models and in psycholinguistic reality. Thus, distributional semantic models do not discriminate between various meanings of words; if occurrences of one of the meanings substantially outnumber the other meanings in discourse and, as a consequence, in the training corpus, the resulting vector reflects the more frequent meaning.

Sentence	Target word	Gold	KKK	Diff	TEXT
399	vision (N)	6.03	1.86	4.17	Check your < vision > to see if you are seeing blurry or double.
353	vision (N)	5.97	1.82	4.15	With retinal migraine, you may experience loss of < vision > in one eye and a headache that starts behind your eyes.
155	spirit (N)	6	2.33	3.67	Gin is an alcoholic < spirit > made from distilled grain or malt.
324	pain (N)	5.2	1.59	3.61	See your doctor if you are experiencing < pain > or discomfort.
61	answer (N)	5.45	1.91	3.54	Be sure to write your final < answer > without the negative sign.
385	war (N)	5.57	2.06	3.51	They have escaped from civil < war > in Liberia or Zimbabwe.
81	answer (N)	5.32	1.92	3.4	Final < answers > for equations are considered wrong unless you have broken them down to their simplest form.
237	heart (N)	6.32	2.98	3.34	The < heart > pumps blood due to an internal electrical system.
163	pain (N)	4.97	1.63	3.34	Take your medications to ease your physical < pain >.
176	agreement (N)	5.16	1.85	3.31	After signing the indemnification < agreement >, you can sign the legally binding bond agreement.

Table 5. Top residuals: KonKretiKa (KKK) vs. Gold standard.

For example, the nearest semantic neighbors of the noun *handmaiden* in the distributional semantic model[3] are: *embodiment*, *personification*, *epitome*, and *paragon* – associating this word with its abstract, metaphoric meaning 'something that supports something else that is more important'[4], whereas for speakers of English the other, concrete meaning 'a woman who is someone's servant' apparently stands out as being more salient. Similarly, among the nearest semantic neighbors of the noun *chasm* in the distributional semantic model are: *disparity*, *schism*, *rich-poor divide*, *mistrust*, *(the) haves*, *divergence*, *antagonism*, and *inequality* – indicating that the distributional vector of *chasm* is biased towards the abstract meaning of this word ('a very big difference that separates one person or group from another') rather than the concrete one ('a very deep crack in rock or ice'), while human subjects see the latter meaning as more salient or prevalent.

As for *nonreactive* and *temptingly*, which are more concrete in the computational data, this could be explained by their perceived vagueness to human subjects, since these words do not have meanings that would be markedly juxtaposed to each other in terms of concreteness-abstractness – thus ranking them rather low in the psycholinguistic data. Meanwhile, the nearest semantic neighbors of *temptingly* in the distributional semantic model are: *strappy sandal*, *capelet*, *knee-length skirt*, *enticingly*, *floral-print*, *high-heeled sandal*, *lace-trimmed*, *harem pants*, and *puffed sleeve* – all rather concrete objects (or the properties of such objects).

The top residuals between KonKretiKa and the gold standard are shown in Table 5. The discrepancy between the abstract meaning of *vision* ('the ability to think about and plan for the future, using intelligence and imagination, especially in politics and business') and its concrete meaning ('the ability to see') can also be attributed to the differences between representation of meanings in distributional semantic models and in psycholinguistic reality – the reason already discussed above. Thus, the nearest distributional semantic neighbors of *vision* are: *worldview*, *ideal*, *visionary*, *thinking*, *perspective*, *idea*, *dream*, and *blueprint* – rather than terms related to eyesight.

The noun *spirit* in Table 5 (Sentence 155) is used in the sense of 'strong alcoholic drink'. However, its nearest neighbors in the distributional semantic model are *ethos*, *ideal*, *idealism*, *tradition*, *essence*, *enthusiasm*, *passion*, *faith*, *chivalric*, *zeal*, *credo*, and *compassion* – indicating that the meaning 'your attitude to life or to other people'

---

[3] Continuous Skip-Gram model (Kutuzov et al., 2017), pre-trained on Gigaword 5th Edition corpus

[4] Definitions are cited according to Macmillan Dictionary (n.d.)

is dominant in the model, and the contextual adjustment we apply is not sufficient for overcoming the abstractness of the dominant meaning.

As for the noun *war*, its nearest neighbors in the distributional semantic model are *conflict*, *warfare*, *invasion*, *1991-95 Serbo-Croatian*, *Israel-Hezbollah*, *genocide*, *Bosnia war*, *Jehad*, *civil-war*, *Croatia war*, *Cold War*, *Iran-Iraq*, *wartime*, *Vietnam-like*, etc. – that is, rather abstract concepts. The only more concrete words referring to physical combat action that occur in the distributional semantic neighborhood of *war* are *battlefield* and *bloodshed*, but this is not enough to outweigh the abstract terms. Thus, the distributional semantic model models warfare in terms of abstract rather than concrete (such as names of weapons, military equipment, military personnel, etc.) concepts. As a result, military action is not sufficiently juxtaposed to the metaphoric meaning of *war* as ‘a situation in which two people or groups of people fight, argue, or are extremely unpleasant to each other’.

In the case of *answer* and *agreement*, their nearest distributional semantic neighbors in the model are fairly abstract concepts: *explanation*, *answer*, *reply*, *solution*, *unanswerable*, *query*, *TV-talkback answer*, *question*, and *yes* (for *answer*), and *accord*, *pact*, *deal*, *treaty*, *initial*, *negotiation*, *memorandum*, *compromise*, and *negotiate* (for *agreement*). Meanwhile, human subjects rank *answer* and *agreement* rather high in concreteness; presumably, this is a consequence of conflating the mental representations of the action of answering / reaching an agreement with their two modes – the spoken and the written, i.e. with the physical actions of speaking and writing. This conflation is not reflected in discourse – it largely exists in the mental representations of *answer* and *agreement* and, therefore, is not very distinguishable on the level of linguistic representation.

Of interest are the cases of *heart* and *pain*, which have much lower concreteness in KonKretiKa than in the gold standard sentences where these words are used in their physical, concrete meanings. The nearest distributional semantic neighbors of *heart* are *heart-related*, *coronary artery*, *kidney*, *liver*, *lung*, *arrhythmia*, *cardiac*, *angina*, and *aneurism*. The nearest neighbors of *pain* are *discomfort*, *ache*, *agony*, *tingling sensation*, *numbness*, *soreness*, *menstrual cramp*, *light-headedness*, *stiffness*, *nausea*, and *arthritis*. It would be quite expected for such semantic neighborhood to entitle *heart* and *pain* to higher concreteness values than what they receive in KonKretiKa. A more in-depth analysis into this contradiction revealed that it stems from the vulnerability in the semantic composition of the concrete paradigm which was used to compute the raw indexes (see Table 1). The words of this paradigm belong to the two major semantic classes – living organisms (animals and plants) and man-made artifacts. The class of words denoting human beings was intentionally excluded when the paradigm was compiled on the grounds that such nouns tend to indicate abstract social roles rather than physical humans. As a consequence, physical organic objects such as body parts and organs, or physical sensations and physiological conditions received non-uniform indexes in KonKretiKa: those that refer to humans as well as to animals (e.g. in veterinary or gastronomic discourse) ranked rather high in concreteness: e.g. *liver* (6.6), *pancreas* (6.4), *foot* (6.3), *encephalitis* (6.25), *kidney* (6.25), *entrails* (6.05), *tummy* (5.92), *womb* (5.6) – whereas those that tend to be primarily associated with humans received lower indexes, e.g. *heart* (2.63), *heartburn* (2.57), *scar* (2.53), *nausea* (2.5), *headache* (1.61), *distress* (1.5), *pain* (1.21), *queasiness* (1.12), etc. Thus, comparison of the KonKretiKa computational indexes with the psycholinguistic data of CONcreTEXT allowed us to detect a potential shortcoming in our approach to the design of the concrete paradigm. As was noted in previous study (Badryzlova, 2020), the class of concrete words seems to be more semantically heterogeneous than of abstract words; therefore, it may reasonable in future experiments to diversify the concrete paradigm and expand it in size by including words that denote human beings.

4 Conclusions

We presented KonKretiKa system for computing concreteness indexes of English words in context; the system was submitted to the English track of the CONcreTEXT shared task. The best modification of KonKretiKa ranked third in the task, with $r_s = .6634$ and $r = .6685$ against the gold standard. We treat concreteness as a bimodal problem and use paradigm lists of concrete and abstract words to compute two indexes for each word, that of concreteness and of abstractness. The single aggregate index indicative of both the word’s concreteness and abstractness is computed as the function of the two respective indexes. The set of raw aggregate indexes is transformed using sigmoid transformation to increase the variance and to attain greater similarity to the psycholinguistic data. To dynamically adjust the concreteness indexes to the context, we apply an adjustment coefficient. Post hoc analysis of the adjustment coefficient indicates that lower coefficients lead to better performance. We hypothesize that the contribution of the adjustment coefficient could be increased by expanding the scope of the context, for example, by considering one or more sentences from the left and the right contexts of the target sentence. According to our analysis, the main source of divergence between the computational and the psycholinguistic indexes lies in the different representation, or salience, of word meanings in distributional semantic models and in psycholinguistic reality. Besides, analysis of divergences between the computational and the psycholinguistic rankings prompted us a potential direction for reducing the bias in composition of the concreteness paradigm, which can be overcome by diversifying the paradigm.

References

Badryzlova, Y., 2020. Exploring Semantic Concreteness and Abstractness for Metaphor Identification and Beyond. Computational Linguistics and Intellectual Technologies 33–47.

Badryzlova, Y., 2019. Automated metaphor identification in Russian texts. National Research University Higher School of Economics, Moscow.

Badryzlova, Y., Panicheva, P., 2018. A Multi-feature Classifier for Verbal Metaphor Identification in Russian Texts, in: Conference on Artificial Intelligence and Natural Language. Springer, pp. 23–34.

Barsalou, L.W., 2008. Grounded cognition. Annu. Rev. Psychol. 59, 617–645.

Basile, V., Croce, D., Di Maro, M., Passaro, L.C., 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian, in: Basile, V., Croce, D., Di Maro, M., Passaro, L.C. (Eds.), Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020). CEUR.org, Online.

Birke, J., Sarkar, A., 2006. A Clustering Approach for Nearly Unsupervised Recognition of Nonliteral Language., in: EACL.

Brysbaert, M., Warriner, A.B., Kuperman, V., 2014. Concreteness ratings for 40 thousand generally known English word lemmas. Behavior research methods 46, 904–911.

Coltheart, M., 1981. The MRC psycholinguistic database. The Quarterly Journal of Experimental Psychology Section A 33, 497–505.

Fellbaum, C., 1998. WordNet: An electronic database. MIT Press, Cambridge, MA.

Gregori, L., Montefinese, M., Radicioni, D.P., Ravelli, A.A., Varvara, R., 2020. CONcreTEXT @ Evalita2020: the Concreteness in Context Task, in: Basile, V., Croce, D., Di Maro, M., Passaro, L.C. (Eds.), Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020). CEUR.org, Online.

Kulagin, D., 2019. Opy`t sozdaniya mashinno-proveryaemoj semanticheskoj razmetki russkix sushhestvitel`ny`x [Developing computationally verifiable semantic annotation of Russian nouns]. Presented at the Annual International Conference “Dialogue,” Moscow.

Kutuzov, A., Fares, M., Oepen, S., Velldal, E., 2017. Word vectors, reuse, and replicability: Towards a community repository of large-text resources, in: Proceedings of the 58th Conference on Simulation and Modelling. Linköping University Electronic Press, pp. 271–276.

Lakoff, G., Johnson, M., 1980. Metaphors we Live by, 2nd ed. The University of Chicago Press, Chicago-London.

Macmillan Dictionary, Free English Dictionary and Thesaurus [WWW Document], n.d. URL https://www.macmillandictionary.com/ (accessed 11.7.20).

Parker, R., Graff, D., Kong, J., Chen, K., Maeda, K., 2011. English Gigaword Fifth Edition LDC2011T07 (Tech. Rep.). Technical Report. Linguistic Data Consortium, Philadelphia.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., 2011. Scikit-learn: Machine Learning in Python Journal of Machine Learning Research.

Tsvetkov, Y., Mukomel, E., Gershman, A., 2013. Cross-lingual metaphor detection using common semantic features, in: Proceedings of the First Workshop on Metaphor in NLP. pp. 45–51.

Turney, P.D., Neuman, Y., Assaf, D., Cohen, Y., 2011. Literal and metaphorical sense identification through concrete and abstract context, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pp. 680–690.

Ghigliottin-AI: Evaluating Artificial Players for the Language Game “La Ghigliottina”

Ghigliottin-AI @ EVALITA2020: Evaluating Artificial Players for the Language Game "La Ghigliottina"

Pierpaolo Basile
Dept. of Computer Science
University of Bari, Italy
pierpaolo.basile@uniba.it

Marco Lovetere
Ghigliottiniamo
marlove@gmail.com

Johanna Monti and **Antonio Pascucci**
UNIOR NLP Research Group
"L'Orientale" University of Naples, Italy
{jmonti, apascucci}@unior.it

Federico Sangati
UNIOR NLP Research Group
"L'Orientale" University of Naples, Italy
OIST Graduate University, Japan
federico.sangati@gmail.com

Lucia Siciliani
Dept. of Computer Science
University of Bari, Italy
lucia.siciliani@uniba.it

Abstract

English. Evaluating Artificial Players for the Language Game "La Ghigliottina" (Ghigliottin-AI) task is one of the tasks organized in the context of the 2020 EVALITA edition, a periodic evaluation campaign of Natural Language Processing (NLP) and speech tools for the Italian language. Ghigliottin-AI participants are asked to build an artificial player able to solve "La Ghigliottina", namely the final game of an Italian TV show called "L'Eredità". The game involves a single player who is given a set of five words unrelated to each other, but related with a sixth word that represents the solution to the game. Fourteen teams registered to Ghigliottin-AI. Nevertheless, only two teams submitted their run. In order to evaluate the submitted systems, we rely on an API base methodology, via a Remote Evaluation Server (RES). In this report we describe the Ghigliottin-AI task, the data, the evaluation and we discuss results.

1 Background and Motivation

Language games draw their challenge and excitement from the richness and ambiguity of natural language, and therefore have attracted the attention of researchers in the fields of Artificial Intelligence and Natural Language Processing. For instance, IBM Watson is a system which successfully challenged human champions of "Jeopardy!", a game in which contestants are presented with clues in the form of answers, and must phrase their responses in the form of a question (Ferrucci et al., 2010; Molino et al., 2015). Another popular language game is solving crossword puzzles. The first experience reported in the literature is Proverb (Littman et al., 2002), that exploits large libraries of clues and solutions to past crossword puzzles. WebCrow is the first solver for Italian crosswords (Ernandes et al., 2008).

Following the first edition of the NLP4FUN task (Basile et al., 2018), proposed at EVALITA 2018, we propose a new edition of the task whose aim is to design a solver for "The Guillotine" (La Ghigliottina, in Italian) game. It is inspired by the final game of an Italian TV show called "L'Eredità". The game, broadcast by Italian national TV, involves a single player, who is given a set of five words - the clues - each linked in some way to a specific word that represents the unique solution of the game. Words are unrelated to each other, but each of them has a hidden association

---

with the solution. Once the clues are given, the player has one minute to find the solution. For example, given the five clues: *pie, bad, Adam, core, eye* the solution is *apple*, because: apple-pie is a kind of pie; bad apple is a way of referring to a trouble maker; Adam's apple is the prominent part of men's throat; apple core is the centre of the apple; apple of someone's eye is way of referring to someone's beloved person. This report is organized as follows: in Section 2 we describe the *Ghigliottin-AI* task. In Section 3 we present the dataset. The task evaluation is in Section 4. Results achieved by participants are shown in Section 5. Conclusions are in Section 6.

2 Task Description

Evaluating Artificial Players for the Language Game "La Ghigliottina" (Ghigliottin-AI) is one of the fourteen EVALITA 2020 tasks (Basile et al., 2020). Ghigliottin-AI participants are asked to build an artificial player able to solve "La Ghigliottina". They can take advantage of solutions adopted by previous systems (Semeraro et al., 2009; Basile et al., 2016; Sangati et al., 2018) and the availability of open repositories on the web.

3 Dataset

We provided a set of 300 games with their solution taken from the last editions of the TV game as training data. The training data was released in JSON format as shown in Figure 1. In this example, the first JSON shows the clues "posto" (literally *place*), "artificiale"(*artificial*), "lavaggio" (*washing*), "allenare" (literally *to train*) and "gallina" (*chicken*) and the solution "cervello" (*brain*): *non avere il cervello a posto* (*to be nutty*), *cervello artificiale* (*artificial brain*), *lavaggio del cervello* (*brainwashing*), *allenare il cervello* (*stretch the brain*) and *cervello da gallina* (*hare-brained*). In the second JSON we find "essere" (*to be*), "comparsa" (*appearance*), "x men", "ronaldo" and "mondiale" (*global*) and the solution "fenomeno" (*phenomenon*): *essere un fenomeno* (*be a phenomenon*), *comparsa di un fenomeno* (*apperance of a phenomenon*), *Fenomeno* is one of the X-men, *Fenomeno* was Ronaldo's nickname and *fenomeno mondiale* (*worldwide phenomenon*).

The test set consists in 350 games instances, provided by a Remote Evaluation Server (RES) *Ghigliottiniamo*[1] at random intervals of time as a request with a single game challenge to registered systems. The RES allowed the systems to reply with a single solution to the game. *Ghigliottiniamo*[2] currently enables both humans and artificial systems to submit solutions to the TV game in real-time.

```
[
  {
     "w1": "posto",
     "w2": "artificiale",
     "w3": "lavaggio",
     "w4": "allenare",
     "w5": "gallina",
     "solution": "cervello"
   },
   {
     "w1": "essere",
     "w2": "comparsa",
     "w3": "x men",
     "w4": "ronaldo",
     "w5": "mondiale",
     "solution": "fenomeno"
   },
   ...
]
```

Figure 1: JSON format of the training set.

4 Task evaluation

In order to evaluate the AI systems, we rely on an API based methodology. During the evaluation period, at random intervals of time (over a period of 7 days), the RES submitted 350 game challenges to the registered systems. The systems had to reply back to the RES with a single solution to the game.

As evaluation measure, we adopt the standard accuracy score:

$$\frac{\text{solved games}}{\text{total games}} \qquad (1)$$

As in the TV game, where players have one minute to provide the solution, the RES will discard system solutions received after 60 seconds from the submitted challenge.

---

[1] `https://quiztime.net`

[2] `https://play.google.com/store/apps/details?id=io.quiztime.game`

5 Results

Fourteen teams registered to the Ghigliottin-AI task. However, only two teams participated to the final test: *GUL.LE.VER* (De Francesco, 2020) and *Il Mago della Ghigliottina* (Sangati et al., 2020). GUiLlotine gLovE resolVER (GUL.LE.VER) is based on the Glove (Pennington et al., 2014) vector representation of the words on the basis of a large collected dataset, containing the Italian Wiktionary, Wikiquote, Wikipedia (only titles), the Italian Collocations Dictionary and other resources scraped on the web containing Italian multiword expressions, proverbs and songs titles. The Glove algorithm was chosen for its intrinsic power in capturing the co-occurrence correlation between two words that are not synonyms, due to the co-occurrence matrix that the algorithm builds before the training. The solution is searched in the vector space near the clues, obtaining a list of solution candidates. This list is descending reordered using a hybrid function composed by two parts: one part is based on the Pointwise Mutual Information; the other one is based on the weighted sum of the cosine similarity between the candidate solutions and the clues, in which the weight is the normalized IDF of the single clue in the corpus (solutions that are correlated with the rarest clues are more important than others). *Il Mago della Ghigliottina* is the same system submitted with the name of *UNIOR4NLP* in the *NLP4FUN* task in 2018 without any changes. The system is based on the observation that most cases clues and solution are connected because they form a multiword expression. In addition, clues are almost always nouns, verbs or adjectives, while solutions are nouns or adjectives. The system is based on a number of freely available corpora, such as: Paisà[3]; itWaC[4]; Wiki-IT-Titles downloaded via WikiExtractor[5]; 1955 proverbs from Wikiquote[6] and 371 from an online collection[7] downloaded on the 24th April 2018. Further lexical resources were developed from "Il Nuovo vocabolario di base della lingua italiana" and from the "De Mauro online dictionary". Technical details about *Il Mago della Ghigliottina* are available in (Sangati et al., 2018), submitted for the *NLP4FUN* task.

Table 1 shows the results of the two systems.

System	Correct	Total	Acc.
GUL.LE.VER	94	350	0.269
Il Mago della Ghigliottina	240	350	**0.686**
Combined (upper bound)	257	350	0.734

Table 1: Results

Both systems were able to provide a solution to all 350 games within a minute. The recorded time of the two systems ranges between 0.316 and 9.988 seconds. It is important to keep in mind that in addition to the response time, the recorded time includes the latency of the network and the time required for the instance to wake-up if it is set to go to sleep when idle. *Il Mago della Ghigliottina* is the system with the highest accuracy (about three solutions out of four correct), followed by *GUL.LE.VER* which on average is able to solve one game out of four.

We have computed the upper bound of the accuracy of the two systems on the test set when used in combination. The resulting accuracy is 73.4%, about 5 percentage points above the best performing system. This means that the two systems have some complementary and could be used in combination with some aggregating strategy.

6 Conclusions

In this report we presented *Ghigliottin-AI*, one of the EVALITA 2020 task. Despite fourteen teams subscribed to the task, just two of them submitted their system, namely *GUL.LE.VER* and *Il mago della Ghigliottina*. This latter achieved the best performances in terms of accuracy (68.6%), while *GUL.LE.VER* obtained 26.9% of accuracy.

Systems have been evaluated through an API methodology conducted by the Remote Evaluation Server (RES) (*Ghigliottiniamo*). To our knowledge, this is the first time that an API based system has been used on a NLP evaluation task. We believe this methodology has a strong advantage compared to a manual evaluation, as systems can be tested more systematically, fairly and continuously in time. We strongly hope that more

---

[3] `https://www.corpusitaliano.it/`

[4] `https://wacky.sslmit.unibo.it/doku.php?id=corpora\#italian`

[5] `http://attardi.github.io/wikiextractor.`

[6] `https://it.wikiquote.org/wiki/Proverbi_italiani`

[7] `http://web.tiscali.it/proverbiitaliani`

tasks will adopt this evaluation strategy in the future. The Ghigliottiniamo system currently enables both humans and artificial systems to submit solutions to the *Ghigliottina* when a new game is broadcasted on TV. This will allow us in the future to compare their results more systematically. The system remains open for new artificial systems to join the live competition[8].

References

Pierpaolo Basile, Marco de Gemmis, Pasquale Lops, and Giovanni Semeraro. 2016. Solving a complex language game by using knowledge-based word associations discovery. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(1):13–26.

Pierpaolo Basile, Marco de Gemmis, Lucia Siciliani, and Giovanni Semeraro. 2018. Overview of the evalita 2018 solving language games (nlp4fun) task. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*, Turin, Italy. CEUR.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Nazareno De Francesco. 2020. Gul.le.ver, a glove based artificial player to solve the language game "la ghigliottina". In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*.

Marco Ernandes, Giovanni Angelini, and Marco Gori. 2008. A web-based agent challenges human experts on crosswords. *AI Magazine*, 29(1):77.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. 2010. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79.

Michael L Littman, Greg A Keim, and Noam Shazeer. 2002. A probabilistic approach to solving crossword puzzles. *Artificial Intelligence*, 134(1-2):23–55.

Piero Molino, Pasquale Lops, Giovanni Semeraro, Marco de Gemmis, and Pierpaolo Basile. 2015. Playing with knowledge: A virtual player for "who wants to be a millionaire?" that leverages question answering techniques. *Artificial Intelligence*, 222:157–181.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Federico Sangati, Antonio Pascucci, and Johanna Monti. 2018. Exploiting multiword expressions to solve "la ghigliottina". In *Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, volume 2263, pages 258–263. Accademia University Press.

Federico Sangati, Antonio Pascucci, and Johanna Monti. 2020. "il mago della ghigliottina"@ghigliottin-ai when linguistics meets artificial intelligence. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*.

Giovanni Semeraro, Pasquale Lops, Pierpaolo Basile, and Marco De Gemmis. 2009. On the tip of my thought: Playing the guillotine game. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence*, IJCAI'09, pages 1543–1548, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

---

[8] `https://quiztime.net`

“Il Mago della Ghigliottina” @ Ghigliottin-AI: When Linguistics meets Artificial Intelligence

Federico Sangati[1,2], Antonio Pascucci[1], and Johanna Monti[1]
[1]L’Orientale University of Naples - UNIOR NLP Research Group, Italy
[2]Okinawa Institute of Science and Technology Graduate University, Japan
`federico.sangati@gmail.com`, `{apascucci,jmonti}@unior.it`

Abstract

English. This paper describes *Il mago della Ghigliottina*, a bot which took part in the *Ghigliottin-AI* task of the Evalita 2020 evaluation campaign. The aim is to build a system able to solve the TV game “La Ghigliottina”. Our system has already participated in the Evalita 2018 task *NLP4FUN*. Compared to that occasion, it improved its accuracy from 61% to 68.6%.

Italiano. *Questo contributo descrive Il mago della Ghigliottina, un bot che ha partecipato a Ghigliottin-AI, uno dei task di Evalita 2020. Scopo del task è mettere in piedi un sistema automatico capace di risolvere il gioco televisivo “La Ghigliottina”. Il nostro sistema ha già partecipato all’edizione del 2018 di Evalita al task NLP4FUN. Rispetto all’edizione del 2018 di NLP4FUN, l’accuratezza è salita dal 61% al 68.6%.*

1 Introduction

In this paper we describe *Il mago della ghigliottina* (Sangati et al., 2020), a bot which participated in *Ghigliottin-AI*, one of the Evalita 2020 tasks (Basile et al., 2020a). Evalita[1] (Basile et al., 2020b) is an initiative of AILC (Associazione Italiana di Linguistica Computazionale) and is a periodic evaluation campaign of Natural Language Processing (NLP) and speech tools for the Italian language, which takes place every two years in conjunction with CLiC-IT[2], the Italian Conference on Computational Linguistics. *Ghigliottin-AI* takes its cue from the Evalita 2018 *NLP4FUN* (Basile et al., 2018) task. Participants are asked to build an artificial player able to solve “La Ghigliottina”, the final game of the popular Italian TV quiz show “L’Eredità”. The game involves a single player, who is given a set of five words (clues), unrelated one to each other, but related with a sixth word that represents the solution to the game. Our system took already part in the 2018 Evalita task *NLP4FUN* as *UNIOR4NLP* (Sangati et al., 2018). *Il mago della Ghigliottina* is identical to *UNIOR4NLP*, being based on the same principles and methodologies: analyzing real game instances we found out that in most cases clues and solution are connected because they form a Multiword Expression (MWE). A MWE can be defined as a sequence of words that presents some characteristic behaviour (at the lexical, syntactic, semantic, pragmatic or statistical level) and whose interpretation crosses the boundaries between words (Sag et al., 2002). MWEs are lexical items which convey a single meaning different from the meanings of the constituents of the MWE, such as in the idiomatic expression *kick the bucket* where the simple addition of the meanings of *kick* and *bucket* does not convey the meaning of *to die*. We have decided to participate as *Il mago della ghigliottina* instead of *UNIOR4NLP* because after participating in the *NLP4FUN* task in 2018 we developed three different versions of the solver *Il mago della ghigliottina* available as i) a Telegram Bot (@Unior4NLPbot)[3], ii) a Twitter bot (@UNIOR4NLP) and finally iii) an Amazon Alexa skill (Mago della Ghigliottina). This paper is organized as follows: in Section 2 we present related work and in Section 3 we provide an overview of the task. In Section 4 we describe our system. Results are shown in Section 5 while in Section 6 we focus on the error anaysis. Conclusions are in Section 7 along with future work.

---

[1] http://www.evalita.it

[2] http://clic2020.ilc.cnr.it/it/home

[3] A short video showing how the bot works is available at https://youtu.be/3fggGlJaSII

2 Related work

Languages have always been a source of inspiration to create games. As the years passed, the possibility to rely on large linguistic resources and artificial intelligence has allowed scholars to build systems able to solve games, which represent an interesting playground to test the results of research (Yannakakis and Togelius, 2018). When we think about linguistics and artificial intelligence it is almost obvious to think to the IBM Watson system, which successfully challenged human champions of Jeopardy!^{TM}, a game where contestants are presented with clues in the form of answers, and must phrase their responses in the form of a question (Ferrucci et al., 2013). Another interesting example is represented by solvers of Italian crosswords (Ernandes et al., 2008; Littman et al., 2002). The scientific community periodically organizes i) shared tasks to evaluate Natural Language Processing (NLP) applications in the solution of linguistic games (*Ghigliottin-AI* is an example) and ii) workshops focused on games and gamification for NLP tasks. The *Games and NLP* (Lukin, 2020) workshop, for instance, was organized this year in the context of the LREC 2020 conference. Fourteen teams presented their research in occasion of this workshop, and according to the submitted papers, we can state that the research moves in two directions: i) the exploitation of NLP techniques to solve linguistic games on the basis of semantic relations between words and ii) the development of *Games With A Purpose (GWAPs)* in order to crowdsource linguistic data from engaged players.

TV games, such as "*Wheel of Fortune*", "*Who wants to be a Millionaire?*" and, indeed, "*La Ghigliottina*" represent an interesting test bench for linguistic knowledge-based systems. (Molino et al., 2015) exploit question answering techniques to build an artificial player for *Who wants to be a Millionaire?*. With regard to our specific case study, other systems were built to solve "*La Ghigliottina*". OTTHO (Semeraro et al., 2009; Basile et al., 2016), the first artificial player of "*La Ghigliottina*", is a system based on i) web resources (e.g. Wikipedia) in order to build a lexicon and a knowledge repository and ii) a knowledge base modeling represented by an association matrix which stores the degree of correlation between any two terms in the lexicon. Word correlations are detected by connecting i) lemmas to the terms in its dictionary definition, pair of words occurring in a proverb, movie or song title, and ii) pair of similar words by exploiting Vector Space Models (Salton et al., 1975). During the *NLP4FUN* Task in 2018 two systems took part in the competition: our system (which is presented in Section 4) and (Squadrone, 2018), that proposed an algorithm based on two steps: i) for each clue of a game, a list of relevant keywords is retrieved from linguistic corpora, so that each clue is associated with keywords representing the concepts having a relation with that clue. Then, words at the intersection of the retrieved sets are considered as candidate solutions; ii) another knowledge source made of proverbs, book and movie titles, word definitions, is exploited to count co-occurrences of clues and candidate solutions. A further system developed to solve "*La Ghigliottina*" game is Robospierre (Cirillo et al., 2019), a system which relies on MWEs automatically extracted through a lexicalized association rules algorithm, on a list of proverbs and on some lists of titles.

3 The *Ghigliottin-AI* task

Ghigliottin-AI is one of the Evalita 2020 tasks. The aim of Evalita (which in 2020 reached its seventh edition) is to promote the development of language and speech technologies for Italian, providing a shared framework where different systems and approaches can be evaluated in a consistent manner. *Ghigliottin-AI* participants are asked to build an artificial player able to solve "La Ghigliottina", the final game of the Italian TV show "L'Eredità". Given a set of five words (clues) the player has to find the solution to the game which is a sixth word related with each one of the five clues. The five clues are unrelated one to each other. For example, given the set of clues *conoscere* (*to know*), *grado* (*degree*), *modello* (*model*), *ideale* (*ideal*) and *divina* (*divine*) the solution is *perfezione* (*perfection*) because: *conoscere alla perfezione* (*to perfectly know*), *grado di perfezione* (*degree of perfection*), *modello di perfezione* (*model of perfection*), *ideale di perfezione* (*ideal of perfection*) and *perfezione divina* (*divine perfection*). In order to train participants' systems, the task organizers provided a set of 300 games with their five clues and their solution in a JSON format. This training set is taken from the last editions of the TV game. The systems have been then evaluated using an API based methodology, namely the Remote Evaluation Server (RES) *Ghigliottiniamo*[4] which currently enables both humans and artificial systems (bots) to submit solutions to the TV game in real-time. The test set consists in 350 games instances, provided by *Ghigliottiniamo* at random intervals of time as a request with a single game challenge to registered systems. The RES allowed systems to reply with a single solution to the game. Similar to the original TV game, where players have 60 seconds to provide the solution, the RES discards solutions received after 60 seconds from the submitted challenge. The same happened in evaluating systems participating in *Ghigliottin-AI*.

4 System description

This section describes *Il mago della Ghigliottina*, which, as already mentioned, is the system submitted in 2018 without any changes. The system is based on the analysis of real game instances: in most cases clues and solution are connected because they form a MWE. A further observation is that clues are always nouns, verbs or adjectives, while solutions are nouns or adjectives. On this basis, we have detected six patterns that identify MWEs connecting clue/solution pairs:

A B pattern: *diario segreto* ('diary secret' → secret diary), *brutta caduta* ('ugly fall' → bad fall), *permesso premio* ('permit price' → good behaviour license), *dare gas* ('give gas' → accelerate).

A det B pattern: *dare il permesso* ('give the permit' → authorize).

A prep B pattern: *colpo di coda* ('flick of tail' → last ditch effort).

A conj B pattern: *stima e affetto* (esteem and affection).

A prepart B or **A prep det B** pattern: e.g. *virtù dei forti*, part of the famous Italian proverb *La calma è la virtù dei forti* (patience is the virtue of the strong).

A+B pattern: compounds such as *radio* + *attività* = *radioattività* (radio + activity = radioactivity).

The system is based on a number of freely available corpora:

Paisà : 225 M words corpus automatically annotated (Lyding et al., 2014).

itWaC : 1.5 B words corpus automatically annotated (Baroni et al., 2009)

Wiki-IT-Titles : Wikipedia-IT titles downloaded via WikiExtractor[5].

Proverbs : 1955 proverbs from Wikiquote[6] and 371 from an online collection[7].

In addition, we have developed the following lexical resources:

DeMauro-Ext : words extracted from "Il Nuovo vocabolario di base della lingua italiana"(De Mauro, 2016b), extended with morphological variations obtained by changing last vowel of the word and checking if the resulting word has frequency ≥ 1000 in *Paisà*.

DeMauro-MWEs : MWEs extracted from the "De Mauro online dictionary" (De Mauro, 2016a) composed of 30,633 entries.

More technical details about our system are available in (Sangati et al., 2018), submitted for the *NLP4FUN* task.

5 Results

In this section, we discuss results and we also compare the performances achieved by our system in *Ghigliottina-AI* with those achieved in the Evalita *NLP4FUN* task. Compared to our participation in *NLP4FUN*, when our system proved to be the best performing one (Basile et al., 2018), the accuracy has increased from 61.0% to 68.6%. This is probably due to the fact that while the 2020 edition only used games from the TV game, in the 2018 edition 39 out of the 105 games in the test set were taken from the board game. This supports what already reported in (Sangati et al., 2018), that is, the board game edition presents different types of word-association as compared to the TV game. The Table 1 provides the performances of our system in both editions of the task.

---

[4] https://quiztime.net

[5] http://attardi.github.io/wikiextractor. Last accessed on the 1st October 2018

[6] https : / / it . wikiquote . org / wiki / Proverbi _ italiani. Downloaded on the 24th April 2018

[7] http://web.tiscali.it/proverbiitaliani. Downloaded on the 24th April 2018

Task	Correct	Total	Accuracy
Ghigliottin-AI (2020)	240	350	68.6%
NLP4FUN (2018)	64	105	61.0%

Table 1: Results on the *Ghigliottin-AI* and the *NLP4FUN* TEST sets. In the column "Total" we show the number of game instances in the test set. Accuracy is computed as the number of correct games divided the total.

6 Error analysis

In the attempt of providing the correct solution to the 350 game instances that compose the *Ghigliottin-AI* test set, 110 errors have been made, which represent 32.4% of the whole test set. In this section we discuss the errors, trying to analyze and justify them. In particular, we try to detect the motivation behind errors, in order to categorize them. The following list presents examples of different categories of errors we detected.

6.1 High correlation between clue(s) and our solution.

One or more clues have a high correlation with the wrong solution provided by the system.

A clues: *fare* (*to do*), *saldo* (two different meanings *sale* and *balance*), *interessato* (*interested*), *grande* (several meanings, such as *big* and *great*) and *attenzione* (*attention*). Our system provided the solution *shopping* (the same in English) instead of the right one *richiesta* (*request*). In this case the system didn't disambiguate correctly the meaning of *saldo* (*richiesta di saldo*, namely *balance request*). The system chose the solution *shopping* instead of *richiesta* due to the high correlation between *shopping* and *saldo* (*sale*). One possible explanation is that *shopping* and *saldo* almost always occur in the same sentence. For this reason the solution *shopping* achieved a higher weight compared to that of other solutions;

B clues: *brutto* (*ugly*), *fare* (*to do*), *morto* (*dead*), *cavaliere* (*kinght*) and *diavolo* (*devil*). The solution is *paura* (*fear*), while our system provided the solution *povero* (*poor*). Considering that our system is also trained with a list of proverbs, in this case the error is due to the high correlation between *povero* and *diavolo* (*povero diavolo*) is a famous way of saying;

C clues: *perdere* (*to lose*), *amicizia* (*frienship*), *bottiglia* (*bottle*), *acqua* (*water*) and *quattro* (*four*). The right solution is *segno* (*sign*), but our system provided the solution *bicchiere* (*glass*) due to the high correlation with *bottiglia* and *acqua*.

6.2 Right kind of reasoning, wrong solution.

Wrong solutions such as singular instead of plural (and vice-versa), or trivial mistakes in the face of a right kind of reasoning.

D clues: *questione* (*question*), *indagine* (*investigation*), *disegno* (*design*), *pagamento* (*payment*) and *lavorare* (*to work*). Instead of *metodo* (*method*), our system provided its plural *metodi*;

E clues: *copertina* (*cover*), *dimensione* (*dimension*), *persona* (*person*), *seno* (*sinus*) and *età* (*age*). The solution is *terza* (*third*), but our system provided the wrong solution *quarta* (*fourth*) which has correlation with all the five clues.

6.3 Clue(s) and solution are synonyms.

The solution provided by our system is a synonym of one or more clue(s).

F clues: *essere* (*to be*), *prezzo* (*price*), *fermo* (*stop*), *capitale* (*capital*) and *regolare* (*regular*). The solution is *partenza* (*departure*), but our system provided the solution *fisso* (*fixed*) which can be intended as a synonym of *fermo*.

6.4 Unclear solutions.

This subsection discusses some strange solutions provided by the system. The solutions that our system detects are listed from the best to the worst one. Then, it chooses the best one. Thanks to a debug function it is possible to analyze the solutions provided by the system in order to understand what is their correlation with one or more clues. The examples provided below concern solutions apparently strange which we analyzed thanks to this function.

G clues: *vecchio* (*old*), *cavallo* (*horse*), *end* (the same in English), *soda* (the same in English) and *conquista* (*conquest*). The solution is *west* (the same in English). Our system provided the solution *polenta* (the same in English), which is a dish as well as the surname of a famous Italian commander lived in the 13th century (Guido da Polenta), also knows as "il Vecchio" (the Elder);

H clues: *gioco* (*game*), *trovare* (*to find*), *fuori* (*out*), *dollaro* (*dollar*) and *quadrato* (*square*). The solurion is *area* (the same in English), but our system provided the solution *straccio* (*shred*), due to the high correlation with *trovare* because of the way of saying *non trovare uno straccio di prova* (*do not have a shred of evidence*);

I clues: *erba* (*grass*), *sangue* (*blood*), *indagine* (*investigation*), *prova* (*evidence*) and *miss* (*Miss*). The solution is *campione* (*champion*), but our system provided the solution *pazienza* (*patience*), because of the high correlation with *erba*: *Erba pazienza* (*Patience Dock*) is the common name for the Rumex patientia plant.

Debugging the system also allows us to observe if the right solution is in the list of best solutions provided by the system and how it is ranked. Statistics based on the 110 errors recorded during the test phase are reported in Table 2, where "best of 5" means: best solutions detected when there is correlation between each one of the solutions and all the five clues. The same reasoning applies to "best of 4" and "best of 3".

Correct solution is	Occurrences
the 2nd best solution	22
in the Best of 5 list	30
in the Best of 4 list	13
in the Best of 3 list	6
not in the list	61

Table 2: Correct solutions in our system list when error solutions are provided as best solution

As we can see, in 22 cases the correct solution is the second best solution detected by our system. In 61 cases the correct one is not in the whole list of possible solutions detected by the system.

6.5 Part-of-speech errors.

We also noticed that some errors are due to the selection of solutions with a wrong part-of-speech (POS). In Table 3 we report the occurrences of POS errors.
In particular, in 26 cases the system selected an adjective as solution instead of a noun, for example:

J - clues: *scrivere* (*to write*), *rosso* (*red*), *luce* (*light*), *colori* (*colors*) and *inchiesta* (*inquiry*). The solution is *film* (*movie*), namely a Noun. In this case while our system provided the solution *giallo* (*yellow*) (an Adjective). We can also note that the error solution has been provided because two of the five clues (*rosso* and *colori*) are related to the same conceptual group of *giallo*, namely colors.

Error POS	Correct POS	Occurrences
Noun	Noun	80
Adjective	Noun	26
Noun	Adjective	2
Verb	Noun	2
Noun	Adjective	-

Table 3: Occurrences of error POS provided by our system instead of correct POS

7 Conclusions and future work

In this paper we described *Il mago della ghigliottina*, a system which took part in the Evalita 2020 *Ghigliottin-AI* task. Our system achieved an accuracy of 0.6857, with 240 correct solutions given on a test set composed of 350 game instances. As already mentioned, our system is the same system which took part in the Evalita 2018 *NLP4FUN* task and is designed on a key observation: clues are connected to the solution because they form a multiword expression (MWE). In order to build our system, we collected linguistic and lexical resources described in Section 4. Since future work will focus on improving the performances of the system, a special focus has been dedicated to error analysis. Section 6, in fact, presents different categories of errors we detected (with examples and clarification of errors) as well as statistics about correct solutions presence in our system list of solutions.

Acknowledgements

This research has been partly supported by the PON Ricerca e Innovazione 2014/20 fund. Authorship contribution is as follows: Federico Sangati is author of Sections 4 and 5, Antonio Pascucci is author of Sections 3 and 6, Johanna Monti is author of Sections 1 and 2, Abstract, Conclusions and future work are in common.

References

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Lang. Resources and Evaluation*, 43(3):209–226.

Pierpaolo Basile, Marco de Gemmis, Pasquale Lops, and Giovanni Semeraro. 2016. Solving a complex language game by using knowledge-based word associations discovery. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(1):13–26.

Pierpaolo Basile, Marco de Gemmis, Lucia Siciliani, and Giovanni Semeraro. 2018. Overview of the evalita 2018 solving language games (nlp4fun) task. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*. CEUR.org, Turin, Italy.

Pierpaolo Basile, Marco Lovetere, Johanna Monti, Antonio Pascucci, Federico Sangati, and Lucia Siciliani. 2020a. Ghigliottin-ai@evalita2020 evaluating artificial playersfor the language game "la ghigliottina". In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020b. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*. CEUR.org, Online.

Nicola Cirillo, Chiara Pericolo, and Pasquale Tufano. 2019. Robospierre, an artificial intelligence to solve" la ghigliottina". In *CLiC-it*.

Tullio De Mauro. 2016a. Il Nuovo De Mauro (Online). https://dizionario.internazionale.it. Last accessed on the 1st October 2018.

Tullio De Mauro. 2016b. Il Nuovo vocabolario di base della lingua italiana (pdf version). https://www.internazionale.it/opinione/tullio-de-mauro/2016/12/23/il-nuovo-vocabolario-di-base-della-lingua-italiana. Last accessed on the 1st October 2018.

Marco Ernandes, Giovanni Angelini, and Marco Gori. 2008. A web-based agent challenges human experts on crosswords. *AI Magazine*, 29(1):77–77.

David A. Ferrucci, Anthony Levas, Sugato Bagchi, David Gondek, and Erik T. Mueller. 2013. Watson: Beyond jeopardy! *Artif. Intell.*, 199:93–105.

Michael L Littman, Greg A Keim, and Noam Shazeer. 2002. A probabilistic approach to solving crossword puzzles. *Artificial Intelligence*, 134(1-2):23–55.

Stephanie M. Lukin, editor. 2020. *Workshop on Games and Natural Language Processing*. European Language Resources Association, Marseille, France.

Verena Lyding, Egon Stemle, Claudia Borghetti, Marco Brunello, Sara Castagnoli, Felice Dell'Orletta, Henrik Dittmann, Alessandro Lenci, and Vito Pirrelli. 2014. The PAISÀ corpus of italian web texts. In *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, pages 36–43. Association for Computational Linguistics, Gothenburg, Sweden.

Piero Molino, Pasquale Lops, Giovanni Semeraro, Marco de Gemmis, and Pierpaolo Basile. 2015. Playing with knowledge: A virtual player for "who wants to be a millionaire?" that leverages question answering techniques. *Artificial Intelligence*, 222:157–181.

Ivan A Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for nlp. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 1–15. Springer.

Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620.

Federico Sangati, Antonio Pascucci, and Johanna Monti. 2018. Exploiting multiword expressions to solve "la ghigliottina". In *Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, volume 2263, pages 258–263. Accademia University Press.

Federico Sangati, Antonio Pascucci, and Johanna Monti. 2020. The challenge of the tv game la ghigliottina to nlp. In *Workshop on Games and Natural Language Processing*, pages 34–38.

Giovanni Semeraro, Pasquale Lops, Pierpaolo Basile, and Marco De Gemmis. 2009. On the tip of my thought: Playing the guillotine game. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence*, IJCAI'09, pages 1543–1548. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Luca Squadrone. 2018. Computer challenges guillotine: how an artificial player can solve a complex language tv game with web data analysis. *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:262.

Georgios N Yannakakis and Julian Togelius. 2018. *Artificial Intelligence and Games*. Springer.

GUL.LE.VER @ GhigliottinAI: A Glove based Artificial Player to Solve the Language Game “La Ghigliottina”

Nazareno De Francesco
Turin, Italy
nazarenodefrancesco@gmail.com

Abstract

The paper describes GUL.LE.VER, GUiLlottine gLovE resolVER, a Glove based system developed to solve the game “La Ghigliottina” which participated in the Evalita 2020 (Basile et al., 2020) task Ghigliottin-AI. The system described positioned #2, with 0.26 of Precision and 0.46 R@10, more than one guillotine is solved every four games, achieving results comparable to human players. The system proved to solve a different kind of guillotines compared to the first classified system 'Il Mago della ghigliottina' (Sangati et al., 2018). An approach based on these two kinds of systems may result in a boost in this field of research.

1 Introduction

“La Ghigliottina” is a language game in which the gamer has to guess the word that is most correlated with other five words, named clues. An example is the guillotine ”Certificate, Son, Tragedy, Star, Venus”, the solution, in this case, is ”Birth”. The game structure is simple, but some complex steps are required in order to solve a guillotine. The gamer's background knowledge has to be rich enough to cover a large variety of fields, such as common culture, proverbs, etc. Additionally, the gamer's reasoning has to be fast enough to give the solution in less than a minute. In this article, an artificial player for The Guillotine has been built: GUL.LE.VER, the GUiLlotine gLovE resolVER. It's mostly based on the Glove (Pennington et al., 2014) vector representation of the words present in a large collected dataset, containing the Italian Wiktionary, Wikiquote, Wikipedia (only titles), the Italian Collocations Dictionary (Tiberi, 2018), and resources scraped on the web containing Italian polirematics, proverbs and songs titles. The Glove algorithm was chosen for its intrinsic power in capturing the co-occurrence correlation between two words that are not synonyms, due to the co-occurrence matrix that the algorithm builds before the training. Other similar algorithms, such as Word2Vec, do not have this characteristic. The solution for the guillotine is searched in the vector space near the clues, obtaining a list of solution candidates. This list is descending reordered using a hybrid function composed by two parts: one part is based on the Pointwise Mutual Information (Sangati et al., 2018), the other one is based on the weighted sum of the cosine similarity between the solution candidate and the clues, in which the weight is the normalized Inverse Document Frequency of the single clue in the corpus (solutions that are correlated with the rarest clues are more important than others).

2 Related work

In order to find the solution for a particular game, a player needs to know the rules that regulate the game and, based on the game type, he also needs to possess a background knowledge that helps him in finding the solution. We can distinguish two types of games based on these two requirements: closed-world games and open-world games. Closed-world games provide the player with all the knowledge necessary for playing the game (like chess), otherwise open-world games can not be solved without additional knowledge. A particular type of open-world games is represented by language games in which word meanings play a central role (like crosswords) (Littman et al., 2000). The challenge in this type of games is found in the intrinsic ambiguity of natural language, in which a word with different meanings

---

may be connected with a word or with another based on its interpretation, which is heavily dependent on the context. The needs of reasoning skills and a background knowledge to solve this type of games is the main reason for which they have attracted the attention of researchers. In this field a language game like Who Wants to be a Millionaire?, in which the player must have a wide background knowledge in order to answer a series of multiple-choice questions, has been shown to be solved mining the web, with the same performance of a human player (Lam et al., 2003). Extract common sense human knowledge from Wikipedia articles is another proposed solution that is able to challenge a human player (Molino et al., 2013). In the same category of open-world language games is set “La Ghigliottina”, an Italian quiz show in which five words are submitted to the player as clues and he has to find the unique word that is correlated with all the clues. In order to find this hidden associations between clues and solution, a human player must possess a wide background knowledge and he has to be able to perform a complex task of reasoning on it in order of finding correlations between different word meanings in different contexts. In literature, a proposed solution to this problem is OTTHO (On the Tip of my THOught) (Semeraro et al., 2009; Semeraro et al., 2012) which achieved performance similar to humans using a network representation of the background knowledge and a spreading algorithm to find the solution. “Il mago della Ghigliottina” (Sangati et al., 2018), based on a co-occurrence matrix obtained from a corpus of patterns mined on web scraped resources and the Pointwise Mutual Information as measure of word correlation, achieved super-human performance. In order to explore a new way to solve this game, GUL.LE.VER is built using similar web scraped resources, Glove algorithm for word representation and a custom word correlation measure based on cosine similarity and inverse document frequency (idf).

2.1 Linguistic Resources

Based on the previous related works, the linguistic resources involved in this project are:

- The italian Wikipedia, only titles, downloaded via WikiExtractor (Attardi, 2012).
- The italian Wiktionary, downloaded via WikiExtractor.
- The italian Wikiquote, downloaded via WikiExtractor.
- The “Dizionario delle Collocazioni” (Tiberi, 2018) containing 200.000 combinations of words in Italian.
- A collection of 369 italian proverbs (Dige, 2016)
- A collection of more than 3700 common sayings, scraped on different websites .
- A collection of more than 6000 italian polirematics, scraped on different websites. 678 italian song titles (Paldo, 2013).

These corpora was preprocessed, using tokenization (single words only) and punctuation removing, obtaining a unique corpus to feed the Glove algorithm.

3 System description

The system can be described in 6 steps:

1. **Glove training:** the corpus is used to train a Glove model that represents the words in corpus in a compact vector space. The best parameters used to train the algorithm are empirically obtained: Vector_size 600, Vocab_min_count 200, Window 10, Iteration 50, Xmax 10, Alpha 0.75, Eta 0.05. They proved to be the best parameters for the Evalita training dataset.
    (a) The Vocab_min_count setted to 200 corresponds to a vocabulary of 28873 unique words represented.
2. **Setting search space, ‘looking into neighbors’:** starting from the clues, a list of $5 \times M$ solution candidates is built finding the M most similar words to each clue in order of cosine similarity. The result search space is significantly smaller than the entire vocabulary. This solution gives faster and more accurate results than the exhaustive search on the vocabulary.
3. **Filtering candidates:** the solution candidates list is filtered by:
    (a) removing all words except Nouns and Adjective (verbs and conjunctions are never solutions for the game).

    (b) removing Adjectives too, if one of the clues is already an adjective.
    (c) removing words that are present in a custom blacklist and not present in a custom whitelist. The blacklist contains lists of non-ambiguous proper nouns, cities names, foreign words, etc.

4. **Reordering, the cosine based score function:** the filtered list is reordered in descending order based on the following formula:
    (a) $F(t) = \frac{\alpha}{\beta} \times \frac{(\sum_{i=1}^{n}(\cos(s,c_i) \times nIDF(c_i)))\div N}{1+\sigma}$
    (b) The first part of the formula are two arbitrary weights that can be manually set up in order to give more importance to the weighted mean of the cosines or the standard deviation.
    (c) The second part of the formula has: as a numerator, the weighted mean of the cosines between the solution candidate and clues. The weight is the normalized Inverse Document Frequency of the clue in the corpus. This gives a boost to the solutions that are correlated to the most rare clues, starting with the assumption that a rare clue has less possible meaningful combination in the corpus, so a candidate solution highly correlated with that may be corresponding to the solution of the game. As a denominator, there is the standard deviation of the cosines (not weighted). This is intended to give a boost to the solutions that are correlated with all the clues in a balanced way, avoiding such solutions that are very highly correlated to a clue but not to the others.
    (d) A cosine threshold can be set in order to discard cosines that are lower than that, penalizing those that are too low. In this case, the cosines lower than zero are penalized automatically to -1 (the lower bound of the cosine similarity function), avoiding solutions that have opposite meaning compared to the clues.

5. **Solution certainty:** if the difference between the first and the second score result is more than a Solution certainty threshold, the first candidate is proposed as a solution for the game. If not, the candidate list is reordered again using the Pointwise Mutual Information (pmi),calculated on the corpus proposed, as the third multiplied part of the formula $F(t)$. This helps in the situation in which the real solution is between the first three/four results before the final reordering.

6. **Solution proposed:** the first candidate of the reorder list is proposed as a solution for the game.

4 System implementation

The system is entirely implemented in Python 3.7. The principal libraries used are:

- gensim (Řehůřek et al., 2011)
- spacy-stanza (Peng et al., 2020)
- nltk (Loper et al., 2002)
- numba (Lam et al., 2015)
- numpy

The Glove algorithm (Pennington et al., 2014) is the C implementation provided by Stanford and the model obtained is loaded through gensim. A Flask python server was setup to respond to the evaluation requests via API.

5 Results

The table 1 shows the results obtained by GUL.LE.VER on the Evalita-GhigliottinAI Dev dataset and Test dataset.

Set	Size	Precision	R@5	R@10	R@100
Dev	300 pt	0,32	0,44	0,51	0,69
Test	350 pt	**0,27**	0,38	**0,46**	0,62
Dev*	300 pt	0,32	0,37	0,44	0,68
Test*	350 pt	**0,28**	0,40	**0,48**	0,65

Table 1: Results on the TEST and DEV set. Evaluations are the Precision (number of correct solutions / the number of guillotines) and R@5, R@10, R@100 (recall at 5, 10, 100).

The 5% difference in the Precision between the Dev set and the Test set is in part due to a blacklist overfitted on the dev set. 10 solutions are found to be erroneously in the blacklist. Putting them in the whitelist gives a more balanced result, slightly higher for the Test dataset and a little lower for the Dev dataset, as shown by the Dev* and Test* rows. The system seems biased by solutions that are very frequent in corpus: it responded 'uno' 23 times and none of them were the correct solution. Another example: it responded 'senza' 9 times, only one time guessing the correct solution. An important point to underline is that almost half of the solutions are found in the first 10 proposed results, with approximately 40% of them in the first 5, with 57% and 56% in the first 20 for Test and Dev set respectively (not reported in Table 1). This seems very promising for future upgrading, finding a better way to clean the candidates list and/or fine tuning the reorder function.

The last point of analysis is a brief comparison between GUL.LE.VER and 'Il Mago della Ghigliottina'. Selecting only the resolved guillotines from the Test Set and submitting them to the Telegram version of 'Il Mago della Ghigliottina', 18 guillotines were not resolved by Sangati et al., 2018 system. These guillotines (in table 2) represent 4.8% of the total test guillotines and can be resolved only by the proposed solution.

Clue1	Clue2	Clue3	Clue4	Clue5	Gullever	Mago
fazzoletto	alto	allungare	braccio	osso	collo	naso
studio	vestire	notte	povero	montalbano	giovane	panni
paradiso	bordo	sud	nino	casa	benvenuti	angolo
vecchio	cavallo	end	soda	conquista	west	polenta
mettere	moto	collo	baffi	brutta	piega	giro
mamma	scena	scuola	re	crudo	nudo	gonna
volo	dare	mezzi	ente	intervento	assistenza	pronto
idee	bocca	isola	sottomarino	spock	vulcano	porto
finestra	vestire	volto	chiara	chiaro	scuro	luna
pari	pace	sosta	motivo	famiglia	senza	apparente
cura	pietre	alto	azzurro	occhi	sole	cielo
acqua	onda	capo	sempre	essere	verde	andata
città	tv	vita	oggi	gioco	ragazzi	frenetico
bandiera	coltelli	caponi	marx	italia	fratelli	regno
dare	camera	consiglio	misura	stato	sicurezza	deciso
regola	parole	alberi	perfetto	fa	tre	quadrato
leggero	barba	togliere	viso	inganno	trucco	velo

Table 2: Guillotine resolved by GUL.LE.VER and not resolved by Il Mago della Ghigliottina.

6 Conclusion and future work

In this paper is described GUL.LE.VER, an artificial player to solve the game "La Ghigliottina", based on the Glove word vector algorithm, whose power is its co-occurrence matrix reduction. An hybrid pmi approach is proposed as fallback in case of uncertainty. The system achieved good performance in the Evalita2020 task, with results comparable to humans. A comparison made with the solutions proposed by the best system, the Sangati et al., 2018 'Il Mago della Ghigliottina', suggests that the proposed approach is capable of solving different kinds of guillotines compared to the first one. As future work, a even more hybrid solution between these two kinds of approaches should be implemented, hoping it will be result in a boost in this field of research.

References

Paola Tiberii, *Dizionario delle collocazioni: le combinazioni delle parole in italiano*, Zanichelli, 2018.

M. L. Littman, Review: *Computer language games*, in Proc. Comput. Games, 2nd Int. Conf., Rev. Papers, T. A. Marsland and I. Frank, Eds., 2000, vol. 2063, pp. 396404, ser. LNCS, Springer.

S. K. Lam, D. M. Pennock, D. Cosley, and S. Lawrence, *Mining the web to play Who wants to be a millionaire?*, in Proc. 19th Conf. Uncert. Artif. Intell., C. Meek and U. Kjrulff, Eds.2003, pp. 337345.

P. Molino, P. Basile, C. Santoro, P. Lops, M. de Gemmis, and G. Semeraro, M.Baldoni, C. Baroglio, G. Boella, and R. Micalizio, Eds., *A virtual player for who wants to be a millionaire? based on question answering*, in Proc. AI*IA 2013: Adv. Artif. Intell.13th Int. Conf. Italian Assoc. Artif. Intell., 2013, vol. 8249, pp. 205216, ser. Lecture Notes in Comput. Sci..

G. Semeraro, M. de Gemmis, P. Lops, and P. Basile, *An artificial player for a language game*, IEEE Intell. Syst., vol. 27, no. 5, pp.3643, Sep.Oct. 2012.

G. Semeraro, P. Lops, P. Basile, and M. de Gemmis, *On the tip of my thought: Playing the Guillotine game*, in Proc. 21st Int. Joint Conf. Artif. Intell., 2009, pp. 15431548, Morgan Kaufmann.

Basile, Pierpaolo and de Gemmis, Marco and Lops, Pasquale and Semeraro, Giovanni *Solving a complex language game by using knowledge-based word associations discovery*. IEEE Transactions on Computational Intelligence and AI in Games 8.1 (2014): 13-26.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Sangati Federico, Antonio Pascucci, and Johanna Monti. *Exploiting Multiword Expressions to solve "La Ghigliottina"*. Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018). Vol. 2263. Accademia University Press, 2018.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. *Glove: Global vectors for word representation*. Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.

Attardi, Giuseppe.*WikiExtractor.* (2012).

Antonio Dige. 2016. *Raccolta di proverbi e detti italiani.* http://web.tiscali.it/proverbiitaliani.

Paldo, Alessandro. 2013. http://alessandro-paldo.blogspot.com/2013/10/1-10-1.html

Lam, Siu Kwan, Antoine Pitrou, and Stanley Seibert. *Numba: A llvm-based python jit compiler.* Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC. 2015.

Řehůřek, Radim, and Petr Sojka. *Gensim—statistical semantics in python.* Retrieved from genism. org (2011).

Loper, Edward, and Steven Bird. *NLTK: the natural language toolkit.* arXiv preprint cs/0205028 (2002).

Qi, Peng and Zhang, Yuhao and Zhang, Yuhui and Bolton, Jason and Manning, Christopher D *Stanza: A python natural language processing toolkit for many human languages.* arXiv preprint arXiv:2003.07082 (2020).

PRELEARN: Prerequisite Relation Learning

PRELEARN @ EVALITA 2020: Overview of the Prerequisite Relation Learning Task for Italian

Chiara Alzetta•◇**, Alessio Miaschi**⋆◇**, Felice Dell'Orletta**◇**,**
Frosina Koceva•**, Ilaria Torre**•
•DIBRIS, Università degli Studi di Genova, ⋆Dipartimento di Informatica, Università di Pisa,
◇CNR, Istituto di Linguistica Computazionale "Antonio Zampolli", Pisa - ItaliaNLP Lab
{`chiara.alzetta,frosina.koceva`}`@edu.unige.it`, `alessio.miaschi@phd.unipi.it`,
`ilaria.torreunige.it`, `felice.dellorletta@ilc.cnr.it`

Abstract

The Prerequisite Relation Learning (PRELEARN) task is the EVALITA 2020 shared task on concept prerequisite learning, which consists of classifying prerequisite relations between pairs of concepts distinguishing between *prerequisite* pairs and *non-prerequisite* pairs. Four sub-tasks were defined: two of them define different types of features that participants are allowed to use when training their model, while the other two define the classification scenarios where the proposed models would be tested. In total, 14 runs were submitted by 3 teams comprising 9 total individual participants.

1 Introduction

The present paper provides an overview of the systems participating to PRELEARN, the first shared task on automatic prerequisite learning between educational concepts.

In the past decades we have witnessed a great revolution in the field of Education: advancement of technologies drastically transformed the teaching method and the setting of the learning process thanks to the raise of e-learning platforms and electronic educational materials. While so far they've been mainly used in lifelong learning, the current pandemic situation made very clear that distant learning is a valuable resource at all educational levels. This new era in education is commonly referred to as Education 4.0 (Saxena et al., 2017; Hussin, 2018; Salmon, 2019) and its main novelty is to put students at the core of every learning activity promoting the mission of fostering and improving personalisation techniques. While there is still much work to do to develop usable and scalable personalisation systems, much of the attention has been devoted to building and testing the building blocks of such applications.

The massive use of distance learning platforms has shed light on the need of developing intelligent agents able to support both students and teachers by, e.g., automatically identifying educational relations between learning concepts. Educational resources are designed to guide students through learning paths consisting of concepts related to each other. Among all pedagogical relations, prerequisite is the most fundamental since it establishes which sequence of concepts allows students to have a full understanding of the domain. In fact, the order in which concepts are presented to the learner plays a crucial role in avoiding student's frustration and misunderstandings while approaching a new topic, so teachers are very careful to organise the content of their learning materials accordingly and to highlight relevant connections to their students. Doing this automatically is still challenging from many perspectives.

The NLP community has tackled automatic prerequisite learning in the past with the goal of integrating prerequisite relations in systems for, e.g., curriculum planning (Agrawal et al., 2016), reading list generation (Gordon et al., 2017; Fabbri et al., 2018), automatic assessment (Wang and Liu, 2016) and automatic educational content creation (Lu et al., 2019). Wikipedia is rightfully considered a rich and freely available resource for training and testing educational applications, and this is also true in the case of prerequisite learning systems, which are often evaluated against manually annotated prerequisite relations between Wikipedia pages (Talukdar and Cohen, 2012; Gasparetti et al., 2018; Zhou and Xiao, 2019).

Based on the works available in the literature, we distinguish prerequisite learning systems in two main categories: 1) those based on re-

---

lational metrics and 2) those on machine learning approaches. Relational metrics are designed to capture the strength of the relation between co-occurring concepts and identify pairs of concepts obtaining low values as non-prerequisites. The *RefD* metric (Liang et al., 2015) is possibly the most popular and measures how differently two concepts refer to each other considering the Wikipedia links of the pages associated with the concepts of the pair. Prerequisite concept learning from textbook concepts is addressed in Adorni et al. (2019), which presents a method based on burst analysis combined with temporal reasoning to identify possible propaedeutic relations and compare it with a concept co-occurrence metric. Among machine learning approaches, we distinguish between those that exploited link-based features (e.g. (Liang et al., 2015; Gasparetti et al., 2018)), text-based features only (e.g. (Miaschi et al., 2019; Alzetta et al., 2019)), or a combination of the two (Liang et al., 2018).

Unfortunately, the results obtained by those systems are not directly comparable: their approaches are based on different assumptions of what a concept is and which are the distinctive features for a prerequisite relation. Moreover, knowledge structures defined by domain experts are not always easily available or are missing for some domains. With PRELEARN, we are proposing the first shared task on automatic prerequisite learning, at least to the best of our knowledge. Located in the context of EVALITA 2020 evaluation campaign (Basile et al., 2020), the task challenges participants to develop prerequisite learning systems that can exploit either only information derived from textual educational resources or that can combine those information with structural properties of knowledge structure. We aim to compare the performances of systems based on these two different approaches and verify if they can obtain similar results or, conversely, one strategy is far better performing than the other. The goal of PRELEARN shared task is not only to offer a setting where different approaches and systems can be directly compared, but also to gather the research teams working on automatic prerequisite learning, which is distributed and doesn't have dedicated venues, and possibly fostering collaborations within the community. More broadly, we expect the outcomes of the task to be relevant to the wider information extraction and knowledge structure construction communities, as it offers the opportunity to test which information – either textual or extracted from a knowledge structure – are more effective for retrieving pedagogical relations in educational data.

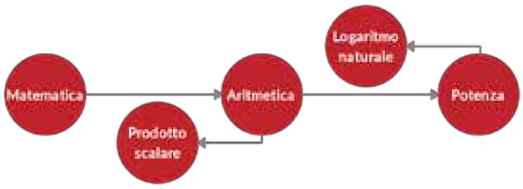


Figure 1: Example of prerequisite relations between concepts.

2 Task Description

PRELEARN (Prerequisite Relation Learning) is a shared task on concept prerequisite learning which consists of classifying prerequisite relations between pairs of concepts. This is the first time, to the best or our knowledge, that *automatic prerequisite learning* is addressed in a shared task. PRELEARN challenges participants to test their models for automatic prerequisite learning on four different domains and four training scenarios.

2.1 Problem Formulation

For the purposes of this task, prerequisite relations learning is proposed as a binary classification problem of concept pairs: given a pair of concepts (A, B), we ask to predict whether or not concept B is a prerequisite of concept A. We define a "*concept*" as single or multi word domain terms corresponding to the title of a page on the Italian Wikipedia: *Prodotto scalare* and *Aritmetica* are both concepts of the precalculus domain and are also the titles of two Italian Wikipedia pages. Prerequisite relations instead are dependency relations that naturally occur between educational concepts determining their learning precedence.

Consider the knowledge structure proposed as an example in Figure 1. Here, nodes represent concepts while links identify the prerequisite relations that connect them. According to the graph, "Aritmetica" is a prerequisite of "Potenza" since, if a student wants to understand what "Potenza" is, he/she has to know "Aritmetica" first. Hence, we formally define a *prerequisite relation* as a relation connecting a target and a prerequisite concept if the second has to be known in order to understand the first. In other words, the Wikipedia page of the prerequisite concept contains the prior knowledge required to understand the content of the Wikipedia page of the target concept.

2.2 Task Settings

We defined four sub-tasks for addressing automatic concepts prerequisite learning: two of them concern the model used by participants for tackling the task, the other two distinguish different classification scenarios where the proposed model can be tested. In order to make a valid submission, we asked participants to submit at least one model complying with at least one of these settings:
i) *Raw features setting* (RF): a model that acquires information only from raw text (e.g. textual content of the Wikipedia pages offered as training set, corpora for acquiring distributional representations, etc.);
ii) *Raw and structured features setting* (RnS): a model that can rely both on raw text and structured information (e.g. Wikipedia graph structure of a domain and metadata of a Wikipedia page, DBpedia, page hierarchical structure in terms of sections and paragraphs, etc.).

Each submitted model was tested in two evaluation scenarios, defined as follows:
i) *In-domain scenario*: the model(s) can be trained on data belonging to any domain, including the one appearing in the test set;
ii) *Cross-domain scenario*: the model(s) can be trained on data belonging to any domain but the domain of the test set.

Overall, we defined a total of four sub-tasks:
1) RF setting in an in–domain scenario;
2) RF setting in a cross–domain scenario;
3) RnS setting in an in–domain scenario;
4) RnS setting in an cross–domain scenario.

Only few work in the literature test their systems in a cross-domain scenario: our previous attempts in this direction (Miaschi et al., 2019) highlighted some issues in transferring the information acquired from one domain to an unknown one. At the same time, although the two proposed settings correspond to the most widely used approaches for automatic prerequisite learning, systems only rarely rely on textual information only, and when they do performances are generally worse than those obtained by exploiting structural information extracted from knowledge bases. This makes, in our view, the RF setting tested in the cross-domain scenario the most challenging sub-task.

2.3 Evaluation

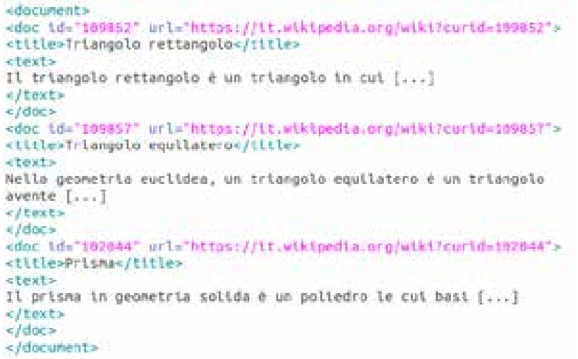
Metrics. Evaluation of participants' systems outputs was carried out on four balanced datasets, one for each domain, used for both in– and cross–domain evaluation. The size of the test sets is reported in Table 1. Each sub-task (i.e. each model on each scenario) was evaluated independently from the others by using standard metrics, such as Accuracy (A), Precision (P), Recall (R) and F$_1$-score (F_1). Since the test sets are balanced, we used Accuracy metric to rank participants' submitted runs.

Baseline. We used for all settings a linear SVM classifier trained using two binary features capturing the presence of a mention of concept *B*/*A* in the text of the Wikipedia page of concept *A*/*B*. Each feature returns 1 if the name of concept *B*/*A* is mentioned in the text of the Wikipedia page of concept *A*/*B*, while it returns 0 otherwise.

3 Data

We relied on ITA-PREREQ dataset (Miaschi et al., 2019), a dataset annotated with prerequisite relations between pairs of concepts in Italian. The dataset was built upon the AL-CPL dataset (Liang et al., 2018), a collection of binary-labelled concept pairs extracted from textbooks on four domains: data mining, geometry, physics and precalculus. In AL-CPL, for each domain, the authors extracted the relevant terms from the textbook: those appearing in the title of a English Wikipedia page were promoted as domain concepts and matched with their corresponding page. Finally, domain experts were asked to manually annotate the presence of absence o a prerequisite relation between all concept pairs. The final dataset consists of both positive and negative concept pairs that can be represented as a concept map, a specific type of knowledge graph where each node is a scientific concept and edges represent pedagogical relations.

The construction of ITA-PREREQ was carried out as follows, as described in (Miaschi et al., 2019). First, we took the Italian version of the Wikipedia pages considered for AL-CPL, excluding from the dataset those concepts (and the relations where they are involved) for which an Italian page was not available. Then, we mapped both positive and negative relations between pairs



```
<document>
<doc id="109852" url="https://it.wikipedia.org/wiki?curid=109852">
<title>Triangolo rettangolo</title>
<text>
Il triangolo rettangolo è un triangolo in cui [...]
</text>
</doc>
<doc id="109857" url="https://it.wikipedia.org/wiki?curid=109857">
<title>Triangolo equilatero</title>
<text>
Nella geometria euclidea, un triangolo equilatero è un triangolo
avente [...]
</text>
</doc>
<doc id="102044" url="https://it.wikipedia.org/wiki?curid=102044">
<title>Prisma</title>
<text>
Il prisma in geometria solida è un poliedro le cui basi [...]
</text>
</doc>
</document>
```

Figure 2: Example of Wikipedia pages (with cut off texts) from the “Wikipedia pages file”.

of the remaining concepts from AL-CPL to ITA-PREREQ. As in AL-CPL, ITA-PREREQ dataset was expanded by creating irreflexive relations (add (B, A) as a negative sample if (A, B) is a positive sample) and transitive pairs (add (A, C) if both (A, B) and (B, C) are positive sample). In summary, ITA-PREREQ consists of pairs of concepts (A, B), labelled as follows: 1 if B is a prerequisite of A and 0 in all other cases. It was not allowed to use any sort of prerequisite-labelled data apart from ITA-PREREQ dataset provided by task organisers as official training set.

3.1 Format

PRELEARN participants were provided, upon request, with five files: a “concept pairs file” for each of the four domains containing the labelled concept pairs and one “Wikipedia pages file” containing the raw text and the link of the Wikipedia pages referring to the concepts appearing in the dataset. Here’s an example of the pairs contained in the “concept pairs file”:

```
Riflessione interna totale,Luce,1
Plasticita’ (fisica),Durezza,0
...
Campo magnetico,Magnete,1
```

Figure 2 on the other hand shows an excerpt of the content of the “Wikipedia pages file”. The content of the Italian Wikipedia pages was extracted using WikiExtractor[1] on a Wikipedia dump from January 2020.

3.2 Train and Test Sets

Table 1 provides a summary of the content of ITA-PREREQ, both for each domain covered by the dataset and overall. The number of concepts and pairs varies for each domain: while Geometry and Data Mining have a comparable amount of concepts, the latter shows a significantly smaller number of labelled pairs. It is interesting to note that, although not being the richer domain in terms of concepts, Physics shows the higher number of relations. As can be noted, regardless of the domain the dataset is strongly unbalanced since the majority of concept pairs do not show a prerequisite relation (*Non-PR Pairs*). For each domain we split the pairs into a portion of training and a portion of test data. For the test portion, we defined a fixed number of pairs to include (i.e. 200 pairs), with the exception of Data Mining where, given the limited number of total pairs, we included only 99 pairs. The distribution of prerequisite and non-prerequisite labels was balanced (50/50) for each domain only in the test datasets.

4 Participants

Following a call for interest, 16 teams registered for the task and thus obtained the training data. Eventually, three teams submitted their predictions, for a total of 14 runs, each executed on all four domains of the dataset. Two teams participated in all four sub-tasks while one team submitted results only for the two sub-tasks involving the RF setting. A summary of participants is provided in Table 2.

4.1 Submitted Systems

NLP-CIC (Angel et al., 2020) presented three different systems trained on both hand-crafted and embedding-based features. In particular, the team developed one model for the RF setting and two models for the RnS setting. Concerning the RF setting, the submitted model corresponds to a single layer Neural Network trained using concept pairs representations extracted from a BERT Italian model[2] fine-tuned on the training datasets. With respect to the RnS setting, the two submitted models are quite similar and differ only for one feature. The first model (Complex) is based on a tree-ensemble learner and trained it using a set of complexity-based features based on those defined by Aroyehun et al. (2018) combined with a feature capturing concept view frequency, i.e. the daily average of unique visits to the concept page by Wikipedia users (including editors, anonymous

---

[1] https://github.com/attardi/wikiextractor

[2] https://huggingface.co/dbmdz/bert-base-italian-cased

Domain	Concepts	Pairs	PR Pairs	non-PR Pairs	Pairs in Train set	Pairs in Test set
Data Mining	76	523	159 (30.40%)	364 (69.59%)	424	99
Geometry	74	1,748	432 (24.71%)	1,316 (75.28%)	1,548	200
Physics	130	2,420	415 (17.14%)	2,005 (82.85%)	2,220	200
Precalculus	177	1,916	508 (26.51%)	1,408 (73.48%)	1,716	200
Total	457	6,607	1,514 (22.91%)	5,093 (77.08%)	5,908	699

Table 1: Number of concepts, pairs, pairs showing a prerequisite [PR Pairs] (absolute and relative) or non-prerequisite relation [non-PR Pairs] (absolute and relative) for each domain of the ITA-PREREQ dataset. We also report the number of pairs (either prerequisite or not) released in the official training and test sets.

Team	Research Group	# Tasks	# Runs
NLP-CIC	Instituto Politécnico Nacional	4	6
B4DS	Università di Pisa	2	4
UNIGE_SE	Università degli Studi di Genova	4	4

Table 2: Teams participating in EVALITA 2020 PRELEARN shared task with number of sub-tasks they particpated in and number of submitted runs.

editors and readers) over the last year. The second model (Complex+wd) is an improved version of the first one: it takes as input the same set of features along with the Wiki-data embedding of each concept appearing in the concept pairs of ITA-PREREQ dataset.

B4DS (Puccetti et al., 2020) presented two different classification models, one based on XGBoost (Chen and Guestrin, 2016) classifier and one based on a Gated Recurrent Unit (GRU) model. The first classifier, Model 1, was trained using a combination of lexical and hand-crafted features. Specifically, lexical features were computed by averaging 300-dimensions pretrained word2vec embeddings (Berardi et al., 2015) of title A and B respectively, with A and B being the two concepts involved in a pair. The set of 14 hand-crafted text-based features, inspired by Miaschi et al. (2019), are extracted for each pair of the datasets and aim at capturing mentions and lexical similarity between the two pages associated with the concepts in the pair. The second classifier (Model 2) was trained with a GRU model (hidden size=8, encoding size=32, learning rate=0.01) that takes as input the first 400 words of each Wikipedia page of the (A, B) pair. The output was computed with a linear layer that takes the concatenation of the two learned vectors.

UNIGE_SE (Moggio and Parizzi, 2020) proposed a classifier based on a two-dense-layers Neural Network trained using a set of features automatically extracted from the Wikipedia pages associated with the concepts appearing in ITA-PREREQ dataset. In particular, the RF model was trained exploiting features that capture concepts co-occurrence and the lexical similarity between the pages referring to the concepts of a pair. On the other hand, the RnS model is trained combining the previous set of features with information based on the hyperlink and category structure of Wikipedia.

5 Results

In this section we provide both a discussion of the approaches and an analysis of the results reported in Tables 3 and 4.

Participants experimented with more classical machine learning algorithm as well as with Neural Networks (NN): we received results computed exploiting 7 different systems, 4 trained using only raw text features (RF setting) and 3 exploiting also structural information (RnS setting). Considering their average performances across all four domains, all systems outperformed the baseline. In this Section, we describe the results obtained by the submitted models and compare their performances on the official test set based on their average accuracy scores over the four domains (column *AVG* in the Tables).

5.1 Comparing Scenarios

In–Domain Scenario. As shown in Table 3, overall the model showing the best performances is Italian BERT, achieving an average accuracy score of 0.887 in the RF setting. Such result is not surprising if we consider the state-of-the-art performances obtained by recent Neural Language Models in the resolution of downstream NLP tasks. However, results obtained by BERT show only a small gap with respect to some of the other models. For instance, B4DS' Model

RF Setting							
Place	**Team**	**Model**	**Data mining**	**Geometry**	**Physics**	**Precalculus**	**AVG**
1	NLP-CIC	BERT	0.838	0.925	0.855	0.930	0.887
2	B4DS	Model 1	0.797	0.920	0.815	0.930	0.866
3	B4DS	Model 2	0.808	0.905	0.810	0.890	0.853
4	UNIGE_SE	NeuralNet	0.595	0.620	0.530	0.675	0.605
5	Baseline	Occurrence	0.494	0.675	0.500	0.675	0.586
RnS Setting							
Place	**Team**	**Model**	**Data mining**	**Geometry**	**Physics**	**Precalculus**	**AVG**
1	NLP-CIC	Complex+wd	0.808	0.905	0.795	0.915	0.856
2	NLP-CIC	Complex	0.828	0.895	0.785	0.885	0.848
3	UNIGE_SE	NeuralNet	0.565	0.755	0.725	0.755	0.700
4	Baseline	Occurrence	0.494	0.675	0.500	0.675	0.586

Table 3: Results in terms of Accuracy of the EVALITA 2020 PRELEARN RF and RnS models in the in–domain evaluation setting for each domain and on average.

RF Setting							
Place	**Team**	**Model**	**Data mining**	**Geometry**	**Physics**	**Precalculus**	**AVG**
1	NLP-CIC	BERT	0.565	0.785	0.635	0.775	0.690
2	B4DS	Model 1	0.505	0.720	0.600	0.765	0.648
3	B4DS	Model 2	0.484	0.710	0.605	0.785	0.646
4	UNIGE_SE	NeuralNet	0.565	0.515	0.465	0.595	0.535
5	Baseline	Occurrence	0.494	0.500	0.605	0.500	0.525
RnS Setting							
Place	**Team**	**Model**	**Data mining**	**Geometry**	**Physics**	**Precalculus**	**AVG**
1	NLP-CIC	Complex+wd	0.535	0.775	0.600	0.760	0.668
2	NLP-CIC	Complex	0.494	0.735	0.595	0.730	0.639
3	UNIGE_SE	NeuralNet	0.545	0.665	0.560	0.710	0.620
4	Baseline	Occurrence	0.494	0.500	0.605	0.500	0.525

Table 4: Results in terms of Accuracy of the EVALITA 2020 PRELEARN RF and RnS models in the cross–domain evaluation setting for each domain and on average.

1, exploiting a decision tree based on XGBoost framework and trained using both word embedding and handcrafted features, achieved 0.866 accuracy thus gaining the second place in the in–domain scenario. Similar competitive results are obtained by the Complex+wd model submitted by NLP-CIC team: this model combines Wiki-data embedding of each concept with a set of manually defined features that measure concept complexity and were designed to solve the task of complex word identification (Aroyehun et al., 2018). B4DS team submitted also a more sophisticated model (i.e. a GRU-based classifier) trained using only Word2vec embeddings with no other handcrafted features. Considering the results, combining lexical features, like word embeddings, with handcrafted features allows to achieve better performances regardless of the model employed for classification, while using these two types of features independently seems a worse strategy. As proof, B4DS' Model 2, despite being more sophisticated, achieved lower scores than Model 1. The fact that these models obtained similar results suggests that automatic prerequisite learning is more affected by predictors rather than the model used for classification.

Among submitted systems, only three didn't exploit word embeddings: NLP-CIC team submitted a tree-ensemble learner trained using only complexity features, and UNIGE_SE team used two versions of a two-layer NN trained with different sets of handcrafted features to comply with settings requirements. The results obtained by these models provide some interesting insights on the role of raw and structural features for solving the task. First, we observe that exploiting raw textual features based on lexical similarity and topic modelling (UNIGE_SE NN in the RF setting) only slightly outperforms the baseline, thus, when no lexical features are available, it seems more useful to rely on structural information. Anyways, complexity-based features exploited by NLP-CIC are more informative for prerequisite learning task than Wikipedia category and link structure. The intuition behind the NLP-CIC team approach is that less complex concepts are prerequisite for the more complex ones and, considering that the results are only slightly below those obtained using word embeddings, the intuition that complexity is involved in the process of defining prerequisite sequences seems confirmed.

Cross–Domain Scenario. Moving to the cross–domain evaluation scenario (see Table 4), we observe only small variations in the ranking of the submitted systems. In spite of this, we also observe a consistent drop of the accuracies obtained by the submitted systems.

Considering again the average accuracy scores, BERT model proved to be the best performing model also in this scenario. Interestingly, this time NLP_CIC's Complex+wd model outperforms B4DS's Model 1: both models are trained using both word embeddings and handcrafted features, with the latter being more useful possibly because capturing domain independent properties. The different performances of the two systems could be again due to the higher effectiveness of complexity-based features for identifying prerequisite relations. Consequently, these results suggest that, unlike the in-domain scenario, lexical information are not enough to identify prerequisite relations. Nevertheless, lexical features proved somehow useful since using handcrafted features only, as in the case of Complex NLP-CIC model and the NN models submitted by UNIGE_SE team, is outperformed by B4DS's Model 2 (based solely on word embeddings).

5.2 Domains Impact

Focusing on the differences between the four domains, we observe that for almost all submitted systems the results obtained on concept pairs belonging to the Data Mining domain are lower than the others. This is especially true for the cross–domain scenario and seems to corroborate what was already stated in Miaschi et al. (2019), namely that Data Mining is a relatively new and more specialised topic that presents shorter pages and, therefore, that contains less clear prerequisite relationships. Nevertheless, the model submitted by the UNIGE_SE team for the RF setting achieved the lowest results when tested on concept pairs belonging to the Physics domain.

With the exception of the UNIGE_SE's RF model in the cross–domain setting, all systems achieved best (and similar) results when classifying Geometry and Precalculus concepts pairs. This might be due to the fact that these two domains are more fundamental and broad subjects and, therefore, present more clear learning dependencies expressed through Wikipedia. Furthermore, since Geometry and Precalculus share more lexicon that the others, we believe that the models can take advantage of this overlap to better classify concept pairs, especially for the cross–domain evaluation setting.

6 Conclusion

Automatic prerequisite learning was for the first time the focus of a dedicated shared task. In particular, PRELEARN task was aimed at comparing the performances of different approaches and models tested within and across the four domains of ITA-PREREQ dataset. Although the results of 14 submitted runs were all above baseline, we observe several differences within the proposed settings and across domains. In particular, results suggests that automatic prerequisite learning is more affected by the predictors rather than by the classification models. Results also confirm that the RF cross–domain setting is the most challenging scenario. Nevertheless, BERT achieved best scores in both RF settings, also outperforming models trained with structural features extracted from the knowledge structure of Wikipedia.

For the future, it would be interesting to test the impact of hand-crafted features combined with a contextual language model such BERT and, considering the effectiveness of complexity–based features, explore the contribution of predictors encoding text readability properties in prerequisite learning systems.

References

Giovanni Adorni, Chiara Alzetta, Frosina Koceva, Samuele Passalacqua, and Ilaria Torre. 2019. Towards the identification of propaedeutic relations in textbooks. In *International Conference on Artificial Intelligence in Education (AIED)*. Springer.

Rakesh Agrawal, Behzad Golshan, and Evangelos Papalexakis. 2016. Toward data-driven design of educational courses: A feasibility study. *Journal of Educational Data Mining*, 8(1):1–21.

Chiara Alzetta, Alessio Miaschi, Giovanni Adorni, Felice Dell'Orletta, Frosina Koceva, Samuele Passalacqua, and Ilaria Torre. 2019. Prerequisite or not prerequisite? that's the problem! an nlp-based approach for concept prerequisites learning. In *6th Italian Conference on Computational Linguistics, CLiC-it 2019*, volume 2481. CEUR-WS.

Jason Angel, Segun Taofeek Aroyehun, and Alexander Gelbukh. 2020. Nlp-cic @ prelearn: Mastering prerequisites relations, from handcrafted features to embeddings. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Segun Taofeek Aroyehun, Jason Angel, Daniel Alejandro Pérez Alvarez, and Alexander Gelbukh. 2018. Complex word identification: Convolutional neural network vs. feature engineering. In *Proceedings of the thirteenth workshop on innovative use of NLP for building educational applications*, pages 322–327.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Giacomo Berardi, Andrea Esuli, and Diego Marcheggiani. 2015. Word embeddings go to italy: A comparison of models and training datasets. In *IIR*.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Alexander Fabbri, Irene Li, Prawat Trairatvorakul, Yijiao He, Weitai Ting, Robert Tung, Caitlin Westerfield, and Dragomir Radev. 2018. TutorialBank: A manually-collected corpus for prerequisite chains, survey extraction and resource recommendation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 611–620, Melbourne, Australia, July. Association for Computational Linguistics.

Fabio Gasparetti, Carlo De Medio, Carla Limongelli, Filippo Sciarrone, and Marco Temperini. 2018. Prerequisites between learning objects: Automatic extraction based on a machine learning approach. *Telematics and Informatics*, 35(3):595–610.

Jonathan Gordon, Stephen Aguilar, Emily Sheng, and Gully Burns. 2017. Structured generation of technical reading lists. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 261–270.

Anealka Aziz Hussin. 2018. Education 4.0 made simple: Ideas for teaching. *International Journal of Education and Literacy Studies*, 6(3):92–98.

Chen Liang, Zhaohui Wu, Wenyi Huang, and C Lee Giles. 2015. Measuring prerequisite relations among concepts. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1668–1674.

Chen Liang, Jianbo Ye, Shuting Wang, Bart Pursel, and C Lee Giles. 2018. Investigating active learning for concept prerequisite learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Weiming Lu, Pengkun Ma, Jiale Yu, Yangfan Zhou, and Baogang Wei. 2019. Metro maps for efficient knowledge learning by summarizing massive electronic textbooks. *International Journal on Document Analysis and Recognition (IJDAR)*, pages 1–13.

Alessio Miaschi, Chiara Alzetta, Franco Alberto Cardillo, and Felice Dell'Orletta. 2019. Linguistically-driven strategy for concept prerequisites learning on italian. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 285–295.

Alessio Moggio and Andrea Parizzi. 2020. Unige_se @ prelearn: Utility for automatic prerequisite learning from italian wikipedia. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Giovanni Puccetti, Luis Bolanos, Filippo Chiarello, and Gualtiero Fantoni. 2020. B4ds @ prelearn: Ensemble method for prerequisite learning. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Gilly Salmon. 2019. May the fourth be with you: Creating education 4.0. *Journal of Learning for Development-JL4D*, 6(2).

Rajan Saxena, Vinod Bhat, and A Jhingan. 2017. Leapfrogging to education 4.0: Student at the core.

Partha Pratim Talukdar and William W Cohen. 2012. Crowdsourced comprehension: predicting prerequisite structure in wikipedia. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 307–315. Association for Computational Linguistics.

Shuting Wang and Lei Liu. 2016. Prerequisite concept maps extraction for automatic assessment. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 519–521. International World Wide Web Conferences Steering Committee.

Yang Zhou and Kui Xiao. 2019. Extracting prerequisite relations among concepts in wikipedia. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

B4DS @ PRELEARN: Ensemble Method for Prerequisite Learning

Giovanni Puccetti
Scuola Normale Superiore
giovanni.puccetti@sns.it

Luis Bolanos
Texty S.r.l.
luis.bolanos@texty.biz

Filippo Chiarello
Universitá di Pisa
filippo.chiarello@unipi.it

Gualtiero Fantoni
Universitá di Pisa
g.fantoni@ing.unipi.it

Abstract

English. In this paper we describe the methodologies we proposed to tackle the EVALITA 2020 shared task PRELEARN. We propose both a methodology based on gated recurrent units as well as one using more classical word embeddings together with ensemble methods. Our goal in choosing these approaches, is twofold, on one side we wish to see how much of the prerequisite information is present within the pages themselves. On the other we would like to compare how much using the information from the rest of Wikipedia can help in identifying this type of relation. This second approach is particularly useful in terms of extension to new entities close to the one in the corpus provided for the task but not actually present in it. With this methodologies we reached second position in the challenge[1].

1 Introduction

The PRELEARN task consists in classifying pairs of concepts according to whether one is a prerequisite for the other or not. The concepts are presented as Wikipedia pages and they are divided into four different domains, physics, precalculus, data mining and geometry.

The task was organized in 4 subtasks: i) two of them concerned with the type of information that can be exploited by the submitted models, either solely textual or including metadata, e.g. Wikipedia hyperlinks; ii) the other two based on different classification scenarios, training and testing could happen either on the same domain or three domain could be used as training set and the fourth as testing. A more extensive description of the task together with all the results and more information is found in the report (Alzetta et al., 2020) which is part of the EVALITA 2020 (Basile et al., 2020). The concept of being a prerequisite is highly complex and can be misunderstood from humans as well. Indeed, this relation can be subtle and depending on the domain it may take a deep level of expertise to recognize. One of the reasons this challenge is very interesting, is the fact that several application can arise from this same setting. Regarding this, we point out how it could be interesting to apply the systems we develop for this task to evaluate teaching modules. Indeed, one could design a quality assessment for courses based on the level of agreement between subsequent chapters and sections and their prerequisite relations. A different application, could be the definition of a new way to move around Wikipedia itself, identifying which links move in the same direction as the prerequisite relation and which on the contrary move against it.

Let us now outline three main aspects common to different works tackling similar tasks. We will take into into account these specifics while developing our own models. The first is that hand crafted features are commonly used, in (Miaschi et al., 2019) they develop these features mostly analysing textual statistics, for example the occurrence of one concept in the page of another one. In (Liang et al., 2015) they also develop top down features, however the information they structure does not come from the body of the pages, instead they use the structure of Wikipedia as a graph with hyperlinks. Following this line, the second aspect is the use of graph structures. In most of the works predicting prerequisites, we see how they interpret pages as nodes and hyperlinks as edges. Both in (Talukdar and Cohen, 2012) and in (Liang et al., 2015) they use this feature, in some cases joining

---

it with textual information, whereas in others as a stand alone one. On the contrary, in (Adorni et al., 2019) they use a bottom up graph structures created to help in the prediction. The third and last is the use of neural networks, as done in (Miaschi et al., 2019), where they are employed to create representations of text that can afterward be fed as features to simpler classifiers. We remark how structuring information into a graph is a practice used also in other tasks involving several documents. One example is topic modeling (Gerlach et al., 2018), it is interesting to notice how this task shares some of the steps needed for prerequisite learning. Indeed, in both cases one needs to crate a hierarchy of concepts which is then exploited in different ways. Since we wish to exploit textual knowledge, we can also employ word embeddings. For the Italian language they are developed in (Berardi et al., 2015). On top of them we will use ensemble methodologies since they can proficiently exploit information in these representations. Notice how in principle more modern techniques, such as transformer models (Devlin et al., 2019) could be used to help performance in this task, however as we will see we preferred not to do so. The main reason supporting this choice is the fact that the dataset provided for this task is not too big and thus we avoided too large models. The systems we developed try to enclose all these pieces of information we reported. Indeed, we try to exploit both knowledge strictly present within the Wikipedia pages provided for this task as well as information coming from the rest of the online encyclopedia.

2 Description of the System

In this report we describe the methodology we developed to tackle the PRELEARN task. We report the choices made and the steps that led us to them. In particular, We focused on the raw-text setting, for which we adopted two systems with the goal of prerequisite learning. Although both use the Wikipedia pages' texts, each one does it in different ways.

2.1 Model 1

This model exploits a combination of pretrained word embeddings, of GloVe type (Pennington et al., 2014), as trained for Italian in (Berardi et al., 2015) and handcrafted features, the latter inspired from (Miaschi et al., 2019). In particular, for each page title in a concept pair (A, B), we computed a 300-dimension vector by averaging the word embeddings of each word in the A/B title. These two resulting vectors were concatenated together with the following 14 handcrafted features.

- Is B(A) in A(B)'s text?
- Number of occurrences of B(A) in A(B)'s text
- Is B(A) in the first sentence of A(B)?
- Is B in A's title?
- Length of A(B)
- Jaccard similarity between the texts
- Jaccard similarity between nouns in the texts
- Difference in length between first paragraphs
- Difference in number of nouns in first paragraphs
- Jaccard similarity between nouns in first paragraphs

Then, for each pair (A,B) the final feature vector of 614 dimensions, was fed to a XGBoost classifier (Chen and Guestrin, 2016), whose model selection was performed via a nested cross validation with grid search.

2.2 Model 2

This model takes as information the first 400 words of each Wikipedia page, and for each pair (A,B) predicts if word B is a prerequisite for word A. It is composed of a Gated Recurrent Unit (Cho et al., 2014) with hidden size of 8 and encoding size 32, and a linear layer taking as input the concatenation of the two vectors representing the two Wikipedia pages to check and predict the prerequisite relation. This model, similar to model M1 in (Miaschi et al., 2019), though simpler, performs well enough and is fast to train. The parameters are chosen based on a grid search selecting the best results achieved on a validation set. The aforementioned values are the best performing choices for all settings and we keep them for the cross domain task as well. We tried different learning rates, though ultimately a constant one of 0.01 for the whole training was the best choice.

3 Discarded Models

We attempted to perform the structured data task as well, in particular adding the Wikipedia link

	Data-mining	Geometry	Physics	Precalculus
In-domain				
GRU + GCNConv[1]	0.74	0.74	0.85	0.84
Model 1	0.80	0.92	0.82	0.93
Model 2	0.81	0.91	0.81	0.89
Cross-domain				
Model 1	0.51	0.72	0.60	0.77
Model 2	0.48	0.71	0.61	0.77

Table 1: Accuracies obtained on the task test set. For the GCN see footnote.

structure to see if it would be useful. In order to exploit this knowledge we tried to use a Graph Convolutional Network (GCN) (Kipf and Welling, 2017). To do so we added the GCN between the Gated recurrent unit and the linear layer in Model 2 so as to perform the prediction based on the concatenation of the embedding of each node (Wikipedia page) in each pair. However this methodology resulted into lower scores in all dataset so we ended up not submitting it. We believe this is due to the fact that this is not the appropriate way to leverage the information present in the Wikipedia structure. Since we know from (Miaschi et al., 2019) that the information itself is relevant.

For Model 1 instead, a variation was tested with a multi-layer perceptron as well, but results were below those reported for the XGBoost ensemble.

An overall different approach we rejected is using transformer models. Indeed to obtain a representation of the text composing each page we could employ a representation extrapolated from BERT. However, after seeing how, much smaller models were overfitting the training set, we concluded that the amount of available textual data is not enough to exploit this model and avoided it.

4 Results

In Table 1 we report the achieved accuracy on the test set. As we can see, Model 1 outperformed Model 2. This is remarkable in the sense that the former is simpler than the one based on recurrent networks. The same can be said about the hand-crafted features, which are mostly statistics of each pair of pages based on occurrences. Indeed, as proven also in (Miaschi et al., 2019), this information does help the model. We believe Model 1 attained a higher score thanks to its pre-trained word embeddings and the larger corpora they are trained upon. Indeed, the dataset used to create those vectors is composed of the whole Italian Wikipedia and of a large amount of novels. This encodes within these representations a wider knowledge than the one provided for this task only. Looking at the accuracy achieved with the GCN layer, we see how performances are systematically lower than the others, that is why we chose not to submit it.

After looking at the challenge results, we proceeded to explore more in general how well our models performed. In order to do so, for each one, we estimated precision, recall, accuracy and f1 score (reported in Table 2).

When comparing Model 1 and 2 between them, we noticed that the latter exhibited higher precision in 3 of the 4 areas, but also lower recall in 3 of them. As a result, there was a systematic difference in accuracy and f1-scores favouring Model 1 over Model 2. If we look closely at Model 1 scores in Table 2 we see how Physics and Precalculus show a broader difference between precision and recall. This underlines how in these two domains there are some concepts that despite being involved in several prerequisite relations are less represented in the general knowledge. Moreover, the same behavior is experienced for Model 2, indicating how the models started to miss some positive samples. The fact that it happens for this second setting makes us believe this phenomenon is also due to the presence of more spread information within the Wikipedia pages of the concepts enclosed in these domains. As we mentioned the second model has higher precision in three cases, whereas the first has higher recall, in two cases the

---

[1] Values from our own validation set split

	Precision	Recall	Accuracy	F1
Model 1				
data_mining	0.80	0.80	0.80	0.80
geometry	0.92	0.92	0.92	0.92
physics	0.84	0.82	0.82	0.81
precalculus	0.93	0.93	0.93	0.93
Model 2				
data_mining	0.82	0.80	0.81	0.81
geometry	0.90	0.91	0.91	0.91
physics	0.87	0.73	0.81	0.79
precalculus	0.95	0.82	0.89	0.88

Table 2: All scores obtained by Models 1 and 2.

difference in recall is much in favor of the latter and indeed it is the better performing one.

5 Discussion

Regarding the first model, we see how the vectorization obtained from the Wikipedia corpus performs well, particularly considering that it represents exclusively the pages' titles. We also notice that the comparison between the two models is not straighforward since the ensemble model we used was not tested on the vectors obtained from the recurrent neural networks. We did not experiment in this mixed setting, since we believe it would not make sense to deploy a methodology with the power of XGBoost on embeddings solely based on the information present in the pages provided for this task. Indeed, there are high chances that the results for such complex model would still be worse than the one with the pretrained embeddings, since, as we mentioned in Section 4, the knowledge available exclusively in the pages proposed for this task is limited.

The other remarkable aspect is that to surpass the performance of the GRU, handcrafted features were helpful, despite them being mostly word occurrences counts. This same information is available to the GRU models, which performs worse. This underlines how the recurrent architecture, though powerful and able to capture long distance relations, can not retain this type of substantial details. Regarding the second model introduced, we remark how the hidden units size and the encoding size are very small. This is coherent with the fact that the dataset is not large enough to exploit the scaling potential of a recurrent neural network with a larger size. However, with this small model the results are better than with a baseline and as we mentioned the training times are all quite small. Thus, the idea of performing more ablation studies where bag of words methodologies are used together with recurrent ones, could lead to further improvements still supporting a more bottom-up solution than hand crafted features.

Following the analysis of the models we used, we can conclude that the property of being a prerequisite is a complex characteristic and thus the use of large amounts of data can be useful. On the other hand, the fact that the model solely based on the data at hand performs only marginally worse than the other underlines how this information is present in the pages themselves. Possibly a mixed dataset contained between the one at hand and the whole Italian Wikipedia could be a solution to move further in prerequisites learning.

References

Giovanni Adorni, Chiara Alzetta, Frosina Koceva, Samuele Passalacqua, and Ilaria Torre. 2019. Towards the identification of propaedeutic relations in textbooks. In Seiji Isotani, Eva Millán, Amy Ogan, Peter M. Hastings, Bruce M. McLaren, and Rose Luckin, editors, *Artificial Intelligence in Education - 20th International Conference, AIED 2019, Chicago, IL, USA, June 25-29, 2019, Proceedings, Part I*, volume 11625 of *Lecture Notes in Computer Science*, pages 1–13. Springer.

Chiara Alzetta, Alessio Miaschi, Felice Dell'Orletta, Frosina Koceva, and Ilaria Torre. 2020. Prelearn @ evalita 2020: Overview of the prerequisite relation learning task for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign* *of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Giacomo Berardi, Andrea Esuli, and Diego Marcheggiani. 2015. Word embeddings go to italy: A comparison of models and training datasets. In *IIR*.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA. Association for Computing Machinery.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Martin Gerlach, Tiago P. Peixoto, and Eduardo G. Altmann. 2018. A network approach to topic models. *Science Advances*, 4(7).

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Chen Liang, Zhaohui Wu, Wenyi Huang, and C. Lee Giles. 2015. Measuring prerequisite relations among concepts. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1668–1674, Lisbon, Portugal, September. Association for Computational Linguistics.

Alessio Miaschi, Chiara Alzetta, Franco Alberto Cardillo, and Felice Dell'Orletta. 2019. Linguistically-driven strategy for concept prerequisites learning on italian. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 285–295.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

Partha Talukdar and William Cohen. 2012. Crowdsourced comprehension: Predicting prerequisite structure in Wikipedia. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 307–315, Montréal, Canada, June. Association for Computational Linguistics.

UNIGE_SE @ PRELEARN: Utility for Automatic Prerequisite Learning from Italian Wikipedia

Alessio Moggio, Andrea Parizzi
DIBRIS, Università degli studi di Genova
{s4062312, s4048705}@studenti.unige.it

Abstract

The present paper describes the approach proposed by the UNIGE_SE team to tackle the EVALITA 2020 shared task on Prerequisite Relation Learning (PRELEARN). We developed a neural network classifier that exploits features extracted both from raw text and the structure of the Wikipedia pages provided by task organisers as training sets. We participated in all four sub–tasks proposed by task organizers: the neural network was trained on different sets of features for each of the two training settings (i.e., raw and structured features) and evaluated in all proposed scenarios (i.e. in– and cross– domain). When evaluated on the official test sets, the system was able to get improvements compared to the provided baselines, even though it ranked third (out of three participants). This contribution also describes the interface we developed to compare multiple runs of our models. [1]

1 Introduction

Prerequisite relations constitute an essential relation between educational items since they express the order in which concepts should be learned by a student in order to allow a full understanding of a topic. Therefore, automatic prerequisite learning is a relevant task for the development of many educational applications.

Prerequisite Relation Learning (PRELEARN) (Alzetta et al., 2020), a shared task organized within EVALITA 2020, the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (Basile et al., 2020), has, as a purpose, automatic prerequisite relation learning between pairs of concepts. For the purposes of the shared tasks, concepts are represented as learning materials written in Italian. In particular, each concept corresponds to a page of the Italian Wikipedia having the concept name as title. The goal of the shared task is to build a system able to automatically identify the presence or absence of a prerequisite relation between two given concepts. The task is divided in four sub-tasks: specifically in order to make a valid submission participants are asked to build at least one model for automatic prerequisite learning to be tested both in in- and cross-domain scenario since task organisers released four official training sets, one for each domain of the dataset. Concerning the model, it can exploit either 1) information extracted from the raw textual content of Wikipedia pages, 2) information acquired from any kind of structured knowledge resource (excluding the prerequisite labelled datasets). Eventually, we submitted our results on the official test sets for all four proposed subtasks. To tackle the problem proposed in the shared task, we propose an approach based on deep learning to classify on different sets of features in order to comply with the sub-tasks requirements. We also developed a user interface to support the comparison between the results obtained running the model trained using different sets of features. Other than selecting which features should be used to train the model, the user can exploit the interface to define the value of a set of parameters in order to customize the classifier structure. The interface reports, for each run, standard evaluation metrics (i.e., accuracy, precision, recall and F-score) and other statistics that allow to explore the model performances.

The remainder of the paper is organised as follows: we present our approach and system in Section 2, then we discuss the results and evaluation (Section 3). Section 4 describes the interface in

---

detail. We conclude the paper in Section 5.

2 System Description

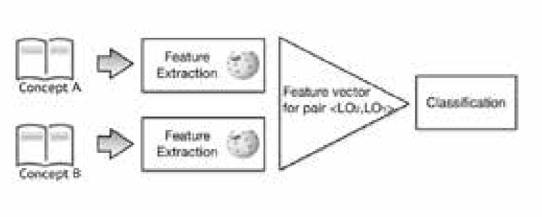


Figure 1: System architecture

In this Section we present our approach for automatic prerequisite learning between Wikipedia pages. We exploited a deep learning model that can be customised by the user on a dedicated GUI. The model was trained and tested on the official dataset of the PRELEARN task.

ITA-PREREQ (Miaschi et al., 2019) is a binary labelled dataset in which the labels stand for the presence or the absence (1 or 0) of the prerequisite relation between a pair of concepts. Each concept is an educational item associated to a Wikipedia page, therefore the concept name matches the title of the equivalent Wikipedia page. Hence, the dataset released for the shared task consists also of the content and the link of the Wikipedia pages referring to the concepts appearing in the dataset. It covers four domains, namely precalculus, geometry, physics and data mining.

2.1 Classifier

The classifier was built with the aim of testing the combination of different hand–crafted features on the automatic prerequisite learning task. More specifically our classifier, whose architecture is described in Figure 1, uses a two–dense–layers Neural Network built using Scikit-Learn and Keras libraries (wrapped for Tensorflow). The activation function for the hidden layer is ReLU while the Adam optimizer (Kingma and Ba, 2014) is used as training algorithm. The output layer consists of one neuron with sigmoid activation function.

Some structural properties of the classifier can be customised by the user from a dedicated GUI. In particular, for what concerns the structure of the neural network the user can define the size of the hidden layer and the number of epochs, while for the evaluation the user can set the number of cross validation folds. Moreover, training can be performed on a customizable set of features (see Section 2.2 for the complete list) since the input layer is set to dynamically match the size of the feature vector. For the specific purposes of this work, we used in every scenario a model exploiting a 20 neurons hidden layer trained on 15 epochs. A 4-fold cross validation was used for the in-domain scenario.

Training The official training set containing concept pairs and their binary labels was formatted as a pair of numpy arrays: one of them has variable length and contains the serialization of the features, which will be the model input, whilst the latter contains the binary labels of the pairs. For the in-domain scenario, the model was trained using stratified random folds of concept pairs that preserve the original proportion of domains' pairs. For the cross-domain evaluation scenario, a "leave one domain out" approach was used, training the model on all domains but the one used for test.

2.2 Features

We defined a set of features extracted from the Wikipedia page content and structure that are available in the GUI and can be selected by the user to train his model. While the pages content was provided in the official release of the training set, we exploited Wikipedia API [2] to extract the Wikipedia metadata and knowledge structure. Depending on the sub-task requirements, we trained our models with a different combination of features.

- Features used for the raw features model:
  - **titleInText**: given a pair (A, B), it checks if the title of page A/B is mentioned in the page of the other concept.
  - **Jaccard similarity**: a concept-based metric that measures the similarity between two pages by the number of words shared between them.
  - **LDA**: the Shannon Entropy of the LDA (Deerwester et al., 1990) of nouns and verbs in *A* and *B*. Nouns and verbs are identified thanks to a morpho-syntactic analysis of the page content performed by UDPipe pipeline (Straka and Straková, 2017).

---

[2] `https://github.com/martin-majlis/Wikipedia-API`

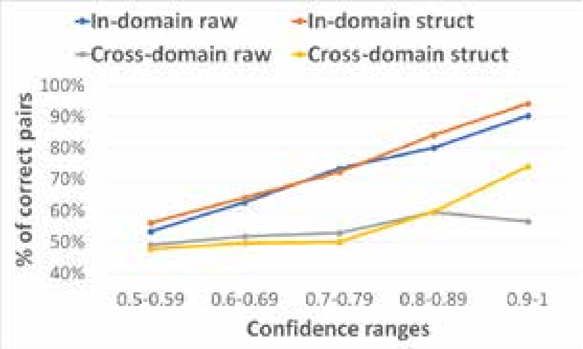


Figure 2: Variation of correctly labelled pairs wrt the classifier confidence for all submitted models.

  - **LDA Cross Entropy**: the cross entropy of the LDA vectors A\B.

- Features used for the raw and structural features model. We exploited all the above features combined with the followings:
  - **extractCategories**: the Wikipedia category(s) to which each page of the pair (A, B) belongs.
  - **extractLinkConnections**: for each pair of concepts (A, B) checks if the Wikipedia page of B contains a link to A.
  - **totalIncoming/OutgoingLinks**: it computes how much a concept is linked to/from other concepts.
  - **Reference distance**: a link-based metric that measures the relation between two pages by the links contained in each of them using the EQUAL weight (Liang et al., 2015).

3 Results and Error Analysis

Table 1 reports the results obtained by our models on the runs submitted for all four sub-tasks. On average, the performances of our systems in the different scenarios show that, as expected, training the model in a in–domain scenario allows to achieve better results. Among the different sets of features used, exploiting structural information extracted from Wikipedia pages' structure is in general more effective than relying only on raw textual data. Thus, our best performing model is the one exploiting both raw and structural features evaluated in-domain scenario which achieves an average accuracy computed across all four domains of 0.700. Interestingly Data Mining constitutes the only case where raw textual features are more effective than a combination of raw and structural features. In fact this domain shows lower accuracies in the structured settings, possibly due to the lower number of entries within the dataset of Data Mining with respect to the other three domains and to the lower coverage of Wikipedia.

If we compare our results obtained for each domain with those obtained by the official baseline, there are only two cases where our models do not outperform the baseline, i.e. Geometry in the raw feats in-domain subtask and physics in both cross-domain subtasks. During error analysis on geometry pairs, we observe that, while pages about geometric figures, e.g. ”Rettangolo” and ”Poligono”, show a prerequisite relation in the gold dataset, our systems always fail to correctly classify them. Concerning Physics, we observe that in both cross-domain settings the classifier did not consider the page ”Fisica” as prerequisite of other pages belonging to the Physics domain, causing the performances to be below baseline.

If we look at the variation of accuracy values for each model with respect to the classifier confidence (see Figure 2), we notice that although the four systems have a similar accuracy when the confidence is low those related to the two in-domain settings show a similar increase in accuracy confidence. Comparing cross-domain settings we notice that only the structured one is able to reach higher accuracy but only when it is highly confident.

4 System Interface

Together with our system we also developed a User Interface aimed at personalizing the network and comparing results obtained with different models. The interface is composed of the following three modules: i) setup module; ii) results module; iii) statistics module.

The setup module, loaded at the start of the program, allows to define:

- The input dataset;
- The parameters to setup the neural network architecture;
- The features for training the model.

Sub-Task	Data Mining	Geometry	Physics	Precalc	AVG
Raw Feats in-domain	0.595	0.620	0.530	0.675	0.605
Raw+Struct in-domain	0.565	0.755	0.725	0.755	**0.700**
Baseline in-domain	0.494	0.675	0.500	0.675	0.586
Raw Feats cross-domain	0.565	0.515	0.465	0.595	0.535
Raw+Struct cross-domain	0.545	0.665	0.560	0.710	**0.620**
Baseline cross-domain	0.494	0.500	0.605	0.500	0.525

Table 1: Results obtained for each sub-tasks by our models and the baseline on PRELEARN official test sets.

The module includes also a table where previously saved Configurations can be selected in order to run them again.

After running the model, the user can reach the results module in which are printed the performance statistics (accuracy, precision, recall, F-score) achieved by the performed configuration. Besides, the result module is composed of different buttons that allows to:

- Save the performed configuration.
- See the results of the classifier on concept pairs labelling.
- Save and download the results as csv file or txt file.

The statistics module plots in four bar charts the values of accuracy, precision, F-score and recall of all configurations saved in the interface. The repository containing the system and its GUI can be consulted on github [3].

5 Conclusion

In the paper we described the approach proposed by the UNIGE_SE team for the EVALITA 2020 PRELEARN shared task. The classifier relied on a set of features that was customised to address the specific requests of each sub-task. The results obtained by our models are all above baseline (if considered averaging the accuracies across all domains), although in some cases the results obtained by the baseline are still highly competitive. This suggests that automatic prerequisite learning is a difficult task requiring many different information to train the models. However, the obtained results suggest that, at least in a in-domain setting, features extracted from raw texts are sufficient to achieve competitive results. In the cross-domain setting exploiting only this type of features is not enough. Nevertheless, using information extracted from knowledge structures allows to achieve better results in all sub-tasks. Although our obtained results are promising, future work will be focused on analyzing the impact of each feature in training the model and exploring the inclusion of new features to improve the performance of the classifier.

References

Chiara Alzetta, Alessio Miaschi, Felice Dell'Orletta, Frosina Koceva, and Ilaria Torre. 2020. Prelearn@evalita 2020: Overview of the prerequisite relation learning task for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Chen Liang, Zhaohui Wu, Wenyi Huang, and C Lee Giles. 2015. Measuring prerequisite relations

---

[3] `https://github.com/mnarizzano/se20-project-16`

among concepts. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1668–1674.

Alessio Miaschi, Chiara Alzetta, Franco Alberto Cardillo, and Felice Dell'Orletta. 2019. Linguistically-driven strategy for concept prerequisites learning on italian. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 285–295.

Milan Straka and Jana Straková. 2017. Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99.

NLP-CIC @ PRELEARN: Mastering Prerequisites Relations, from Handcrafted Features to Embeddings*

Jason Angel
Instituto Politécnico Nacional
Mexico City, Mexico
`ajason08@gmail.com`

Segun Taofeek Aroyehun
Instituto Politécnico Nacional
Mexico City, Mexico
`aroyehun.segun@gmail.com`

Alexander Gelbukh
Instituto Politécnico Nacional
Mexico City, Mexico
`www.gelbukh.com`

Abstract

We present our systems and findings for the prerequisite relation learning task (PRELEARN) at EVALITA 2020. The task aims to classify whether a pair of concepts hold a prerequisite relation or not. We model the problem using handcrafted features and embedding representations for in-domain and cross-domain scenarios. Our submissions ranked first place in both scenarios with average F1 score of 0.887 and 0.690 respectively across domains on the test sets. We made our code freely available[1].

1 Introduction

A prerequisite relation is a pedagogical relation that indicates the order in which concepts can be presented to learners. The relation can be used to guide the presentation sequence of topics and subjects during the design of academic programs, lectures, and curricula or instructional materials.

In this work, we present our systems to automatically detect prerequisite relations for Italian language in the context of the PRELEARN shared task (Alzetta et al., 2020) at EVALITA 2020 (Basile et al., 2020). The evaluation of submissions considers: (1) in-domain and cross-domain scenarios defined by either the inclusion (in-domain) or exclusion (cross-domain) of the target domain in the training set. The four domains are 'data mining' (DM), 'geometry' (Geo), 'precalculus' (Prec), and 'physics' (Phy). (2) the type of resources (features) used to train the model – raw text VS. structured information.

The combination of these settings defined the four PRELEARN subtasks. Formally, a prerequisite relation exists between two concepts if one has to be known beforehand in order to understand the other. For the PRELEARN task, given a pair of concepts, the relation exists only if the latter concept is a prerequisite for the former. Therefore, the task is a binary classification task.

We approach the problem from two perspectives: handcrafted features based on lexical complexity and pre-trained embeddings. We employed static embeddings from Wikipedia and Wikidata, and contextual embeddings from Italian-BERT model.

2 Related works

Prerequisite relation learning has been mostly studied for the English language (Liang et al., 2018; Talukdar and Cohen, 2012). Adorni et al. (2019) performed unsupervised prerequisite relations extraction from textbooks using word co-occurrence and order of words appearance in the text. In the case of Italian language there is *ITA-PREREQ* (Miaschi et al., 2019), the first dataset for prerequisite learning, and actually the one used for the present work. It was automatically built as a projection of *AL-CPL* (Liang et al., 2018) from the English Wikipedia to the Italian Wikipedia. In addition, Miaschi et al. (2019) examines the utility of lexical features for individual concepts and features derived from the concept pairs.

3 Methodology

This section describes the data analysis, the features we used to model the task, and the system we finally submitted to the PRELEARN competition.

---

[1] `https://github.com/ajason08/EVALITA2020_PRELEARN`

3.1 Dataset

The dataset provided by the organizers includes the concept pairs splitted into the following domains: 'data mining', 'geometry', 'precalculus' and 'physics'. The dataset contains the list of concepts with a link to the corresponding Wikipedia article. The first paragraph of such article is named the concept description. All concept descriptions are cleaned in order to facilitate the extraction of information from the text, e.g. the mathematical expressions are already tagged using this pattern formula_<number>.

Table 1 displays the number of samples and the distribution over the prerequisite relations (positive samples) across domains for the training set. The test sets in turn exhibits a 50-50 distribution over positive and negative samples.

The only preprocessing we did was lowercase the concept description and remove line-breaks.

Domain	Samples	Prerequisites rel.
Data mining	424	0.257
Geometry	1548	0.214
Precalculus	2220	0.142
Physics	1716	0.238

Table 1: Training set number of samples and distribution of prerequisite relations (positive samples) across domain

3.2 Features

The following are the set of features we experiment with:

Complexity-based: a set of handcrafted features intended to measure how complex a concept is. The rationale is that less complex concepts are prerequisites for the more complex ones. We used some features that have been found effective for the task of complex word identification (Aroyehun et al., 2018), specifically they are:

- Age of acquisition of concept: we use *ItAoA* (Montefinese et al., 2019), a dataset of age of acquisition norms (we average the values for the different entries per word), to derive the age of acquisition for each concept we compute the geometric mean of values from ItAoA for words which occur in the concept description after replacing outliers (by the closest permitted value). In addition, we use the number of matches as a feature.

- Age of acquisition of related concepts: We derived a list of concepts related to each concept by matching which of them appears in the concept description. Then, we average the age of acquisition of those concepts. We also took the count of the related concepts.

- Description length: we count the number of words in the concept description.

- Number of mathematical expressions: we count the occurrence of mathematical expressions. We assume that more complex concepts will have a higher occurrence of mathematical expressions in their descriptions.

- Concept view frequency: the average of the daily unique visits by Wikipedia users (including editors, anonymous editors, and readers) over the last year. We think that the number of visitors will be correlated with the degree of complexity of a concept. To gather this information we used the Pageviews Analysis of Wikipedia [2].

Concept-to-Concept features: they aim to model the relation between the concept pairs, specifically we evaluate whether a concept appears as a sub-string in the title or description of the other concept. We did this in both directions resulting in two features. We also represent the domain they belong to as a one-hot vector.

Wiki-embeddings: We map each concept identifier to their corresponding Wikipedia title and Wikidata identifier using the Wikidata Query Service[3]. Then, we obtain the 100 dimensional vector for each Wikipedia title from a pre-trained Wikipedia embedding[4] (Yamada et al., 2020). Similarly, we use the Wikidata embedding[5] (Lerer et al., 2019) to represent the Wikidata identifiers as 200 dimensional vectors.

Italian-BERT features: We used a pre-trained uncased version of Italian BERT (base model)[6] provided by the MDZ Digital Library team (dbmdz) trained on 13GB of text mainly from

---

[2] `https://pageviews.toolforge.org`

[3] `query.wikidata.org`

[4] `http://wikipedia2vec.s3.amazonaws.com/models/it/2018-04-20/itwiki_20180420_100d.pkl.bz2`

[5] `https://dl.fbaipublicfiles.com/torchbiggraph/wikidata_translation_v1.tsv.gz`

[6] `https://huggingface.co/dbmdz/bert-base-italian-uncased`

Scenario	Resources	System	DM	Geo	Phy	Prec	AVG
in-domain	raw-text	Italian-BERT	0.838	0.925	0.855	0.930	0.887
in-domain	structured	Complex+wd	0.808	0.905	0.795	0.915	0.856
in-domain	structured	Complex	0.828	0.895	0.785	0.885	0.848
cross-domain	raw-text	Italian-BERT	0.565	0.785	0.635	0.775	0.690
cross-domain	structured	Complex+wd	0.535	0.775	0.600	0.760	0.668
cross-domain	structured	Complex	0.494	0.735	0.595	0.730	0.639

Table 2: Test set results for the four PRELEARN subtasks using F1-score

Settings	In-domain	Cross-domain
raw-text	+2.1%	+4.2%
structured	**+15.6%**	**+4.8%**

Table 3: Performance advantage over the next best participant on average across domains

Wikipedia and other text sources. With this model, we get the 768 dimensional vector representation for a sequence corresponding to the [CLS] token as in the original implementation of BERT (Devlin et al., 2019). The sequence consists of the combination of the concept and its Wikipedia description.

3.3 Systems

Considering the proposed features and our experimental results at Section 5, we proposed the following three systems to address both, in-domain and cross-domain scenarios. For the in-domain scenario we trained with a combination of all the training samples per domain. In the same way, we combined the remaining three domains for each cross-domain experiment (i.e. excluding samples from the target domain).

Complex: a completely handcrafted machine learning system, it uses all the complexity-based and Concept-to-Concept features (except the domain vector for cross-domain scenario), and we normalize the features using Z-score normalization. This system uses a tree-ensemble learner as classifier[7] with the default parameters provided by Breiman (2001)[8]. This system participated under the structured resource setting because the "concept view frequency" feature is structured information.

Complex+wd: an improved version of the *Complex* system by only concatenating the Wikidata embedding of each concept in the concept pair to the feature set. This system participated under the structured resource setting as well. We decided to not include the Wikipedia embeddings considering the ablation analysis we present in Table 4.

Italian-BERT: a single layer neural network mapping the 768 features from the [CLS] to the output space of dimension 2 as a sequence pair classification task. In addition, the pre-trained weights of the base model are fine-tuned on the training dataset. We fine-tune the base model using the huggingface transformers library (version 3.1) for Pytorch (Wolf et al., 2019). In the in-domain scenario, we use the following training parameters: the number epochs is 10, learning rate is $5e{-}5$, weight decay is 0.01, batch size is 32, warm up steps is 100, optimizer is AdamW with a linear schedule after a period of warm up steps. We find that the model exhibits high variance across runs in our cross-domain experiments. Hence, in addition to the parameter settings for the in-domain experiments, we choose the number of training steps using a validation set for the unseen target domain. Accordingly, we set the maximum training step to 400 and the warm up steps to 100, 200, 150, and 200 for data mining, geometry, physics, and precalculus cross-domain scenarios respectively.

4 Results

Table 2 shows our per-domain results for our systems indicating the kind of scenario and resources they used. We observe the clear superiority of Italian-BERT which only relies on raw-text resources. This suggest that just fine-tuning BERT

---

[7] Other classifiers were tested and obtained lower peformance

[8] `https://cran.r-project.org/web/packages/randomForest/index.html`

Scenario	Resources	Feature set	DM	Geo	Phy	Prec	AVG
in-domain	raw	complexity	0.646	0.817	0.622	0.792	0.720
in-domain	raw	wp_embedding	0.705	**0.818**	0.670	0.827	0.755
in-domain	raw	Italian-BERT	**0.947**	0.746	**0.829**	**0.842**	**0.841**
in-domain	structured	complexity +page_view	0.648	0.805	0.629	0.804	0.721
in-domain	structured	wd_embedding	0.660	0.814	0.674	0.838	0.746
in-domain	structured	wd+wp_embedding	0.694	**0.824**	0.672	0.831	0.755
in-domain	structured	complexity +page_view +wd_embedding	**0.697**	0.823	**0.686**	**0.845**	**0.763**
cross-domain	raw	complexity	0.072	0.592	0.258	**0.586**	0.377
cross-domain	raw	wp_embedding	0.000	0.622	0.079	0.344	0.261
cross-domain	raw	Italian-BERT	**0.145**	**0.646**	**0.460**	0.570	**0.455**
cross-domain	structured	complexity +page_view	**0.107**	0.588	0.297	0.577	0.392
cross-domain	structured	wd_embedding	0.000	**0.661**	0.355	0.608	0.406
cross-domain	structured	wd+wp_embedding	0.000	0.660	0.332	0.605	0.399
cross-domain	structured	complexity +page_view +wd_embedding	0.064	0.645	**0.366**	**0.630**	**0.426**

Table 4: Ablation analysis results using F1-score (validation set for Italian-BERT and 10-fold for the others)

is enough for gaining a notion of prerequisite relations on concepts. Still, the systems based on handcrafted features and non-contextual embedding exhibit competitive results, with a good enough performance to rank first in the structured resource setting, while being faster, more interpretable and simpler than the Italian-BERT counterpart.

The results showed that there is a huge performance reduction for the cross-domain scenario. The largest performance drop is on the "data mining" domain. Given that we train our models on the combination of examples from all other domains, it is likely that the probable cause is the domain mismatch. Yet, the reduction on the test sets are smaller than what we observe in our K-fold experiments and validation sets.

In addition, we show in Table 3 the performance advantage we obtained over the next best participant based on the ranking released by the organizers.

One can see that the greater performance advantage is from the structured resource setting. This suggests that the "Concept view frequency" and the Wikidata embedding features are effective.

5 Discussion: ablation analysis

During the creation our systems we perform several experiments over the possible features to use. We did 10-fold cross validation for the in-domain experiments except with the Italian-BERT[9], for which we used a stratified split of 30% for validation set. Table 4 shows the experimental results over the training (validation) set for both, in-domain and cross-domain scenarios. The "Resources" column serves to identify the type of resources used for the current feature.

We observe that the "data mining" domain appears to be difficult in the cross-domain scenario, models based on the non-contextual embedding features obtain results of zero. We suspect that this difficulty is due to the domain mismatch.

Based on these results, we select the Italian-BERT for the raw-text setting, and the "complexity +page_view" and the addition of Wikidata embeddings ("wd_embedding") for the structured resource setting for our submissions.

---

[9] Due to its high computational requirements

6 Conclusion

We tackle the task of prerequisite relation learning using a variety of systems that explore three set of features: handcrafted features based on complexity intuitions, embedding models from Wikipedia and Wikidata, and contextual embedding from Italian-BERT model. We examine the capabilities of our models in in-domain and cross-domain scenarios. Our models ranked first in all the subtask of the PRELEARN competition at EVALITA 2020. We found that although our Italian-BERT model outperformed the others, the simpler models show competitive results.

We plan to further examine the impact of using a combination of all possible domains as training set on the performance of our models.

Acknowledgments

The authors thank CONACYT for the computer resources provided through the INAOE Supercomputing Laboratory's Deep Learning Platform for Language Technologies.

References

Giovanni Adorni, Chiara Alzetta, Frosina Koceva, Samuele Passalacqua, and Ilaria Torre. 2019. Towards the identification of propaedeutic relations in textbooks. In *International Conference on Artificial Intelligence in Education*, pages 1–13. Springer.

Chiara Alzetta, Alessio Miaschi, Felice Dell'Orletta, Frosina Koceva, and Ilaria Torre. 2020. Prelearn @ evalita 2020: Overview of the prerequisite relation learning task for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Segun Taofeek Aroyehun, Jason Angel, Daniel Alejandro Pérez Alvarez, and Alexander Gelbukh. 2018. Complex word identification: Convolutional neural network vs. feature engineering. In *Proceedings of the thirteenth workshop on innovative use of NLP for building educational applications*, pages 322–327.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Adam Lerer, Ledell Wu, Jiajun Shen, Timothee Lacroix, Luca Wehrstedt, Abhijit Bose, and Alex Peysakhovich. 2019. PyTorch-BigGraph: A Large-scale Graph Embedding System. In *Proceedings of the 2nd SysML Conference*, Palo Alto, CA, USA.

Chen Liang, Jianbo Ye, Han Zhao, Bart Pursel, and C Lee Giles. 2018. Active learning of strict partial orders: A case study on concept prerequisite relations. *arXiv preprint arXiv:1801.06481*.

Alessio Miaschi, Chiara Alzetta, Franco Alberto Cardillo, and Felice Dell'Orletta. 2019. Linguistically-driven strategy for concept prerequisites learning on italian. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 285–295.

Maria Montefinese, David Vinson, Gabriella Vigliocco, and Ettore Ambrosini. 2019. Italian age of acquisition norms for a large set of words (itaoa). *Frontiers in psychology*, 10:278.

Partha Talukdar and William Cohen. 2012. Crowdsourced comprehension: predicting prerequisite structure in wikipedia. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 307–315.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: an efficient toolkit for learning and visualizing the embeddings of words and entities from wikipedia. *arXiv preprint 1812.06280v3*.

TRACK
“TIME AND DIACHRONY”

DaDoEval: Dating Documents

DaDoEval @ EVALITA 2020:
Same-Genre and Cross-Genre Dating of Historical Documents

Stefano Menini[*], **Giovanni Moretti**[**], **Rachele Sprugnoli**[**], **Sara Tonelli**[*]
[*]DH Research Group, Fondazione Bruno Kessler
Via Sommarive 18, 38123 Trento
[**]CIRCSE Research Centre, Università Cattolica del Sacro Cuore
Largo Agostino Gemelli 1, 20123 Milano
`{menini,satonelli}@fbk.eu`
`{giovanni.moretti,rachele.sprugnoli}@unicatt.it`

Abstract

English. In this paper we introduce the DaDoEval shared task at EVALITA 2020, aimed at automatically assigning temporal information to documents written in Italian. The evaluation exercise comprises three levels of temporal granularity, from coarse-grained to year-based, and includes two types of test sets, either having the same genre of the training set, or a different one. More specifically, DaDoEval deals with the corpus of Alcide De Gasperi's documents, providing both public documents and letters as test sets. Two systems participated in the competition, achieving results always above the baseline in all subtasks. As expected, coarse-grained classification into five periods is rather easy to perform automatically, while the year-based one is still an unsolved problem also due to the lack of enough training data for some years. Results showed also that, although De Gasperi's letters in our test set were written in standard Italian and in a style which was not too colloquial, cross-genre classification yields remarkably lower results than the same-genre setting.[1]

1 Introduction

In the context of EVALITA 2020 (Basile et al., 2020), we propose the task of assigning a temporal span to a document, i.e. recognising when a document was issued. The task has already been addressed in other languages, namely French, English, Polish, also in the framework of shared tasks, see for example the DÉfi Fouille de Textes (DEFT) 2010 and 2011 challenges (Grouin et al., 2010; Grouin et al., 2011), the SemEval-2015 task on Diachronic Text Evaluation (Popescu and Strapparava, 2015) and the RetroC challenge (Graliński et al., 2017). This task is relevant because it can play a role in document retrieval, summarisation, event detection, etc. It is also an important task per se, since it can be used to process large archival collections. In particular, when some documents in a collection have not been dated, supervised approaches could be applied to learn from the documents with a date which time span can be assigned to those who are not provided with temporal metadata. Along this line, we proposed our task taking Alcide De Gasperi's corpus of public documents (Tonelli et al., 2019) as a use case. To our knowledge, this task for Italian has never been proposed before to the NLP community, which means that all participating systems have been built from scratch.

All information related to the task, the official scorer and the training, test and gold data are available on the task website `https://dhfbk.github.io/DaDoEval/`.

2 Task Description

The goal of the DaDoEval shared task is to foster the development of systems able to automatically assign temporal information to unseen documents with different granularity. Therefore, we foresee three types of temporal spans, from coarse-grained to year-based, corresponding to different classification difficulty. Furthermore, we want to assess the impact of out-of-domain data on classification quality. We therefore propose the six following subtasks:

1a **Coarse-grained classification on same-genre data**: participants are asked to assign each document in the test set to one of the main time periods that historians have identi-

---

A	B	C	D	E
Habsburg years	Beginning of political activity	Internal exile	From fascism to the Italian Republic	Building the Italian Republic
1901-1918	1919-1926	1927-1942	1943-1947	1948-1954

Table 1: Time periods for the coarse-grained tasks.

fied in De Gasperi's life, reported in Table 1. Each document in the training set is labeled with one of the five periods and test data are of the same genre of the training data, both taken from the corpus of De Gasperi's public documents (Tonelli et al., 2019).

1b **Coarse-grained classification on cross-genre data**: participants are asked to assign each document in the test set to one of the main time periods that historians have identified in De Gasperi's life, reported in Table 1. Each document in the training set is labeled with one of the five periods and taken from the corpus of De Gasperi's public documents, while the test set contains letters from De Gasperi's correspondence (Tonelli et al., 2020).

2a **Fine-grained classification on same-genre data**: participants are asked to assign each document in the test set to one temporal slice of 5 years. Each document in the training set is labeled with a temporal slice and test data are of the same genre of the training data, both taken from De Gasperi's public documents.

2b **Fine-grained classification on cross-genre data**: participants are asked to assign each document in the test set to one temporal slice of 5 years. Each document in the training set is labeled with a temporal slice and test data are extracted from De Gasperi's correspondence.

3a **Year-based classification on same-genre data**: participants are asked to assign each document in the test set to its exact year of publication. Each document in the training set is labeled with the year of publication and test data are of the same genre of the training data, both taken from De Gasperi's public documents.

3b **Year-based classification on cross-genre data**: participants are asked to assign each document in the test set to its exact year of publication. Each document in the training set is labeled with the year of publication and test data are extracted from De Gasperi's correspondence.

Subtask 1 is the easiest task of the challenge, since the five time periods were defined by history scholars based also on the different roles and events involving De Gasperi during his career. We expect therefore that the documents grouped together for each time period present a high degree of similarity concerning topics, mentioned people and events. Also different document types should vary over time, with more news articles dated between 1901 and 1918, when De Gasperi worked as a journalist, and more telegrams written towards the end of his career, when De Gasperi was Minister of Foreign Affairs.

Subtask 2 includes 11 classes, each comprising 5 years. In this case, however, the division is arbitrary and purely based on the document date, therefore documents in the same class do not necessarily have anything in common concerning the topic, De Gasperi's role, etc. Finally, subtask 3 is the most challenging one, also because for some years only few training examples were available. More details on the document distribution in the training set are reported in Section 3.

The aforementioned subtasks can be addressed in several ways. For example, researchers interested in historical content analysis can infer temporal information by looking at persons, places and time expressions, possibly integrating linking techniques. For those interested in studying semantic shifts, a purely lexical analysis may highlight changes in the lexical choices made by De Gasperi over time and give hints for document dating (Kulkarni et al., 2018). Also deep learning techniques, which proved effective on larger English corpora for document dating, could be tested (Vashishth et al., 2018). As an alternative, the subtasks could be addressed using document similarity techniques, so to assess to which training documents those in the test set are most similar, assuming that similar documents have been written in the same years.

3 Dataset

The corpus of De Gasperi's public documents contains 2,759 documents, manually tagged with a date, written by De Gasperi and issued between 1901 and 1954. All the documents have been written by the same person, thus removing the effects that different author styles can have on the dating process. Since we proposed a supervised task, the corpus was split into a training and a test set following an 80:20 ratio, thus having 2,210 documents for training and the remaining 549 for testing.

In addition to the in-domain test set, we also provide a cross-genre out-of-domain test set of 100 private letters, written by De Gasperi in the same time span of the corpus of public documents within the Epistolario project[2]. This out-of-domain test set allowed DaDoEval organisers to evaluate the robustness of the proposed approaches, and measure how the specific characteristics of correspondence affect the dating process.

We report in Table 3 the document distribution in the training and test set for the coarse- and the fine-grained subtasks. In general, the classes are not well-balanced, with some periods having only few training documents. For example, in the fine-grained subtask the span 1926 – 1930 has only 16 documents vs. 599 documents belonging to the period 1946 – 1950.

In Figure 1 and 2 we show also the year-based distribution of documents in the training and in the test set. While the same-genre distribution is similar, the letters in the test set (red line in the graph) are more homogeneous, with no year-based peaks like for public documents. On the contrary, some years that are barely represented in the training set (for example 1927) present several instances in the cross-genre test set, making classification particularly challenging.

For both corpora, there are no privacy issues and the documents can be made freely to task participants.

4 Evaluation Procedure and Baseline

Each participating team is allowed to submit two runs for each subtask. The evaluation is performed by computing class-based Precision, Recall and F1, and then the macro-averaged F1, upon which the final ranking is based. The task scorer is available on the task website[3].

		Same-genre		Cross-genre
		Train	Test	Test
Coarse-grained	class1	572	140	20
	class2	342	109	20
	class3	150	37	20
	class4	514	98	20
	class5	632	165	20
Fine-grained	1901-1905	85	21	3
	1906-1910	256	65	6
	1911-1915	211	48	5
	1916-1920	109	42	11
	1921-1925	246	73	12
	1926-1930	16	2	10
	1931-1935	76	22	4
	1936-1940	62	13	8
	1941-1945	191	36	15
	1946-1950	599	129	16
	1951-1955	399	98	10

Table 2: Document distribution for the coarse-grained and the fine-grained subtasks.

As a baseline, we adopt for all tasks the same Logistic Regression configuration. As features to represent the document content, we calculate tf-idf for each term (unigram) in the dataset, without removing stopwords or performing any preprocessing on the text. For computing tf-idf and training the Logistic Regression classifier we rely on the scikit-learn library (Pedregosa et al., 2011).

5 Participants and Results

Eighteen teams registered to participate, but only two actually submitted the results for the evaluation for a total of 16 runs. Both participants come from the academia: one from Italy (University of Pisa) and one from Germany (University of Tübingen). A short description of each system follows:

matteo-brv (University of Tübingen) participated only in subtask 1 and 2 with two runs for each subtask (Brivio, 2020). Both subtasks have been treated as classification problems and modeled with a linear Support Vector Machine multi-class

---

[2] `https://www.epistolariodegasperi.it/`

[3] `https://github.com/dhfbk/DaDoEval/blob/master/DaDoEval_Eval.py`

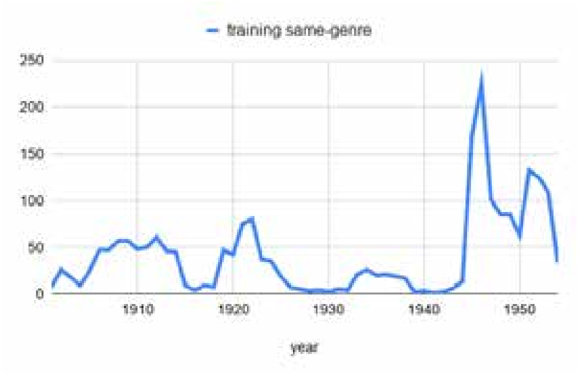


Figure 1: Per year document distribution in the training set.

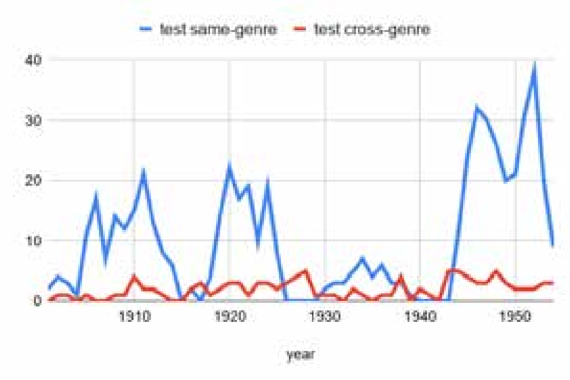


Figure 2: Per year document distribution in the test set.

Same-genre								
subtask 1a			subtask 2a			subtask 3a		
TEAM	#RUN	MACRO F1	TEAM	#RUN	MACRO F1	TEAM	#RUN	MACRO F1
matteo-brv	1	0.934	rmassidda	2	0.638	rmassidda	2	0.274
matteo-brv	2	0.934	rmassidda	1	0.579	rmassidda	1	0.256
rmassidda	1	0.858	BASELINE		0.485	BASELINE		0.126
rmassidda	2	0.855						
BASELINE		0.827						

Cross-genre								
subtask 1b			subtask 2b			subtask 3b		
TEAM	#RUN	MACRO F1	TEAM	#RUN	MACRO F1	TEAM	#RUN	MACRO F1
matteo-brv	1	0.413	rmassidda	2	0.177	rmassidda	1	0.074
matteo-brv	2	0.413	BASELINE		0.171	rmassidda	2	0.035
rmassidda	2	0.392	rmassidda	1	0.158	BASELINE		0.02
BASELINE		0.368						
rmassidda	1	0.366						

Table 3: Results of six subtasks in terms of macro-average F1.

classifier, implemented through the scikit-learn library (Pedregosa et al., 2011). The model was trained on a set of style-based features: TF-IDF weighted character and word n-grams, and number of word tokens per document. Features have been extracted without any form of data set pre-processing. N-gram size has been determined empirically and found to yield the best results in a range of 3 to 5 and 1 to 2 for character and word n-grams, respectively. On the other hand, TF-IDF parameters and model parameters were tuned using a 5-fold cross validation Bayesian optimization strategy, an algorithm implemented in the Scikit-Optimize library[4].

rmassidda (University of Pisa) participated in all subtasks with 2 runs for each of them (Massidda, 2020). Two representations are generated for each document with no fine-tuning: (i) a sequence of sentence embeddings using Sentence-BERT (Reimers and Gurevych, 2019), and (ii) a bag-of-entities obtained using the spaCY Named Entity Recognition system[5]. Since the performance obtained on a validation set showed that the first representation yields better results on the coarse-grained task, while the bag-of-entities performed better on the fine- and year-based tasks, the two representations are combined in an architecture where the sentence embeddings are fed to a transformer block containing a multi-headed self-attention layer. Its output is then averaged and concatenated with the bag-of-entities representation of the document before being fed to a multi-layer neural network. The output of each

---

[4] `https://scikit-optimize.github.io/stable/`

[5] `https://github.com/explosion/spacy-models`

layer of this network is also fed to a dedicated neural network that produces the output of each subtask.

6 Discussion

6.1 System comparison

The two submitted systems are based upon different paradigms: **matteo-brv** relies on an SVM-based classifier with simple linguistic features, while **massidda** uses recent transformer-based models and neural networks. Despite being more computationally intensive and complex, the second approach yields a lower performance than the first one. The difference in performance, however, is smaller in the cross-genre subtask (0.02 F1) than in the same-genre one (0.07 F1). As a comparison, we show in Fig. 3 the average F1 obtained by each participant's best run for the five classes (i.e. time periods) in the same-genre coarse-grained task. The results across the five classes are rather balanced and do not reflect the number of training examples for each class (see Table 3). Indeed, Class 3 (from 1927 to 1942) has the least number of training documents but both systems achieve the best results. This probably depends on the fact that in those years De Gasperi does not participate in public life and has no political role, therefore the tone, topics and mentioned people are probably different from those in the rest of the document collection, therefore they are easily identifiable.

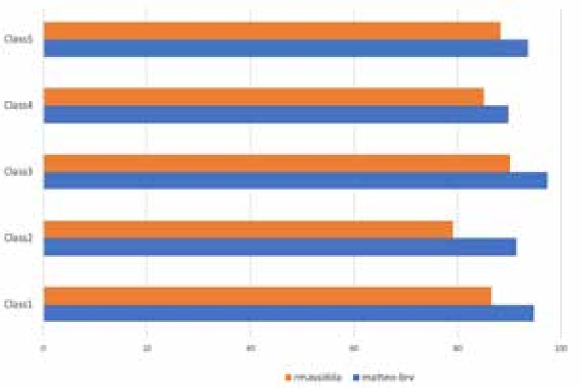


Figure 3: Comparison of participating systems on same-genre coarse-grained task

In Figure 4 we report the same comparison but in the cross-genre coarse-grained task. In this case, the two systems show a completely different behaviour, obtaining the worse results on Class 3. Furthermore, no system achieves the best result on all classes, like for the same-genre task. Interestingly, on Class 2 and 3, containing the least training documents, the neural approach by rmassidda clearly outperforms the SVM-based one.

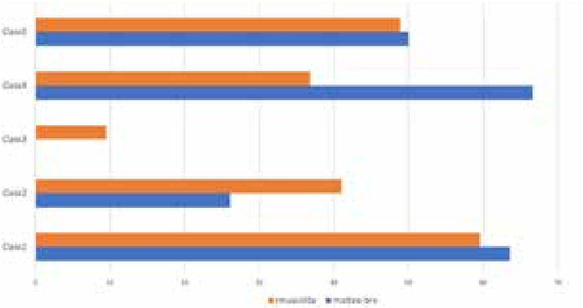


Figure 4: Comparison of participating systems on cross-genre coarse-grained task

Overall, there are huge performance differences with different classification granularity: while the coarse-grained subtask on same-genre data achieves a macro F1 above 0.82 even with a simple logistic regression baseline, performance drops dramatically with the fine-grained classification, and in the year-based task every presented approach yields insufficient results for any practical application. The presence of 55 classes (i.e. years) as well as an unbalanced distribution of training instances in the different classes make it indeed very difficult to build a robust supervised system.

After the competition deadline, matteo-brv submitted with the same SVM-based configuration the runs for subtasks 2 and 3, which were missing in the original submission. If regularly submitted to the competition, the system performance would be top-ranked with 0.702 in subtask 2a, 0.403 in subtask 3a, 0.240 on subtask 2b and 0.086 on subtask 3b. This confirms that, when dealing with middle-sized datasets, non-neural approaches can still be the best option, beside being easier to tune and less computationally intensive than neural classifiers.

6.2 Dataset comparison

In order to understand the impact of genre on classification performance, we randomly select 20 documents for each time period in the same-genre test set so to obtain a subcorpus similar in size (100 documents) and distribution as the cross-genre test set. Then, we process both corpora by running the Tint NLP Suite (Aprosio and Moretti, 2018), using in particular the modules computing complexity and readability indices.

From a lexical point of view, the two test sets do not differ much. For instance, type-token ratio is 0.81 in the same-genre subcorpus and 0.79 in the cross-genre one. In both cases, the value is rather high, confirming the careful selection of terms and expressions performed by De Gasperi, who was well-known for formal, sometimes archaic use of the language. This is evident also in the letters, even if they concerned people and events from his private sphere. Also the lexical density, i.e. the proportion between content words and the total number of words, is very similar, being 0.58 in same-genre subcorpus and 0.59 in the cross-genre one. Also in this case, the higher the value, the more 'conceptually dense' the text is, requiring more cognitive effort to read and understand the document content.

Although from a lexical point of view the two subcorpora are aligned, we observe a difference from the syntactic point of view. Indeed, while the average sentence length in the same-genre subcorpus is 21 tokens, it is 13 in the letters. This difference is confirmed also by the Gulpease score (Lucisano and Piemontese, 1988), which is the standard readability metric for Italian taking into account word and sentence length as a proxy for complexity. Gulpease is 61 for the letters and 50 for the same-genre subcorpus, corresponding to a higher readability for the former (the higher, the easier to read). Overall, this analysis shows that the more informal style usually associated with letters is expressed by De Gasperi through the use of simpler syntactic structures rather than through a simpler vocabulary. Also, classification approaches that rely on sentence-based units, for example sentence embeddings, may perform worse when the sentence characteristics are very different in the training and the test set.

If we consider semantic information, we observe also in this case some differences. For instance, the use of named entities is less frequent in letters than in the same-genre test set (0.44 avg. NER per sentence vs. 0.58). This holds for all the NER types considered, from persons (0.19 per sentence vs. 0.21) to locations (0.14 vs. 0.21). This again may affect the performance of systems using NER-based analysis like bag-of-entities, when the use of NER varies a lot between the training and the test set.

7 Conclusions

In this paper we have presented the DaDoEval task, which has been proposed for the first time at EVALITA 2020, with the goal to automatically date Italian documents. The task includes three different classification granularities, from five broad time spans to fifty-five years. Two subtasks are also foreseen, i.e. same-genre and cross-genre classification. The corpus used is the collection of De Gasperi's public documents, plus 100 letters being the test set for the cross-genre task.

Two systems have participated in the DaDoEval evaluation exercise, but only for the coarse-grained setting. In the other subtasks, there has been only one participant. A comparison between the two approaches has showed that a classifier based on SVM has consistently achieved better results than a neural one even if using a much simpler architecture. We also observed that cross-genre classification is still problematic, as is fine-grained classification. In order to have a better understanding of fine-grained classification, and provide more insightful system comparisons, it would be interesting to modify the scorer so to take into account how close misclassified examples are from the correct year or time period. This would provide a partial recognition to wrong instances when the assigned date is not far from the correct one.

The datasets and the scorer have been made available to the research community through the DaDoEval website, so that researchers will be able to deal with this task in the future, which is far from being solved.

Acknowledgements

We thank the President of the National Edition of De Gasperi's Letters Giuseppe Tognon and Stefano Malfatti for giving us access to the letters used in cross-genre classification task.

References

Alessio Palmero Aprosio and Giovanni Moretti. 2018. Tint 2.0: an All-inclusive Suite for NLP in Italian. In Elena Cabrio, Alessandro Mazzei, and Fabio Tamburini, editors, *Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Torino, Italy, December 10-12, 2018*, volume 2253 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Matteo Brivio. 2020. matteo-brv @ DaDoEval: An SVM-based Approach for Automatic Document Dating. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Filip Graliński, Rafał Jaworski, Łukasz Borchmann, and Piotr Wierzchoń. 2017. The RetroC challenge: how to guess the publication year of a text? In *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage*, pages 29–34.

Cyril Grouin, Dominic Forest, Patrick Paroubek, and Pierre Zweigenbaum. 2010. Présentation et résultats du défi fouille de texte deft2011 quand un article de presse a t-il été écrit? à quel article scientifique correspond ce résumé? In *Actes du septième DÉfi Fouille de Textes*.

Cyril Grouin, Dominic Forest, Patrick Paroubek, and Pierre Zweigenbaum. 2011. Présentation et résultats du défi fouille de texte deft2011 quand un article de presse a t-il été écrit? à quel article scientifique correspond ce résumé? In *Actes du septième DÉfi Fouille de Textes*.

Vivek Kulkarni, Yingtao Tian, Parth Dandiwala, and Steven Skiena. 2018. Simple neologism based domain independent models to predict year of authorship. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 202–212.

Pietro Lucisano and Maria Emanuela Piemontese. 1988. Gulpease. Una formula per la predizione della difficoltà dei testi in lingua italiana. *Scuola e Città*, 3:57–68.

Riccardo Massidda. 2020. rmassidda @ DaDoEval: Document Dating Using Sentence Embeddings at EVALITA 2020. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Octavian Popescu and Carlo Strapparava. 2015. Semeval 2015, task 7: Diachronic text evaluation. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 870–878.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3973–3983.

Sara Tonelli, Rachele Sprugnoli, and Giovanni Moretti. 2019. Prendo la Parola in Questo Consesso Mondiale: A Multi-Genre 20th Century Corpus in the Political Domain. In *In Proceedings of CLiC-it 2019*.

Sara Tonelli, Rachele Sprugnoli, Moretti Giovanni, Malfatti Stefano, and Odorizzi Marco. 2020. Epistolario De Gasperi: National Edition of De Gasperi's Letters in Digital Format. In *IX Convegno Annuale AIUCD*, pages 253–259. Alma Mater Digital Library.

Shikhar Vashishth, Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. Dating documents using graph convolution networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1605–1615.

matteo-brv @ DaDoEval: An SVM-based Approach for Automatic Document Dating

Matteo Brivio
University of Tübingen
Department of Linguistics
`matteo.brivio@student.uni-tuebingen.de`

Abstract

English. This paper describes our contribution to the EVALITA 2020 shared task DaDoEval – Dating Document Evaluation. The solution we present is based on a linear multi-class Support Vector Machine classifier trained on a combination of character and word n-grams, as well as number of word tokens per document. Despite its simplicity, the system ranked first both in the coarse-grained classification task on same-genre data and in the one on cross-genre data, achieving a macro-average F1 score of 0.934 and 0.413, respectively. The system implementation is available at `https://github.com/matteobrv/DaDoEval`.

1 Introduction

Temporal information, such as the publication date of a document, is of major relevance in a number of domains, like historical linguistics and digital humanities (Niculae et al., 2014). This is arguably even more true for a wide range of information retrieval tasks, such as document exploration, similarity search, summarisation and clustering, where the temporal dimension plays a major role in improving search results (Alonso et al., 2007; Alonso et al., 2011).

Such information, however, is not always readily available and must therefore be inferred, relying either on qualitative or quantitative methods, if not both (Ciula, 2017). Nonetheless, despite their significance, methods for temporal text classification and automatic document dating are still rather unexplored compared to other text classification tasks (Niculae et al., 2014). This, however, is most likely bound to change as the increasing availability of large-scale, time-annotated digital resources, such as Google n-grams[1], is promoting research in this direction. Two recent examples of this new trend, in line with the present task, are the Diachronic Text Evaluation shared task organised by Popescu et al. (2015) at SemEval 2015 and the RetroC Challenge presented by Graliński et al. (2017).

In this work we propose a simple, yet effective, approach for automatic document dating based on a linear multi-class Support Vector Machine classifier, trained on a combination of character and word n-grams, as well as document length in word tokens.

The solution is evaluated in the context of the DaDoEval – Dating Document Evaluation – shared task at EVALITA 2020 (Menini et al., 2020; Basile et al., 2020). The task is based on the Alcide De Gasperi's corpus of public documents (Tonelli et al., 2019) and is organised into six sub-tasks: (I) coarse-grained classification on same-genre data, (II) coarse-grained classification on cross-genre data, (III) fine-grained classification on same-genre data, (IV) fine-grained classification on cross-genre data, (V) year-based classification on same-genre data, (VI) year-based classification on cross-genre data.

The proposed solution tackles the first two sub-tasks, coarse-grained classification on same-genre and cross-genre data. Both sub-tasks require to correctly assign document samples to one of the main five time periods identified in De Gasperi's political life, spanning a range of over fifty years from 1901 to 1954.

The paper is structured as follows: in section 2 we provide a brief overview of the training data set, in section 3 we go over the system setup and describe the feature space, section 4 is dedicated to results analysis and discussion, in section 5 we

---

[1] `http://books.google.com/ngrams`

	1901-1918	*1919-1926*	*1927-1942*	*1943-1947*	*1948-1954*
SAMPLES PER CLASS	572	342	150	514	632
AVG. SAMPLE LENGTH	867	1033	3044	633	1209

Table 1: Training set overview, showing the number of document samples per class and the average number of word tokens per sample, rounded up to the nearest integer.

consider possible improvements while section 6 is reserved for final remarks.

2 Data

The training data set released for the shared task includes 2,210 document samples extracted from the Alcide De Gasperi's corpus of public documents, a multi-genre collection of 2,759 texts written or transcribed between 1901 and 1954 (Tonelli et al., 2019).

With respect to the coarse-grained classification sub-tasks, the given samples are organised into five classes (see Table 1) corresponding to the main time periods historians identified in De Gasperi's political life: *Habsburg years* 1901-1918, *Beginning of political activity* 1919-1926, *Internal exile* 1927-1942, *From fascism to the Italian Republic* 1943-1947, *Building the Italian Republic* 1948-1954.

A preliminary analysis of the data set reveals an imbalanced class distribution, with a significantly lower number of samples in the third class, corresponding to the 1927-1942 interval. This, however, is partially mitigated by the markedly higher average number of word tokens per sample observed in this class compared to the other ones.

3 System Description

The proposed solution is based on a Support Vector Machine (SVM) classifier implemented using the Scikit-learn library (Pedregosa et al., 2011).

To account for the rather imbalanced data set, the SVM is tuned in such a way that classes are assigned weights inversely proportional to their frequency in the input data.

Following the assumption that most text categorisation problems are linearly separable (Joachims, 1998) the model uses a linear kernel implemented in terms of `libsvm` (Chang and Lin, 2011) while relying on a `one-versus-one` decision strategy to handle both sub-tasks as multi-class, single label, classification problems.

3.1 Feature space

The system relies solely on the data provided by the task organisers and is split into training set (80%) and development set (20%). No preprocessing is applied, as measures such as case normalisation and punctuation removal do not seem to improve the classification result on the development set, but rather to worsen it.

Each document in the data set is represented using three sets of features: document length in terms of word tokens as well as character and word n-grams. In this respect, we explore the idea that SVMs trained on combinations of character and word n-grams are particularly effective in tackling text classification tasks (Çöltekin and Rama, 2017; Çöltekin and Rama, 2018).

Character n-grams are extracted for $n \in \{3, 4, 5\}$ and span across word boundaries, thus capturing punctuation and space characters occurring at the beginning and at the end of each word token. Word n-grams, on the other hand, are extracted for $n \in \{1, 2\}$. Both feature sets are weighted using term-frequency, inverse-document frequency (`TF-IDF`) to scale down the impact of the most frequent n-grams.

The number of word tokens per document is computed in a naive way, splitting each sample at every white space. Similarly to n-gram features, tokens count are scaled down to a 0-1 range in an attempt to avoid numerical problems and prevent features in higher numeric ranges from dominating those in smaller ones (Hsu et al., 2003).

3.2 Optimisation and Tuning

The system hyper-parameters are optimised to obtain the best F1 score on the development set.

A subset of the hyper-parameters is tuned empirically through several experiments or on the basis of existing literature. This is the case for kernel type, decision strategy, class balancing, tolerance for stopping criterion (`tol`) and n-grams size.

The remaining hyper-parameters considered during optimisation are the regularisation parameter (`C`) together with the maximum and minimum document frequency (`max_df`, `min_df`), which in the present approach are used to set an acceptance threshold for high and low frequency n-grams.

COMPONENT	PARAMETER	VALUE
TfidfVectorizer	analyzer	word
	max_df	0.9
	min_df	0.004
	ngram range	(1, 2)
	lowercase	False
TfidfVectorizer	analyzer	char
	max_df	0.3
	min_df	0.001
	ngram range	(3, 5)
	lowercase	False
SVM	kernel	linear
	decision function	ovo
	tol	1e-12
	C	0.881
	class weight	balanced

Table 2: Final hyper-parameters setup for each system component.

These hyper-parameters are tuned through the `BayesSearchCV` algorithm implemented in the scikit-optimize library (Head et al., 2020), using a 5-fold-shuffled cross validation. `BayesSearchCV` relies on Bayesian Optimisation and explores the hyper-parameters search space exploiting the information available from previous evaluations. This is in contrast to other approaches, such as grid and random search, which move across the search space either in an exhaustive or completely random manner.

Table 2 summarises the best hyper-parameters setup obtained from the tuning process.

4 Results

In this section we present the results for the two sub-tasks the system participated to. Results are summarised in Table 3 and reported in terms of macro-average F1 score.

The system ranked first both in the same-genre and in the cross-genre coarse-grained classification task, obtaining a macro-average F1 score of 0.934 and 0.413, respectively.

SUB-TASK	TEAM	RUN	MACRO F1
same-genre	matteo-brv	1	0.934
		2	0.934
	team 1	1	0.858
		2	0.855
	baseline	-	*0.827*
cross-genre	matteo-brv	1	0.413
		2	0.413
	team 1	1	0.392
	baseline	-	*0.368*
	team 1	2	0.366

Table 3: Final rankings for sub-task 1 and 2 in terms of macro-average F1 scores.

4.1 Classification on same-genre data

The runs submitted for the first sub-task are based on test samples of the same genre as the ones in the training set. The system scored well above the baseline, which was computed with a Logistic Regression model trained on `TF-IDF`-weighted word unigrams, without performing any preprocessing.

Overall, the results registered on the test set are in line with those observed during training. This is confirmed by the data summarised in Table 4 and by the confusion matrix in Figure 1.

The confusion matrix depicts a run on the development set which achieved a macro-average F1 score of 0.95, while Table 4 reports the per-class results of the best test run submitted for the sub-task. In both cases 1919-1926, 1943-1947 and 1948-1954 are the classes showing the highest number of misclassifications and, incidentally, are also the ones corresponding to the shortest time periods.

CLASS	PRECISION	RECALL	F1
1901-1918	0.914	0.986	0.948
1919-1926	0.96	0.872	0.913
1927-1942	0.973	0.973	0.973
1943-1947	0.898	0.898	0.898
1948-1954	0.939	0.933	0.936

Table 4: Per-class results of the best test run for sub-task 1.

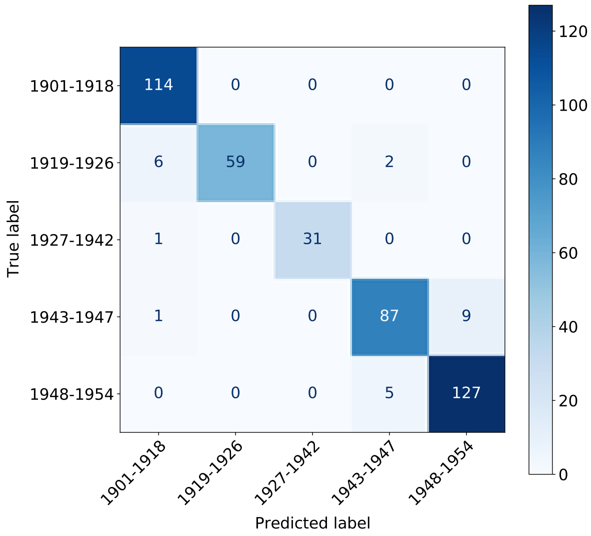


Figure 1: Confusion matrix for a development set run with a macro-average F1 score of 0.95.

4.2 Classification on cross-genre data

The runs submitted for the second sub-task are based on samples coming from a cross-genre, out-of-domain test data set. These samples are a subset of the documents collected for the Epistolario project (Tonelli et al., 2020), an ongoing effort to create a digital archive of Alcide De Gasperi's private and public correspondence.

CLASS	PRECISION	RECALL	F1
1901-1918	0.583	0.7	0.636
1919-1926	1.0	0.15	0.261
1927-1942	0.0	0.0	0.0
1943-1947	0.6	0.75	0.667
1948-1954	0.354	0.85	0.5

Table 5: Per-class results of the best test run for sub-task 2.

As expected, despite scoring above the baseline, cross-genre results are significantly lower than those obtained in the same-genre task. Per-class results summarised in Table 5 show how promising system performances registered in the same-genre task do not transfer to the cross-genre one, suggesting a poor ability of the model to generalise. Particularly interesting and worth investigating are the results registered for the third class, corresponding to the 1927-1942 interval. With respect to this class precision and recall values are equal to 0, indicating that model did not recognise any sample as belonging to this time period.

5 Possible improvements

Results for the same-genre task are quite encouraging and in line with those obtained on the development set, where the F1 score ranges between 0.92 and 0.96. However, with the current data and setup, there might not be much room for further improvement. Nonetheless, additional features like richness measures and linguistically motivated features (e.g. POS tags) are explored in other contributions (Štajner and Zampieri, 2013; Zampieri et al., 2016) and could help achieve more stable results.

On the other hand, results for the second subtask suggest a lack of generalisation on cross-genre, out-of-domain data. In this respect, even though SVM-based systems for text classification should be able to perform well and take advantage of high dimensional feature spaces (Joachims, 1998), it might still be worthwhile experimenting with some feature selection methods. Another angle worth considering is that the system might be too sensitive to the shallow n-gram features used to represent the training data. In this case, including deeper text features, such as those encoding syntactic information, might help the system to abstract away from the lexical level. A first step in this direction is attempted by Szymanski and Lynch (2015) who employ Google Syntactic N-grams in an SVM-based system that participated to the Diachronic Text Evaluation shared task (Popescu et al., 2015) at SemEval 2015.

6 Conclusions

In this paper we describe a simple, yet effective, approach for automatic document dating implemented for the DaDoEval shared task at EVALITA 2020. The system is based on a linear Support Vector Machine and is trained on a small set of stylistic and lexical features, resulting in a fast and efficient classification model.

In particular, the approach achieves top scores in both coarse-grained classification sub-tasks, thus confirming that SVM-based systems trained on character and word n-grams are indeed well suited to tackle text classification problems.

Nonetheless, results observed in the second task suggest that the model does not generalise well on cross-genre data, leaving room for further improvements.

Acknowledgments

We thank Dr. Çağrı Çöltekin for his patient encouragement and valuable suggestions throughout this project.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Arianna Ciula. 2017. Digital palaeography: What is digital about it? *Digital Scholarship in the Humanities*, 32(2):ii89–ii105.

Çağrı Çöltekin, Taraka Rama. 2018. Tübingen-oslo at SemEval-2018 task 2: SVMs perform better than RNNs in emoji prediction. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, 34-38.

Çağrı Çöltekin, Taraka Rama. 2017. Tübingen system in VarDial 2017 shared task: experiments with language identification and cross-lingual parsing. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, 146-155.

Chih-chung Chang, Chih-jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27.

Chih-Wei Hsu, Chih-Chung Chang and Chih-Jen Lin. 2003. A practical guide to support vector classification. *Technical report, Department of Computer Science, National Taiwan University.*

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Filip Graliński, Rafał Jaworski, Łukasz Borchmann and Piotr Wierzchoń. 2017. The RetroC Challenge: How to Guess the Publication Year of a Text?. In *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage*, 29–34.

Marcos Zampieri, Shervin Malmasi and Mark Dras. 2016. Modeling Language Change in Historical Corpora: The Case of Portuguese. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC'16)*, 4098–4104.

Octavian Popescu and Carlo Strapparava. 2015. Semeval 2015, task 7: Diachronic text evaluation. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 870–878.

Omar Alonso, Strötgen Jannik, Baeza Y. Ricardo and Gertz Michael. 2011. Temporal Information Retrieval: Challenges and Opportunities. In *Proceedings of the 1st International Temporal Web Analytics Workshop*, 11:1–8.

Omar Alonso, Gertz Michael and Baeza Y. Ricardo. 2007. On the value of temporal information in information retrieval. *SIGIR Forum*, 41:35–41.

Sanja Štajner and Marcos Zampieri. 2013. Stylistic Changes for Temporal Text Classification. In *Proceedings of the 16th International Conference on Text, Speech and Dialogue (TSD), Lecture Notes in Artificial Intelligence - LNAI 8082, Springer*, 519-526.

Sara Tonelli, Rachele Sprugnoli and Giovanni Moretti. 2019. Prendo la Parola in Questo Consesso Mondiale: A Multi-Genre 20th Century Corpus in the Political Domain. In *Proceedings of CLIC-it 2019.*

Sara Tonelli, Rachele Sprugnoli, Giovanni Moretti, Stefano Malfatti and Marco Odorizzi. 2020. Epistolario De Gasperi: National Edition of De Gasperi's Letters in Digital Format. In *Proceedings of AIUCD.*

Stefano Menini, Giovanni Moretti, Rachele Sprugnoli and Sara Tonelli. 2020. DaDoEval @ EVALITA 2020: Same-Genre and Cross-Genre Dating of Historical Documents. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020).*

Terrence Szymanski and Gerard Lynch. 2015. UCD: Diachronic Text Classification with Character, Word, and Syntactic N-grams. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 879–883.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*, 1398:137–142.

Tim Head, Manoj Kumar, Holger Nahrstaedt, Gilles Louppe and Iaroslav Shcherbatyi. 2020. scikit-optimize/scikit-optimize (Version v0.8.1). Zenodo *http://doi.org/10.5281/zenodo.4014775.*

Vlad Niculae, Marcos Zampieri, Liviu Dinu and Alina M. Ciobanu. 2014. Temporal Text Ranking and Automatic Dating of Texts. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2:17–21.

rmassidda @ DaDoEval: Document Dating Using Sentence Embeddings at EVALITA 2020

Riccardo Massidda
Università di Pisa
r.massidda@studenti.unipi.it

Abstract

This report describes an approach to solve the DaDoEval document dating subtasks for the EVALITA 2020 competition. The dating problem is tackled as a classification problem, where the significant length of the documents in the provided dataset is addressed by using sentence embeddings in a hierarchical architecture. Three different pre-trained models to generate sentence embeddings have been evaluated and compared: USE, LaBSE and SBERT. Other than sentence embeddings the classifier exploits a bag-of-entities representation of the document, generated using a pre-trained named entity recognizer. The final model is able to simultaneously produce the required date for each subtask.

1 Introduction

To solve the DaDoEval task (Menini et al., 2020) for the EVALITA 2020 competition (Basile et al., 2020) a model should be able to assign a temporal span from a discrete set of candidates to a document, i.e. recognizing when the document was issued. As many other NLP tasks, like author identification or topic assignment, this task can be reduced to a classification problem.

The provided dataset contains documents written by the Italian statesman Alcide De Gasperi in the time span 1901-1954, labeled with the year in which they were issued. The dating task is divided into different subtasks of increasing granularity. The first subtask requires to classify a document into one of five representative periods in De Gasperi's life as identified by historians. (Table 1) The second and the third subtasks require to date a document more precisely, using a five-year span for the former and the precise year for the latter. These subtasks are referred to as the same-genre subtasks.

ID	Period description	Time span
A	Habsburg years	1901-1918
B	Beginning of political activity	1919-1926
C	Internal exile	1927-1942
D	From fascism to the Italian Republic	1943-1947
E	Building the Italian Republic	1948-1954

Table 1: Historical periods of De Gasperi's life

Other than on a blind test set kept from the same-genre dataset, the model has been also evaluated on three additional cross-genre subtasks. In this case, documents coming from a De Gasperi's epistolary archive were used to build an external blind test set. The cross-genre subtasks require to classify documents with the same increasing time granularity as the same-genre ones.

The tasks are evaluated using macro-averaged F1. Baseline results using logistic regression and tf-idf on a bag-of-word representation are provided by the task proponents in table 2.

Subtask	Macro-Average F1
Historical	0.827
Five-years	0.485
Single-year	0.126

Table 2: Proponents baseline

All of the results and the described experiments have been implemented using TensorFlow and executed on the platform Google Colab. The limitations of the platform regarding continuous usage are not negligible and had an acknowledgeable weight in multiple decisions.

In section 2 different approaches to deal with long text classification are described and the various sentence embeddings models are presented. In section 3 the peculiarities of the dataset are discussed. In section 4 the different sentence embeddings models are evaluated and compared with alternative approaches over a single subtask. In section 5 the architecture of the final model used to solve all the subtasks is described, its results are reported in section 6 and discussed in section 7.

2 Methodological survey

The use of pre-trained transformers such as BERT (Devlin et al., 2019) has remarkably improved the state of the art in many NLP tasks, text classification included. Furthermore contextual word embeddings produced by pre-trained transformers are preferable when dealing with polysemy. Documents from a wide time span could manifest lexical change, so polysemy may significantly emerge (Blank, 1999).

When dealing with text classification using the transformer model the first architectural issue is given by the length of the documents. To classify a text a special symbol is usually inserted at the start of the input sequence, then the output corresponding to that symbol is fed into a neural network to retrieve the predicted class. Since the maximum input size for a BERT transformer is 512 tokens, it is unlikely that the whole document will fit. Different architectures are available to overcome this problem.

For certain domains it has been studied that not all of the text is needed to achieve good classification accuracy. For instance Sun et al. (2020) propose to select only part of the text, like the head, or the tail or both, up to reducing the text size to fit the input layer of the transformer. The random selection of tokens inside a document has also proven to be effective for topic classification of academic papers (Liu et al., 2018).

Recently different solutions started to exploit hierarchical architectures, segmenting the text to consequently analyze it in its entirety. The use of sentences may be intuitively perceived as more meaningful than fixed-length segments. Accordingly, three different sentence embeddings solutions have been selected to be implemented and evaluated for the DaDoEval task. All of them provide pre-trained multilingual models, satisfying so the computational constraints and the task requirements.

Sentence-BERT, also known as SBERT, produces sentence embeddings by stacking a pooling layer on the top of a BERT transformer. A pre-trained BERT model is fine-tuned using Siamese networks, back-propagating over the cosine similarity of supposedly semantically related sentences. (Reimers and Gurevych, 2019) A monolingual model can be then distilled and expanded to other languages by training a student model to replicate the behavior of the teacher model, and under the assumption that the vector representation of translated sentences should coincide. (Reimers and Gurevych, 2020). The authors of SBERT published `distiluse-base-multilingual-cased`, a distilled model pre-trained on many languages including Italian.

The Universal Sentence Encoder, or USE, comprises different architectures trained on the same set of tasks to enable transfer learning for many NLP tasks with different requirements. (Cer et al., 2018) The original USE has then been expanded for multilingual applications providing two pre-trained models, a transformer and a CNN, both available on Tensorflow HUB. (Yang et al., 2019)

Lastly, the Language-agnostic BERT Sentence Embedding model, or LaBSE, produces sentence embeddings by using a fine-tuned BERT model. The LaBSE model is designed similarly to SBERT, using two sharing-weights transformers initialized by a pre-trained BERT model. The main difference lies in the datasets and the tasks used for fine-tuning. The authors report the remarkable results of LaBSE for languages unseen but somehow related to those in the training set. (Feng et al., 2020) This result may be useful to fill the gaps between contemporary Italian and the XX-century Italian language in the dataset.

3 Data Analysis

The overall dataset contains 2759 manually labeled documents of variable length written by Alcide De Gasperi during its political life. However, the development dataset provided by the proponents contains only 2210 of them, since the remaining ones are kept for the blind same-genre test set. The dataset is extremely unbalanced since the number of elements per time period varies considerably. For instance by analyzing figure 1 it is evident how some years contribute to the dataset with few documents. The lack of data for these periods remarkably impacts the overall accuracy of the learning process. The development set provided by the proposers has been split into a training set and a validation set to assess the capabilities of the different tested models. The training set was composed by sampling the 80% of the development dataset, leaving the remaining 20% to the

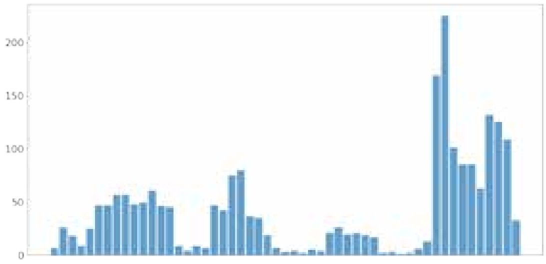


Figure 1: Number of documents per year from 1901 to 1954.

validation split. This choice reflects the proportion between the size of the provided development set and the overall dataset.

Without altering the validation split for the assessment, the training data can be augmented to contrast the unbalancing. The hierarchical solution highly increases the number of tokens that can be used to classify a document, nonetheless the number of sentences per document should be constrained under a fixed constant. When truncating a document to limit the number of sentences, the remaining part is then inserted in the dataset as a new document instead of discarding it. The data augmentation procedure described has been implemented under the assumption that the less represented years contain the longest documents. While this holds for some classes, the effect of data augmentation didn't impact on the overall distribution.

Method	Time
SBERT	223.068s
LaBSE	3364.272s
USE$_{\text{TRANS}}$	154.277s
USE$_{\text{CNN}}$	29.681s

Table 3: Time required by each sentence embedding technique to process the training set.

The tokenizer for the Italian language included in the NLTK library has been used to split each document into a list of sentences (Bird et al., 2009). The content of each sentence has been tokenized instead with a custom tokenizer for each one of the sentence embeddings techniques, since they may require different configurations and their vocabulary must be used. A common issue in this scenario is given by the rate of out-of-vocabulary tokens (Wang et al., 2019), but this hasn't been evaluated since the interfaces offered by the selected models don't offer insights over the OOV rate or other token-level statistics. The time required to produce the embeddings over the training set is reported in table 3.

4 Building blocks selection

Because of the computational limitations, many experiments have been conducted only on one subtask, relegating the others to a subsequent phase. The historical subtask has been chosen because of the better balancing of the dataset and the foreseeable and more promising results. The provided dataset has been split using stratified sampling and data augmentation in a consistent training set and a smaller validation set. The training split covers the 80% of the provided development set, leaving the remaining 20% to the validation one. All of the results are produced by averaging multiple runs, to overcome the non-deterministic and unpredictables effects of the GPUs used for training.

4.1 Truncation based classification

The first experiments used a pre-trained BERT multilingual model for text classification. To overcome the constraint over the input size the documents were truncated up to their first 512 tokens. As expected the truncation has proven to be ineffective since, even after fine-tuning, the model didn't converge on the training set for any subtask. The results aren't significant and therefore not reported.

4.2 Sentence embeddings

Once each document is represented as a sequence of sentence embeddings, two different classification models have been implemented and evaluated. The first is a Recurrent Neural Network with two bidirectional LSTM layers followed by a combination of dropout and dense layers of reducing width. The other classifier is based on the transformer architecture, where a transformer block composed of a multi-headed self-attention layer with 128 heads, dropout and layer normalization is followed by a combination of dropout and dense layers as in the previous solution.

The results of the experiments over the combination of sentence embeddings and the two classifiers are reported in table 4, showing how the combination of SBERT and the transformer-based classifier is the most adequate. With the exception of LaBSE, all the other sentence embeddings models gave better results when coupled with a

	TR			VL		
Top	Loss	Acc	F1	Loss	Acc	F1
LaBSE						
RNN	0.356	0.875	0.884	**0.663**	0.778	0.781
Trans	0.559	0.771	0.697	0.960	0.713	0.616
SBERT						
RNN	0.143	0.955	0.975	0.690	0.824	0.829
Trans	**0.060**	**0.982**	**0.987**	1.235	**0.850**	**0.851**
USE_{CNN}						
RNN	0.193	0.937	0.959	0.780	0.775	0.780
Trans	0.217	0.920	0.937	0.850	0.821	0.819
$\text{USE}_{\text{Transformer}}$						
RNN	0.105	0.969	0.978	0.780	0.815	0.823
Trans	0.192	0.923	0.972	0.773	0.822	0.830

Table 4: Results for the historical periods subtask over training and validation set using different sequence embeddings.

transformer block than with a recurrent neural network. Also, the two variants of USE manifested a more significant gap when coupled with the RNN classifier than with the transformer-based one. Finally, the performance drop of the LaBSE model may reflect a condition also explored by Reimers and Gurevych (2020), where a comparable performance gap with SBERT occurs in semantic textual similarity tasks.

4.3 Bag-of-entities

Another approach to tackle the subtasks consists of exploiting the knowledge of a pre-trained named entity recognizer. It is reasonable to suppose that the entities extracted by a document will produce a good representation for the document itself. In the context of document dating this could be meaningful by assuming that the issues discussed by the author will vary during the years, consequently influencing the entities contained. By building a vocabulary of unique entities it is possible to represent each document as a bag-of-entities, then a multi-layer dense classifier with dropout can be trained to predict the correct time span.

Named entity recognition is achieved using one pre-trained CNN for the Italian language distributed by spaCy (Honnibal and Montani, 2017). Three variants of the same model are provided but, since their differences heavily impact on the model size rather than on the performances (Table 5), the medium sized model has been chosen without further validation. Because of this it is not possible to assess how the performances of the NER alone influence the performances of the overall system.

The NER model returns for each entity a pair containing its content and a label regarding its role. It is possible to consider as a member of the entities vocabulary only the textual content or the unique pair of text and label, both methods were implemented and compared but finally only the label was chosen as representative of the entity.

	Small	Medium	Large
F1	86.57	88.54	89.40
Precision	86.85	88.76	89.56
Recall	86.29	88.33	89.24
Size	13MB	43MB	544MB

Table 5: Model size and benchmark as provided by spaCy for the Italian language pre-trained models. (Explosion.ai, 2020)

4.4 Results

The transformer classifier using sentence embeddings provided by SBERT is chosen as the final candidate since it's the best performing model on the validation set. As previously discussed, the model selection procedure only considered the first subtask because of the magnitude and the balancing of its dataset. To roughly estimate the behavior on all the subtasks both the sentence embeddings classifier and the bag-of-entities solution have been retrained from scratch on the specific subtasks labels and evaluated on the validation set. The results are reported in table 6.

		SBERT+Trans		Bag-of-entities	
Task	Baseline	TR	VL	TR	VL
Historical	0.827	0.930	**0.846**	0.997	0.841
Five-years	0.485	0.482	0.354	0.996	**0.563**
Single-year	0.126	0.086	0.040	0.990	**0.211**

Table 6: Macro-averaged F1 for all the subtasks

5 Model Architecture

It is therefore clear that both the approaches have their advantages on different subtasks. More precisely the sentence embeddings one has proven to be more effective when dealing with the historical periods subtask, while the bag-of-entities obtains better results on the finer ones. The problem of combining these two solutions is now tackled.

The trivial solution would be to hardwire in a single model the different approaches, producing so the output for the first subtask using a sentence embeddings model and for the other subtasks with

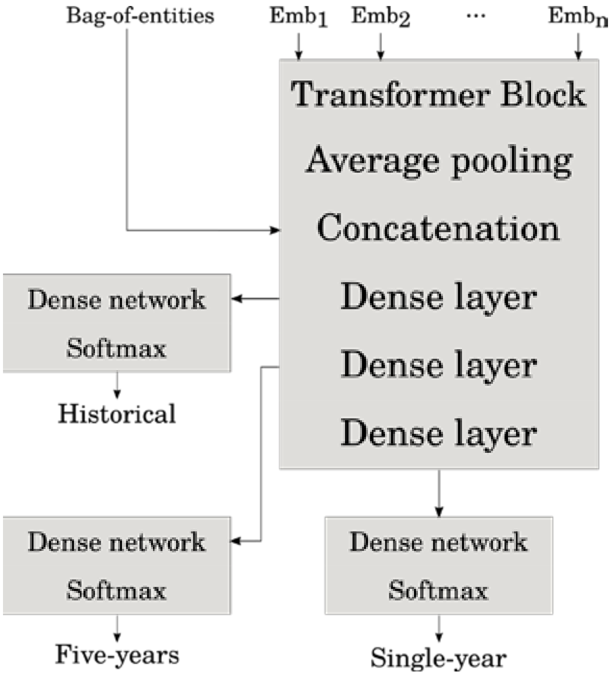


Figure 2: Architecture of the final model.

a bag-of-entities one. While this solution would be acceptable, and seemingly over the baseline according to the estimates on the validation set, it is reasonable to assume that the representations for these subtasks could be shared, improving the performances. Different variations of the same architecture are therefore evaluated on the validation set to monitor such improvement.

In the final model, the sentence embeddings produced by SBERT are fed to a transformer block containing a multi-headed self-attention layer, its output is then averaged and concatenated with the bag-of-entities representation of the document before being fed to a multi-layer neural network. The output of each layer of this network is also fed to a dedicated neural network that produces the output of each subtask. The selected order for the subtasks in the multi-layer dense classifier places the historical classification first, followed by the five-years and then the single-year classification. A graphical representation of the architecture is in figure 2.

Both the reverse of the subtasks order and the absence of hierarchy, by connecting all the classification networks directly to the transformer block, have been tested. Also, the supposed additional value of the concatenation with the entities representation has been experimentally evaluated. The results of these variations are reported in table 7, where the selected final model for the competition is on the fourth row.

		Historical		Five-years		Single-year	
BoE	Order	TR	VL	TR	VL	TR	VL
N	F	0.987	0.828	0.961	0.554	0.577	0.144
N	B	0.988	0.828	0.930	0.566	0.871	0.204
N	A	0.983	0.813	0.973	0.560	0.920	0.228
Y	F	0.991	**0.842**	0.980	**0.599**	0.852	0.236
Y	B	**0.993**	**0.842**	0.988	0.578	0.897	**0.247**
Y	A	0.991	0.820	**0.994**	0.560	0.967	0.242

Table 7: Results for the different subtasks over the training and the validation sets using different architectures. The first column refers to the use of the bag-of-entities representation in the model as in **Y**es or **N**o, the second to the order of the subtasks as in **B**ackward, **F**orward and **A**bsent.

6 Results

The model has been evaluated by using two independent test sets: same-genre and cross-genre. The first one is a blind test set, containing documents from the same source of the provided development dataset. The cross-genre set is instead an external test set, containing documents from a different source, specifically from an archive of epistolary documents of the same subject.

For each subtask two runs per test set were submitted, for brevity in table 8 only the average result of the submitted runs is reported. The model performs over the baseline in the same-genre evaluation for each subtask, also improving the performances with respect to the validation set. Instead, concerning the cross-genre evaluation, the model replicates the results of the baseline and shows a significant drop in respect to the validation set.

		Same-genre		Cross-genre	
	VL	BL	TS	BL	TS
Historical	0.842	0.827	**0.857**	0.368	0.379
Five-years	0.599	0.458	**0.609**	0.171	0.168
Single-year	0.236	0.126	**0.265**	0.020	0.055

Table 8: F1 macro-averaged results for the different subtasks over the validation set (VL), the test sets (TS) and the respective baselines (BL).

7 Conclusions

The contribution of the bag-of-entities representation was certainly helpful, but this should not overshadow the performance improvement given by the introduction of the hierarchical model. The first three rows in the already discussed table 7 report the results of the model without any contribution from the bag-of-entities representation. Whilst neither of these was elected as the best candidate, there is a remarkable improvement over the independent use of the very same building blocks of the final architecture for each subtask.

The described architecture is prone to multiple variations and only some of them have been formally evaluated and compared. Nonetheless, the selected final model was able to surpass the same-genre baseline for all of the different subtasks. Anyhow the performance drop in the cross-genre test should be interpreted as a limit to the generalization power of the chosen model. A wider exploration of the models may increase the overall performances for both the same-genre and the cross-genre tasks.

Also, targeting multiple subtasks at the same time made nontrivial the choice of a final model, therefore it has been carried out intuitively considering the results over the validation set for each subtask. A formal approach to this issue may result in a finer model selection.

Despite the discussed approximations, the use of sentence embeddings models has proven to be effective also on tasks different from the ones they were originally conceived for, and compatible with other representations such as bag-of-entities.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. "O'Reilly Media, Inc.", June. Google-Books-ID: KGIbfiiP1i4C.

Andreas Blank. 1999. Why do new meanings occur? A cognitive typology of the motivations for lexical semantic change. *Historical semantics and cognition*, 61.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder. *arXiv:1803.11175 [cs]*, April. arXiv: 1803.11175.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*, May. arXiv: 1810.04805.

Explosion.ai. 2020. Italian · spaCy Models Documentation. https://spacy.io/models/it.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic BERT Sentence Embedding. *arXiv:2007.01852 [cs]*, July. arXiv: 2007.01852.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

Liu Liu, Kaile Liu, Zhenghai Cong, Jiali Zhao, Yefei Ji, and Jun He. 2018. Long Length Document Classification by Local Convolutional Feature Aggregation. *Algorithms*, 11(8):109, August. Number: 8 Publisher: Multidisciplinary Digital Publishing Institute.

Stefano Menini, Giovanni Moretti, Rachele Sprugnoli, and Sara Tonelli. 2020. DaDoEval @ EVALITA 2020: Same-Genre and Cross-Genre Dating of Historical Documents. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv:1908.10084 [cs]*, August. arXiv: 1908.10084.

Nils Reimers and Iryna Gurevych. 2020. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. *arXiv:2004.09813 [cs]*, April. arXiv: 2004.09813.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2020. How to Fine-Tune BERT for Text Classification? *arXiv:1905.05583 [cs]*, February. arXiv: 1905.05583.

Hai Wang, Dian Yu, Kai Sun, Janshu Chen, and Dong Yu. 2019. Improving Pre-Trained Multilingual Models with Vocabulary Expansion. *arXiv:1909.12440 [cs]*, September. arXiv: 1909.12440.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Multilingual Universal Sentence Encoder for Semantic Retrieval. *arXiv:1907.04307 [cs]*, July. arXiv: 1907.04307.

DIACR-Ita: Diachronic Lexical Semantics

DIACR-Ita @ EVALITA2020: Overview of the EVALITA2020 Diachronic Lexical Semantics (DIACR-Ita) Task

Pierpaolo Basile
Dept. of Computer Science
University of Bari, Italy
`pierpaolo.basile@uniba.it`

Annalina Caputo
ADAPT Centre
School of Computing, Dublin City University
`annalina.caputo@dcu.ie`

Tommaso Caselli
CLCG
University of Groningen, Netherlands
`t.caselli@rug.nl`

Pierluigi Cassotti
Dept. of Computer Science
University of Bari, Italy
`pierluigi.cassotti@uniba.it`

Rossella Varvara
DILEF
University of Florence, Italy
`rossella.varvara@unifi.it`

Abstract

English. This paper describes the first edition of the “Diachronic Lexical Semantics” (DIACR-Ita) task at the EVALITA 2020 campaign. The task challenges participants to develop systems that can automatically detect if a given word has changed its meaning over time, given contextual information from corpora. The task, at its first edition, attracted 9 participant teams and collected a total of 36 submission runs.

1 Background and Motivation

The Diachronic Lexical Semantics (DIACR-Ita) task focuses on the automatic recognition of lexical semantic change over time, combining together computational and historical linguistics. The aim of the task can be shortly described as follows: given contextual information from corpora, systems are challenged to detect if a given word has changed its meaning over time.

Word meanings can evolve in different ways. They can undergo *pejoration* or *amelioration* (when meanings become respectively more negative or more positive) or they can be object of *broadening* (also referred to as *generalization* or *extension*) or *narrowing* (also known as *restriction* or *specialization*). For instance, the English word *dog* is a clear case of broadening, since its more general meaning came from the late Old English “dog of a powerful breed” (Traugott, 2006). On the contrary, the Old English word *deor* with the general meaning of “animal” became *deer* in present-day English. Semantic changes can be further classified on the basis of the cognitive process that originated them, i.e. either from *metonymy* or *metaphor*. Lastly, it is possible to distinguish among changes due to language-internal or language-external factors (Hollmann, 2009). The latter usually reflects a change in society, as in the case of technological advancements (e.g. *cell*, from the meaning of “prisoner cell” to “cell phone”).

The problem of the automatic analysis of lexical semantic change is gaining momentum in the Natural Language Processinng (NLP) and Computational Linguistics (CL) communities, as shown by the growing number of publications on the diachronic analysis of language and the organisation of related events such as the 1st International Workshop on Computational Approaches to Historical Language Change[1] and the project “Towards Computational Lexical Semantic Change Detection”[2]. Following this trend, SemEval 2020 has hosted for the first time a task on automatic recognition of lexical semantic change: the SemEval 2020 Task 1 - Unsupervised Lexical Semantic Change Detection[3] (Schlechtweg et al.,

---

[1] `https://languagechange.org/events/2019-acl-lcworkshop/`

[2] `https://languagechange.org/`

[3] `https://competitions.codalab.org/competitions/20948`

2020). While this task targets a number of different languages, namely Swedish, Latin, and German, Italian is not present.

Many are the existing approaches, data sets, and evaluation strategies used to detect semantic change, or drift. Most of the approaches rely on diachronic word embeddings, some of these are created as post-processing of static word embeddings, such as Hamilton et al. (2016); while others create dynamic word embeddings where vectors share the same space for all time periods (Del Tredici et al., 2016; Yao et al., 2018; Rudolph and Blei, 2018; Dubossarsky et al., 2019). Recent work exploits word sense induction algorithms to discover semantic shifts (Tahmasebi and Risse, 2017; Hu et al., 2019) by analyzing how induced senses change over time. Finally, Gonen et al. (2020) propose a simple approach based on the neighbors' intersection between two corpora. The neighborhood of a word is separately computed in each corpus, then the intersection is exploited to compute a measure of the semantic shift. The neighborhood in each corpus can be computed using the cosine similarity between word embeddings built on the same corpus without using vectors alignment. A more complete state of the art is described in a critical and concise way in the latest surveys (Tahmasebi et al., 2018; Kutuzov et al., 2018; Tang, 2018).

Almost all of the previously mentioned methods use English as the target language for the diachronic analysis, leaving the other languages still under-explored. To date, only one evaluation has been carried out on Italian using the Kronos-it dataset (Basile et al., 2019).

The DIACR-Ita task at the EVALITA 2020 campaign (Basile et al., 2020b) fosters the implementation of new systems purposely designed for the Italian language. To achieve this goal, a new dataset for the evaluation of lexical semantic change on Italian has been developed based on the "L'Unità" corpus (Basile et al., 2020a). This is the first Italian dataset manually annotated with semantic shifts between two different time periods.

2 Task Description

The goal of DIACR-Ita is to establish if a set of *target* words change their meaning across two time periods, T_1 and T_2, where T_1 precedes T_2.

Following the SemEval 2020 Task 1 settings, we focus on the comparison of two time periods. In this way, we tackle two issues:

1. We reduce the number of time periods for which data has to be annotated;
2. We reduce the task complexity, allowing for the use of different models' architectures, and thus widening the range of potential participants.

During the test phase, participants have been provided with two corpora C_1 and C_2 (for the time periods T_1 and T_2, respectively), and a list of target words. For each target word, systems have to decide whether the word changed or not its meaning between T_1 and T_2, according to its occurrences in sentences in C_1 and C_2. For instance, the meaning of the word "imbarcata" is known to have expanded[4], i.e, it has acquired a new sense, from T_1 to T_2. This will be reflected in different occurrences of the word usage in sentences between C_1 and C_2.

The task is formulated as a closed task, i.e. participants must train their model only on the data provided in the task. However, participants may rely on pre-trained word embeddings, but they cannot train embeddings on additional diachronic Italian corpora, they can use only synchronic corpora.

3 Data

This section provides an overview of the datasets that were made available to the participants in the two different stages of the evaluation challenge, namely **trial** and **test**.

3.1 Trial data

The trial phase corresponds to the evaluation window in which the participants have to build their systems before the official test data are release. The following data were provided:

- An example of 5 trial target words for which predictions are needed;
- An example of gold standard for the trial target words;
- A sample submission file for the trial target words;

---

[4] The word originally referred to an acrobatic manoeuvre of aeroplanes. Nowadays, it is also used to refer to the state of being deeply in love with someone.

- Two trial corpora that participants could use to develop their models and check the compliance of the generated output to the required format;
- An evaluation and some additional utility scripts for managing corpora.

Trial data do not reflect the actual data from C_1 and C_2. The sample training corpora and target words were artificially built just to provide an example of the data format for developing their systems. Since the training corpus is publicly available on the Internet, we decided not to release these data during the trial phase to prevent participants from identifying the source data and consequently potential set of target words.

3.2 Test data

For the test phase, the following data were provided:

- A diachronic split of the "L'Unità" corpus into the two sub-corpora, C_1 and C_2, each belonging to a specific time period;
- 18 target words, among which 6 were identified as target of semantic meaning change between the two time periods.

Corpus Creation The "L'Unità" diachronic corpus (Basile et al., 2020a) is a collection of documents extracted from the digital archive of the newspaper "L'Unità".[5]

For the task, the corpus has been initially split into two sub-corpora, C_1, corresponding to the time period $T_1 = [1945 - 1970]$, and C_2, corresponding to the time period $T_2 = [1990 - 2014]$.

To facilitate participants in the closed-task formulation, the corpora were provided in a preprocessed format. In particular, we adopted a tab separated format, with one token per line. For each token, we provided its corresponding part-of-speech and lemma. Sentences are separated by empty lines. Data were pre-processed with UDPipe[6] using the ISDT-UD v2.5 model. An example of the data format is illustrated below.

```
Questa  PRON questo
è  AUX essere
una DET uno
frase  NOUN frase
. PUNCT .

Questa  PRON questo
è  AUX essere
un′ DET uno
altra  ADJ altro
frase NOUN frase
. PUNCT .
```

Participants are free to combine the available information as they want. Furthermore, to facilitate the generation of word embeddings, we made available a script for generating a format containing one sentence per line.

The whole "L'Unità" diachronic corpus has been built, cleaned and annotated automatically. This process consisted of several steps, namely:

Step 1: Downloading All PDF files are downloaded from the source site and stored into a folder structure that mimics the publication year of each article.

Step 2: Text extraction The text is extracted from the PDF files by using the Apache Tika library.[7] First, the library tries to extract the embedded text if present in the PDF. If this process fails, the internal OCR system is used. It is important to notice that during this step several OCR errors may occur due to different reasons. The processing of the early years of publications, i.e., between 1945–1948, represented a non trivial challenge for the extraction of the textual data. In particular, we noticed that the page format had a major impact on the quality of the OCR. In these period, the newspaper has quite an unconventional format where a few large pages contain many articles scattered into several columns. This affected the performance of the OCR due to its failure in properly identifying the column boundaries.

Step 3: Cleaning In this step, we try to fix some text extraction issues. We identified two lines of actions, the first dealing with paragraph splits and the second with noisy text. In the text extraction process, paragraphs are separated by means of an empty line. However, word hyphenation can trigger errors in the paragraph segmentation phase by wrongly adding empty lines. We addressed this issue by reconstructing the paragraph on a single text line, thus ensuring that empty lines are

---

[5] https://archivio.unita.news/

[6] http://lindat.mff.cuni.cz/services/udpipe/run.php

[7] https://tika.apache.org/

only used to delimit the actual paragraphs. In our case, noisy text corresponds to tokens whose composing characters are wrongly interpreted by the OCR mixing together alphabetical characters with numbers or symbols. Two heuristics were implemented to limit the amount of noisy text. The first heuristic requires that paragraphs must contain at least five tokens composed by only alphabetical characters. The second heuristic requires that at least 60% of each paragraph must contain words that are attested in a dictionary. For this, we did not use a reference dictionary, but we automatically created it by extracting tokens from the Paisà corpus (Lyding et al., 2014). Numbers were excluded and only alphabetical strings were retained. The output of the cleaning process is a plain text file for each year where each paragraph is separated by an empty line.

Step 4: Processing All plain text files produced by the cleaning step are processed by a Python script that splits each paragraph into sentences and analyses each sentence with UDPipe [8] ISDT-UD v2.5 model. In this way, we obtain tokens, part-of-speech tags, and lemmas. The processed data are then stored in a vertical format as illustrated is Section 3.

After these preparation steps, the valid and retained data for the task span over a temporal period between 1948 and 2014. We revised the initial split of the two sub-corpora as follows: C_1 ranges between $T_1 = [1948 - 1970]$, and C_2 between $T_2 = [1990 - 2014]$. Table 1 illustrates the distributions of the tokens across the two time periods for the sub-corpora. The difference in the number of tokens between C_1 and C_2 reflects differences in the trends in the number of daily published articles, due to cheaper printing costs and the availability of new technologies such as the World Wide Web.

Corpus	Period	#Tokens
L'Unità	1948-1970	52,287,734
L'Unità	1990-2014	196,539,403

Table 1: Official Training Corpora: Occurrence of Tokens.

Creation of the Gold Standard The selection of the target words that compose the Gold Standard data required a manual annotation. Identifying words that have undergone a semantic change is not an easy task. To boost the identification of candidate target words, we adopted a semi-automatic method. In the following paragraphs we illustrate in detail our approach.

Step 1: Selection of candidate words. The initial selection of potential candidate words was based on Kronos-IT (Basile et al., 2019). Kronos-IT is a dataset for the evaluation of semantic change point detection algorithms for the Italian language automatically built by using a web scraping strategy. In particular, it exploits the information presents on the online dictionary "Sabatini Colletti"[9] to create a pool of words that have undergone a semantic change. In the dictionary, some lemmas are tagged with the year of the first attestation of its sense. In some cases, associated with the lemma there are multiple years attesting the introduction of new senses for that word. Kronos-IT uses this information to identify the set of semantic changing words. We retained those words that were predicted to have changed their meaning after 1970, so as to match the temporal periods of the sub-corpora. In this way, we obtained 106 candidate lemmas.

Step 2: Filtering candidate targets. A challenging issue is the attestation of the potential candidate words in both sub-corpora with a relatively high number of occurrences to account for different contexts of use. Frequency, indeed, plays a quite relevant role for the task: infrequent tokens must be discarded because they affect the quality of word representations. The initial list of candidate targets has been further cleaned by removing all tokens that occur less than 20 times in each corpora. Moreover, we conducted a further analysis by manually inspecting some randomly sampled lemma contexts. The aim of this analysis was to remove targets for which the lemmas occurrences are affected by OCR errors. This analysis was performed by the means of the Sketch Engine[10], in particular we analyze concordances of the target word in order to discover OCR errors. One of such words was "toro" derived from the mistaken

---

[8] `http://lindat.mff.cuni.cz/services/udpipe/run.php`

[9] `https://dizionari.corriere.it/dizionario_italiano/`

[10] `https://www.sketchengine.eu/`

OCR of "loro". At the end of this process, we obtained a list of 27 candidate targets for the annotation.

Step 3: Manual Annotation. For each target, we randomly extracted up to 100 sentences from each of the sub-corpus[11]. Each sentence was then annotated by two annotators: they were asked to assign each occurrence to one of the meaning of the lemma according to those reported in the Sabatini-Coletti dictionary. In case the meaning of the word in a sentence was not present in the list of senses reported in the reference dictionary, the annotators were allowed to add the sense to the word. In total, we annotated 2,336 occurrences of the candidate target words.

Step 4: Annotation check. All cases of disagreement were collectively discussed among all of the annotators to reach a final decision. We observed that some disagreements were also due to a biased interpretation of the context of occurrence by one of the annotators. These cases mainly concerned short ambiguous sentences that prevented a clear identification of the word meaning. As a result of this step, a few candidates were removed from the pool of candidates because occurring in too ambiguous context.

Step 5: Creation of the gold standard. We retained as valid instances of lexical semantic change all those targets that had occurrences of one specific sense only in T_2, and never in T_1. In other words, in the context of this task, a valid lexical semantic change corresponds to the acquisition of a new meaning by a target word. Out of the 23 candidate target words, only 6 of them show a semantic change in T_2. All the other targets did not show a diachronic meaning change. In the final Gold Standard, we kept 12 candidate target words that did not change meaning obtaining a final set of 18 target words.

The Gold Standard contains 18 targets listed as lemmas, one lemma per line, with an accompanying label to mark whether the lemmas has undergone semantic change (label 1) or not (label 0). Participants were given a file containing the 18 target lemmas, one per each line, without annotation. The expected system output is a modification of this file where the participant had to annotate each target lemma with the system prediction (0 or 1).

4 Evaluation

The task is formulated as a binary classification problem. Systems predictions are evaluated against the change labels annotated in the Gold Standard by using accuracy.

The test set (G) contains both positive (P) and negative (N) examples, i.e. $G = P \cup N$. For example:

$$P = \{pilotato, lucciola, ape, rampante\}$$

$$N = \{brama, processare\}$$

Negative words are those that did not undergo a change in their meaning. Systems' predictions involve both positive and negative classified targets $Pr = Pr_{pos} \cup Pr_{neg}$. Then, true positives (positive targets classified as positive) are $TP = P \cap Pr_{pos}$, true negatives (negative targets classified as negative) are $TN = N \cap Pr_{neg}$, false negatives (positive targets classified as negative) are $FN = P \cap Pr_{neg}$ and false positives (negative targets classified as positive) are $FP = N \cap Pr_{pos}$. We can then compute the accuracy as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

4.1 Baselines

We provided two baseline models:

- Frequencies: The absolute value of the difference between the word frequencies in the two sub-corpora;
- Collocations: For each word, we build two vector representations consisting of the Bag-of-Collocations related to the two different time periods (T_0 and T_1). Then, we compute the cosine similarity between the two BoCs. It is the same approach evaluated in (Basile et al., 2019).

In both baselines, we use a threshold to predict if the word has changed its meaning. While for the frequencies, a change is detected when the difference is higher than the average. For the collocations a semantic change occurs when the similarity between the two time periods drops under the average plus the variance. Both the average and the variance are computed on the set of target words.

---

[11] This means that in case a target words occurs less than 100 times, all occurrences were annotated.

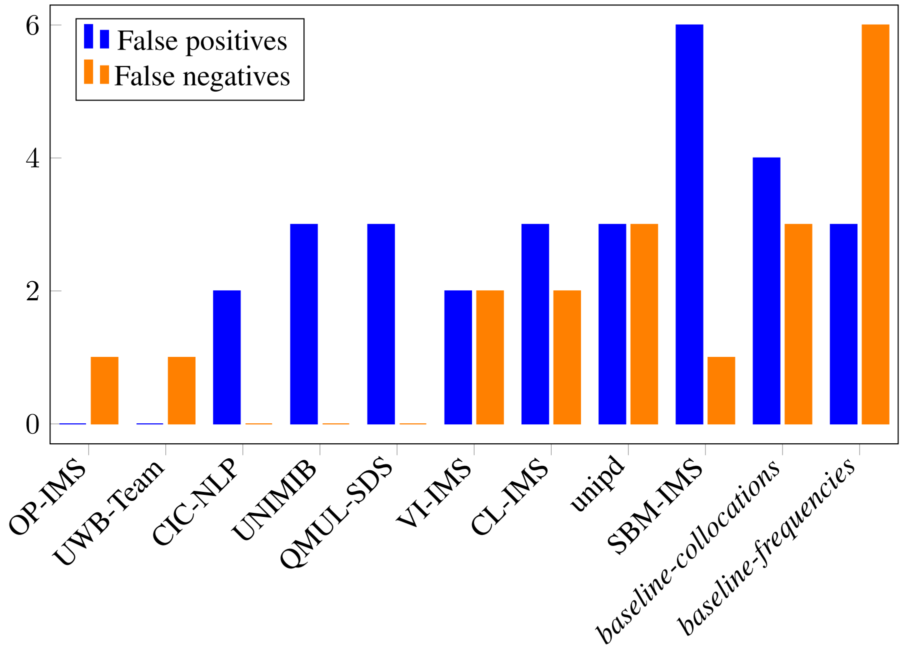


Figure 1: Number of false positives and false negatives for each system.

System	Type
OP-IMS	Post-alignement
UWB Team	Post-alignement
CIC-NLP	PoS tag features
UNIMIB	Jointly alignment
QMUL-SDS	Jointly alignment
VI-IMS	Jointly alignment
CL-IMS	Contextual Embeddings
unipd	Contextual Embeddings
SBM-IMS	Graph

Table 2: Systems types.

5 Systems

21 teams registered to the DIACR-Ita task. However, 9 teams participated in the final task for a total of 36 submitted runs. Based on the algorithms employed, we can group systems into four categories: Post-alignment, Joint Alignment, Contextual Embeddings, Graph-based and PoS tag features (see Table 2). The first two classes are characterised by the type of alignment used. Post-alignment systems first train static word embeddings for each time periods, and then align them. Joint Alignment systems train word embeddings and jointly align vectors across all time slices. Contextual Embeddings systems use contextualized embeddings, such as BERT (Devlin et al., 2019); while Graph-based systems rely on graph algorithms. PoS tag features system rely on the distribution of targets PoS tags across the two time periods. The majority of participating systems use cosine distance as a measure of semantic change, i.e. compute the cosine distance between the vectors of the target lemmas among time periods. Other systems use the Average Pairwise Cosine Distance or the Average Canberra Distance, since the cosine distance does not fit contextual embeddings representations. The last group of systems uses graph-based measures.

We report a short description of each team (best submission) as follows:

OP-IMS (Kaiser et al., 2020) This team uses Skipgram model with Negative sampling (SGNS) to compute word embeddings, the resulting matrices are mean-centred. Word embeddings are aligned using Orthogonal Procrustes. They choose cosine similarity to compare vectors of different word spaces and a threshold based on mean and standard deviation to classify target words.

UWB Team (Pražák et al., 2020) The team maps semantic spaces using linear transformations, such as Canonical Correlation Analysis and Orthogonal Transformation and cosine similarity as a measure to decide if a target word is stable or not. They use a threshold based on mean.

CIC-NLP (Angel et al., 2020) This team analyses the Part-Of-Speech distribution over the two corpora and create vectors with information about the most common word POS-tags. Then, they obtain a score using pairs of vectors of the two time periods and the sum of Euclidean, Manhattan and cosine distance. They rank targets in discerning order. Finally, they label first upper-third targets as changed words.

UNIMIB (Belotti et al., 2020) The team creates temporal word embeddings using Temporal Word Embeddings with a Compass (TWEC) (Di Carlo et al., 2019). They use the move measure, i.e. a weighted linear combination of the cosine and Local Neighbors, introduced by (Hamilton et al., 2016). They label targets as stable if the move measure is greater than 0.7.

QMUL-SDS (Alkhalifa et al., 2020) The team uses TWEC (Di Carlo et al., 2019) to compute temporal word embeddings with TWEC C-BoW model (Continuous Bag of Words) default settings. They use a cosine similarity as measure of change and a threshold based on mean.

VI-IMS The team uses SGNS to create word embeddings exploiting Vector Initialization (Kim et al., 2014). They use cosine distance as a measure of semantic change and a threshold based on the mean and the standard deviation to classify targets words.

CL-IMS (Laicher et al., 2020) The team creates word vectors using different combinations of the first and last four layers of BERT. They rank targets according to Average Pairwise Cosine Distance, and label the first 7 targets as changed words.

unipd (Benyou et al., 2020) This team uses contextualised word embeddings and an linear combination of distances metrics to measure semantic change, namely Euclidean Distance, Average Canberra distance, Hausdorff distance, as well as Jensen–Shannon divergence between cluster distributions. They rank targets according to the score obtained, and label the first half as changed words.

SBM-IMS The team compute token vectors using BERT. They create a graph where the vertices are the vectors extracted from BERT, while the edges are the cosine distance between word vectors. They cluster the graph with Weighted Stochastic Block Model. Then, they consider the number of incoming edges from the first and second period as a measure of semantic change.

Team	Accuracy
OP-IMS	0.944
UWB Team	0.944
CIC-NLP	0.889
UNIMIB	0.833
QMUL-SDS	0.833
VI-IMS	0.778
CL-IMS	0.722
unipd	0.667
SBM-IMS	0.611
baseline-collocations	0.611
baseline-frequencies	0.500

Table 3: Results.

6 Results

Table 3 reports the final results. The best result has been achieved by two systems: *OP-IMS* and *UWB-Team*. Both systems exploit post-alignment strategy. The second system *CIC-NLP* uses an approach based on PoS tag features. QMUL-SDS and VI-IMS are based on joint alignment, while *unipd* and *SBM-IMS* use contextual embeddings. The last system *SBM-IMS* is the only graph-based approach. Moreover, we report both false negative and false positives in Figure 1. Both post-alignment systems share the same unique false negative: the target “tac”, while *CIC-NLP* detects two false positives. Joint-alignment systems have a number of false positives higher or at least equal to the number of false negatives. *CL-IMS* and *unipd* produce respectively 2 and 3 false negatives and both misclassify three stable words. The only graph-based approach, *SBM-IMS*, reports the highest number of false positives. In conclusion, the results show that systems based on post/joint alignment and PoS tag features achieve the best performance, while contextual embeddings do not perform as good in this type of task. However all the systems outperform both the baselines.

7 Conclusions

We proposed for the first time the “Diachronic Lexical Semantics” (DIACR-Ita) task. The goal of the task is to develop systems able to automatically detect if a given word has changed its meaning over time, given contextual information from corpora. We created two corpora for two different time periods T_1 and T_2, and we manually annotated a set of target words that change/do not change meaning across these two periods. This is the first Italian dataset of this type. 9 teams participated in the task for a total of 36 submitted runs. All the systems are able to outperform the two baselines. The results suggests that methods based on post-alignment are the most suitable for this type of task, resulting in better performance even when compared to contextual embedding methods, such as BERT.

References

Rabab Alkhalifa, Adam Tsakalidis, Arkaitz Zubiaga, and Maria Liakata. 2020. QMUL-SDS @ DIACR-Ita: Evaluating Unsupervised Diachronic Lexical Semantics Classification in Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Jason Angel, Carlos A. Rodriguez-Diaz, Alexander Gelbukh, and Sergio Jimenez. 2020. CIC-NLP @ DIACR-Ita: POS and Neighbor Based Models for Lexical Semantic Change in Diachronic Italian Corpora. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Pierpaolo Basile, Giovanni Semeraro, and Annalina Caputo. 2019. Kronos-it: A dataset for the Italian semantic change detection task. In *CEUR Workshop Proceedings*, volume 2481.

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Casotti, and Rossella Varvara. 2020a. A Diachronic Italian Corpus based on “L’Unità”. In *CEUR Workshop Proceedings*.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020b. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Federico Belotti, Federico Bianchi, and Matteo Palmonari. 2020. UNIMIB @ DIACR-Ita: Aligning Distributional Embeddings with a Compass for Semantic Change Detection in the Italian Language. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Wang Benyou, Emanuele Di Buccio, and Massimo Melucci. 2020. University of Padova at DIACR-Ita. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Marco Del Tredici, Malvina Nissim, and Andrea Zaninello. 2016. Tracing metaphors in time through self-distance in vector spaces. In *CEUR Workshop Proceedings*. 3rd Italian Conference on Computational Linguistics, CLiC-it 2016 and 5th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian, EVALITA 2016 ; Conference date: 05-12-2016 Through 07-12-2016.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Valerio Di Carlo, Federico Bianchi, and Matteo Palmonari. 2019. Training temporal word embeddings with a compass. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6326–6334.

Haim Dubossarsky, Simon Hengchen, Nina Tahmasebi, and Dominik Schlechtweg. 2019. Time-Out: Temporal Referencing for Robust Modeling of Lexical Semantic Change. In *57th Annual Meeting of the Association for Computational Linguistics*, pages 457–470. Association for Computational Linguistics (ACL), sep.

Hila Gonen, Ganesh Jawahar, Djamé Seddah, and Yoav Goldberg. 2020. Simple, interpretable and stable method for detecting words with usage change across corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 538–555, Online, July. Association for Computational Linguistics.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. In *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, volume 3, pages 1489–1501, may.

Willem Hollmann. 2009. Semantic change. In *English Language: Description, Variation and Context*, pages 301–313. Basingstoke: Palgrave.

Renfen Hu, Shen Li, and Shichen Liang. 2019. Diachronic Sense Modeling with Deep Contextualized Word Embeddings: An Ecological View. In *57th Annual Meeting of the Association for Computational Linguistics*, pages 3899–3908.

Jens Kaiser, Dominik Schlechtweg, and Sabine Schulte Im Walde. 2020. OP-IMS @ DIACR-Ita: Back to the Roots: SGNS+OP+CD still rocks Semantic Change Detection. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Yoon Kim, Yi-I Chiu, Kentaro Hanaki, Darshan Hegde, and Slav Petrov. 2014. Temporal analysis of language through neural language models. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 61–65.

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: a survey. *27th International Conference on Computational Linguistics*.

Severin Laicher, Dominik Schlechtweg, Gioia Baldissin, Enrique Castaneda, and Sabine Schulte Im Walde. 2020. CL-IMS @ DIACR-Ita: Volente o Nolente: BERT does not outperform SGNS on Semantic Change Detection. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Verena Lyding, Egon Stemle, Claudia Borghetti, Marco Brunello, Sara Castagnoli, Felice Dell'Orletta, Henrik Dittmann, Alessandro Lenci, and Vito Pirrelli. 2014. The paisa' corpus of italian web texts. In *9th Web as Corpus Workshop (WaC-9)@ EACL 2014*, pages 36–43. EACL (European chapter of the Association for Computational Linguistics).

Ondřej Pražák, Pavel Přibáň, , and Stephen Taylor. 2020. UWB @ DIACR-Ita: Lexical Semantic Change Detection with CCA and Orthogonal Transformation. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Maja Rudolph and David Blei. 2018. Dynamic Embeddings for Language Evolution. In *WWW '18: Proceedings of the 2018 World Wide Web Conference*, pages 1003–1011. Association for Computing Machinery (ACM).

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. Semeval-2020 task 1: Unsupervised lexical semantic change detection. *arXiv preprint arXiv:2007.11464*.

Nina Tahmasebi and Thomas Risse. 2017. Finding IndividualWord Sense Changes and their Delay in Appearance. In *International Conference Recent Advances in Natural Language Processing*, pages 741–749. Assoc. for Computational Linguistics Bulgaria, nov.

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of Computational Approaches to Lexical Semantic Change. *1st International Workshop on Computational Approaches to Historical Language Change 2019*.

Xuri Tang. 2018. A state-of-the-art of semantic change computation. *Natural Language Engineering*, 24(5):649–676, sep.

Elizabeth Closs Traugott. 2006. Semantic change: Bleaching, strengthening, narrowing, extension. In *Encyclopedia of Language and Linguistics*. Elsevier.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic word embeddings for evolving semantic discovery. In *WSDM 2018 - Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, volume 2018-Febua, pages 673–681.

University of Padova @ DIACR-Ita [*]

Benyou Wang and **Emanuele Di Buccio** and **Massimo Melucci**
Department of Information Engineering
University of Padova, Padova, Italy
`{wang,dibuccio,melo}@dei.unipd.it`

Abstract

Semantic change detection task in a relatively low-resource language like Italian is challenging. By using contextualized word embeddings, we formalize the task as a distance metric for two flexible-size sets of vectors. Various distance metrics like average Euclidean Distance, average Canberra distance, Hausdorff distance, as well as Jensen–Shannon divergence between cluster distributions based on K-means clustering and Gaussian mixture model are used. The final prediction is given by an ensemble of top-ranked words based on each distance metric. The proposed method achieved better performance than a frequency and collocation based baselines.

1 Introduction

Lexical Semantic Change detection aims at identifying words that change meaning over time; this problem is of great interest for NLP, lexicography, and linguistics. A semantic change detection task in English, German, Latin, and Swedish was proposed by Schlechtweg et al. (2020). Recently, Basile et al. (2020a) organized a lexical semantic change detection task in Italian called DIACR-Ita at EVALITA 2020 (Basile et al., 2020b). This technical report describes the methodology designed and developed by the University of Padova for the participation to DIACR-Ita.

Some previous approaches for semantic change modelling were based on static word embedding, where word vectors were trained for each time-stamped corpus and then were aligned, e.g. by orthogonal projections (Hamilton et al., 2016), vector initialization (Kim et al., 2014), and temporal teferencing (Dubossarsky et al., 2019). This work relies on contextualized word embeddings as the basic word representation component (Hu et al., 2019), since they have been shown to be effective in many NLP tasks including document classification and question answering. The methods relying on contextualized word embeddings performed worse than those based on static word embedding in Semantic Change detection tasks in many languages (Kutuzov and Giulianelli, 2020; Pömsl and Lyapin, 2020; Schlechtweg et al., 2020; Vani et al., 2020; Giulianelli et al., 2020; Giulianelli, 2019). However, it is our opinion that the use of contextualized word embeddings for this task is worth investigating because (1) they have highly expressive power as demonstrated in many downstream tasks e.g., document classification and question answering, and (2) they could handle fine-grained representations of individual context at the level of tokens.

By using contextualized word embedding, each word in a specific sentence is represented as a vector depending on the neighboring words which form the context of the word; a word appearing many times in a corpus is therefore represented as a set of vectors since one vector corresponds to each occurrence). In this paper, semantic change detection is addressed by computing the distance between two flexible-size sets consisting of vectors with respect to two time-stamped corpora. We investigated several distance metrics: average Euclidean Distance, average Canberra distance, and Hausdorff distance. Our methodology also relies on a clustering algorithm (e.g. K-means clustering and Gaussian Mixture Model) on the joint set and calculates a Jensen–Shannon divergence between cluster distributions in the two sub-corpora. We aggregate top-ranked words based on each distance metric as the final prediction. The proposed method achieved better perfor-

---

mance than frequency and collocation based baselines and finally ranked the 8-th among 9 participanting teams.

2 Problem definition

Unlike the static word embedding like Word2vec (Mikolov et al., 2013) [1], contextualized word embeddings like ELMO (Peters et al., 2018) and BERT (Devlin et al., 2018) generate word representation based on the context of a word which does in this way not have a unique mapping with a fixed word vector.

Let us denote a corpus with m sentences as $\mathcal{C}$. In this paper, $\mathcal{C}$ is related to a time span t because of the task characteristics; however, the corpus can be tailored to any specific aspect, e.g. a specific domain such as news or books. For a word w_i appearing in $\mathcal{C}$, its contextualized word representation in the k-th sentence [2] is denoted by $e_{i,k}^{(\mathcal{C})}$. The word representation in the corpus is a set

$$\Phi_i^{\mathcal{C}} = \{e_{i,1}^{(\mathcal{C})}, e_{i,2}^{(\mathcal{C})}, \cdots e_{i,k}^{(\mathcal{C})}, \cdots, e_{i,m}^{(\mathcal{C})}\} \tag{1}$$

To examine whether a word w_i exhibits a semantic change between two corpora $\mathcal{C}_1$ (in t_1) and $\mathcal{C}_2$ (in t_2), we check the difference between two sets $\Phi_i^{\mathcal{C}_1}$ and $\Phi_i^{\mathcal{C}_2}$. Let l_i be a human-annotated label indicating the semantic change degree; l_i usually ranges from 0 to 1, where 1 denotes a full semantic change. Let D be the dimension of the word vector. We define the distance metric as a function

$$f : \{\mathbb{R}^D\}^m, \{\mathbb{R}^D\}^n \to \mathbb{R}. \tag{2}$$

to obtain a semantic change degree based on the representation of a word in two corpora denoted as $\Phi_i^{\mathcal{C}_1}, \Phi_i^{\mathcal{C}_2}$. When labels are binary, one may simply use a threshold on the values of $f(\cdot,\cdot)$ to predict the binary label. Let δ be a function to generate a binary output, e.g., based on a hand-crafted threshold. We can predict whether w_i exhibits a semantic change between $\mathcal{C}_1$ and $\mathcal{C}_2$ as follows

$$\bar{l}_i = \delta(f(\Phi_i^{\mathcal{C}_1}, \Phi_i^{\mathcal{C}_2})) \tag{3}$$

where $\bar{l}_i$ is the predicted binary label.

In conclusion, in our work the semantic change detection task is formalized as follows

$$\arg\max_{f,\delta} \sum_{w_i} \left(\delta(f(\Phi_i^{\mathcal{C}_1}, \Phi_i^{\mathcal{C}_2})) == l_i \right) \tag{4}$$

Since this is a closed task, we may not have enough annotated samples to train a f using gradient descent. Therefore, a well-selected f will be crucial.

3 Methodology

3.1 Contextualized Word Embedding

Using contextualized word embeddings like ELMO and BERT has be shown to improve performance in various downstream tasks due to its expressive power for words. In this paper, we use a multilingual-BERT[3]. Uncased models are adopted since we assume that semantic change detection is insensitive to word case. All models are in *base* settings with 12 layers, 12 heads, and a hidden state dimension of 768. Only last-layer output of BERT is used as word representation.

3.2 Measuring Semantic Change Degree

3.2.1 Distance-based methods

In this section, we introduce various methods to calculate the semantic change degree.

Average Geometric Distance. Average Geometric Distance (AGD) (also can be seen in (Kutuzov and Giulianelli, 2020; Giulianelli, 2019)) is defined as below:

$$\mathrm{AGD}(\Phi_i^{\mathcal{C}_1}, \Phi_i^{\mathcal{C}_2}) = \frac{1}{mn} \sum_{\boldsymbol{x} \in \Phi_i^{\mathcal{C}_1}, \boldsymbol{y} \in \Phi_i^{\mathcal{C}_2}} d(\boldsymbol{x}, \boldsymbol{y})$$

The distance function $d(\cdot,\cdot)$ can be the *Euclidean Distance* [4], the *Canberra distance* (Lance and Williams, 1966) [5] or any distance function. In this paper, we also use the negative cosine similarity as a normalized distance metric.

Hausdorff distance. Hausdorff distance (Rockafellar and Wets, 2009) is denoted as HD in short and is generally used to measure the distance between two non-empty sets, namely,

$$\begin{aligned}\mathrm{HD}(\Phi_i^{\mathcal{C}_1}, \Phi_i^{\mathcal{C}_2}) = \max(&\sup_{\boldsymbol{x} \in \Phi_i^{\mathcal{C}_1}} \inf_{\boldsymbol{y} \in \Phi_i^{\mathcal{C}_2}} ||\boldsymbol{x} - \boldsymbol{y}||_2, \\ &\sup_{\boldsymbol{x} \in \Phi_i^{\mathcal{C}_2}} \inf_{\boldsymbol{y} \in \Phi_i^{\mathcal{C}_1}} ||\boldsymbol{x} - \boldsymbol{y}||_2)\end{aligned} \tag{5}$$

---

[1] An overview on word vectors is in Wang et al. (2019).

[2] If a word appears in a sentence more than once, we take the average.

[3] `https://storage.googleapis.com/bert_models/2018_11_03/multilingual_L-12_H-768_A-12.zip`.

[4] Euclidean Distance : $d(\boldsymbol{x}, \boldsymbol{y}) = ||\boldsymbol{x} - \boldsymbol{y}||_2$

[5] Canberra distance is a normalized version of the Manhattan distance, $d(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{D} \frac{|x_i - y_i|}{|x_i| + |y_i|}$

3.2.2 Clustering-based Methods

By clustering the union set between $\Phi_i^{\mathcal{C}_1}$ and $\Phi_i^{\mathcal{C}_2}$ in K clusters/categories, we obtained the category distributions $\boldsymbol{p}, \boldsymbol{q}$ for $\Phi_i^{\mathcal{C}_1}$ and $\Phi_i^{\mathcal{C}_2}$, respectively. We adopted two commonly used clustering methods: the K-means clustering method and the Gaussian Mixture Model method. As for the distance between distributions, we adopted the Jensen–Shannon Divergence (JSD), which is a symmetrized and smoothed version of the Kullback–Leibler divergence:

$$\text{JSD} = \frac{1}{2}\text{KL}(\boldsymbol{p}, \boldsymbol{q}) + \frac{1}{2}\text{KL}(\boldsymbol{q}, \boldsymbol{p})$$

where $\text{KL}(\boldsymbol{p}, \boldsymbol{q}) = \sum_{i=1}^{K} \boldsymbol{p}_i \log \frac{\boldsymbol{p}_i}{\boldsymbol{q}_i}$.

3.3 Threshold and Ensemble

We took the top-K ranked target words of each metric and aggregated them for the final submission. The K was decided when the aggregated target words reached the half of total words numbers, since we assumed that the annotated labels are balanced. See (Schlechtweg et al., 2020) for detailed discussions about thresholds.

4 Experiments

4.1 Dataset and Evaluation Methodology

DIACR-Ita is the first task on lexical semantic change for Italian. DIACR-Ita aims to automatically detect whether a word semantically change over time. The task is to detect if a set of words, called target words, change their meaning across two periods, t_1 and t_2, where t_1 precedes t_2. Participants are provided with two corpora $\mathcal{C}_1$ and $\mathcal{C}_2$ (corresponding to t_1 and t_2, respectively), and a set of target words. For instance, the meaning of the word 'imbarcata' has changed from t_1 to t_2; originally, the word referred to an 'acrobatic manoeuvre of aeroplanes', but it is nowadays used to refer to the state of being deeply in love (Basile et al., 2020a) although the latter meaning is much less used than the former meaning. The task is formulated as a closed task, namely, models must be trained solely on the provided data. The occurrence about target words is reported in Table 1.

Labels in this task are binary and the task is considered as a binary classification problem. The evaluation is based on accuracy:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

word	# corpus $\mathcal{C}_1$	# corpus $\mathcal{C}_2$
egemonizzare	11	37
lucciola	64	226
campanello	109	628
trasferibile	7	60
brama	17	93
polisportiva	74	134
palmare	19	88
processare	39	594
pilotato	34	285
cappuccio	60	198
pacchetto	274	5690
ape	123	252
unico	4524	29620
discriminatorio	110	262
rampante	26	462
campionato	3918	11871
tac	88	438
piovra	30	621

Table 1: '#1' and '#2' denote the number of sentences where the target word occurs in two time-stamped corpora $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively.

methods	accuracy
Frequencies	0.50
Collocations	0.61
Aggregated results (submitted)	**0.67**
Average negative cosine similarity	0.67
Average distance with Euclidean distance	0.61
Average distance with Canberra distance	0.61
Hausdorff distance	0.50
JS divergence with K-means Clustering	0.61
JS divergence with Gaussian Mixture Model	0.61

Table 2: Results of the proposed methods.

T, F refers to 'True' and 'False', P, N refers to 'positive' and 'negative'. For example, TP is the number of Truly-predicted Positive samples.

The task735680 organizers provided two baselines: **Frequencies**: the absolute value of the difference between the words' frequencies is computed; **Collocations**: for each word, it computes the cosine similarity between two Bag-of-Collocations (BoCs) vector representations related to $\mathcal{C}_1$ and $\mathcal{C}_2$. In both baseline models, a threshold is used to predict if the word has changed its meaning.

4.2 Experimental Results

Experimental results are reported Table 2 and show that the proposed method achieved better performance than frequency and collocation based baselines.

4.3 Post-hoc Analysis

In this section, we will provide a bi-dimensional visualization of word representation to intuitively understand how the contextualized word vectors work. For each word, we get all contextualized word vectors (with a dimension of 768) based on its context. To visualized word in a 2D plane, we used a typical dimension reduction algorithm called T-SNE (Maaten and Hinton, 2008) to reduce word vectors from 768 to 2. Red and blue points denote the low dimensional representation of vectors when considering the two time-stamped corpora $\mathcal{C}_1$ (blue) and $\mathcal{C}_2$ (red).

For example, 'rampante' and 'palmare' are the predicted positive samples while 'cappuccio' and 'campanello' are predicted negative samples. As shown in Figure 1, the predicted semantically-shifted words exhibit a clear difference between red points an blue points with respect to two time-stamped corpora. For the predicted semantically-unshifted words (see Figure 2), it looks slightly indistinguishable.

5 Limitations

In (Schlechtweg et al., 2020), semantic representations are mainly divided to two categories: average embeddings ('type embeddings') and contextualized embeddings ('token embeddings'). Schlechtweg et al. (2020) illustrated the performance of token-based models are much lower than type-based embedding models. In this section, we will discuss some limitations of currently-used contextualized embedding based methods for semantic change detection.

There are typically two kinds of methods to use contextualized embeddings for semantic change detection: *embedding-based distance metrics* and *clustering-based distance metrics* (Schlechtweg et al., 2020; Vani et al., 2020; Giulianelli et al., 2020; Giulianelli, 2019). The former are directly calculated on the raw contextualized word embeddings while the latter are based on the clustering results of contextualized word embeddings.

5.1 Embedding-based Distance Metrics

Can distance metrics distinguish semantic shift patterns? Many typical patterns of semantic shifts have been investigated (Grossmann and Rainer, 2013; Basile et al., 2020a): 1) pejoration or amelioration (when word meanings become more negative or more positive); 2) broadening or narrowing (when it evolves as a generalized/extended object or a restricted or specialized one); 3) adding/deleting a sense; 4) totally shifted. The patterns of semantic change are multifaceted and we are questioning that a single distance metric could precisely distinguish all the above typical semantic shift patterns.

Normalization. Most of distance metrics are not normalized except for negative cosine similarity. Absolute values of unnormalized distance metrics may differ a lot among individual words; they are sometimes unexpectedly affected by the number of samples, leads to that the values of metrics may not be comparable among words.

Outliers. Some distance metrics (e.g., Hausdorff distance) are sensitive to outliers. For example, since the calculation of Hausdorff distance is based on infimum and supremum, an outlier point may largely affect the final Hausdorff distance. As seen in Table 3, frequently-appearing words e.g., 'campionato' and 'unico' have the highest Hausdorff distance between $\mathcal{C}_1$ and $\mathcal{C}_2$, this is probably biased by the fact that the two words appear frequently (see Table 1) and therefore likely have more unexpected outliers.

Model Fine-tuning. The contextualized word embedding that is based on pre-trained language models like BERT achieved much better results compared to static word embedding with a two-stage training paradigm, where the two stages are pre-training in language model (e.g., mask language model) and fine-tuning in downstream tasks (e.g., classifications). However, in the semantic change detection task, fine-tuning in downstream tasks is currently impossible because the annotated labels are insufficient to this aim; to some extent, the lack of fine-tuning stage may harm the performance of the pre-trained language models.

5.2 Clustering-based Distance Metrics

After clustering, we used the Jensen–Shannon divergence (JSD) which is affected by the issues mentioned in Section 5.1 like other distance metrics. Plus, the clustering algorithm may introduce some errors of semantic change detection. First, typical clustering algorithms may not necessarily converge to an identical clustering result when the seed centroids are changed. Moreover, the number of clusters is crucial since the optimal number of clusters cannot easily be decided before clustering.

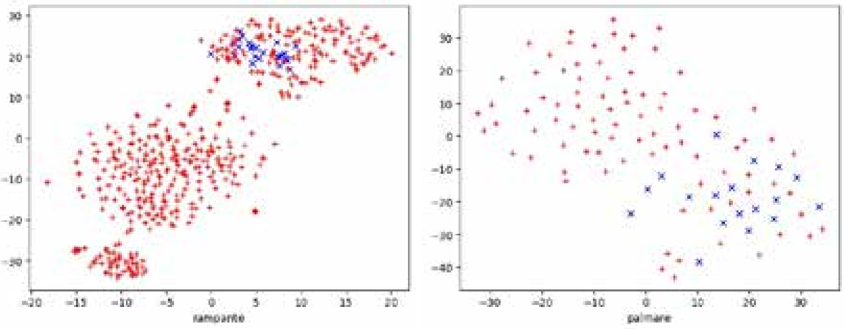


Figure 1: Examples (i.e., ‘rampante’ and ‘palmare’) of predicted ”semantically-shifted” words. Red and blue points denote dimensionally-reduced vectors of two time-stamped corpora respectively.

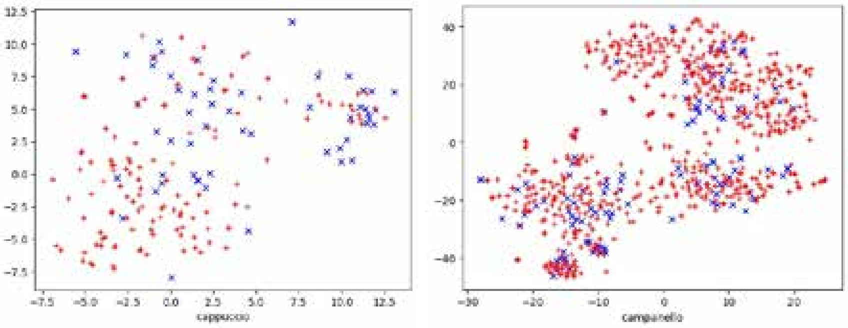


Figure 2: Examples (i.e., ‘cappuccio’ and ‘campanello’) of predicted ”semantically-unshifted” words. Red and blue points denote dimensionally-reduced vectors of two time-stamped corpora respectively.

6 Conclusions

This paper formalizes semantic change detection as a distance metric between two variable-sized sets of vectors. The final prediction is based on an ensemble of different distance metrics. The proposed method outperformed weak frequency and collocation baselines, but it performed less well than SOTA baselines. As a future work, this task may be largely improved via a supervised task in a unified multi-lingual framework; thus, any human-annotated labels in other languages could be used in this task since currently the number of annotated semantically-shift words in a single language is limited.

Acknowledgments

This work is supported by the Quantum Access and Retrieval Theory (QUARTZ) project, which has received funding from the European Union‘s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 721321.

A Appendix

Table 3 reports the predictions based on various distance metrics.

References

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020a. DIACR-Ita @ EVALITA2020: Overview of the EVALITA 2020 Diachronic Lexical Semantics (DIACR-Ita) Task. In *EVALITA 2020*, Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro (Eds.). CEUR.org, Online.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020b. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro (Eds.). CEUR.org, Online.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

Haim Dubossarsky, Simon Hengchen, Nina Tahmasebi, and Dominik Schlechtweg. 2019. Time-out: Temporal referencing for robust

word	AGD-cosine	AGD-euclidean	AGD-canberra	Hausdorff distance	JSD-GMM	JSD-Kmeans
matematica	0.996	1.02	86.6	10.0	0.004	0.025
dettagliato	**0.895**	**6.09**	**290.9**	7.5	**0.693**	**0.693**
sanità	0.990	1.86	130.8	**10.9**	0.025	0.052
senatore	0.997	0.79	79.1	7.7	0.009	0.002
istruzione	**0.854**	**6.14**	**333.7**	**14.4**	**0.275**	**0.279**
egemonizzare	0.988	1.62	136.6	5.6	0.003	0.033
lucciola	0.970	2.58	187.3	8.4	**0.414**	**0.154**
campanello	0.990	1.13	131.7	10.8	0.003	0.003
trasferibile	**0.873**	**4.25**	**300.7**	7.2	0.059	0.073
brama	**0.830**	**5.80**	**346.2**	8.3	**0.420**	**0.406**
polisportiva	**0.921**	**4.42**	**285.7**	7.5	**0.293**	**0.291**
palmare	0.955	2.55	220.5	8.0	0.130	**0.154**
processare	0.986	1.76	159.9	6.9	0.105	0.067
pilotato	0.970	2.27	198.9	**12.1**	0.108	0.128
cappuccio	0.973	1.78	183.6	**12.2**	0.015	0.016
pacchetto	0.984	1.67	149.6	10.5	0.011	0.009
ape	0.953	2.09	216.7	**15.3**	0.033	0.031
unico	0.985	1.89	149.9	**16.2**	0.035	0.032
discriminatorio	0.987	1.56	150.5	10.2	0.007	0.007
rampante	**0.888**	**4.78**	**302.7**	6.5	**0.293**	**0.299**
campionato	0.978	2.51	183.1	**16.0**	0.074	0.071
tac	**0.815**	**5.25**	**366.2**	9.9	**0.301**	**0.391**
piovra	0.976	2.27	189.6	9.7	0.033	0.033

Table 3: Calculated scores of various distance metrics. Top ranked scores are in bold.

modeling of lexical semantic change. *arXiv preprint arXiv:1906.01688* (2019).

Mario Giulianelli. 2019. Lexical semantic change analysis with contextualised word representations. *Unpublished master's thesis, University of Amsterdam, Amsterdam* (2019).

Mario Giulianelli, Marco Del Tredici, and Raquel Fernández. 2020. Analysing Lexical Semantic Change with Contextualised Word Representations. *arXiv preprint arXiv:2004.14118* (2020).

Maria Grossmann and Franz Rainer. 2013. *La formazione delle parole in italiano*. Walter de Gruyter.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. In *ACL*. 1489–1501.

Renfen Hu, Shen Li, and Shichen Liang. 2019. Diachronic sense modeling with deep contextualized word embeddings: An ecological view. In *ACL*. 3899–3908.

Yoon Kim, Yi-I Chiu, Kentaro Hanaki, Darshan Hegde, and Slav Petrov. 2014. Temporal Analysis of Language through Neural Language Models. *ACL 2014* (2014), 61.

Andrey Kutuzov and Mario Giulianelli. 2020. UiO-UvA at SemEval-2020 Task 1: Contextualised Embeddings for Lexical Semantic Change Detection. *arXiv preprint arXiv:2005.00050* (2020).

Godfrey N Lance and William T Williams. 1966. Computer programs for hierarchical polythetic classification ("similarity analyses"). *Comput. J.* 9, 1 (1966), 60–64.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 9, Nov (2008), 2579–2605.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*. 2227–2237.

Martin Pömsl and Roman Lyapin. 2020. CIRCE at SemEval-2020 Task 1: Ensembling Context-Free and Context-Dependent Word Representations. *arXiv preprint arXiv:2005.06602* (2020).

R Tyrrell Rockafellar and Roger J-B Wets. 2009. *Variational analysis*. Vol. 317. Springer Science & Business Media.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. *arXiv preprint arXiv:2007.11464* (2020).

K Vani, Sandra Mitrovic, Alessandro Antonucci, and Fabio Rinaldi. 2020. SST-BERT at SemEval-2020 Task 1: Semantic Shift Tracing by Clustering in BERT-based Embedding Spaces. *arXiv preprint arXiv:2010.00857* (2020).

Benyou Wang, Emanuele Di Buccio, and Massimo Melucci. 2019. Representing Words in Vector Space and Beyond. In *Quantum-Like Models for Information Retrieval and Decision-Making*. Springer, 83–113.

UWB @ DIACR-Ita: Lexical Semantic Change Detection with CCA and Orthogonal Transformation

Ondřej Pražák[* 1,2], **Pavel Přibáň**[* 1,2], and **Stephen Taylor**[* 2]

[1]NTIS – New Technologies for the Information Society,
[2]Department of Computer Science and Engineering,
Faculty of Applied Sciences, University of West Bohemia, Czech Republic
`{ondfa, pribanp, taylor}@kiv.zcu.cz`
`http://nlp.kiv.zcu.cz`

Abstract

In this paper, we describe our method for detection of lexical semantic change (i.e., word sense changes over time) for the DIACR-Ita shared task, where we ranked 1^{st}. We examine semantic differences between specific words in two Italian corpora, chosen from different time periods. Our method is fully unsupervised and language independent. It consists of preparing a semantic vector space for each corpus, earlier and later. Then we compute a linear transformation between earlier and later spaces, using CCA and Orthogonal Transformation. Finally, we measure the cosines between the transformed vectors.

1 Introduction

Language evolves with time. New words appear, old words fall out of use, and the meanings of some words shift. There are changes in topics, syntax, and presentation structure. Reading the natural philosophy musings of aristocratic amateurs from the eighteenth century, and comparing with a monograph from the nineteenth century, or a medical study from the twentieth century, we can observe differences in many dimensions, some of which need a deep historical background to study. Changes in word senses are both a visible and a tractable part of language evolution.

Computational methods for researching the stories of words have the potential of helping us understand this small corner of linguistic evolution. The tools for measuring these diachronic semantic shifts might also be useful for measuring whether the same word is used in different ways in synchronic documents. The task of finding word sense changes over time is called diachronic *Lexical Semantic Change (LSC)* detection. The task is getting more attention in recent years (Hamilton et al., 2016b; Schlechtweg et al., 2017; Schlechtweg et al., 2020). There is also the *synchronic LSC* task, which aims to identify domain-specific changes of word senses compared to general-language usage (Schlechtweg et al., 2019).

1.1 Related Work

Tahmasebi et al. (2018) provide a comprehensive survey of techniques for the *LSC* task, as do Kutuzov et al. (2018). Schlechtweg et al. (2019) evaluate available approaches for *LSC* detection using the *DURel* dataset (Schlechtweg et al., 2018). Schlechtweg et al. (2020) present results of the first shared task that addresses the LSC problem and provide an evaluation dataset that was manually annotated for four languages.

According to Schlechtweg et al. (2019), there are three main types of approaches. (1) Semantic vector spaces approaches (Gulordava and Baroni, 2011; Eger and Mehler, 2016; Hamilton et al., 2016a; Hamilton et al., 2016b; Rosenfeld and Erk, 2018; Pražák et al., 2020) represent each word with two vectors for two different time periods. The change of meaning is then measured by some distance (usually by the cosine distance) between the two vectors. (2) Topic modeling approaches (Bamman and Crane, 2011; Mihalcea and Nastase, 2012; Cook et al., 2014; Frermann and Lapata, 2016; Schlechtweg and Walde, 2020) estimate a probability distribution of words over their different senses, i.e., topics and (3) Clustering models (Mitra et al., 2015; Tahmasebi and Risse, 2017).

1.2 The DIACR-Ita task

The goal of the DIACR-Ita task (Basile et al., 2020a; Basile et al., 2020b) is to establish if a set of Italian words (target words) change their meaning from time period t_1 to time period t_2 (i.e., bi-

---

[*] Equal contribution.

nary classification task). The organizers provide corresponding corpora C_1 and C_2 and a list of target words. Only these inputs may be used to train systems, which judge for each target word, whether it is changed or not. The task is the same as the binary sub-task of the SemEval-2020 Task 1 (Schlechtweg et al., 2020) competition.

2 Data

The DIACR-Ita data consists of many randomly ordered text samples that have no relationship to each other. Most of the text samples are complete sentences, but some are sentence fragments.

The 'early' corpus, C_1 has about 2.4 million text samples and 52 million tokens; the 'later' corpus, C_2 has about 7.8 million text samples and 738 million tokens. Each token is given in the corpora with its part-of-speech tag and lemma. The target word list consists of 18 lemmas. The POS and lemmas of the corpora are generated with the UDPipe (Straka, 2018) model ISDT-UD v2.5, which has an error rate of about 2%.

3 System Description

3.1 Overview

Because language is evolving, expressions, words, and sentence constructions in two corpora from different time periods about the same topic will be written in languages that are quite similar but slightly different. They will share the majority of their words, grammar, and syntax. We can observe a similar situation in languages from the same family, such as *Italian-Spanish* in Romance languages or *Czech-Slovak* in Slavic languages. These pairs of languages share a lot of common words, expressions and syntax. For some pairs, native speakers can understand and sometimes even actively communicate through a (low) language barrier.

Our system follows the approach from (Pražák et al., 2020)[1]. The main idea behind our solution is that we treat each pair of corpora C_1 and C_2 as different languages L_1 and L_2 even though the text from both corpora is written in Italian. We believe that these two languages L_1 and L_2 will be extremely similar in all aspects, including semantic. We train a separate semantic space for each corpus, and subsequently, we map these two spaces into one common cross-lingual space. We use methods for cross-lingual mapping (Brychcín et al., 2019; Artetxe et al., 2016; Artetxe et al., 2017; Artetxe et al., 2018a; Artetxe et al., 2018b) and thanks to the large similarity between L_1 and L_2 the quality of transformation should be high. We compute cosine similarity of the transformed word vectors to classify whether the target words changed their sense.

3.2 Semantic Space Transformation

First, we train two semantic spaces from corpus C_1 and C_2. We represent the semantic spaces by a matrix $\mathbf{X}^s$ (i.e., a source space s) and a matrix $\mathbf{X}^t$ (i.e., a target space t)[2] using word2vec Skip-gram with negative sampling (Mikolov et al., 2013). We perform a cross-lingual mapping of the two vector spaces, getting two matrices $\hat{\mathbf{X}}^s$ and $\hat{\mathbf{X}}^t$ projected into a shared space. We select two methods for the cross-lingual mapping *Canonical Correlation Analysis (CCA)* using the implementation from (Brychcín et al., 2019) and a modification of the *Orthogonal Transformation* from *VecMap* (Artetxe et al., 2018b). Both of these methods are linear transformations. The transformations can be written as follows:

$$\hat{\mathbf{X}}^s = \mathbf{W}^{s\to t}\mathbf{X}^s \tag{1}$$

where $\mathbf{W}^{s\to t}$ is a matrix that performs linear transformation from the source space s (matrix $\mathbf{X}^s$) into a target space t and $\hat{\mathbf{X}}^s$ is the source space transformed into the target space t (the matrix $\mathbf{X}^t$ does not have to be transformed because $\mathbf{X}^t$ is already in the target space t and $\mathbf{X}^t = \hat{\mathbf{X}}^t$).

Finally, in all transformation methods, for each word w_i from the set of target words T, we select its corresponding vectors $\mathbf{v}^s_{w_i}$ and $\mathbf{v}^t_{w_i}$ from matrices $\hat{\mathbf{X}}^s$ and $\hat{\mathbf{X}}^t$, respectively ($\mathbf{v}^s_{w_i} \in \hat{\mathbf{X}}^s$ and $\mathbf{v}^t_{w_i} \in \hat{\mathbf{X}}^t$), and we compute cosine similarity between these two vectors. The cosine similarity is then used to generate a final classification output using different strategies, see Section 3.5 and 3.6.

3.3 Canonical Correlation Analysis

Generally, the CCA transformation transforms both spaces $\mathbf{X}^s$ and $\mathbf{X}^t$ into a third shared space o (where $\mathbf{X}^s \neq \hat{\mathbf{X}}^s$ and $\mathbf{X}^t \neq \hat{\mathbf{X}}^t$). Thus, CCA computes two transformation matrices $\mathbf{W}^{s\to o}$ for the source space and $\mathbf{W}^{t\to o}$ for the target space. The transformation matrices are computed by

---

[1] The source code is available at `https://github.com/pauli31/SemEval2020-task1`

[2] The source space $\mathbf{X}^s$ is created from the corpus C_1 and the target space $\mathbf{X}^t$ is created from the corpus C_2.

minimizing the negative correlation between the vectors $\mathbf{x}_i^s \in \mathbf{X}^s$ and $\mathbf{x}_i^t \in \mathbf{X}^t$ that are projected into the shared space o. The negative correlation is defined as follows:

$$\underset{\mathbf{W}^{s\to o},\mathbf{W}^{t\to o}}{\operatorname{argmin}} -\sum_{i=1}^{n} \rho(\mathbf{W}^{s\to o}\mathbf{x}_i^s, \mathbf{W}^{t\to o}\mathbf{x}_i^t) = -\sum_{i=1}^{n} \frac{cov(\mathbf{W}^{s\to o}\mathbf{x}_i^s, \mathbf{W}^{t\to o}\mathbf{x}_i^t)}{\sqrt{var(\mathbf{W}^{s\to o}\mathbf{x}_i^s) \times var(\mathbf{W}^{t\to o}\mathbf{x}_i^t)}} \tag{2}$$

where cov is the covariance, var is the variance and n is the number of vectors used for computing the transformation. In our implementation of CCA, the matrix $\hat{\mathbf{X}}^t$ is equal to the matrix $\mathbf{X}^t$ because it transforms only the source space s (matrix $\mathbf{X}^s$) into the target space t from the common shared space with a pseudo-inversion, and the target space does not change. The matrix $\mathbf{W}^{s\to t}$ for this transformation is then given by:

$$\mathbf{W}^{s\to t} = \mathbf{W}^{s\to o}(\mathbf{W}^{t\to o})^{-1} \tag{3}$$

The submissions that use CCA are referred to as **cca-bin** and **cca-ranking** in Table 1. The **-bin** and **-ranking** parts refer to a strategy used for the final classification decision, see Section 3.5 and 3.6.

3.4 Orthogonal Transformation

In the case of the Orthogonal Transformation, the submission is referred to as **ort-bin**. We use Orthogonal Transformation with a supervised seed dictionary consisting of all words common to both semantic spaces. The transformation matrix $\mathbf{W}^{s\to t}$ is given by:

$$\underset{\mathbf{W}^{s\to t}}{\operatorname{argmin}} \sum_{i}^{|V|} (\mathbf{W}^{s\to t}\mathbf{x}_i^s - \mathbf{x}_i^t)^2 \tag{4}$$

under the hard condition that $\mathbf{W}^{s\to t}$ needs to be orthogonal, where V is the vocabulary of correct word translations from source space $\mathbf{X}^s$ to target space $\mathbf{X}^t$ and $\mathbf{x}_i^s \in \mathbf{X}^s$ and $\mathbf{x}_i^t \in \mathbf{X}^t$. The reason for the orthogonality constraint is that linear transformation with an orthogonal matrix does not squeeze or re-scale the transformed space. It only rotates the space, thus it preserves most of the relationships of its elements (in our case, it is important that orthogonal transformation preserves angles between the words, so it preserves the cosine similarity).

3.5 Binary Strategy

We use different strategies for the binary classification output, but all have in common that they use continuous scores. The continuous score for each target word is computed as the cosine similarity between the two vectors from the earlier and later corpus.

In the case of the *binary strategy*, we assume a threshold t for which the target words with a continuous score greater than t changed meaning and words with the score lower than t did not. We know that this assumption is generally wrong (because using the threshold, we introduce some error into the classification), but we still believe it holds for most cases and it is the best choice. To estimate the threshold t, we used an approach called *binary-threshold* (**cca-bin** and **ort-bin** in Table 1). For each target word w_i we compute cosine similarity of its vectors $\mathbf{v}_{w_i}^s$ and $\mathbf{v}_{w_i}^t$, then we average these similarities for all words. The resulting averaged[3] value is used as the threshold.

3.6 Ranking Strategy

The *ranking strategy* is the second approach for generating a classification output (the submission result **cca-ranking** in Table 1). It uses the mean rank of repeated runs of each embedding pair. For each run, the target words are scored with a cosine distance. Then the distances for each embedding pair are sorted and a rank-order is assigned to each target. The rank-orders are averaged, to get a mean rank (and a standard deviation) for each target for each pair. Finally, ranks for all embedding pairs are averaged. The composite rank is used, along with an estimate of the associated cosine distance and its corresponding angle, to divide the target list into changed and unchanged sets. This does not work well; there are competing gaps in rank and distance estimates.

We use the number of embeddings, and not the total number of runs, to compute the standard error of the mean (which is standard deviation divided by the square root of samples).

4 Experimental Setup

To obtain the semantic spaces, we employ Skip-gram with negative sampling (Mikolov et al., 2013). For the final submission, we trained the semantic spaces with 100 (the **ort-bin** submission)

---

[3] The **ort-bin** submission sets the threshold to be in the largest gap between the similarity values

and 150 (the **cca-bin** submission) dimensions for five iterations with five negative samples and window size set to five. Each word has to appear at least five times in the corpus to be used in the training. To train the semantic space, we used the lemmatized corpora. The dimensions 100 and 150 are selected based on our previous experiences with these methods (Pražák et al., 2020). Since we were able to submit four different submissions, we did not use the same dimension for both methods.

The **cca-ranking** submission uses the same settings and dimensions 100-105, 110-115, etc. up to 210-215, resulting in 72 different dimension sizes. It combines 40 runs on each of 72 embedding pairs, a total of 2880 runs.

For the **cca-bin** submission, we build the translation dictionary for the transformation of the two spaces by removing the target words from the intersection of their vocabularies. In the case of the **cca-ranking** submission, the dictionary in each run consists of up to 5000 randomly chosen common words for each semantic space.

The **random** submission represents output that was generated completely randomly.

4.1 Corpus variants

The organizers provided the corpora already tokenized in four different versions: original tokens; lemmatized tokens; original tokens with POS tag; lemmatized tokens with POS tag. We experimented with each of these variants, although in the end, we used results based only on lemmas. Figure 1 shows the mean standard deviation of rank for target words over forty runs for each of 72 different embedding sizes. The most consistent variant is the *lemmas* only.

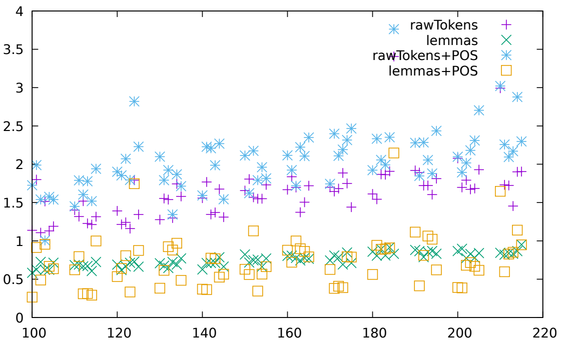


Figure 1: Standard deviation (of rank) versus embedding size for four versions of the corpora.

5 Results

We submitted four different submissions. The accuracy results for each submission are shown in Table 1. The **ort-bin** system achieved the best accuracy of 0.944 and ranked first[4] among eight other teams in the shared task, classifying 17 out of 18 target words correctly. The **cca-bin** system achieved an accuracy of 0.889 (16 correct classifications out of 18). After releasing the gold labels, we performed an additional experiment with the **cca-bin** system achieving also an accuracy of 0.944 when the same word embeddings (with embeddings dimension 100 instead of 150) are used as for the **ort-bin** system. We found an optimal threshold for both systems, which makes them classify all the words correctly[5].

We believe that the key factor of the success of our system is the sufficient size of the provided corpora. Thanks to that, we were able to train semantic spaces of good quality and thus achieve good results.

System	Accuracy
cca-bin	.889
ort-bin	.944
cca-ranking	.778
random	.500

Table 1: Results for our final submissions.

6 Conclusion

Our systems based on Canonical Correlation Analysis and Orthogonal Transformation achieved the best accuracy of 0.944 in the shared task and ranked first among eight other teams. We showed that our approach is a suitable solution for the *Lexical Semantic Change* detection task. Applying a threshold to semantic distance is a sensible architecture for detecting the binary semantic change in target words between two corpora. Our *binary-threshold* strategy succeeded quite well.

This task provided plenty of text to build good word embeddings. Corpora with much smaller amounts of data might have increased the random variation between the earlier and later embeddings, which would have given our method problems. A flaw in our technique is that semantic vec-

---

[4] We share the first place with another team that achieved the same accuracy.

[5] That is, 100% accuracy was possible with the continuous scores of both methods if we only had an oracle to set the threshold.

tors are based on all senses of a word in the corpus. We do not yet have tools to tease out what kinds of changes are implied by a particular semantic distance between vectors. We considered using the part of speech data in the corpora since different parts of speech for the same lemma are likely different senses. But placing the POS in the token, like using inflections instead of lemmas, results in many more, less well-trained semantic vectors, as suggested by Figure 1.

Acknowledgements

This work has been partly supported by ERDF "Research and Development of Intelligent Components of Advanced Technologies for the Pilsen Metropolitan Area (InteCom)" (no.: CZ.02.1.01/0.0/0.0/17 048/0007267); by the project LO1506 of the Czech Ministry of Education, Youth and Sports; and by Grant No. SGS-2019-018 Processing of heterogeneous data and its specialized applications. Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures" (CESNET LM2015042), is greatly appreciated.

References

[Artetxe et al.2016] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2289–2294, Austin, Texas, November. Association for Computational Linguistics.

[Artetxe et al.2017] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Vancouver, Canada, July. Association for Computational Linguistics.

[Artetxe et al.2018a] Mikel Artetxe, Gorka Labaka, , and Eneko Agirre. 2018a. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 5012–5019.

[Artetxe et al.2018b] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018b. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia, July. Association for Computational Linguistics.

[Bamman and Crane2011] David Bamman and Gregory Crane. 2011. Measuring historical word sense variation. In *Proceedings of the 11th Annual International ACM/IEEE Joint Conference on Digital Libraries*, JCDL '11, page 1–10, New York, NY, USA. Association for Computing Machinery.

[Basile et al.2020a] Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020a. DIACR-Ita @ EVALITA2020: Overview of the EVALITA2020 Diachronic Lexical Semantics (DIACR-Ita) Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

[Basile et al.2020b] Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020b. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

[Brychcín et al.2019] Tomáš Brychcín, Stephen Taylor, and Lukáš Svoboda. 2019. Cross-lingual word analogies using linear transformations between semantic spaces. *Expert Systems with Applications*, 135:287–295.

[Cook et al.2014] Paul Cook, Jey Han Lau, Diana McCarthy, and Timothy Baldwin. 2014. Novel word-sense identification. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1624–1635, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.

[Eger and Mehler2016] Steffen Eger and Alexander Mehler. 2016. On the linearity of semantic change: Investigating meaning variation via dynamic graph models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 52–58, Berlin, Germany, August. Association for Computational Linguistics.

[Frermann and Lapata2016] Lea Frermann and Mirella Lapata. 2016. A Bayesian model of diachronic meaning change. *Transactions of the Association for Computational Linguistics*, 4:31–45.

[Gulordava and Baroni2011] Kristina Gulordava and Marco Baroni. 2011. A distributional similarity approach to the detection of semantic change in the Google books ngram corpus. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 67–71, Edinburgh, UK, July. Association for Computational Linguistics.

[Hamilton et al.2016a] William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016a. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2116–2121, Austin, Texas, November. Association for Computational Linguistics.

[Hamilton et al.2016b] William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016b. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501, Berlin, Germany, August. Association for Computational Linguistics.

[Kutuzov et al.2018] Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: a survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

[Mihalcea and Nastase2012] Rada Mihalcea and Vivi Nastase. 2012. Word epoch disambiguation: Finding how words change over time. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 259–263, Jeju Island, Korea, July. Association for Computational Linguistics.

[Mikolov et al.2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of workshop at ICLR*. arXiv1301.3781.

[Mitra et al.2015] Sunny Mitra, Ritwik Mitra, Suman Kalyan Maity, Martin Riedl, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2015. An automatic approach to identify word sense changes in text media across timescales. *Natural Language Engineering*, 21(5):773–798.

[Pražák et al.2020] Ondřej Pražák, Pavel Přibáň, Stephen Taylor, and Jakub Sido. 2020. Uwb at semeval-2020 task 1: Lexical semantic change detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2020)*, Barcelona, Spain, Sep. Association for Computational Linguistics.

[Rosenfeld and Erk2018] Alex Rosenfeld and Katrin Erk. 2018. Deep neural models of semantic shift. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 474–484, New Orleans, Louisiana, June. Association for Computational Linguistics.

[Schlechtweg and Walde2020] Dominik Schlechtweg and Sabine Schulte im Walde. 2020. Simulating lexical semantic change from sense-annotated data. In A. Ravignani, C. Barbieri, M. Martins, M. Flaherty, Y. Jadoul, E. Lattenkamp, H. Little, K. Mudd, and T. Verhoef, editors, *The Evolution of Language: Proceedings of the 13th International Conference (EvoLang13)*.

[Schlechtweg et al.2017] Dominik Schlechtweg, Stefanie Eckmann, Enrico Santus, Sabine Schulte im Walde, and Daniel Hole. 2017. German in flux: Detecting metaphoric change via word entropy. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 354–367, Vancouver, Canada, August. Association for Computational Linguistics.

[Schlechtweg et al.2018] Dominik Schlechtweg, Sabine Schulte im Wlade, and Stefanie Eckmann. 2018. Diachronic usage relatedness (durel): A framework for the annotation of lexical semantic change. In *Proceedings of NAACL-HLT 2018*, pages 169–174.

[Schlechtweg et al.2019] Dominik Schlechtweg, Anna Hätty, Marco Del Tredici, and Sabine Schulte im Walde. 2019. A wind of change: Detecting and evaluating lexical semantic change across times and domains. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 732–746, Florence, Italy, July. Association for Computational Linguistics.

[Schlechtweg et al.2020] Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval 2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2020)*, Barcelona, Spain, Sep. Association for Computational Linguistics.

[Straka2018] Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium, October. Association for Computational Linguistics.

[Tahmasebi and Risse2017] Nina Tahmasebi and Thomas Risse. 2017. Finding individual word sense changes and their delay in appearance. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 741–749, Varna, Bulgaria, September. INCOMA Ltd.

[Tahmasebi et al.2018] Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of computational approaches to lexical semantic change. *arXiv preprint arXiv:1811.06278*.

QMUL-SDS @ DIACR-Ita: Evaluating Unsupervised Diachronic Lexical Semantics Classification in Italian

Rabab Alkhalifa[1,2], **Adam Tsakalidis**[1,3], **Arkaitz Zubiaga**[1], and **Maria Liakata**[1,3]

[1]Queen Mary University of London, United Kingdom
[2]Imam Abdulrahman bin Faisal University, Saudi Arabia
[3]Alan Turing Institute, United Kingdom

Abstract

In this paper, we present the results and main findings of our system for the DIACR-Ita 2020 Task. Our system focuses on using variations of training sets and different semantic detection methods. The task involves training, aligning and predicting a word's vector change from two diachronic Italian corpora. We demonstrate that using Temporal Word Embeddings with a Compass C-BOW model is more effective compared to different approaches including Logistic Regression and a Feed Forward Neural Network using accuracy. Our model ranked 3rd with an accuracy of 83.3%.

1 Introduction

The quantitative analysis of language evolution over time is a new emerging research area within the domain of Natural Language Processing (Turney and Pantel, 2010; Hamilton et al., 2016; Dubossarsky et al., 2017). The study of Diachronic Lexical Semantics (Tahmasebi et al., 2018; Kutuzov et al., 2018), which contributes towards detecting word-level language evolution, brings together researchers with broadly varying backgrounds from computational linguistics, cognitive science, statistics, mathematics, and historical linguistics, since the identification of words whose lexical semantics have changed over time has numerous downstream applications in various domains such as historical linguistics and NLP. Despite the increase in research interest, few tasks that track word meaning change over time have focused on non-English languages, while the comparison of different approaches in the same experimental and evaluation setting is still limited (Schlechtweg et al., 2020). The DIACR-Ita 2020 Task (Basile et al., 2020a; Basile et al., 2020b) aims to fill these gaps by focusing on the Italian language used during two different time periods and providing a single evaluation framework to researchers for testing their methods.

This work presents our approach towards detecting Italian words with altered lexical semantics during the two distinct time periods studied in the DIACR-Ita 2020 Shared Task. Our contribution focuses on evaluating findings from previous studies, exploring evaluation approaches for different methods and comparing their performance. We contrast several variants of training-testing words with different alignment approaches across two word embedding models, namely Skip-gram and Continuous Bag-of-Words (Mikolov et al., 2013). Our submission consisted of four models that showed the best average cosine similarity, calculated on the basis of their ability to accurately reconstruct the representations of Italian stop-words across the two periods of time under study. Our best performing model uses a Continuous Bag-of-Words temporal compass model, adapted from the model introduced by (Carlo et al., 2019). Our system ranked third in the task.

2 Related Work

Work related to unsupervised diachronic lexical semantics detection can be divided into different approaches depending on the type of word representations used in a diachronic model (e.g., based on graphs or probability distributions (Frermann and Lapata, 2016; Azarbonyad et al., 2017), temporal dimensions (Basile and McGillivray, 2018), frequencies or co-occurrence matrices (Sagi et al., 2009; Cook and Stevenson, 2010), neural- or Transformer-based (Hamilton

---

et al., 2016; Boleda et al., 2019; Shoemark et al., 2019; Schlechtweg et al., 2019; Giulianelli et al., 2020), etc.). In our work, we focus on dense word representations (Mikolov et al., 2013), due to their high effectiveness that has been demonstrated in prior work.

Systems operating on representations such as those derived from Skip-gram or Continuous Bag-of-Words leverage in most cases deterministic approaches using mathematical matrix transformations (Hamilton et al., 2016; Azarbonyad et al., 2017; Tsakalidis et al., 2019), such as Orthogonal Procrustes (Schönemann, 1966), or machine learning models (Tsakalidis and Liakata, 2020). The goal of these approaches is to learn a mapping between the word vectors that have been trained independently by leveraging textual information from two or more different periods of time. The common standard for measuring the level of diachronic semantic change of a word under this setting is to use a similarity measure (e.g., cosine distance) on the aligned space – i.e., after the mapping step is complete (Turney and Pantel, 2010).

(Dubossarsky et al., 2017) argue that using cosine distance introduces bias in the system triggered by word frequency variations. (Tan et al., 2015) only use the vectors of the top frequent terms to find the transformation matrix, and then they calculate the similarity for the remaining terms after applying the transformation to the source matrix. Incremental update (Kim et al., 2014; Boleda et al., 2019) used the intersection of words between datasets in each time frame by initializing the word embedding from the previous time slice to compare the word shift cross different years instead of using matrix transformation. Temporal Word Embeddings with a Compass (TWEC) (Carlo et al., 2019) approach uses an approach of freezing selected vectors based on model's architecture, it learn a parallel embedding for all time periods from a base embedding frozen vectors.

Our approaches, detailed in Section 4, follow and compare different methodologies from prior work based on (a) Orthogonal Procrustes alignment, (b) machine learning models and (c) aligned word embeddings across different time periods.

3 Task Description

The task was introduced by (Cignarella et al., 2020) and is defined as follows:

> Given two diachronic textual data, an unsupervised diachronic lexical semantics classifier should be able to find the optimal mapping to compare the diachronic textual data and classify a set of test words to one of two classes: 0 for stable words and 1 for words whose meaning has shifted.

We were provided with the two corpora in the Italian language, each from a different time period, and we developed several methods in order to classify a word in the given test set as "semantically shifted" or "stable" across the two time periods. The test set included 18 observed words – 12 stable and 6 semantically shifted examples.

4 Our Approach

Here we outline our approaches for detecting words whose lexical semantics have changed.

4.1 Generating Word Vectors

Word representations W_i at the period T_i were generated in two ways:

(a) IND: via Continuous Bag of Words (CBOW) and Skip-gram (SG) (Mikolov et al., 2013) applied to each year independently;

(b) CMPS: via the Temporal Word Embeddings with a Compass (TWEC) approach (Carlo et al., 2019), where a single model (CBOW or SG) is first trained over the merged corpus; then, SG (or CBOW) is applied on the representations of each year independently, by initialising and freezing the weights of the model based on the output of the first base model pass and learning only the contextual part of the representations for that year.

In both cases, we used gensim with default settings.[1] Sentences were tokenised using the simple split function for flattened sentences provided by the organisers, without any further pre-processing. Although there are many approaches to generate word representations (e.g., using syntactic rules), we focused on 1-gram rep-

---

[1] `https://radimrehurek.com/gensim/`

resentations using CBOW and SG, without considering words lemmas and Part-of-Speech tags.

4.2 Measuring Semantic Change

We employ the cosine similarity for measuring the level of semantic change of a word. Given two word vectors w^{T0}, w^{T1}, semantic change between them is defined as follows:

$$\cos(w^{T0}, w^{T1}) = \frac{w^{T0} \cdot w^{T1}}{\|w^{T0}\| \|w^{T1}\|} = \frac{\Sigma_{i=1} w_i^{T0} w_i^{T1}}{\sqrt{\Sigma_{i=1} {w_i^{T0}}^2} \sqrt{\Sigma_{i=1} {w_i^{T1}}^2}} \quad (1)$$

Though alternative methods have been introduced in the literature (e.g., neighboring by pivoting the top five similar words (Azarbonyad et al., 2017)), we opted for the similarity metric which is most widely used in related work (Hamilton et al., 2016; Shoemark et al., 2019; Tsakalidis et al., 2019).

4.3 Evaluation Sets

The challenge is expecting the lexical change detection to be done in an unsupervised fashion (i.e., no word labels have been provided). Thus, we considered stop words[2] (*SW*) and all of the other common words (*CW*) in T_0 and T_1 as our training and evaluation sets interchangeably.

4.4 Semantic Change Detection Methods

We employed the following approaches for detecting words whose lexical semantics have changed:

(a) Orthogonal Procrustes (OP): Due to the stochastic nature of CBOW/SG, the resulting word vectors W_0 and W_1 in *IND* were not aligned. Orthogonal Procrustes (Hamilton et al., 2016) tackles this issue by aligning W_1 based on W_0. The level of semantic shift of a word is calculated by measuring the cosine similarity between the aligned vectors. For evaluation purposes, we measured the cosine similarity of the stop words between the two aligned matrices. Higher values indicate a better model (i.e., stop words retain their meaning over time).

(b) Feed-Forward Neural Network (FFNN): We trained a FFNN that leverages *IND* to predict W_1 based on W_0. The level of semantic shift of a word in a test set is calculated by measuring the cosine similarity between the predicted W_1^* and W_1. For evaluation purposes, we measure the cosine similarity between the actual and predicted representations of words in T_1. Higher values for stop-words indicate a better model.

(c) Linear Regression (LR): We employed an ordinary linear mapping with least square error objective function.[3] The task and the evaluation setting was identical to FFNN.

(d) Temporal Word Embeddings with a Compass (TWEC) (Carlo et al., 2019): Working on the *CMPS* vectors, the level of semantic shift of a word is calculated by measuring the cosine similarity between T_0 and T_1 directly.

Notation In the rest of this paper, we denote a model M trained on CW (SW) as M_CW (M_SW). For the case of OP, the training process involves learning an alignment based on a specific word set (CW or SW). Note that this notation does not apply for $TWEC$, since the word vectors in the two time periods can be directly compared against each other – thus the level of semantic change can be calculated directly (i.e., there is no need to learn any mapping between W_0 and W_1). Finally, we add a subscript $_{CBOW}$ or $_{SG}$ to our models, denoting the type of algorithm that was used for generating the respective embeddings that are fed to our model.

Model Selection We select to apply the models on the test set providing high average cosine similarity with stop words.

4.5 Word Classification

As per the task guidelines (Cignarella et al., 2020), words can fall into one of the two categories: **0**: the target word does not change meaning between T_0 and T_1 and **1**: the target word changes its meaning between T_0 and T_1. For all of our submitted models, we considered all the words with cosine similarity below the mean as shifted words and labelled them with 1. We further investigate the model's ability to detect words laying two standard deviations below the mean ($\mu - 2\sigma$), a.k.a variance. Interestingly, some of the models including LR and FFNN_CW$_{CBOW}$ showed an increase in accuracy.

5 Results

The results are shown in Table 1, where we split our results based on model #M ar-

---

[2] `https://github.com/stopwords-iso/stopwords-it`

[3] `https://scikit-learn.org/stable/`

IND		**SG**						**C-BOW**					
		Accuracy			**Ranking**			**Accuracy**			**Ranking**		
train.	**M**	CS_{avg}^{SW}	$\%\mu$	$\%\mu-2\sigma$	$\%\mu_{rank}$	R_{p50}	$R_{\downarrow 6}$	CS_{avg}^{SW}	$\%\mu$	$\%\mu-2\sigma$	$\%\mu_{rank}$	R_{p50}	$R_{\downarrow 6}$
SW	***OP***	0.748	**0.778**	**0.667**	**0.222**	**1.000**	**0.667**	0.784	**0.778**	0.667	**0.270**	**1.000**	**0.833**
	LR	**0.854**	0.333	0.389	0.373	0.833	0.500	**0.795**	0.500	**0.778**	**0.278**	0.833	0.500
	FFNN	**0.769**	0.333	0.333	0.373	0.833	0.500	0.709	0.556	**0.722**	0.341	0.833	0.500
CW	***OP***	0.464	0.389	**0.778**	0.381	0.667	0.500	0.289	0.611	0.667	0.397	0.833	0.333
	LR	0.409	0.333	0.444	0.508	0.500	0.333	0.146	0.333	0.444	0.381	0.667	0.667
	FFNN	0.658	0.333	0.389	**0.317**	**1.000**	0.500	0.621	0.333	**0.722**	0.317	0.833	0.500
	TWEC	0.722	**0.722**	**0.667**	**0.317**	0.833	**0.667**	**0.833**	**0.833**$_*$	0.667	0.286	**1.000**	**0.667**

Table 1: Performance of our models using different evaluations methods. *($_*$) best submission.*

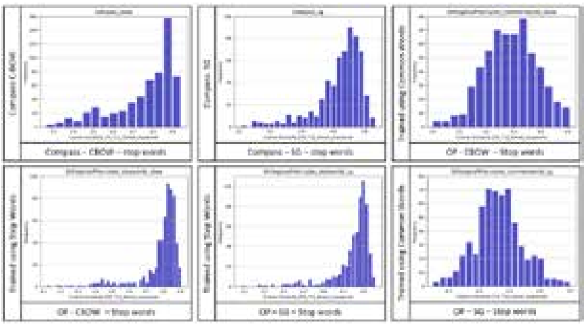


Figure 1: Frequency of stop words by their cosine similarity scores, where each subfigure pertains to a different model.

chitecture, $_{SG}$ and $_{CBOW}$ and model's training word sets, Stop-Words (SW) and Common-Words (CW). For models based on linear transformation, our top performing models scored below average cosine similarity, TWEC$_{CBOW}$ (0.833), OP_SW$_{SG}$(0.778), OP_SW$_{CBOW}$ (0.778), TWEC$_{SG}$(0.722). As shown in Figure 5, we observe that these models tend to have skewed distributions for stop words, where the vast majority of stop words are assigned high cosine similarity scores. However, other models did not show this skewness, e.g. OP_CW$_{SG}$(0.389) and OP_CW$_{CBOW}$(0.611). When labeling the change based on variance ($\mu-2\sigma$), as in outlier detection, some models showed an increase from the dummy classifier's performance. For instance, OP_CW$_{sg}$ showed an increase on performance from (0.389) to (0.778) showing that those with low average cosine similarity lay out in the tail from majority similarity. Similarly, models based on reducing the similarity error between the predicted and actual vectors, e.g. LR and FFNN considering the outlier detection methodology, tend to achieve better performance, including LR_SW$_{CBOW}$, FFNN_SW$_{CBOW}$ and FFNN_CW$_{CBOW}$ where LR_SW$_{CBOW}$ showed an increase from frequency classifier's baseline (0.500) to (0.778), and LR_SW$_{CBOW}$ showed an increase from dummy classifier performance (0.333) to (0.722).

Ranking methods, average ranking (μ_{rank}) and Recall (R), expect prior knowledge about the evaluation labels to make them useful for evaluating the reliability of the model of interest. For that, we further investigate the reliability of our experiment models, using μ_rank and R at %50 (R$_{p50}$) and %30 (R$_{\downarrow 6}$). Although using (R$_{p50}$) signal OP_SW$_{SG}$, OP_SW$_{CBOW}$, FFNN_CW$_{SG}$, TWEC$_{CBOW}$ as equalliy good, μ_{rank} ranked top models as OP_SW$_{SG}$, OP_SW$_{CBOW}$, LR_SW$_{CBOW}$ then TWEC$_{CBOW}$ with (0.222, 0.270, 0.278 and 0.286), respectively. Additionally, under extreme conditions, OP_SW$_{CBOW}$ ranked better than all including TWEC$_{CBOW}$. This shows that under extreme conditions, a good method is the one which keeps providing out of distribution signals to changing words and that needs to take a careful consideration about the distribution of the words before and after the alignments as in OP. In general, CBoW-based models showed better performance than SG-based models with average accuracy of (%μ 0.564 and %$\mu-2\sigma$ 0.667) compared to (%μ 0.460 and $\mu-2\sigma$ 0.524) for words labelled by mean and variance, respectively. Further, alignment using non-changing words (e.g. *stop-words*) yields higher performance than using all common words with average cosine similarity for stop words as (CS_{avg}^{SW} 0.777) compared to (CS_{avg}^{SW} 0.431), which is expected because SW-based models learns the optimal mapping with less noise than CW-based models.

6 Discussion

Our work provides a comprehensive analysis for Italian lexical diachronic methods introduced from previous work. For models that are based on matrix linear transformation including TWEC and OP, we find a relation between high average stop words similarity and accuracy. Further, C-BOW tends to achieve better results than the SG architecture for most experiments. Visually, we find that a visibly skewed distribution showing the tendency of stop words to have high cosine similarity scores leads to effective means for capturing semantic shift. We also showed that by evaluating the models using different methods, TWEC_{CBOW} achieved top performance. Followed by OP_SW and $\mathrm{OP\_CW}_{SG}$, and LR using outlier detection methodology. Further, FFNN showed high recall (R_{p50}) by ranking changed words with lowest cosine similarity on testing set similar to OP_SW and TWEC_{CBOW}. This provides promising insights encouraging further investigation of neural network models using different languages and larger datasets.

7 Conclusions

In this report, we describe and compare our models submitted to the DIACR-Ita 2020 shared task, which assessed the ability to classify semantic-shift of words in Italian. We show that the TWEC model yields better performance than Orthogonal Procrustes, labelling all words scored below average cosine similarity as semantically shifted words, i.e. words with altered semantics over the two time periods. Additionally, we showed that using an outlier detection methodology yields better results in prediction-based models such as Linear Regression and Feed-Forward Neural Network, boosting the performance significantly compared to the baselines and dummy classifier.

In the future we aim to focus on fine tuning SoTa pre-trained language models such as ELMo and BERT for word level semantics-shift detection as well as investigating the ability of dynamic graph models on capturing word evolution.

8 Acknowledgments

This research utilised Queen Mary's Apocrita HPC facility, supported by QMUL Research-IT.

References

Hosein Azarbonyad, Mostafa Dehghani, Kaspar Beelen, Alexandra Arkut, Maarten Marx, and Jaap Kamps. 2017. Words are malleable: Computing semantic shifts in political and media discourse. *International Conference on Information and Knowledge Management, Proceedings*, Part F1318(3):1509–1518.

Pierpaolo Basile and Barbara McGillivray. 2018. Exploiting the web for semantic change detection. In *International Conference on Discovery Science*, pages 194–208. Springer.

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020a. DIACR-Ita @ EVALITA2020: Overview of the EVALITA 2020 Diachronic Lexical Semantics (DIACR-Ita) Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020b. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Gemma Boleda, Marco Del Tredici, and Raquel Fernández. 2019. Short-term meaning shift: a distributional exploration. *Proceedings of the 2019; 2019 Jun 2-7; Minneapolis, United States of America. Stroudsburg (PA): ACL; 2019. p. 2069–75.*

Valerio Di Carlo, Federico Bianchi, and Matteo Palmonari. 2019. Training temporal word embeddings with a compass. *CoRR*, abs/1906.02376.

Alessandra Teresa Cignarella, Mirko Lai, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2020. Overview of the EVALITA 2020 Task on Stance Detection in Italian Tweets (SardiStance). In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2020)*. CEUR-WS.org.

Paul Cook and Suzanne Stevenson. 2010. Automatically Identifying Changes in the Semantic Orientation of Words. In *Proceedings of the Seventh conference on International Language Resources and Evaluation.*

Haim Dubossarsky, Daphna Weinshall, and Eitan Grossman. 2017. Outta control: Laws of semantic change and inherent biases in word representation models. In *Proceedings of the 2017 conference on* *empirical methods in natural language processing*, pages 1136–1145.

Lea Frermann and Mirella Lapata. 2016. A bayesian model of diachronic meaning change. *Transactions of the Association for Computational Linguistics*, 4:31–45.

Mario Giulianelli, Marco Del Tredici, and Raquel Fernández. 2020. Analysing Lexical Semantic Change with Contextualised Word Representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July. Association for Computational Linguistics.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*.

Yoon Kim, Yi-I Chiu, Kentaro Hanaki, Darshan Hegde, and Slav Petrov. 2014. Temporal analysis of language through neural language models. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 61–65.

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: a survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Eyal Sagi, Stefan Kaufmann, and Brady Clark. 2009. Semantic Density Analysis: Comparing Word Meaning across Time and Phonetic Space. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 104–111. Association for Computational Linguistics.

Dominik Schlechtweg, Anna Hätty, Marco Del Tredici, and Sabine Schulte im Walde. 2019. A wind of change: Detecting and evaluating lexical semantic change across times and domains. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 732–746.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. Semeval-2020 task 1: Unsupervised lexical semantic change detection. *arXiv preprint arXiv:2007.11464*.

Peter H Schönemann. 1966. A Generalized Solution of the Orthogonal Procrustes Problem. *Psychometrika*, 31(1):1–10.

Philippa Shoemark, Farhana Ferdousi Liza, Dong Nguyen, Scott Hale, and Barbara McGillivray. 2019. Room to glo: A systematic comparison of semantic change detection approaches with word embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 66–76.

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of computational approaches to lexical semantic change. *arXiv preprint arXiv:1811.06278*.

Luchen Tan, Haotian Zhang, Charles Clarke, and Mark Smucker. 2015. Lexical comparison between wikipedia and twitter corpora by using word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 657–661.

Adam Tsakalidis and Maria Liakata. 2020. Autoencoding word representations through time for semantic change detection. *arXiv preprint arXiv:2004.13703*.

Adam Tsakalidis, Marya Bazzi, Mihai Cucuringu, Pierpaolo Basile, and Barbara McGillivray. 2019. Mining the UK web archive for semantic change detection. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1212–1221.

Peter D Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

CL-IMS @ DIACR-Ita:
Volente o Nolente: BERT does not Outperform SGNS on Semantic Change Detection

Severin Laicher, Gioia Baldissin, Enrique Castañeda
Dominik Schlechtweg, Sabine Schulte im Walde
Institute for Natural Language Processing, University of Stuttgart
`{laichesn,baldisga,medinaeo,schlecdk,schulte}@ims.uni-stuttgart.de`*

Abstract

We present the results of our participation in the DIACR-Ita shared task on lexical semantic change detection for Italian. We exploit Average Pairwise Distance of token-based BERT embeddings between time points and rank 5 (of 8) in the official ranking with an accuracy of .72. While we tune parameters on the English data set of SemEval-2020 Task 1 and reach high performance, this does not translate to the Italian DIACR-Ita data set. Our results show that we do not manage to find robust ways to exploit BERT embeddings in lexical semantic change detection.

1 Introduction

Lexical Semantic Change (LSC) Detection has drawn increasing attention in the past years (Kutuzov et al., 2018; Tahmasebi et al., 2018). Recently, SemEval-2020 Task 1 provided a multi-lingual evaluation framework to compare the variety of proposed model architectures (Schlechtweg et al., 2020). The DIACR-Ita shared task extends parts of this framework to Italian by providing an Italian data set for SemEval's binary subtask (Basile et al., 2020a; Basile et al., 2020b). We present the results of our participation in the DIACR-Ita shared task on lexical semantic change for Italian. We exploit Average Pairwise Distance of token-based BERT embeddings (Devlin et al., 2019) between time points and rank 5 (of 8) in the official ranking with an accuracy of .72. While we tune parameters on the English data set of SemEval-2020 Task 1 and reach high performance, this does not transfer to the Italian DIACR-Ita data set. Our results show that we do not manage to find robust ways to exploit BERT embeddings in lexical semantic change detection.

2 Related Work

Most existing approaches for LSC detection are type-based (Schlechtweg et al., 2019; Shoemark et al., 2019). This means that not every word occurrence is considered individually (token-based) but a general vector representation that summarizes every occurrence of a word (including ambiguous words) is created. The results of the SemEval-2020 Task 1 (Martinc et al., 2020; Schlechtweg et al., 2020) showed that type-based approaches (Pražák et al., 2020b; Asgari et al., 2020) achieved better results than token-based approaches (Beck, 2020; Kutuzov and Giulianelli, 2020a). This is somewhat surprising since in the last years contextualized token-based approaches have achieved significant improvements over the static type-based approaches in several NLP tasks (Ethayarajh, 2019). Schlechtweg et al. (2020) suggest a range of possible reasons for this: (i) Contextual embeddings are new and lack proper usage conventions. (ii) They are pre-trained and may thus carry additional, and possibly irrelevant, information. (iii) The context of word uses in the SemEval data set was too narrow (one sentence). (iv) The SemEval corpora were lemmatized, while token-based models usually take the raw sentence as input. In the DIACR-Ita challenge (iii) and (iv) are irrelevant because raw corpora with sufficient context are made available to participants. We tried to tackle (i) by excessively tuning parameters and system modules on the English SemEval data set. (ii) can be tackled by fine-tuning BERT on the target corpora. However, our experiments on the English SemEval data set show that exceptionally high performances can be reached even without fine-tuning.

---

3 Experimental setup

The DIACR-Ita task definition is taken from SemEval-2020 Task 1 Subtask 1 (binary change detection): Given a list of target words and a diacronic corpus pair C_1 and C_2, the task is to identify the target words which have changed their meanings between the respective time periods t_1 and t_2 (Basile et al., 2020a; Schlechtweg et al., 2020).[1] C_1 and C_2 have been extracted from Italian newspapers and books. Target words which have changed their meaning are labeled with the value '1', the remaining target words are labeled with '0'. Gold data for the 18 target words is semi-automatically generated from Italian online dictionaries. According to the gold data, 6 of the 18 target words are subject to semantic change between t_1 and t_2. This gold data was only made public after the evaluation phase. During the evaluation phase each team was allowed to submit up to 4 predictions for the full list of target words, which were scored using classification accuracy between the predicted labels and the gold data. The final competition ranking compares only the highest of the scores achieved by each team.

4 System Overview

Our model uses BERT to create token vectors and the average pairwise distance to compare the token vectors from two times. The following chapter presents our model, how we have trained it and how we have chosen our submissions.

4.1 BERT

In 2018 Google has released a pre-trained model that ran over Wikipedia and books of different genres (Devlin et al., 2019): BERT (Bidirectional Encoder Representations from Transformer) is a language representation model, designed to find representations for text by analysing its left and right contexts (Devlin et al., 2019). Peters et al. (2018) show that contextual word representations derived from pre-trained bidirectional language models like BERT and ELMo yield significant improvements to the state-of-the-art for a wide range of NLP tasks. BERT can be used to analyse the semantics of individual words, by creating contextualized word representations, vectors that are sensitive to the context in which they appear (Ethayarajh, 2019). BERT can either create one vector for an input sentence (sentence embedding) or one vector for each input token (token embedding).[2]

Different pre-trained BERT models across languages can be downloaded. In this task, we have used the *bert-base-italian-xxl-cased* model for the Italian language[3] to create token embeddings.

The basic BERT version is transformer-based and processes text in 12 different layers. In each layer a contextualized token vector representation can be created for each word in an input sentence. It has been claimed that each layer captures different aspects of the input. Jawahar et al. (2019) suggest that the lower layers capture surface features, the middle layers capture syntactic features and the higher layers capture semantic features of the text. Each layer can serve as representation for the corresponding token by itself, or within a combination of multiple layers.

4.2 Average Pairwise Distance

Given two sets of token vectors from two time periods t_1 and t_2, the idea of Average Pairwise Distance (APD) is to randomly pick a number of vectors from both sets and measure their pair-wise distance (Sagi et al., 2009; Schlechtweg et al., 2018; Giulianelli et al., 2020; Beck, 2020; Kutuzov and Giulianelli, 2020b). The LSC score of the word is the mean average distance of all comparisons:

$$\mathrm{APD}(V, W) = \frac{1}{n_V * n_W} \sum_{v \in V, w \in W} d(v, w)$$

where V and W are two sets of vectors, n_V and n_W denote the number of vectors to be compared, and $d(v, w)$ refer to a distance measure (we used cosine distance (Salton and McGill, 1983)).

4.3 Tuning

The choice of BERT layers and the measure used to compare the resulting vectors (e.g. APD, COS or clustering) strongly influence the performance (Kutuzov and Giulianelli, 2020a). Hence, we tuned these parameters/modules on the English SemEval data (Schlechtweg et al., 2020). For the 40 English

---

[1] The time periods t_1 and t_2 were not disclosed to participants.

[2] The code of our system is available at `https://github.com/Garrafao/TokenChange`.

[3] `https://huggingface.co/dbmdz/bert-base-italian-xxl-cased`

target words we had access to the sentences that were used for the human annotation (in contrast to task participants who had only access to the lemmatized larger corpora containing more target word uses than just the annotated ones).

We tested several change measures regarding their ability to find the actual changing words. As part of our tuning, the APD measure produced the binary and graded LSC scores that best matched the actual LSC scores. We also tested the token vectors from different layers in order to check which one fits best to our task. The best layer combinations were the average of the last four layers and the average of the first and last layer of BERT. The highest F1-score for the binary subtask was .75 and a Spearman correlation of .65 for the graded subtask. Our results outperformed all official submissions of the shared tasks, of which the best were all type-based.

4.4 Threshold Selection

We created four predicted change rankings for the target words with BERT+APD. By experience and consideration of the shared tasks (Schlechtweg et al., 2020), we assumed that maximum half of all target words are actual words with a change. Therefore we always annotated at most 9 of 18 words with 1. First, we extracted for each target word a maximum of 200 sentences that contain the word in any token form. We limited the number of uses to 200 for computational efficiency reasons. Then, for each occurrence, we extracted and averaged the token vectors of (i) the last four layers of BERT, and (ii) the first and last layer. For our first submission (‘Last Four, 7’) we labeled those 7 words with ‘1’ that achieved the highest APD scores in layer combination (i). For our second submission (‘First + Last, 7’) we labeled those 7 words with ‘1’ that achieved the highest APD scores in layer combination (ii). In (i) and (ii) the same 9 words had the highest APD scores. Therefore, in our third submission (‘Average, 9’) exactly these 9 words were labeled with ‘1’. And for our last submission (Lemma, Average, 6’) we extracted only sentences in which the target words were present in their lemma form. Again we created the token vectors for the two layer combinations of BERT mentioned above. In both mentioned layer combinations the same 6 words had the highest APD scores. Therefore in our last submission exactly these 6 words were labeled with ‘1’ (similar as in submission 1).

5 Results

Table 1 shows the accuracy scores for the different submissions. The best result was achieved by combining the first and last layer of BERT (’First + Last, 7’ with .72), just like on the SemEval data. The second-best result was obtained by using the sentences where the target word occurred in its lemma form (’Lemma, Average, 6’ with .67). Only these two submissions outperformed the task baselines and the majority class baseline. The two lowest results were achieved by combining the last four layers of BERT (’Last Four, 7’ with .61) and by averaging the two layer combinations (’Average, 9’ with .61). The accuracy of our best submission (.72) was ranked at position 5 of the shared task, where the best task result was achieved by two different submissions and reached an accuracy of .94. Both submissions were based on type-based embeddings (Pražák et al., 2020a; Kaiser et al., 2020), clearly outperforming our system.

Submission	**Thresh.**	**Acc.**
First + Last	7	**.72**
Lemma, Average	6	.67
Majority Class Baseline	-	.66
Average	9	.61
Last Four	7	.61
Collocations Baseline	-	.61
Frequency Baseline	-	.61

Table 1: Overview accuracy scores for the four submissions with official task baselines. We also report a majority class baseline of a classifier predicting ‘0’ for all target Words.

6 Analysis

As aforementioned, the best performance of our system, achieved with ’First + Last, 7’, has an accuracy of .72. It erroneously predicts a meaning change for *cappuccio*, *unico* and *campionato*, while for *palmare* and *rampante* it does not detect the change as given by the gold standard.

We compared both corpora in order to find out if the target words are correctly labeled by the gold standard as well as to identify the possible reasons behind the wrong predictions of our model.

According to our analysis, we can state that the data matches the gold standard. *Cappuccio* is polysemous across both time periods t_0 and t_1 (“hood”, “cap”). However, 31% of the uses in t_1 are upper-cased, namely proper nouns (in contrast to the 4% in t_0), which might imply a different sense compared to the above-mentioned ones:

(1) BENEVENTO Il desiderio di il potere , il potere di il desiderio : ruota intorno a questo inquietante (e attualissimo) spunto il Festival di Benevento diretto da Ruggero **Cappuccio** .

*'BENEVENTO The desire of the power, the power of the desire: the Festival di Benevento directed by Ruggero **Cappuccio** revolves around this unsettling (and current) cue.'*

This skewed distribution of proper names in the two corpora is a possible reason for the wrong prediction of our model.

Throughout all target words, we noticed that the context provided by the previous and the following sentences (as given as input to our model) is often not related topic-wise; in some instances it seems as if the sentences are headlines, since they refer to different topics:

(2) M ROMA Sono quindici gli articoli in cui è suddiviso il provvedimento « antiracket » [...]. Roberta Serra ha vinto ieri lo slalom gigante di il **campionati** italiani femminili .

*'M ROMA The «antiracket» measure is divided into fifteen articles [...]. Roberta Serra won yesterday the giant slalom of the Italian female **championship**.'*

(3) ... le **uniche** azioni pericolose fiorentine sono arrivate quando il pallone e statu giocato su i lati di il Campo . costruzione di centrali idroelettriche , di miniere , canali e strade ...

*'...the **only** dangerous Florentine actions arrived when the ball was played on the sides of the field. Construction of hydroelectric power plants, mines, channels and streets...'*

This "headlines effect" occurs across the whole corpus. It can be traced back to the extraction process of the original corpus and may be a main source of error in our model. Despite not being representative, the following example shows that in some cases no centric window of any size would avoid considering unrelated context.

(4) REPARTO CONFEZIONI UOMO GIACCA cameriere bianca , in tessuto L' **unica** cosa certa è che il governo ha ricevuto una dura lezione da i professori .

*'MEN'S TAILORING DEPARTMENT white textile waiter JACKET The **only** certain thing is that the government has received a hard lesson by the professors.'*

Unico is another example of a word that was erroneously predicted as changing. Due to its abstract meaning ("only", "single", "unique"), it exhibits heterogeneous context across both time periods. Additionally, it can belong to different word classes (noun and adjective in (5) and (6), respectively).

(5) Rischiamo di rimanere gli **unici** a non aver dato mano a la ristrutturazione di le Forze Armate .

*'We risk remaining the **only ones** not having helped in the reorganization of the Armed Forces.'*

(6) ... è chiaro che l' **unica** cosa da fare sarebbe l' unificazione di le due aziende comunali ...

*'...it is clear that the **only** thing to do would be the unification of the two municipal companies...'*

With regards to the undetected changes, the term *palmare* (polysemous within and across word classes) acquires a novel sense in t_1. While it mostly has the meaning of "evident" in the 22 sentences of t_0 (see (7)), it additionally denotes "palmtop" in t_1 (see (8)).

(7) ... con evidenza **palmare** , la impossibilità di difendere una causa perduta ...

*'with **undeniable** evidence, the impossibility of defending a lost cause'*

(8) Per i palestinesi occorre una sistemazione provvisoria in attesa che gli europei si accordino per accoglier li . Potremmo citare in il lungo elenco il **palmare** Apple Newton troppo in anticipo su i tempi

*'A temporary arrangement is needed for the Palestinians while waiting for the Europeans to agree on hosting them. We could quote in the long list the **palmtop** Apple Newton too far ahead of its time'*

Note that also in (8), the topic of the previous and the target sentence is unrelated.

Rampante is a further case of undetected change. The phrase *cavallino rampante*, which metonymically denotes "Ferrari", dominates the usage of the word in t_0 (70%) and covers a (slightly) relevant share of the uses in t_1 (19%). We hypothesize that this leads to a large number of homogenous usage pairs masking the change from “rampant”, “unbridled” to “extremely ambitious” of *rampante*.

7 Conclusion

Our system comprising BERT+APD was ranked 5 in the DIACR-Ita shared task. The combination of BERT and APD did not perform as well as expected and much lower than the best type-based embeddings, but our best submission still outperformed all baselines. The high tuning results achieved on the SemEval data could not be transferred to the Italian data. One reason for this may be that a different BERT model was applied, trained on text of a different language. We have not tuned the Italian BERT model. It is therefore possible that the decrease in performance may be due to the change of the underlying BERT model. Furthermore, given that our model considers as input also the previous and the following sentences, the presence of semantically unrelated context could have played a significant role in mislabeling the target words.

Acknowledgments

Dominik Schlechtweg was supported by the Konrad Adenauer Foundation and the CRETA center funded by the German Ministry for Education and Research (BMBF) during the conduct of this study. We thank the task organizers and reviewers for their efforts.

References

Ehsaneddin Asgari, Christoph Ringlstetter, and Hinrich Schütze. 2020. EmbLexChange at SemEval-2020 Task 1: Unsupervised Embedding-based Detection of Lexical Semantic Changes. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020a. DIACR-Ita @ EVALITA2020: Overview of the EVALITA 2020 Diachronic Lexical Semantics (DIACR-Ita) Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020b. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Christin Beck. 2020. DiaSense at SemEval-2020 Task 1: Modeling sense change via pre-trained BERT embeddings. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Mario Giulianelli, Marco Del Tredici, and Raquel Fernández. 2020. Analysing lexical semantic change with contextualised word representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3960–3973, Online, July. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy, July. Association for Computational Linguistics.

Jens Kaiser, Dominik Schlechtweg, and Sabine Schulte im Walde. 2020. OP-IMS @ DIACR-Ita: Back to the Roots: SGNS+OP+CD still rocks Semantic Change Detection. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Andrey Kutuzov and Mario Giulianelli. 2020a. UiO-UvA at SemEval-2020 Task 1: Contextualised Embeddings for Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Andrey Kutuzov and Mario Giulianelli. 2020b. UiO-UvA at SemEval-2020 Task 1: Contextualised Embeddings for Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: A survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Matej Martinc, Syrielle Montariol, Elaine Zosa, and Lidia Pivovarova. 2020. Discovery Team at SemEval-2020 Task 1: Context-sensitive Embeddings not Always Better Than Static for Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium, October-November. Association for Computational Linguistics.

Ondřej Pražák, Pavel Přibákň, and Stephen Taylor. 2020a. UWB @ DIACR-Ita: Lexical Semantic Change Detection with CCA and Orthogonal Transformation. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Ondřej Pražák, Pavel Přibákň, Stephen Taylor, and Jakub Sido. 2020b. UWB at SemEval-2020 Task 1: Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Eyal Sagi, Stefan Kaufmann, and Brady Clark. 2009. Semantic density analysis: Comparing word meaning across time and phonetic space. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 104–111, Athens, Greece, March. Association for Computational Linguistics.

Gerard Salton and Michael J McGill. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company, New York.

Dominik Schlechtweg, Sabine Schulte im Walde, and Stefanie Eckmann. 2018. Diachronic Usage Relatedness (DURel): A framework for the annotation of lexical semantic change. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 169–174, New Orleans, Louisiana, USA.

Dominik Schlechtweg, Anna Hätty, Marco del Tredici, and Sabine Schulte im Walde. 2019. A Wind of Change: Detecting and evaluating lexical semantic change across times and domains. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 732–746, Florence, Italy. Association for Computational Linguistics.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Philippa Shoemark, Farhana Ferdousi Liza, Dong Nguyen, Scott Hale, and Barbara McGillivray. 2019. Room to Glo: A systematic comparison of semantic change detection approaches with word embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 66–76, Hong Kong, China. Association for Computational Linguistics.

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of computational approaches to diachronic conceptual change. *arXiv:1811.06278*.

OP-IMS @ DIACR-Ita:
Back to the Roots: SGNS+OP+CD still Rocks Semantic Change Detection

Jens Kaiser, Dominik Schlechtweg, Sabine Schulte im Walde
Institute for Natural Language Processing, University of Stuttgart
`{jens.kaiser,schlecdk,schulte}@ims.uni-stuttgart.de`

Abstract

We present the results of our participation in the DIACR-Ita shared task on lexical semantic change detection for Italian. We exploit one of the earliest and most influential semantic change detection models based on Skip-Gram with Negative Sampling, Orthogonal Procrustes alignment and Cosine Distance and obtain the winning submission of the shared task with near to perfect accuracy (.94). Our results once more indicate that, within the present task setup in lexical semantic change detection, the traditional type-based approaches yield excellent performance.

1 Introduction

Lexical Semantic Change (LSC) Detection has drawn increasing attention in recent years (Kutuzov et al., 2018; Tahmasebi et al., 2018). Recently, SemEval-2020 Task 1 provided a multilingual evaluation framework to compare the variety of proposed model architectures (Schlechtweg et al., 2020). The DIACR-Ita shared task extends parts of this framework to Italian by providing an Italian data set for SemEval's binary subtask (Basile et al., 2020a; Basile et al., 2020b).

We present the results of our participation in the DIACR-Ita shared task exploiting one of the earliest and most established semantic change detection models based on Skip-Gram with Negative Sampling, Orthogonal Procrustes alignment and Cosine Distance (Hamilton et al., 2016a). Based on our previous research (Schlechtweg et al., 2019; Kaiser et al., 2020) we optimize the dimensionality parameter assuming that high dimensionalities reduce alignment error. With our setting win the shared task with near to perfect accuracy (.94). Our results once more demonstrate that, within the present task setup in lexical semantic change detection, the traditional type-based approaches yield excellent performance.

2 Related Work

As evident in Schlechtweg et al. (2020) the field of LSCD is currently dominated by Vector Space Models (VSMs), which can be divided into type-based (Turney and Pantel, 2010) and token-based (Schütze, 1998) models. Prominent type-based models include low-dimensional embeddings such as the Global Vectors (Pennington et al., 2014, GloVe) the Continuous Bag-of-Words (CBOW), the Continuous Skip-gram as well as a slight modification of the latter, the Skip-gram with Negative Sampling model (Mikolov et al., 2013a; Mikolov et al., 2013b, SGNS). However, as these models come with the deficiency that they aggregate all senses of a word into a single representation, token-based embeddings have been proposed (Peters et al., 2018; Devlin et al., 2019). According to Hu et al. (2019) these models can ideally capture complex characteristics of word use, and how they vary across linguistic contexts. The results of SemEval-2020 Task 1 (Schlechtweg et al., 2020), however, show that contrary to this, the token-based embedding models (Beck, 2020; Kutuzov and Giulianelli, 2020) are heavily outperformed by the type-based ones (Pražák et al., 2020; Asgari et al., 2020). The SGNS model was not only widely used, but also performed best among the participants in the task. Its fast implementation and combination possibilities with different alignment types further solidify SGNS as the standard in LSCD. A common and surprisingly robust (Schlechtweg et al., 2019; Kaiser et al., 2020) practice is to align the time-specific SGNS embeddings with Orthogonal Procrustes (OP) and measure change with Cosine Distance (CD) (Kulka-

---

rni et al., 2015; Hamilton et al., 2016b). This has been shown in several small but independent experiments (Hamilton et al., 2016b; Schlechtweg et al., 2019; Kaiser et al., 2020; Shoemark et al., 2019) and SGNS+OP+CD has produced two of three top-performing submissions in Subtask 2 in SemEval-2020 Task 1 including the winning submission (Pömsl and Lyapin, 2020; Arefyev and Zhikov, 2020).

3 System overview

Most VSMs in LSC detection combine three subsystems: (i) creating semantic word representations, (ii) aligning them across corpora, and (iii) measuring differences between the aligned representations (Schlechtweg et al., 2019). Alignment is needed as columns from different vector spaces may not correspond to the same coordinate axes, due to the stochastic nature of many low-dimensional word representations (Hamilton et al., 2016b). Following the above-described success, we use SGNS to create word representations in combination with Orthogonal Procrustes (OP) for vector space alignment and Cosine Distance (CD) (Salton and McGill, 1983) to measure differences between word vectors. From the resulting graded change predictions we infer binary change values by comparing the target word distribution to the full distribution of change predictions between the target corpora. For our experiments we use the code provided by Schlechtweg et al. (2019).[1]

3.1 Semantic Representation

SGNS is a shallow neural network trained on pairs of word co-occurrences extracted from a corpus with a symmetric window. It represents each word w and each context c as a d-dimensional vector to solve

$$\arg\max_{\theta} \sum_{(w,c)\in D} \log \sigma(v_c \cdot v_w) + \sum_{(w,c)\in D'} \log \sigma(-v_c \cdot v_w),$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$, D is the set of all observed word-context pairs and D' is the set of randomly generated negative samples (Mikolov et al., 2013a; Mikolov et al., 2013b; Goldberg and Levy, 2014). The optimized parameters θ are v_{w_i} and v_{c_i} for $i \in 1, ..., d$. D' is obtained by drawing k contexts from the empirical unigram distribution $P(c) = \frac{\#(c)}{|D|}$ for each observation of (w, c), cf. Levy et al. (2015). After training, each word w is represented by its word vector v_w.

Previous research on the influence of parameter settings on SGNS+OP+CD lays the foundation for our parameter choices (Schlechtweg et al., 2019; Kaiser et al., 2020). Although this subsystem combination is extremely stable regardless of parameter settings, subtle improvements can be achieved by modifying the window size and dimensionality. A common hurdle in LSC detection is the small corpus size, increasing the standard setting for window size from 5 to 10 leads to the creation of more word-context pairs used for training the model. In addition, we also experiment with dimensionalities of 300 and 500. Higher dimensionalities alleviate the introduction of noise during the alignment process (Kaiser et al., 2020). We keep the rest of the parameter settings at their default values (learning rate α=0.025, #negative samples k=5 and sub-sampling t=0.001).

3.2 Alignment

SGNS is trained on each corpus separately, resulting in matrices A and B. To align them we follow Hamilton et al. (2016b) and calculate an orthogonally-constrained matrix W^*:

$$W^* = \arg\min_{W\in O(d)} \|BW - A\|_F$$

where the i-th row in matrices A and B correspond to the same word. Using W^* we get the aligned matrices $A^{OP} = A$ and $B^{OP} = BW^*$. Prior to this alignment step we length-normalize and mean-center both matrices (Artetxe et al., 2017; Schlechtweg et al., 2019).

3.3 Threshold

The DIACR-Ita shared task requires a binary label for each of the target words. However, CD produces graded values between 0.0 and 2.0 when measuring differences in word vectors between the two time periods. We tackle this problem by defining a threshold parameter, similar to many approaches applied in SemEval-2020 Task 1 (Schlechtweg et al., 2020). All words with a CD greater or equal than the threshold are labeled ‘1’, indicating change. Words with a CD less than the threshold are assigned ‘0’, indicating no change.

A simplified approach is to set the threshold such that the number of words is equal in both groups. This has many disadvantages: Mainly, it

---

[1] `https://github.com/Garrafao/LSCDetection`

relies on the assumption that the two groups are of equal size. This is rarely given in real world applications, especially if the focus is in one word at a time. Thus a more sophisticated approach is needed. In SemEval-2020's Subtask 1 many participants faced the same problem and developed various methods to solve it. Similar to the simplified approach, Zhou and Li (2020) only look at target words, and after fitting the histogram of CDs to a gamma distribution, set the threshold at the 75% density quantile. This approach resulted in good performance but is not always applicable due to its dependence on underlying properties of the test set. Amar and Liebeskind (2020) avoid the dependence on target words by randomly selecting 200 words and setting the threshold such that 90% of the 200 words have a lower distance than the threshold. A more careful selection of words is taken by Martinc et al. (2020), they look at the CD of semantically stable stop words, accumulate them in different bins and set the threshold to the upper limit of the bin containing fewer than $\#stopwords/\#bins$ words. Pražák et al. (2020) propose several methods. One of them is setting the threshold at the mean of the distances of all words in the corpus vocabulary. Our method for determining a threshold is very similar to Pražák et al. (2020), but instead of taking the mean, we use the mean + one standard deviation ($\mu + \sigma$) of all words in the corpus vocabulary.

4 Experimental setup

The DIACR-Ita task definition is taken from SemEval-2020 Task 1 Subtask 1 (binary change detection): Given a list of target words and a diacronic corpus pair C_1 and C_2, the task is to identify the respective target words which have changed their meaning between the time periods t_1 and t_2 (Basile et al., 2020a; Schlechtweg et al., 2020).[2] C_1 and C_2 have been extracted from Italian newspapers and books. Target words which have changed their meaning are labeled with the value '1', the remaining target words are labeled with '0'. Gold data for the 18 target words is semi-automatically generated from Italian online dictionaries. According to the gold data, 6 of the 18 target words are subject to semantic change between t_1 and t_2. This gold data was only made public after the evaluation phase. During the evaluation phase each team was allowed to submit 4 predictions for the full list of target words, which were scored using classification accuracy between the predicted labels and the gold data. The final competition ranking compares only the highest of the 4 scores achieved by each team.

entry	dim	threshold		ACC	AP
#2	300	$(\mu+\sigma)$	.76	**.944**	.915
#4	500	$(\mu+\sigma)$	.78	.889	.915
#1	300	(50:50)	.57	.833	.915
#3	500	(50:50)	.64	.833	.915
major. baseline		-		.667	.333
freq. baseline		unk.		.611	.418
colloc. baseline		unk.		.500	unk.

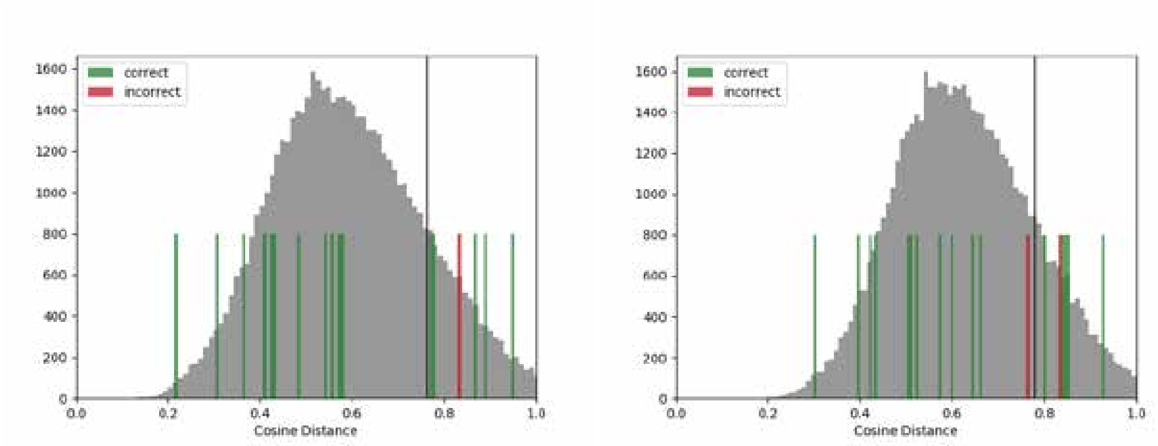
Table 1: Accuracy (ACC) and Average Precision (AP) for various parameter settings and thresholds and baselines; *freq. baseline*: Absolute frequency difference between the words in C_1 and C_2 and an unknown threshold; *colloc. baseline*: Bag of Words + CD and an unknown threshold; *major. baseline*: Every word labeled with '0'.

5 Results

We created target word rankings using SGNS+OP+CD with a dimensionality of 300 and 500 as described above. From these rankings our predictions are calculated using two different thresholding methods: (i) Splitting the targets into two equally-sized groups (50:50) and (ii) using the mean + one standard deviation (μ+σ) as threshold, refer to Section 3.3. The accuracy scores achieved in this way are listed in Table 1, alongside the official baselines *freq.* and *colloc.* and an additional *major.* baseline. Submission #2 is our highest scoring submission and won the DIACR-Ita task together with one other undisclosed submission. For both of our rankings the 50:50 threshold yielded lower accuracy than the μ+σ threshold. This is due to the imbalance of changed to unchanged target words in the test set. Using μ+σ as threshold resulted in an optimal split for the ranking created with d=300. For d=500 this threshold was slightly too high with a value of 0.78. The target word *palmare* which, according to the gold data, has undergone semantic change (label '1') has CD of 0.76 and was thus incorrectly labeled by our system. Figure 1 shows the histogram of CD values for all words of the corpus dictionary in gray. The green and

---

[2] The time periods t_1 and t_2 were not disclosed to participants.



(a) d=300

(b) d=500

Figure 1: Background shows histogram (in gray) of CDs for all words in the corpus vocabulary. The colored bars show the CDs of target words, green indicates that the target word was correctly labeled, red indicates incorrect labeling. Vertical line marks threshold value (mean + standard deviation).

red colored bars correspond target words. If the target word was correctly labeled the bar is green, incorrect labeled target words have red bars. From this visualisation we can see that there is a pronounced gap between the CDs of target words which have changed and those which have not. Our proposed threshold method of μ+σ tends to slightly overshoot this gap. This has lead to the lower accuracy of submission #4, despite the ranking allowing for a higher accuracy. In order to measure the quality of the rankings independent from the threshold we also report AP (Shwartz et al., 2017) in Table 1, confirming the potential equal performance.

The method of using the mean + one standard deviation of the CDs of all words in the corpus dictionary resulted in good accuracy, but leaves room for improvement. It tends to over-shoot the gap between unchanged and changed words slightly. Only using the mean shifts the tendency towards under-shooting the gap. The optimal threshold seems to lie somewhere in between. Though, this needs the be confirmed on other, larger, data sets. Furthermore, not all binary classification tasks are suitable for the approach of first creating a ranked list of graded change predictions and then choosing a threshold. The data set of SemEval-2020 Task 1 comprises two tasks, a binary and a ranked task for the same target words. It is not possible to achieve an accuracy of 1 on the binary task even if all the ranks are predicted correctly for the graded task, i.e., binary change is not just high graded change (Schlechtweg et al., 2020).

The one target word which our model labels incorrectly, across a variety of parameter settings, is *piovra*. According to the gold data this word has not undergone semantic change between t_1 and t_2, while our system labels it as changed. A possible explanation for the error may be differences in frequency: In C_1 *piovra* appears 35 times and in C_2 it appears 643 times. SGNS often struggles to create reliable embeddings for low frequency words (Kaiser et al., 2020). Alternatively, the error could be caused by discrepancies between gold labels and corpora. Basile et al. (2020a) state that the gold data is initially based on Italian online dictionaries such as 'Sabatini Coletti'. In a manual annotation process the gold data is further refined by providing human judges with up to 100 occurrences of each target word, for which they have to identify the used meaning according to the meanings listed in the dictionaries. A target word is labeled as changed if a meaning is observed in C_2 which has not been observed in C_1. Although not very likely, it is possible that this annotation method fails to detect novel senses in C_2. Sabatini Coletti reports that in addition to the sense "squid" *piovra* acquired a new sense "a secret criminal organisation deeply rooted in society" in 1983. This might explain why we detect *piovra* as a word which has undergone semantic change given that C_1 comprises texts from 1948 to 1970 and C_2 comprises texts from 1990 to 2014 (Basile et al., 2020a).

The DIACR-Ita task dataset is a very valuable contribution to the research field of LSC detection and extends the variety of available data sets to the Italian language. Nonetheless, two points are important when interpreting or results this data set: (i) it contains a small number of target words in combination with binary classification. This makes the data set vulnerable to randomness. (ii) The nature of the gold labels, in addition to possibly not being directly related to the corpus, it is unclear if they reflect semantic change as sense gain and sense loss as in SemEval's Subtask 1. The online dictionaries which create the basis for the gold data only state sense gains. Thus, it might possible for a word to completely lose a sense but still be labeled as unchanged.

6 Conclusion

We participated in the DIACR-Ita shared task using well-established type-based methods for diacronic semantic representations in combination with a carefully calculated threshold. We were able to reach the first place with a nearly perfect accuracy of .94 confirming once more the reliability of the type-based embeddings created by SGNS, OP as an alignment method and CD to measure differences between word vectors. The presented approach is very suitable for similar tasks as no fine-tuning of parameters is needed. Yet, the system relies on the assumption that graded change is indicative of binary classes.

Acknowledgments

Dominik Schlechtweg was supported by the Konrad Adenauer Foundation and the CRETA center funded by the German Ministry for Education and Research (BMBF) during the conduct of this study. We thank the task organizers and reviewers for their efforts.

References

Efrat Amar and Chaya Liebeskind. 2020. JCT at SemEval-2020 Task 1: Combined Semantic Vector Spaces Models for Unsupervised Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Nikolay Arefyev and Vasily Zhikov. 2020. BOS at SemEval-2020 Task 1: Word Sense Induction via Lexical Substitution for Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 451–462. Association for Computational Linguistics.

Ehsaneddin Asgari, Christoph Ringlstetter, and Hinrich Schütze. 2020. EmbLexChange at SemEval-2020 Task 1: Unsupervised Embedding-based Detection of Lexical Semantic Changes. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020a. DIACR-Ita @ EVALITA2020: Overview of the EVALITA2020 Diachronic Lexical Semantics (DIACR-Ita) Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020b. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Christin Beck. 2020. DiaSense at SemEval-2020 Task 1: Modeling sense change via pre-trained BERT embeddings. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Yoav Goldberg and Omer Levy. 2014. Word2vec explained: Deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv:1402.3722*.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016a. Cultural shift or linguistic drift? Comparing two computational measures of semantic change. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages

2116–2121, Austin, Texas. Association for Computational Linguistics.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016b. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1489–1501, Berlin, Germany. Association for Computational Linguistics.

Renfen Hu, Shen Li, and Shichen Liang. 2019. Diachronic sense modeling with deep contextualized word embeddings: An ecological view. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3899–3908, Florence, Italy. Association for Computational Linguistics.

Jens Kaiser, Dominik Schlechtweg, Sean Papay, and Sabine Schulte im Walde. 2020. IMS at SemEval-2020 Task 1: How low can you go? Dimensionality in Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2015. Statistically significant detection of linguistic change. In *Proceedings of the 24th International Conference on World Wide Web, WWW*, pages 625–635, Florence, Italy.

Andrey Kutuzov and Mario Giulianelli. 2020. UiO-UvA at SemEval-2020 Task 1: Contextualised Embeddings for Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: a survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Matej Martinc, Syrielle Montariol, Elaine Zosa, and Lidia Pivovarova. 2020. Discovery Team at SemEval-2020 Task 1: Context-sensitive Embeddings not Always Better Than Static for Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119, Lake Tahoe, Nevada, USA.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, Doha, Qatar.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237, New Orleans, LA, USA.

Martin Pömsl and Roman Lyapin. 2020. CIRCE at SemEval-2020 Task 1: Ensembling Context-Free and Context-Dependent Word Representations. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Ondřej Pražák, Pavel Přibákň, Stephen Taylor, and Jakub Sido. 2020. UWB at SemEval-2020 Task 1: Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Gerard Salton and Michael J McGill. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company, New York.

Dominik Schlechtweg, Anna Hätty, Marco del Tredici, and Sabine Schulte im Walde. 2019. A Wind of Change: Detecting and evaluating lexical semantic change across times and domains. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 732–746, Florence, Italy. Association for Computational Linguistics.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, March.

Philippa Shoemark, Farhana Ferdousi Liza, Dong Nguyen, Scott Hale, and Barbara McGillivray. 2019. Room to Glo: A systematic comparison of semantic change detection approaches with word embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 66–76, Hong Kong, China. Association for Computational Linguistics.

Vered Shwartz, Enrico Santus, and Dominik Schlechtweg. 2017. Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Valencia, Spain*, pages 65–75.

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of computational approaches to diachronic conceptual change. *CoRR*, abs/1811.06278.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *J. Artif. Int. Res.*, 37(1):141–188, January.

Jinan Zhou and Jiaxin Li. 2020. TemporalTeller at SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection with Temporal Referencing. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

UNIMIB @ DIACR-Ita: Aligning Distributional Embeddings with a Compass for Semantic Change Detection in the Italian Language

Federico Belotti
University of Milano-Bicocca
Viale Sarca 336, 20126
Milan, Italy
f.belotti8@campus.unimib.it

Federico Bianchi
Bocconi University
Via Sarfatti 25, 20136
Milan, Italy
f.bianchi@unibocconi.it

Matteo Palmonari
University of Milano-Bicocca
Viale Sarca 336, 20126
Milan, Italy
matteo.palmonari@unimib.it

Abstract

In this paper, we present our results related to the EVALITA 2020 challenge, DIACR-Ita, for semantic change detection for the Italian language. Our approach is based on measuring the semantic distance across time-specific word vectors generated with Compass-aligned Distributional Embeddings (CADE). We first generate temporal embeddings with CADE, a strategy to align word embeddings that are specific for each time period; the quality of this alignment is the main asset of our proposal. We then measure the semantic shift of each word, combining two different semantic shift measures. Eventually, we classify a word meaning as changed or not changed by defining a threshold over the semantic distance across time.

1 Introduction

Semantic change detection is the task of detecting if a word has shifted in meaning between different periods of time (Tahmasebi et al., 2018; Kutuzov et al., 2018). The DIACR-Ita (Basile et al., 2020a) challenge (at EVALITA (Basile et al., 2020b)) is meant to evaluate approaches for semantic change detection for the Italian Language.

The task is described as follows: for training, two corpora $t1$ and $t2$, consisting of text coming from different periods are given, for testing, a set of unlabeled target words is given, where for each of them a binary scores has to be predicted: 1 identifies lexical change between $t1$ and $t2$ while 0 does not.

In this paper, we present our approach to semantic change detection that is based on two components: 1) an alignment procedure to generate distributional vector spaces that are comparable for $t1$ and $t2$ and 2) the use of distance metrics to compute the degree of semantic change for a given word. Our alignment procedure is based on Compass Aligned Distributional Embeddings (CADE) proposed by Bianchi et al. (2020) (note the approach was introduced as Temporal Word Embeddings with a Compass by Di Carlo et al. (2019), but the name was changed to enforce the idea that the embeddings can be used to align more general corpora and not just diachronic ones). Given the aligned embeddings, we use two measures to compute the degree of change based on the similarities of the vectors in the embedded space. Our results show that our methodology for aligning spaces can be useful in detecting lexical semantic change.

2 Description of the System: Semantic Change Detection with Compass Aligned Embeddings

Our approach is based on measuring the semantic distance across time of time-specific word vectors generated with CADE and on the use of two measures for detecting semantic shifts i.e., the semantic distance between word vectors across time. This distance can be interpreted as a function of the words' self-similarity across time, where the similarity is measured by a linear combination of cosine and second-order similarity (Hamilton et al., 2016a).

Finally, a threshold over this self-similarity is used to classify a word as changed or not changed.

This methodology was applied also in the semantic shift detection challenge presented at SemEval2020 (Schlechtweg et al., 2020) (to which we participated after the end of the challenge). The challenge allowed us to explore and understand how the alignment and our self-similarity behaved. In the classification task of the SemEval2020 challenge (the one similar to this task),

---

we eventually achieved 0.703, 0.771, 0.725, 0.742, in accuracy for respectively the English, German, Latin and Swedish languages; these results have been obtained with extensive parameter search given the gold standard available in the post-evaluation.[1] In DIACR-Ita, the threshold and few other hyper parameters can be heuristically set to account for the limited number of possible submissions. In the next subsections we provide more details about the alignment methodology and the similarity function; more details about how we set the hyper parameters are provided in Section 3.

2.1 Aligning Embeddings

Word2vec (Mikolov et al., 2013) is a useful methodology to generate vectors of words allowing us to study word similarity through vector similarity. However, due to the stochasticity of the training procedure, running word2vec on different corpora creates word vectors that are not comparable. Thus, an alignment procedure that puts the temporal word vectors in the same space is needed.

There are different approaches to generate these aligned embeddings (see for example the work by (Hamilton et al., 2016b) and (Yao et al., 2018)). In this paper, we generate aligned embeddings with Compass Aligned Distributional Embeddings (CADE) (Bianchi et al., 2020) (See Figure 1 for a schematic description of the model). CADE is a strategy to align word embeddings that are specific for each time period that extends the word2vec Continuous Bag Of Word (CBOW) model proposed by Mikolov et al. (2013). CADE can be used to generate aligned temporal word embeddings (i.e., time-specific vectors of words, like "$amazon^{1974}$") from the different slices.

Given in input a set of slices of text, where each slice corresponds to text coming from a specific period of time, the alignment procedure is as follows:

First, the text from all the slices is concatenated and CBOW is run on this corpus in order to obtain a "compass" model, i.e., a model defining the embedding space. The CBOW model uses two matrices to generate the embeddings (**U** and **C** in Figure 1), one for the context words and one for the target words. The target word matrix of the compass is then used to initialize the target matrices for each new CBOW model fitted on each of the slices. During training, these new target matrices are frozen, i.e., they are not updated during the training on the slice. This ensures that at the end of the training process, the various temporal embeddings are all aligned in the same embedding space, making them comparable without losing their individual temporal distinctions. We use the publicly available online implementation of CADE.[2]

2.2 Computing Semantic Change

Once the embeddings are aligned, we need measures to evaluate the degree of semantic change. We compute the semantic shift of each word, i.e. the semantic distance between word vectors across time using the combination of two different measures: Local Neighbors (ln), introduced by Hamilton et al. (2016a) and cosine similarity (cos), merging them with a weighted linear combination into a new measure called *Move*.

Local Neighbors *ln* is based on the similarity between a word and its neighbor words in the two different time periods. Essentially we compute the degree of semantic change of the word w in two slices by first collecting the nearest neighbors (NNs) of $\mathbf{w}^t$ and $\mathbf{w}^{t+1}$ in the two respective slices, then given the embeddings at time t the similarities between the vector of $\mathbf{w}^t$ and the vectors of all the neighbors are computed.[3] The same process is run for time $t+1$ with $\mathbf{w}^{t+1}$, eventually giving us two vectors of similarity scores. These two vectors are again compared using cosine similarity. The higher the value of this measure the less the vector has changed with respect to its neighbors and thus the less the word should have shifted in meaning.

Cosine Similarity The second measure we use is simply the cosine similarity of the vectors of a word in two different time periods. Similarly as before , the higher the value the less the vector has changed and thus the less the word should have shifted in meaning.

The *Move* Measure We merge these measures together using a weighted linear combination, that is:

$$\begin{aligned} s(w^t, w^{t+1}) = (1-\lambda) \cdot \text{ln}(w^t, w^{t+1}) \\ +\lambda \cdot \text{cos-sim}(\mathbf{w}^t, \mathbf{w}^{t+1}) \end{aligned}$$

---

[1] Check the *belerico* entry in the challenge leaderboard at `https://competitions.codalab.org/competitions/20948#results`

[2] `http://github.com/vinid/cade`

[3] When a neighbor is missing in one time slice, we replace it with the average vector of the space.

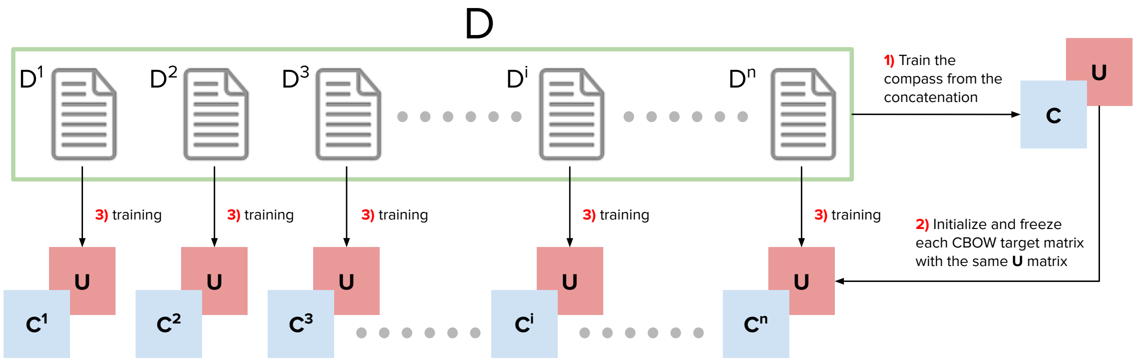

D
D1
D2
D3
Di
Dn
1) Train the compass from the concatenation
U
C
3) training
3) training
3) training
3) training
3) training
2) Initialize and freeze each CBOW target matrix with the same U matrix
U
U
U
U
U
C1
C2
C3
Ci
Cn

Figure 1: An high level overview of the Compass Aligned Distributional Embeddings model.

with $\lambda \in [0, 1]$. In particular λ express the usage strength of the two measures: a high λ will shift *Move* towards the cosine similarity, while a low one towards the *ln* measure. As introduced before we classify if the meaning has changed by defining a threshold over s (more details about this are presented in the next Section).

3 Experimental Evaluation

The dataset provided by the challenge's organizers (Basile et al., 2020a) is a collection of documents extracted by newspapers written in the Italian language labeled with temporal information. Participants must train their models only on the data provided, so a pre-processed corpus is given: tab separated, with one token per line, where for each token there are its corresponding part-of-speech (POS) tag and lemma, with sentences separated by empty lines. The corpus is split into two slices, each belonging to a specific period of time, $t1$ and $t2$, where $t1 < t2$.

3.1 Dataset

For the training data we used the *flat* version with only the lemmas, obtained by the organizers' script (Basile et al., 2020a); in addition we applied a pre-processing step, in which we removed punctuation and non alpha-numeric symbols and we kept only those sentences with at least two tokens.

3.2 Models Considered

We use the embeddings aligned with CADE and the *move* measure. The parameters of the moving average we need to consider are: the number of nearest neighbors (NNs) to be collected by *ln*, λ for the moving average and the threshold for the similarity. We set the threshold to decide if a word is stable or not is set to 0.7, with the decision given by:

$$\begin{cases} 0 & \text{if } s(w^t, w^{t+1}) \geq 0.7 \\ 1 & \text{otherwise} \end{cases}$$

Essentially, the less changed are the two vectors of the words (for *cos*) and the neighbors (for *ln*) the more the word has been stable between the two time periods. As heuristics we chose $\lambda \in \{0.3, 0.5, 0.7\}$ to evaluate the relationship between the two measures used to build *move*, and we set to 22 the number of nearest neighbors to be considered by the *ln*; this is the general setup that gave the results that have been submitted to the challenge.

We trained CADE for 10 epochs to learn 100-dimensional vectors, with the window size set to 5, 10 negative examples for every positive one, with the initial learning rate set to 0.025 and decreased linearly during training.

As other models, in the post evaluation we also considered one that only uses the *cos* (CADE (cos)) similarity measure and one that uses only the *ln* metric CADE (ln)) (again with 0.7 as threshold and with the number of NNs for *ln* set to 22).

As baselines, the authors propose to use *baseline-freq*, that is the absolute value of the difference between the words' frequencies and *baseline-colloc*, where the Bag-of-Collocations of the two words in the two different periods is built and then cosine similarity is applied. A threshold is used on both metrics to define semantic change (Basile et al., 2020a). We report also the results of the other participants.

	λ	Acc.
team$_1$	/	0.944
team$_2$	/	0.944
team$_3$	/	0.889
CADE (move)†	0.3	**0.833**
team$_4$	/	0.833
team$_5$	/	0.833
team$_6$	/	0.778
team$_7$	/	0.722
team$_8$	/	0.667
team$_9$	/	0.611
baseline-colloc	/	0.611
baseline-freq	/	0.500
CADE (move)†	0.5	0.722
CADE (move)†	0.7	0.722
CADE (cos)	/	0.722
CADE (ln)	/	**0.889**

Table 1: Accuracy scores for the binary classification w.r.t. the other participants to the challenge. † identifies our submitted results.

3.3 Results

The evaluation metric used in this challenge is the accuracy, that is, the number of correct predictions over the target data. Table 1 shows the results. Our model was the third most accurate. However, in the post-evaluation we discovered that just using the *ln* metric and ignoring the use of *cos* (this is equivalent to using $\lambda = 0$ in our *move* measure) improves the performance leading to the second best accuracy score in the leaderboard.

4 Discussion

Our results show that CADE (Bianchi et al., 2020) is an effective method to generate aligned embeddings for the Italian language. This result, together with those obtained on the SemEval2020 data, suggest that CADE can support models of semantic shift detection in several languages. Indeed, we show that in combination with some simple semantic change measures it is possible to provide a good model for semantic change detection that can be subsequently extended with more features. Appendix A contains some more detailed examples of the words that CADE (ln) and CADE (move), with lambda set to 0.3, could not classify correctly. Also, we show the neighborhood for some of those words to give more context on why we get those errors. A more precise use of pre-processing techniques with the combination of other metrics to compute semantic change might help in reducing these errors.

References

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020a. DIACR-Ita @ EVALITA2020: Overview of the EVALITA2020 Diachronic Lexical Semantics (DIACR-Ita) Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020b. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Federico Bianchi, Valerio Di Carlo, Paolo Nicoli, and Matteo Palmonari. 2020. Compass-aligned distributional embeddings for studying semantic differences across corpora. *arXiv preprint arXiv:2004.06519*.

Valerio Di Carlo, Federico Bianchi, and Matteo Palmonari. 2019. Training temporal word embeddings with a compass. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6326–6334.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016a. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2116–2121, Austin, Texas, November. Association for Computational Linguistics.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016b. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501, Berlin, Germany, August. Association for Computational Linguistics.

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: a survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. Semeval-2020 task 1: Unsupervised lexical semantic change detection. *arXiv preprint arXiv:2007.11464*.

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of computational approaches to lexical semantic change. *arXiv preprint arXiv:1811.06278*.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic word embeddings for evolving semantic discovery. In *Proceedings of the eleventh acm international conference on web search and data mining*, pages 673–681.

A CADE Misclassifications

We report in Tables 2 and 3 CADE's misclassifications with the two best metrics, namely CADE (move) with λ = 0.3 and CADE (ln). Eventually, we also show in Tables 4 and 5 some examples of neighborhood for the target words.

Word	Pred	True
trasferibile	changed	not changed
pacchetto	changed	not changed
piovra	changed	not changed

Table 2: Wrong predictions done by CADE (move) with λ = 0.3.

Word	Pred	True
pacchetto	changed	not changed
rampante	not changed	changed

Table 3: Wrong predictions done by CADE (ln).

Table 4 shows the top 10 nearest neighbors of the target word "pacchetto" and we think CADE classifies its meaning as changed because during time t_1 the meaning is more focused in the economic area, as one can see from neighbors like "azionario", "obbligazione" or "contante" (translated to "stock" as referred to the market, "bond" and "cash" resp.); while at time t_2 shifts to a more political sense, as shown by words such as "decreto" or "emendamento" ("decree" and "amendment" resp.).

t_1	t_2
azionario	maxiemendamento
obbligazione	finanziaria
azionista	decretone
azionano	decreto
edison	ddl
casseforte	emendamento
contante	liberalizzazioni
siap	decretere
shell	maxidecreto
prestire	ecobonus

Table 4: First 10 nearest neighbors by cosine similarity of the word "pacchetto" from t_1 and t_2

The same it seems to happen for the target word "piovra", as one can see from Table 5, where at time t_1 CADE gathers senses from both considering it as the animal, for example from the word "tentacle", or as someone tied to crime in general, given words such as "profittatore" or "ruberia" ("profiteer" and "robbery" resp.); while at time t_2 captures a shift towards the Italian crime TV series "La piovra", as emerge from words such as "fiction", "camorra" or "retequattro", which is an Italian television channel.

t_1	t_2
tentacolo	fiction
ingordigia	sceneggiato
profittatore	tentacolo
somaro	camorrere
feudatario	retequattro
insaziabile	raidue
impere	puntato
ruberia	camorra
zanne	gomorra
putrido	miniserie

Table 5: First 10 nearest neighbors by cosine similarity of the word "piovra" from t_1 and t_2

NLP-CIC @ DIACR-Ita: POS and Neighbor Based Distributional Models for Lexical Semantic Change in Diachronic Italian Corpora*

Jason Angel
CIC, Instituto Politécnico Nacional
Mexico City, Mexico
`ajason08@gmail.com`

Carlos A. Rodriguez-Diaz
CIC, Instituto Politécnico Nacional
Mexico City, Mexico
`amnet04@gmail.com`

Alexander Gelbukh
CIC, Instituto Politécnico Nacional
Mexico City, Mexico
`www.gelbukh.com`

Sergio Jimenez
Instituto Caro y Cuervo
Bogota, Colombia
`sergio.jimenez@caroycuervo.gov.co`

Abstract

We present our systems and findings on unsupervised lexical semantic change for the Italian language in the DIACR-Ita shared-task at EVALITA 2020. The task is to determine whether a target word has evolved its meaning with time, only relying on raw-text from two time-specific datasets. We propose two models representing the target words across the periods to predict the changing words using threshold and voting schemes. Our first model solely relies on part-of-speech usage and an ensemble of distance measures. The second model uses word embedding representation to extract the neighbor's relative distances across spaces and propose "the average of absolute differences" to estimate lexical semantic change. Our models achieved competent results, ranking third in the DIACR-Ita competition. Furthermore, we experiment with the $k\_neighbor$ parameter of our second model to compare the impact of using "the average of absolute differences" versus the cosine distance used in (Hamilton et al., 2016).

1 Introduction

Lexical semantic change has recently gained interest in the intersection of natural language processing and historical linguistics[1], therefore several datasets have been proposed for different languages (Schlechtweg et al., 2020a). This work take place in the context of DIACR-Ita (Basile et al., 2020a) at EVALITA 2020 (Basile et al., 2020b), which sets the task for the Italian language in a fully unsupervised fashion. From DIACR-Ita we received 18 target words[2], and two time-specific and preprocessed Italian corpora, namely *T0* and *T1*, which include part-of-speech tagging and lemmatization information.

We present two perspectives to approach the problem, regarding how we represent target words and estimate the lexical-semantic change across datasets. (1) uses the POS distribution of target words as representation, and employee an ensemble of distance measures for the estimation. (2) uses the target words neighbor similarities as representation and one (of two proposed) similarity measure for estimation.

The following three sections describe the previous works, modeling, and results we obtained using these approaches. Following that, section 5 (Discussion) focuses on examine the second approach to illustrate the impact of the k parameter in similarity measures and the discriminatory performance of our embedding-based model.

2 Related works

Previous works have employed similar approaches to address the unsupervised lexical-semantic-change task, mostly for the English language (Schlechtweg et al., 2020a; Asgari et al., 2020; Schlechtweg et al., 2020b). Our first approach follows the idea of "syntactic models" (Kulkarni et al., 2015), which supposes that some semantic changes could imply a new syntactic functionality, such as acquiring a new part-of-speech category, as Kulkarni et al. (2015) exemplify: the word "ap-

---

[1] see `https://languagechange.org/`

[2] 'egemonizzare', 'lucciola', 'campanello', 'trasferibile', 'brama', 'polisportiva', 'palmare', 'processare', 'pilotato', 'cappuccio', 'pacchetto', 'ape', 'unico', 'discriminatorio', 'rampante', 'campionato', 'tac', 'piovra'

ple" increased his use as a proper name in the '80s.

On the other hand, our second approach follows the idea of "embedding-based models" (Kulkarni et al., 2015; Hamilton et al., 2016; Shoemark et al., 2019), which compares word vector representations from each period using an aligned space, which can be computed either globally (for the full model) or locally (only for a target words). A common strategy for local aligning is to perform a new transformation representing the target words (the same from different spaces) through neighborhood structures, under the assumption that independent training of embedding algorithms on comparable corpora will still produce similar neighborhood structures (Kulkarni et al., 2015).

Our second approach align the space locally using the nearest neighbors of target words as shared feature.

3 Methodology

In this section we provide a detailed description of our systems, each of them composed of two stages, the model and the voting scheme.

3.1 Models

We represented the target words as vectors for each time of period using two perspectives that originate our submitted systems: the POS-model and the embedding-model. The word representations are comparable across spaces, and serve to estimate the lexical semantic change through similarity and distance measures, from which we finally predict the changing words using thresholds and voting schemes.

POS-model: we simply analyzes the Part-Of-Speech distribution as the relative frequency over the datasets taking the top 4 most common POS-tags, namely ADJ, NOUN, PROPN and VERB. The produced four-dimensional vector pairs are then used to assess the lexical semantic change of each target word from the perspective of their Euclidean, Manhattan and Cosine distances[3].

Embedding-model: We lowercase and concatenate each word form with its corresponding POS to build embedding models for each dataset T, namely $T0$ and $T1$. Specifically, we used Word2Vec models(Mikolov et al., 2013) with the CBOW version from gensim[4] with the following parameters: size of 256, window of 5, min_count of 3. Then we take the common vocabulary of both $V_c = V_{(TO)} \cap V_{(T1)}$, and use it to constraint the set of top k nearest neighbors of the target word only from $T0$[5], i.e., $N_k = \{n_1, n_2 ... n_k\}, n_k \in V_c$, to build the representation of the target word for each space based on its neighbor proximity, i.e. $\vec{W}^T = [cos\_sim(\vec{w}, \vec{n_k}) | n_k \in T]$, and estimate the lexical semantic change using the following two formulas[6]:

$$\textit{avg.abs.diff} = Avg(|\vec{W}^{T0} - \vec{W}^{T1}|) \qquad (1)$$

$$\textit{cosine\_similarity} = cos\_sim(\vec{W}^{T0}, \vec{W}^{T1}) \qquad (2)$$

The average of absolute point-wise differences (avg.abs.diff for short) works under the assumption that the neighbors a non-changing word preserves their relative distance each other across diachronic representations. Therefore, the value of this measure increases according to the lexical semantic change a target word underwent. In our submission we used $k = 10$.

3.2 Threshold and voting schemes

Given that DIACR-Ita is an unsupervised task we experiment with different threshold and voting schemes to aggregate the measure ranks and determine which target words have underwent a lexical semantic change. As a result, we propose three voting schemes from which we derive our results.

System1: Upper-third of distance ranks (used for POS model): we sorted the target words in descending order and rank their positions according to the Euclidean, Manhattan and Cosine distances. We then sum all these ranks and sort in descending order again. Finally we label the first upper-third part of this list as changing words.

System2: Half intersection (used for the embedding model): We sort the target words in descending and ascending order for the lineal-difference scores (1) and the cosine-similarity (2) respectively. Then we take the top 50% of each group, and intersect them to obtained the words that we predicted as changing words.

System3: Union of Upper-third and Half intersection: This is just the union of results from System1 and System2.

---

[3] we noticed that at this point Kulkarni et al. (2015) uses Jenssen-Shannon divergence measure

[4] https://radimrehurek.com/gensim/models/word2vec.htm

[5] Unlike Hamilton et al. (2016) that takes the top-k neighbors from each model and union them ($N_k = N_k^{T0} \cup N_k^{T1}$).

[6] Hamilton et al. (2016) only uses cosine distance.

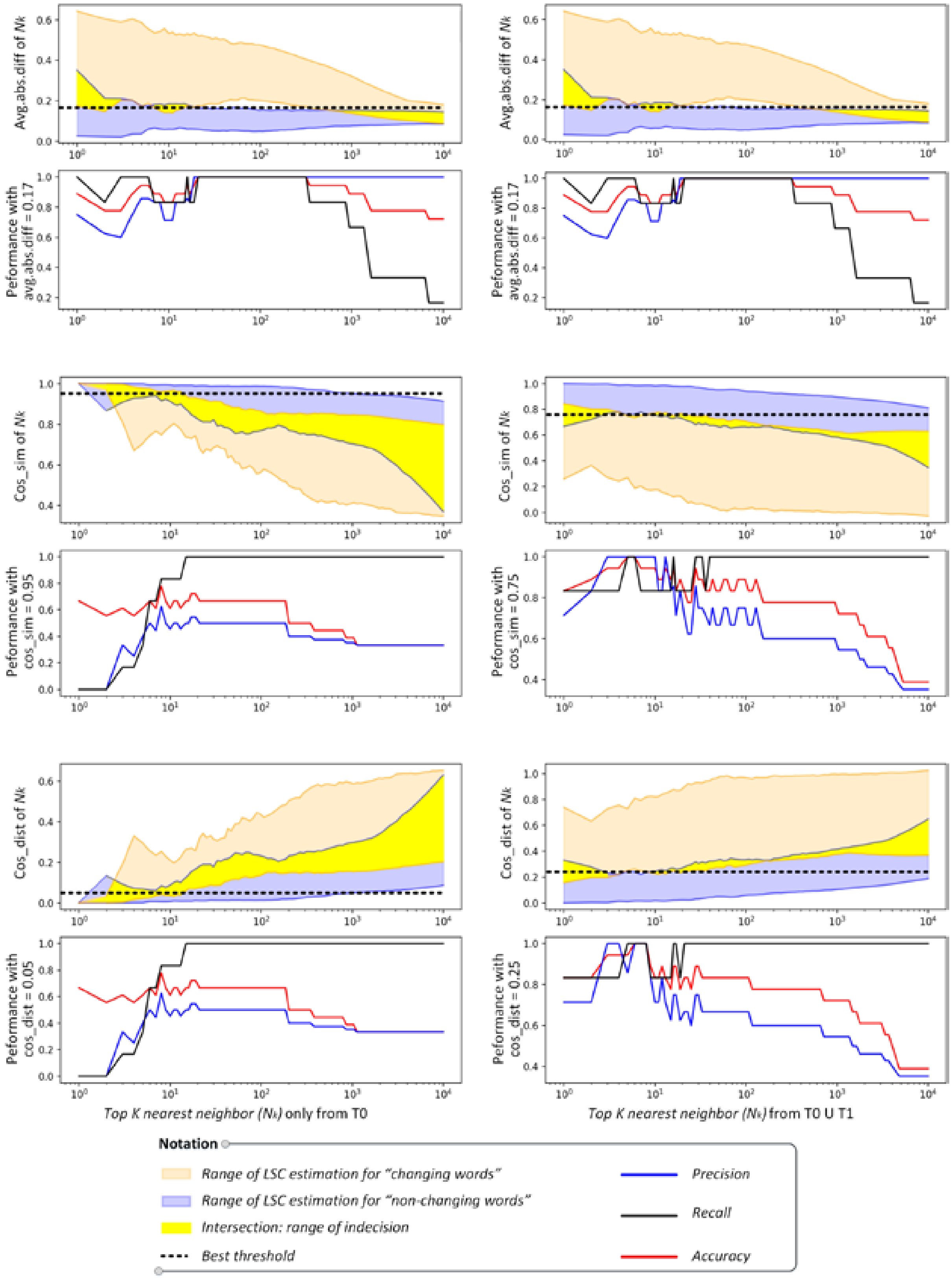


Figure 1: Analysis of estimation ranges of lexical semantic change by neighbor-based distributional models using several measures, and two aggregation methods: only from T0 (at left) and the union of T0 and T1 (at right).

4 Results

Table 1 summarize the results we obtained during the competition. One can see that the system3 which combine system 2 and 3 also combine its false positive results while removing the False negative ones. We officially ranked third place with the System1, which in spite of exhibit equal results than System3, is much simpler. We also made error analysis over the system 1 for the case of "polisportiva" at Table 2, the results show that there is a large difference in the POS usage of "polisportiva" across the time periods, NOUN and PROPN seems to invert their distribution usage. We also made the code[7] publicly available for the systems reproduction.

S (#)	Acc.	False positive	False negative
1	0.88	polisportiva	rampante
2	0.83	egemonizzare	lucciola, ape
3	0.88	polisportiva, egemonizzare	–

Table 1: Submission results using Accuracy

Corpus	ADJ	NOUN	PROPN	VERB
T0	0.04	0.18	0.76	0.02
T1	0.02	0.61	0.34	0.02

Table 2: POS usage of "polisportiva" over the time periods

5 Discussion: Post-evaluation analysis

In this section we employee the gold-standard labels of the target words to analyze at Figure 1 the capabilities of our neighbor-based embedding-model using several settings. To this end, we divide the Figure 1 into vertical and horizontal views. The vertical view defines 3 groups (from top to bottom), that serves to compare the three proposed measures to estimate the lexical semantic change, namely the average of absolute differences, cosine similarity and cosine distance. At the same time, the horizontal view serves to compare the strategy of only use T0 (at left), versus the union of T0 and T1 (at right), to define the top nearest neighbors N_k.

Next, each of the charts shows an analysis of the model for the given measure across the k parameter. The area charts represent by color regions the ranges that discriminate the lexical semantic change of target words: "changing words" (orange region) and "non-changing words" (purple region). The yellow region in the middle marks the intersection of these ranges, thus, words falling into the yellow region are difficult to estimate, according to the used measure. We also identified the threshold that best discriminate changing and non-changing target words, and draw a dashed line at that point. On the other hand, the line charts throw light on all the possible performance that the model could obtain by changing the k parameter while using the best possible discriminator threshold.

These results suggest that the "average of absolute difference" is the best proposed measure because it obtains a better performance for a larger number of k values as displayed in the line charts. Moreover, the "average of absolute difference" offers a larger range for possible discriminator thresholds (as shown in the area charts), and it is tolerant to the N_k election, since it remains almost unchanged while using either the union of T0 and T1, or only T0. One can also note that the area charts for the cosine similarity versus cosine distance mirror each other, as expected, and their performance is the same when using N_k only from T0 (at left), but slightly differ when using N_k as the union of T0 and T1 (at right).

6 Conclusion

We tackle the problem of unsupervised lexical semantic change on two time-specific datasets for 18 target words in Italian language. Our two approaches focus on the representation of target words across the provided diachronic datasets, they use part-of-speech usage and nearest neighbors respectively, and a number of measures between these representation to estimate the lexical semantic change. Then, this estimation serves to decide which target words underwent a change by the use of proposed threshold and voting schemes. Afterward, in the last part of this work, we analyzed the nearest neighbor model through the impact of deciding the k parameter and the similarity measure that estimates the lexical semantic change. Our results for the DIACR-Ita datasets suggest that the estimations of "the average of ab-

---

[7] `https://github.com/ajason08/evalita2020_diacrita`

solute differences" measures have a better performance for a larger number of k values than the cosine similarity and the cosine distance used in Hamilton et al. (2016).

As for future work, we plan to investigate different mechanism for deciding the threshold, and explore other diachronic datasets for other languages such as English, German and Spanish. We also believe that further experiments on a larger number of target words will benefit the reliability of models to judge the lexical semantic change in an unsupervised fashion.

Acknowledgments

The authors thank CONACYT for the computer resources provided through the INAOE Supercomputing Laboratory's Deep Learning Platform for Language Technologies.

References

Ehsaneddin Asgari, Christoph Ringlstetter, and Hinrich Schütze. 2020. Unsupervised embedding-based detection of lexical semantic changes. *arXiv preprint arXiv:2005.07979*.

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020a. DIACR-Ita @ EVALITA2020: Overview of the EVALITA2020 Diachronic Lexical Semantics (DIACR-Ita) Task. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020b. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2116–2121, Austin, Texas, November. Association for Computational Linguistics.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2015. Statistically significant detection of linguistic change. In *WWW '15: Proceedings of the 24th International Conference on World Wide Web)*, page 625–635, Florence, Italy, May. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020a. Semeval-2020 task 1: Unsupervised lexical semantic change detection.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020b. Semeval-2020 task 1: Unsupervised lexical semantic change detection. *arXiv preprint arXiv:2007.11464*.

Philippa Shoemark, Farhana Ferdousi Liza, Dong Nguyen, Scott Hale, and Barbara McGillivray. 2019. Room to Glo: A systematic comparison of semantic change detection approaches with word embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 66–76, Hong Kong, China, November. Association for Computational Linguistics.

TRACK
“NEW CHALLENGES IN LONG-STANDING TASKS”

AcCompl-it: Acceptability & Complexity evaluation

AcCompl-it @ EVALITA2020: Overview of the Acceptability & Complexity Evaluation Task for Italian

Dominique Brunato[1], **Cristiano Chesi**[2], **Felice Dell'Orletta**[1],
Simonetta Montemagni[1], **Giulia Venturi**[1], **Roberto Zamparelli**[3]
[1]ILC-CNR, Via G. Moruzzi 1, Pisa, Italy
[2]NETS-IUSS, P.zza Vittoria 15, Pavia, Italy
[3]CIMeC-UNITRENTO, Corso Bettini 31, Rovereto Italy
`[name.surname]@ilc.cnr.it`, `cristiano.chesi@iusspavia.it`,
`roberto.zamparelli@unitn.it`

Abstract

The Acceptability and Complexity evaluation task for Italian (AcCompl-it) was aimed at developing and evaluating methods to classify Italian sentences according to Acceptability and Complexity. It consists of two independent tasks asking participants to predict either the acceptability or the complexity rate (or both) of a given set of sentences previously scored by native speakers on a 1-to-7 points Likert scale. In this paper, we introduce the datasets distributed to the participants, we describe the different approaches of the participating systems and provide a first analysis of the obtained results.

1 Motivation

The availability of annotated resources and systems aimed at predicting the level of grammatical acceptability or linguistic complexity of a sentence (see, among others, (Warstadt et al., 2018; Brunato et al., 2018)) is becoming increasingly relevant for different research communities that focus on the study of language. From the Natural Language Processing (NLP) perspective, the interest has been recently prompted by automatic generation systems (e.g. Machine Translation, Text Simplification, Summarization) mostly based on Deep Neural Networks algorithms (Gatt and Krahmer, 2018). In this scenario, resources and methods able to assess the quality of automatically generated sentences or devoted to investigate the ability of artificial neural networks to score linguistic phenomena on the acceptability and complexity scales are of pivotal importance. From the theoretical linguistics perspectives, controlled datasets containing acceptability judgments and analyzed with machine learning techniques can be useful to test the extent to which syntactic and semantic deviance can be induced from corpus data alone, especially for low frequency phenomena (Chowdhury and Zamparelli, 2018; Gulordava et al., 2018; Wilcox et al., 2018), while the same data, seen from a psycholinguistic angle, can shed light on the relation between complexity and acceptability (Chesi and Canal, 2019), and on the extent to which measures of on-line perplexity in artificial language models can track human parsing preferences (Demberg and Keller, 2008; Hale, 2001).

The Acceptability & Complexity evaluation task for Italian (AcCompl-it) at EVALITA 2020 (Basile et al., 2020) is in line with this emerging scenario. Specifically, it is aimed at developing and evaluating methods to classify Italian sentences according to Acceptability and Complexity, which can be viewed as two simple numeric measures associated with linguistic productions. Among the outcomes of the task, we also include the creation of a set of sentences annotated with acceptability and complexity human judgments that we are going to share with the linguistic community. While datasets annotated for acceptability exist for English, see in particular the COLA dataset (Warstadt et al., 2018), to our knowledge the present dataset is a first for Italian, and is also the first one to combine judgments of acceptability and complexity.

2 Definition of the task

We conceived AcCompl-it as a prediction task where participants were asked to estimate the average acceptability and complexity score of a set of sentences previously rated by native speakers on a 1-7 Likert scale and, if possible, to predict the actual standard error (SE) among the annotations. SE gives an estimation of the actual agreement between human annotators: the highest the SE, the

---

lowest the agreement. The task is articulated in three subtasks, as follows:

- the *Acceptability prediction* task (*ACCEPT*), where participants have to estimate the acceptability score of sentences (along with their standard error); in this case, 1 corresponds to the lowest degree of acceptability, while 7 corresponds to the highest level. The assignment of a score on a gradual scale is in line with the definition of perceived acceptability that we intend to empirically inspect. According to the literature, in fact, acceptability is a concept closely related to grammaticality but with some major differences (see, among others, (Sprouse, 2007; Sorace and Keller, 2005)). While the latter is a theoretical construction corresponding to syntactic wellformedness and it is typically interpreted as a binary property (i.e., a sentence is either grammatical or ungrammatical), acceptability can depend on many factors, such as syntactic, semantic, pragmatic, and non–linguistic factors;
- the *Complexity prediction* task (*COMPL*), where participants have to estimate the complexity score of sentences (along with their standard error); in this case, 1 corresponds to the lowest possible level of complexity, while 7 indicates the highest degree. Similarly to the Acceptability prediction task, the use of the Likert scale as a tool to collect perceived values is motivated by the assumption that sentence complexity is a gradient, rather than binary, concept;
- the *Open task*, where participants are requested to model linguistic phenomena correlated with the human ratings of sentence acceptability and/or complexity in the datasets provided.

The three subtasks were independent and participants could decide to participate in any one of them, though we encouraged participation in multiple subtasks, since the complexity metrics might be influenced by the grammatical status of an expression and vice versa. In line with this intuition, we distributed a subset of sentences annotated with both acceptability and complexity scores in order to investigate whether and to what extent there is a correlation between the two phenomena.

In all subtasks, participants were free to use external resources, and they were evaluated against a blind test set.

3 Dataset

3.1 Composition

Acceptability dataset: it contains 1,683 Italian sentences annotated with human judgments of acceptability on a 7–point Likert scale. The number of annotations per sentence ranges from 10 to 85, with an average of 16.38. The dataset is constructed by merging the data of four psycholinguistic studies on minimal variations of controlled linguistic oppositions with different levels of grammaticality with a subset of 672 sentences generated from templates.

The first subset (128 sentences), taken from (Chesi and Canal, 2019), focuses on person features oppositions in object clefts dependencies where Determiner Phrases (DPs) are either introduced by determiners or by pronouns used as determiner as in (1).

(1) {Sono | siete} {gli | voi} architetti che {gli |
{are$_{3Ppl}$ | are$_{2Ppl}$} {the | you} architects that {the |
voi} ingegneri {hanno | avete} consultato.
you} engineers {have$_{3Ppl}$ | have$_{2Ppl}$} consulted
'it is {the|you} architects that {the|you} engineers have consulted'

The second subset (515 sentences) is taken from the studies presented in (Greco et al., 2020) involving copular constructions (e.g. canonical (2a) vs. inverse (2b) (Moro, 1997).

(2) a. Le foto del muro sono la causa della
the pictures of_the wall are the cause of_the
rivolta.
riot

b. La causa della rivolta sono le foto
the cause of_the riot are the pictures
del muro.
of_the wall

This subset also contains declarative and interrogative (yes/no) sentences with a minimal verbal structure (contrasting preverbal vs postverbal subject position in unergatives (3a), unaccusatives (3b) and transitive predicates (3c))

(3) a. I cani hanno abbaiato | Hanno abbaiato i
the dogs have barked | have barked the
cani.
dogs

b. Gli autobus sono partiti | Sono partiti gli
The buses have left | have left the
autobus.
buses

c. Le bambine hanno mangiato il dolce |
the girls have eaten the dessert |
Hanno mangiato le bambine il dolce
have eaten the girls the dessert

The third set (320 sentences) is based on a study in which number and person subject-verb agreement and unagreement cases are tested (Mancini et al., 2018):

(4) Qualcuno ha detto che io$_{1Psg}$ {scrivo$_{1Psg}$ |*
Somebody has said that I$_{1Psg}$ {write$_{1Psg}$ |
scriviamo$_{1Ppl}$} una lettera.
*write$_{1Psg}$} a letter

The fourth one (48 sentences) contains experimental items from (Villata et al., 2015) involving different types of wh-islands violations.

(5) {Cosa | Quale edificio}$_i$ ti chiedi {chi |
{What | Which building}$_i$ do you wonder {who |
quale ingegnere} abbia costruito _$_i$?
which engineer} has built _$_i$?

The last set of 672 sentences was generated by creating all the possible content word combinations from various structural templates designed to test acceptability patterns due to: **(i)** extra or missing gaps in Wh-extractions (6a) vs. topic constructions (6b).

(6) a. {Cosa | Quale problema}$_i$ lo studente
{what | which problem}$_i$ the student
dovrebbe descriver(e) {_$_i$| -lo$_i$ | questo
should describe {_$_i$| it | this
problema}?
problem}

b. Questo problema$_i$, lo studente dovrebbe
this problem, the student should
descriver(e) {_$_i$| -lo$_i$ | questo problema}
describe {_$_i$| it$_i$ | this problem}

(ii) Wh- and relative clauses with gaps inside VP conjunctions (in all conjuncts, i.e. "Across the Board", in only one conjunct, or not at all, see e.g. (7)).

(7) Chi$_i$ ... Maria vuole chiamar(e) {_$_i$| -lo} e
who$_i$ ... Mary wants call$_{inf}$ {_$_i$| him} and
il dottore medicar(e) {_$_i$| -lo}?
the doctor cure {_$_i$| him}?

(iii) embedded Wh-clauses and the possibility of subextractions from them (similar to (5)).

(8) Quale provvedimento Maria ha saputo {che |
Which measure M. has heard {that |
dove | perché | quando} il ministro prenderà?
where | why | when} il ministro prenderà?

(iv) extractions from VPs in subject vs. object positions (9) (cf. (2)).

(9) Carlo conosceva bene il compagno$_i$ di classe
Carlo knew well the classmate$_i$
che {incontrare _$_i$ divertiva sempre Anna | Anna
that {meet$_{inf}$ _$_i$ amused always Anna | Anna
voleva sempre incontrare _$_i$}
wanted always meet$_{inf}$ _$_i$}

(v) NEGPOLs (*nessuno, alcunché, mai* 'any, anything, ever') that are licensed by a higher negation, by a question, or not licensed, in simple or (deeply) embedded sentences (e.g. (10)).

(10) {Maria | Nessuno} si aspetta che qualcuno
{M. | No-one} self expects that someone
possa aver {già | mai} finito questo
could have {already | never} completed this
esercizio (?)
exercise (?)

The use of expanded templates was designed to minimize the potential effect of collocations or specific lexical choices.

Whenever possible each sentence was also manually annotated according to the linguistic-theoretic expectations for "grammaticality", on a 4-points scale: * (ungrammatical, coded as 0), ?? (very marginal, coded as 0.66), ? (marginal, coded as 0.33) and OK (grammatical, coded as 1).

Complexity dataset: it comprises 2,530 Italian sentences annotated with human judgments of perceived complexity on a 7–point Likert scale as for the acceptability dataset. The number of annotations per sentence ranged from 11 to 20, with an average of 16.753. The corpus was internally subdivided into two subsets representative of two different typologies of data, i.e. 1,858 naturalistic sentences extracted from corpora and 672 artificially-generated sentences drawn from the *Acceptability* dataset, and chosen to cover the range of linguistic phenomena represented in its templates. The first subset contains sentences taken from the Universal Dependency (UD) treebanks (Nivre et al., 2016) available for Italian, representative of different text genres and domains. In this regard, the largest portion contains 1,128 sentences taken from the newswire section of the Italian Stanford Dependency Treebank (ISDT) (Bosco et al.,), annotated with complexity judgments by Brunato et al. (2018). Beside these, we chose to include in this corpus smaller subsets of sentences representative of a non-standard language variety and of specific constructions, i.e. Wh-questions and direct speech. Non-standard sentences (for a total of 323) are in the form of generic tweets and tweets labelled for irony taken from two representative treebanks, i.e. PoSTWITA and TWITTIRÒ (Sanguinetti et al., 2018; Cignarella et al., 2019). Wh-questions (164 sentences) were extracted from a dedicated section (prefixed by the string 'quest') included in ISDT. Direct speech sentences (243) mainly include transcripts of European parliamentary debates (taken from the 'europarl' section of ISDT) and extracts from literary texts (mostly contained in the UD Italian VIT (Delmonte et al., 2007)). The choice of annotating a shared portion of data with both acceptability and complexity scores was explicitly motivated by the attempt to empirically investigate whether there is a correlation between the two sentence properties, and whether complexity is judged differently in the case of ill-formed constructions.

For the purpose of the task, both datasets were split into training and validation samples with a proportion of 80% to 20%, respectively.

3.2 Annotation with Human Judgments

For the collection of judgments of sentence acceptability and complexity by Italian native speakers we relied on crowdsourcing techniques using different platforms. More specifically, for *Acceptability*, the set of sentences drawn from the psycholinguistic studies described in Section 3.1 was annotated using an on-line platform based on jsPsych scripts (De Leeuw, 2015). For the *Complexity* dataset, the annotation of the subcorpus of sentences taken from (Brunato et al., 2018) was performed through the CrowdFlower platform[1] (more details are reported in the reference paper), while the remaining sentences in this dataset were annotated using Prolific[2]. To make the annotation process comparable to the one followed by (Brunato et al., 2018), the whole process was split into different tasks, each one consisting in the annotation of about 200 sentences randomly mixed for the various typologies. For all tasks, workers were asked to read each sentence and answer the following question:

> *"Quanto è complessa questa frase da 1 (semplicissima) a 7 (molto difficile)?" 'How difficult is this sentence from 1 (very easy) to 7 (very difficult)?'*

Beyond complexity, the 672 artificially-generated sentences were also labelled for perceived acceptability according to the following question:

> *"Quanto è accettabile questa frase da 1 (completamente agrammaticale) a 7 (perfettamente grammaticale)?" 'How acceptable is this sentence from 1 (completely ungrammatical) to 7 (completely grammatical)?'*

After collecting all annotations, we excluded workers who performed the assigned task in less than 10 minutes, which we set as the minimum threshold to accurately complete the survey.

3.3 Analysis of Judgments across Corpora

Table 1 shows the average value, standard deviation and minimum and maximum score of complexity and acceptability labels for the whole dataset. As it can be noticed, complexity values are on average lower and less scattered than the acceptability ones. For this corpus, the lowest value on the Likert scale (1) – which should have been used to label sentences perceived as very easy, in line to the task question – was given only twice, specifically to the following sentences:

(11) Dimmi il nome di una città finlandese.
tell me the name of a town Finnish
'Tell me the name of Finnish town'

(12) Quali uve si usano per produrre vino?
Which grapes PRT they_use to make wine?

Conversely, for the acceptability corpus, the highest value on the Likert scale (i.e. 7, meaning in this case *completely acceptable*) was attributed to 26 sentences. For space reasons, we report here only two examples:

(13) Le sorelle sono sopravvissute.
The sisters are survived.
'the sisters have survived'

(14) I lupi hanno ululato.
The wolves have howled.

---

[1] Now known as Figure Eight, https://appen.com/

[2] www.prolific.co

With respect to the ‘worst’ values, two sample sentences judged respectively as the most complex (i.e. 6.46 on the Likert scale) and (among) the least acceptable (1.55) in each dataset are the following ones, respectively:

(15) Chi è che lui ha affermato che il professore
who is that he has claimed that the professor
aveva detto che lo studente avrebbe dovuto
had said that the student had_{subj} must
considerare questo candidato?
consider this candidate?

(16) Il falegname è arrivato mentre noi
The carpenter has arrived while we
montavo la mensola.
$\text{were\_assembling}_{\text{1Psg}}$ the shelf.

	COMPL		ACCEPT	
	SCORE	SE	SCORE	SE
μ	3.12	0.332	4.45	0.36
σ	1.04	0.08	1.7	0.14
min	1	0	1.13	0
max	6.46	0.63	7	0.74

Table 1: Statistics collected for the two corpora of the AcCompl-it dataset.

If we consider the internal composition of the two datasets we can see a more articulated picture depending on its various subparts (see Table 2 and 3). For complexity, average scores are higher for sentences created to display specific acceptability patterns, thus proving that acceptability does affect the perception of complexity. Note that the most complex sentence (reported in (15)) is contained in this set, and is ungrammatical (no gap).

Among the treebank sentences, those extracted from journalistic texts (ISDT_news) were judged on average as the most complex, questions as the easiest ones. Twitter and direct speech sentences obtained scores in between the highest and the lowest value and very close to each other. This is in line with stylistic and linguistic analysis showing that the language of social media inherits many features from spoken language.

For the whole acceptability dataset, the Spearman’s rank correlation coefficient between theoretically-driven grammaticality and mean acceptability labels is very strong (*r*(656)=.83, *p*<.001). While this could be somehow expected, when we focus only on the 672 sentences annotated for both complexity and acceptability, we still observe a significant but lower correlation between expected grammaticality and mean complexity (*r*(656)=.34, *p*<.001). Still considering this subset, an additional outcome is the moderate (and negative) correlation between the two metrics (*r*(672)=.49, *p*<.001), further suggesting that the more a sentence is perceived as complex, the less acceptable it is.

	SCORE	SE	MIN	MAX
ISDT_news	3.28	0.33	1.25	5.7
Twitter	2.59	0.31	1.13	4.69
DirectSpeech	2.68	0.31	1.14	6
Wh-Quest	1.61	0.22	1	2.94
ArtifSent	3.63	0.37	1.42	6.47

Table 2: Average complexity score, standard error and minimum and maximum value across the different subsets of the **Complexity** dataset.

4 Evaluation measures

For both the ACCEPT and COMPL Task, the evaluation metric was based on Spearman’s rank correlation coefficient between the participants’ scores and the test set scores. For each task, two different ranks were produced according to the prediction of the relative scores and to standard errors. In each task a different baseline was defined:

- in the ACCEPT task, it corresponds to the score assigned by a SVM linear regression using unigram and bigram of words as features;
- in the COMPL task, it corresponds to the score assigned by a SVM linear regression using sentence length as its sole feature.

5 Participation and results

The AcCompl-it task received three submissions for each subtask from two different participants, for a total of 6 runs. Unfortunately, neither participant took part in the Open Task. Results for the other ones are reported in Tables 4 and 5.

The systems from the two participants in the task follow very different approaches: one is based on deep learning and trained on raw texts (Sarti, 2020), the other relies on (heuristic) rules applied to semantic and syntactic features automatically extracted from sentences (Delmonte, 2020). In spite of their very different nature, the two approaches also present some commonalities, such

	SCORE	SE	MIN	MAX
clefts (1)	4.27	0.39	1.36	6
copular	5.01	0.48	2.90	6.5
canonical (2a)	5.47	0.44	3.58	6.5
inverse (2b)	4.56	0.51	2.90	5.9
unerg V (3a)	5.91	0.30	3.70	7
SV	6.64	0.21	5.94	7
VS	5.18	0.40	3.70	6.2
unacc V (3b)	6.28	0.27	4.86	7
SV	6.61	0.20	5.82	7
VS	5.96	0.33	4.86	6.72
trans V (3c)	4.91	0.34	2	7
SV	6.47	0.24	5.06	7
VS	3.34	0.43	2	4.52
S V agree (4)	3.81	0.31	1.25	6.93
match	5.87	0.30	3.14	6.92
mismatch	1.74	0.32	1.25	2.72
wh-island (arg) (5)	3.85	0.17	1.68	5.63
filler-gap dep.	3.56	0.51	1.5	6.69
doubly filled (6)	3.28	0.42	1.5	6.69
coord (7)	4.25	0.47	2.5	6
wh-island (adj) (8)	3.02	0.41	1.38	5.6
no extraction	6.26	0.29	5.26	7
subj/obj (9)	3.70	0.51	2.66	5.15
NPIs (10)	4.75	0.45	2.27	6.6
bad fillers	1.13	0.06	1.13	1.13
good fillers	6.76	0.07	6.76	6.76
medium fillers	4.07	0.18	4.07	4.07

Table 3: Average acceptability score, standard error and minimum and maximum value across the different linguistic phenomena of the **Acceptability**. Numbers in (·) refer to examples in the text.

PARTICIPANT	SCORE	SE
UmBERTO-MTSA (Sarti)	**0.88****	**0.52****
ItVenses-run1 (Delmonte)	0.44**	0.25**
ItVenses-run2 (Delmonte)	0.49**	0.41**
Baseline	0.30**	0.35**

Table 4: ACCEPT task results.**p value<0.001; *p value <0.05

PARTICIPANT	SCORE	SE
UmBERTO-MTSA (Sarti)	**0.83****	**0.51****
ItVenses-run1 (Delmonte)	0.31**	0.09*
ItVenses-run2 (Delmonte)	0.31**	0.07
Baseline	0.50**	0.33**

Table 5: COMPL task results. **p value<0.001; *p value <0.05

as the reliance on external resources. In particular, both make use of additional sentences taken from existing Italian treebanks, either to enrich the original training sets with additional annotated examples (Sarti's case) or to check the frequency of a given construction and use this info among the features of the proposed system (ItVenses).

Sarti's systems obtained the best performance on both tasks using a similar multi-task learning (MTL) approach, which consists in leveraging the predictions of a state-of-the-art neural language model for Italian (i.e. UmBERTO[3]) fine-tuned on the two downstream tasks to augment the original development sets with a large set of unlabeled examples extracted from available Italian treebanks. The bigger dataset was then split into different portions to train an ensemble of classifiers. The resulting MTL model was finally used to predict the complexity/acceptability labels on the original test sets.

Delmonte's *ItVenses* system parses the sentences to obtain a sequence of constituents and a set of sentence-level semantic features (presence of agreement, negation markers, speech act and factivity). These features, along with constituent triples and their frequency in the training set and in the Venice Italian Treebank are weighed with various heuristics and used to derive a prediction. Agreement mismatches were checked using morphological analysis of verb and subject, while the argumental structure is inferred using a deep parser. The two versions of the system (*run1* and *run2*) differ only in their use of features (*run2* dispenses with proposional negation and certain verb agreement features).

As it can be seen, ItVenses's performance were considerably lower than Sarti's system (lower, in fact, than the baseline based on sentence length, in the COMPL prediction task). However, as better explained in the following section, in the artificial data subset, which has complex but far less diverse structures, the gap with the winning system is reduced in the COMPL task (cfr. Table 7) and, even more robustly, in the ACCEPT task (Table 6).

6 Discussion

The extremely good performance of the winning system in both tasks is not wholly unexpected in light of the impressive results obtained by current neural networks models across a variety of NLP tasks. In this regard, it is worth noticing that, in his report, the author compared the performance of the best system based on multi-task learning to the one obtained by a simpler version of the UmBERTO-based model with standard fine-tuning on the two downstream tasks, achieving al-

---

[3] https://github.com/musixmatchresearch/umberto

ready very good results (.90 and .84 for acceptability and complexity predictions on the training corpus, respectively). Similarly, and especially for the automatic assessment of sentence acceptability, the scores obtained by the winning system (.88) are in line with those reported in (Linzen et al., 2016), who train a classifier to detect subject-verb agreement mismatches from the hidden states of an LSTM, achieving a .83 score. Most other systems at work on the ability of neural models to detect acceptability or grammaticality in a broader range of cases report much lower scores, but they try to read (minimal pair) judgments from metrics associated to the performance of systems that have not been expressly trained on giving judgments, reasoning that 'judgment giving' is not a task humans have a life-long training for, but which is nonetheless feasible.

To have a better understanding of the potential impact of different types of data on the predictive capabilities of the two systems, we further inspected the final results by testing each system on sentences representative of diverse linguistic phenomena and textual genres. To this end, we split the whole test set into the distinct subsets defined in the corpus collection process (cfr. Section 3.1) and we assessed the correlation score between predicted and real labels for each type: note that, for the ACCEPT predictions, this analysis was performed considering only two 'macro–classes', i.e. artificial vs psycholinguistics-related data, in order to have a significant number of examples in the test set. Similarly, for COMPL, we distinguished the artificially–generated sentences from sentences drawn from all treebanks. Results of this fine-grained analysis are shown in Tables 6 and 7.

Interestingly, although the gap between the two systems is still evident, we observed that artificial data have an opposite effect on their performance. In particular, as anticipated in the previous section, *ItVenses* is more accurate in predicting both the complexity and, especially, the acceptability level of this group of sentences. The opposite holds for Sarti's system, which although still very good in both tasks, achieves lower correlation scores when tested against artificial data.

Running an exploratory analysis based on expected grammaticality, we observed that Sarti's system performs much better in predicting the acceptability score on expected grammatical sentences (r=.80, p<.001) than on expected ungrammatical ones (r=.76, p<.001). Similarly, but less robustly, the same numerical asymmetry is observed in both Delmonte's runs: for grammatical predictions, RUN1 r=.33, RUN2 r=.35; for ungrammatical ones RUN1 r=.32, RUN2 r=.34, all correlations being equally significant (p<.001).

PARTICIPANT	**SCORE**	**SE**
Psycholinguistics related		
UmBERTO-MTSA (Sarti)	0.90**	0.55**
ItVenses-run1 (Delmonte)	0.42**	0.24**
ItVenses-run2 (Delmonte)	0.50**	0.48**
Artificial data		
UmBERTO-MTSA (Sarti)	0.74**	0.33**
ItVenses-run1 (Delmonte)	0.50**	0.20*
ItVenses-run2 (Delmonte)	0.46**	0.25*

Table 6: ACCEPT task results on different subsets of the official test set. **p value<0.001; *p value <0.05

PARTICIPANT	**SCORE**	**SE**
Treebank sentences		
UmBERTO-MTSA (Sarti)	0.86**	0.61**
ItVenses-run1 (Delmonte)	0.25**	0.13*
ItVenses-run2 (Delmonte)	0.24**	0.10*
Artificial data		
UmBERTO-MTSA (Sarti)	0.70**	0.06
ItVenses-run1 (Delmonte)	0.44**	-0.07
ItVenses-run2 (Delmonte)	0.51**	-0.11

Table 7: COMPL task results on different subsets of the official test set. **p value<0.001; *p value <0.05

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

C. Bosco, S. Montemagni, and M. Simi. Converting Italian Treebanks: Towards an Italian Stanford Dependency Treebank. In *Proceedings of the ACL Linguistic Annotation Workshop & Interoperability with Discourse*.

Dominique Brunato, Lorenzo De Mattei, Felice

Dell'Orletta, Benedetta Iavarone, and Giulia Venturi. 2018. Is this sentence difficult? do you agree? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2690–2699.

Cristiano Chesi and Paolo Canal. 2019. Person features and lexical restrictions in Italian clefts. *Frontiers in Psychology*, 10:2105.

Shammur Absar Chowdhury and Roberto Zamparelli. 2018. Rnn simulations of grammaticality judgments on long-distance dependencies. In *Proceedings of the 27th international conference on computational linguistics*, pages 133–144.

Alessandra Teresa Cignarella, Cristina Bosco, and Paolo Rosso. 2019. Presenting TWITTIRÒ-UD: An italian twitter treebank in universal dependencies. In *Proceedings of the Fifth International Conference on Dependency Linguistics (Depling, SyntaxFest 2019).*

Joshua R De Leeuw. 2015. jspsych: A javascript library for creating behavioral experiments in a web browser. *Behavior research methods*, 47(1):1–12.

Rodolfo Delmonte, Antonella Bristot, and Sara Tonelli. 2007. VIT - Venice Italian Treebank: Syntactic and quantitative features. In *Proceedings of the Sixth International Workshop on Treebanks and Linguistic Theories*.

Rodolfo Delmonte. 2020. Venses@AcCompl-it: Computing complexity vs acceptability with a constituent trigram model and semantics. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Vera Demberg and Frank Keller. 2008. Data from eye-tracking corpora as evidence for theories of syntactic processing complexity. *Cognition*, 109(2):193–210.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Matteo Greco, Paolo Lorusso, Cristiano Chesi, and Andrea Moro. 2020. Asymmetries in nominal copular sentences: Psycholinguistic evidence in favor of the raising analysis. *Lingua*, 245:102926.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. *arXiv preprint arXiv:1803.11138*.

John Hale. 2001. A probabilistic earley parser as a psycholinguistic model. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

T. Linzen, E. Dupoux, and Y. Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. In *Transactions of the Association for Computational Linguistics*, volume 4, pages 521–535.

Simona Mancini, Paolo Canal, and Cristiano Chesi. 2018. The acceptability of person and number agreement/disagreement in Italian: an experimental study. *Lingbuzz preprint: https://ling.auf.net/lingbuzz/005514*.

Andrea Moro. 1997. *The raising of predicates: Predicative noun phrases and the theory of clause structure*, volume 80. Cambridge University Press.

Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666.

Manuela Sanguinetti, Cristina Bosco, Alberto Lavelli, Alessandro Mazzei, and Fabio Tamburini. 2018. PoSTWITA-UD: an Italian Twitter Treebank in universal dependencies. In *Proceedings of the Eleventh Language Resources and Evaluation Conference (LREC 2018).*

Gabriele Sarti. 2020. UmBERTo-MTSA @ AcCompl-it: Improving complexity and acceptability prediction with multi-task learning on self-supervised annotations. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Antonella Sorace and Frank Keller. 2005. Gradience in linguistic data. *Lingua*, 115:1497–1524.

Jon Sprouse. 2007. Continuous acceptability, categorical grammaticality, and experimental synta. *Biolinguistics*, pages 1123–134.

Sandra Villata, Paolo Canal, Julie Franck, Andrea Carlo Moro, and Cristiano Chesi. 2015. Intervention effects in wh-islands: An eye-tracking study. In *Architectures and Mechanisms for Language Processing (AMLaP 2015)*, pages 195–195.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Ethan Wilcox, Roger Levy, Takashi Morita, and Richard Futrell. 2018. What do rnn language models learn about filler-gap dependencies? *arXiv preprint arXiv:1809.00042*.

UmBERTo-MTSA @ AcCompl-It: Improving Complexity and Acceptability Prediction with Multi-task Learning on Self-Supervised Annotations

Gabriele Sarti
Department of Mathematics and Geoscience, University of Trieste
International School for Advanced Studies (SISSA), Trieste, Italy
`gsarti@sissa.it`

Abstract

English. This work describes a self-supervised data augmentation approach used to improve learning models' performances when only a moderate amount of labeled data is available. Multiple copies of the original model are initially trained on the downstream task. Their predictions are then used to annotate a large set of unlabeled examples. Finally, multi-task training is performed on the parallel annotations of the resulting training set, and final scores are obtained by averaging annotator-specific head predictions. Neural language models are fine-tuned using this procedure in the context of the AcCompl-it shared task at EVALITA 2020, obtaining considerable improvements in prediction quality.

Italiano. *Questo articolo descrive un approccio di self-supervised data augmentation utilizzabile al fine di migliorare le performance di algoritmi di apprendimento su task aventi solo una modesta quantità di dati annotati. Inizialmente, molteplici copie del modello originale vengono allenate sul task prescelto. Le loro previsioni vengono poi utilizzate per annotare grandi quantità di esempi non etichettati. In conclusione, un approccio di multi-task training viene utilizzato, con le annotazioni del dataset risultante in veste di task indipendenti, per ottenere previsioni finali come medie dei i punteggi dei singoli annotatori. Questa procedura è stata utilizzata per allenare modelli del linguaggio neurali per lo shared task AcCompl-it a EVALITA 2020, ottenendo ampi miglioramenti nella qualità predittiva.*

1 Introduction

In recent times, pre-trained neural language models (NLMs) have become the preferred approach for language representation learning, pushing the state-of-the-art in multiple NLP tasks (Devlin et al. (2019); Radford et al. (2019); Yang et al. (2019); Raffel et al. (2019) *inter alia*). These approaches rely on a two-step training process: first, a *self-supervised pre-training* is performed on large-scale corpora; then, the model undergoes a *supervised fine-tuning* on downstream task labels using task-specific prediction heads. While this method was found to be effective in scenarios where a relatively large amount of labeled data are present, researchers highlighted that this is not the case in low-resource settings (Yogatama et al., 2019).

Recently, *pattern-exploiting training* (PET, Schick and Schutze (2020a,b) tackles the dependence of NLMs on labeled data by first reformulating tasks as cloze questions using task-related patterns and keywords, and then using language models trained on those to annotate large sets of unlabeled examples with soft labels. PET can be thought of as an offline version of *knowledge distillation* (Hinton et al., 2015), which is a well-established approach to transfer the knowledge across models of different size, or even between different versions of the same model as in *self-training* (Scudder, 1965; Yarowsky, 1995). While effective on classification tasks that can be easily reformulated as cloze questions, PET cannot be easily extended to regression settings since they cannot be adequately verbalized. Contemporary work by Du et al. (2020) showed how self-training and pre-training provide complementary information for natural language understanding tasks.

In this paper, I propose a simple self-supervised data augmentation approach that can be used to improve the generalization capabilities of NLMs on regression and classification tasks for modest-sized labeled corpora. In short, an ensemble of fine-tuned models is used to annotate a large corpus of unlabeled text, and new annotations are leveraged in a multi-task setting to obtain final predictions over the original test set. The method was tested on the AcCompl-it shared tasks of the EVALITA 2020 campaign (Brunato et al., 2020b; Basile et al., 2020), where the objective was to predict respectively *complexity* and *acceptability* scores on a 1-7 Likert scale for each test sentence, alongside an estimation of its standard error. Results show considerable improvements over regular fine-tuning performances on COMPL and ACCEPT using the UmBERTo pre-trained model (Francia et al., 2020), suggesting the validity of this approach for complexity/acceptability prediction and possibly other language processing tasks.

2 Description of the Approach

Let:

- $\mathcal{L} = [(x_1, y_1), \dots (x_n, y_n)]$ be the initial labeled corpus containing sentence-annotation pairs $x_i \in X, y_i \in Y_x$. [1]
- $\mathcal{U} = [x'_1, \dots x'_m]$ be a large unlabeled corpus such that $m \gg n$
- $M : x_i \rightarrow \hat{y}_i$ be a pre-trained neural language model with a single task-specific heads, taking sentence x_i as input and predicting label y_i at inference time.

For some $k \in \mathbb{N}_1$, we begin by splitting $\mathcal{L}$ in k equal-sized segments $\mathcal{L}_1, \dots, \mathcal{L}_k$ and fine-tune k identical versions of M using k-fold cross-validation. We call the resulting models $M^1, \dots, M^k$ "NLMs with standard fine-tuning on the y target task", with M^i being trained on the subset $\mathcal{L} - \mathcal{L}_i$ and evaluated on $\mathcal{L}_i$. Then, each sentence of $\mathcal{U}$ is passed to each model, obtaining the corpus

$$\mathcal{U}' = [(x'_1, \hat{y}'^1_1 \dots \hat{y}'^k_1), \dots, (x'_m, \hat{y}'^1_m \dots \hat{y}'^k_m)] \quad (1)$$

labeled with expert annotations from fine-tuned models. Predicted values are taken instead of probability distributions after the softmax, which are typically used in the knowledge distillation literature, to keep the approach simple while making it viable in the context of regression tasks.

Now that the large corpus is annotated, a *multi-task NLM* $MTM : x_i \rightarrow \dot{y}^1_i \dots \dot{y}^k_i$ is fine-tuned on $\mathcal{U}'$ by treating each annotation in the set $\hat{y}'^1 \dots \hat{y}'^k$ as a separate task, using 1-layer feed-forward neural networks as task-specific heads while performing hard parameter sharing (Caruana, 1997) on underlying model parameters. Intuitively, the k models used to produce annotations were trained on different folds of the original corpus, and as such, they provide complementary viewpoints on the modeled phenomenon when k is small.

As a final step, MTM is fine-tuned on a training portion of $\mathcal{L}$, using as prediction scores $f(\dot{y}^1_i \dots \dot{y}^k_i)$, where f is a task and context-dependent aggregation function. For example, in the case of a classification task, one can select the majority vote from the ensemble of model heads as the final prediction, while in a regression setting this can be done by averaging scores across heads. Once fine-tuned, the model can be tested on the test portion of $\mathcal{L}$ using the same f as the aggregator. I refer to this approach as *Multi-Task Self-Annotation (MTSA)* in the following sections.

3 Experimental Evaluation

For the experimental evaluation part:

- The ACCEPT and COMPL training corpora, containing respectively 1339 and 2012 sentences labeled with average scores and standard error across annotators, were used as labeled datasets $\mathcal{L}_A, \mathcal{L}_C$. The two tasks were learned separately, following the same approach described in the previous section.
- A set of multiple Italian treebanks including train, dev, and test sets of the Italian Stanford Dependency Treebank (Bosco et al., 2013), the Turin University Parallel Treebank (Sanguinetti and Bosco, 2015), PoSTWITA-UD (Sanguinetti et al., 2018) and the Venice Italian Treebank (Delmonte et al., 2007) was used as unlabeled corpus $\mathcal{U}$. The final corpus contains 37,344 unlabeled sentences and spans multiple textual genres.
- The UmBERTo model (Francia et al., 2020) available through the HuggingFace's Transformers framework (Wolf et al., 2019) was

---

[1] y_i can be either discrete or continuous in this context.

Model	**Score** (ρ)	**Error** (ρ)
UmBERTo surprisal	-0.36	0.17
Length (# of tokens)	-0.39	0.17
Length (characters)	-0.39	0.21
UmBERTo fine-tuned	0.90	0.50
UmBERTo-STSA	**0.91**	0.53
UmBERTo-MTSA	**0.91**	**0.54**
UmBERTo surprisal	0.49	0.28
Length (# of tokens)	0.55	0.36
Length (characters)	0.60	0.39
UmBERTo fine-tuned	0.84	0.54
UmBERTo-STSA	0.87	0.62
UmBERTo-MTSA	**0.88**	**0.63**

Table 1: Spearman's correlation scores on the ACCEPT (top) and COMPL (bottom) subtasks' training portions. Models are evaluated using 5-fold cross-validation. All scores have $p < 0.001$

used both for fine-tuning $M^{1 \dots k}$ during the annotation part and for fine-tuning MTM. The model is based on the RoBERTa architecture (Liu et al., 2019) and was pre-trained on the Italian portion of the OSCAR CommonCrawl corpus (Ortiz Suárez et al., 2020), containing roughly 210M sentences and over 11B tokens.

Since both tasks involve predicting both averaged scores and the original standard error across participants, the approach presented in the previous section was adapted to account for multi-task learning of scores and errors from the beginning, with each model M^i producing both a predicted score $\hat{y}'^i$ and a predicted error $\hat{\epsilon}'^i$ for the annotation step. The k parameter was set to 5 to prevent excessive overlapping of training data across models, with the final multi-task model $MTM : x_i \to \dot{y}_i^1 \dots y_i^5, \epsilon_i^1 \dots \epsilon_i^5$ returning prediction for scores and errors for all the five sets of fine-tuned model annotations.

Models $M^{1 \dots k}$ were trained for a maximum of 15 epochs on the labeled training sets using early stopping (5 patience steps, 20 evaluation steps using a 10% slice as dev set), learning rate $\lambda = 1e^{-5}$, batch size $b = 32$ and embedding dropout $\delta = 0.1$. The model's base variant was used, having a hidden size $|h| = 768$, and a maximum sequence length of 128. Notably, the representations at the last layer of the UmBERTo model were averaged to obtain a sentence-level representation instead of using the [CLS] token. During the training on the whole unlabeled corpus, the evaluation steps were increased to 100 to balance evaluation time with the corpus's increased size.

4 Results

Table 1 presents methods for which the correlation between values and complexity scores was tested on the training portion of the ACCEPT and COMPL tasks with 5-fold cross validation, leading to the selection of MTSA as the top-performing approach:

- **UmBERTo surprisal**: Sentence-level surprisal estimates are produced using the pre-trained model without fine-tuning as:

$$P(x) = \prod_{i=1}^{m} P(w_i \,| w_{1:i-1}, w_{i+1:m}) \qquad (2)$$

- **Length (# of tokens)**: Length of the sentence in number of tokens
- **Length (characters)**: Length of the sentence in number of characters (including whitespaces)
- **UmBERTo fine-tuned**: Predictions produced by Umberto with standard fine-tuning on complexity corpus annotations.
- **UmBERTo-STSA**: A variant of the MTSA approach where instead of performing multi-task learning over model annotations on $\mathcal{U}$, we average them in a single score, and the model is trained on it with single-task fine-tuning.
- **UmBERTo-MTSA**: The approach presented in this work.

From Table 1, it can be observed that, although length alone is already correlated with acceptability complexity scores, UmBERTo can leverage additional information from its representation to produce much stronger predictions. Interestingly, both the STSA and MTSA self-annotation approaches consistently outperform regular fine-tuning, especially for what concerns standard error scores. This fact suggests that self-annotation leads to better generalization capabilities in the model over downstream tasks when relatively few

Model	**Score** (ρ)	**Error** (ρ)
SVM 2-gram baseline	0.30	0.35
UmBERTo-MTSA	**0.88**	**0.52**
SVM length baseline	0.50	0.33
UmBERTo-MTSA	**0.83**	**0.51**

Table 2: Correlation scores with gold labels on the ACCEPT (top) and COMPL (bottom) subtasks' test portions. All scores have $p < 0.001$.

	Acceptability		**Complexity**	
	$\rho(y_\epsilon)$	$\rho(\epsilon_\epsilon)$	$\rho(y_\epsilon)$	$\rho(\epsilon_\epsilon)$
avg. score (y)	-25%	10%	41%	-2%
std. error (ϵ)	12%	2%	23%	27%
upos_dist_PROPN	19%	-3%	4%	6%
dep_dist_nmod	19%	-8%	4%	1%
avg_max_depth	16%	-3%	7%	-7%
n_prep_chains	16%	-8%	4%	-2%
prep_chain_len	16%	-6%	9%	-4%
upos_dist_PRON	1%	20%	8%	9%
dep_dist_root	-9%	18%	-4%	23%
dep_dist_punct	-9%	17%	1%	-3%
aux_mood_dist_Imp	7%	6%	17%	7%
n_tokens	9%	-13%	5%	-18%
avg_links_len	-3%	1%	-6%	-17%
max_links_len	-1%	-9%	-1%	-16%

Table 3: Pearson's correlation scores between prediction errors and various linguistic features. Orange and cyan cells contain respectively positive and negative scores for which $p < 0.001$.

annotations are available. While the contribution of multi-task learning is modest, the MTSA approach may prove especially beneficial when training models $M^{1 \dots k}$ on scores produced by different annotators instead of using different folds of the same corpus, as in this case. In both cases, predicted surprisal scores act as poor predictors for downstream tasks. It should also be noted that length appears to be negatively correlated to acceptability scores (i.e. longer sentences are generally less acceptable), while the relation is positive in the case of complexity (i.e. longer sentences are generally more complex).

Table 2 reports the scores obtained by MTSA over the test sets for the ACCEPT and the COMPL shared tasks. The organizers' baseline scores correspond to the correlation among gold labels and acceptability and complexity predictions produced by an SVM model trained on 1-grams and bigrams of sentences and an SVM trained on sentence length, respectively. The MTSA approach achieved the first rank in both tasks, with considerable improvements over baseline scores.

5 Error Analysis

Finally, some error analysis is performed to gain additional insights on which factors influence the predictability of complexity and acceptability judgments. The Profiling-UD tool by Brunato et al. (2020a) is used to produce linguistic annotations on test sentences for both tasks. Given an input sentence, Profiling-UD produces roughly $\sim$ 100 numeric scores representing different phenomena and properties at different language levels.[2] I then correlate the value of all features with y_ϵ and ϵ_ϵ, representing the mean absolute error between true and predicted values for scores and standard errors, respectively. Table 3 presents the results of the error analysis.

Strongly correlated values in Table 3 correspond to features that highly influence, either positively or negatively, the prediction capabilities of the MTSA model. Extreme task scores (avg. score), denoting either not very acceptable or highly complex sentences, are less predictable than their average counterparts by MTSA. Sentences for whose the standard deviation of scores is high across participants appear to be less predictable in the context of complexity scores, while this does not affect acceptability predictions.

Concerning acceptability, I found a significant correlation between acceptability prediction errors and the presence of multilevel syntactic structures, (*avg_max_depth*) multiple long prepositional chains (*n_prep_chains*, *prep_chain_len*) and nominal modifiers (*dep_dist_nmod*). From the complexity viewpoint, instead, the presence of inflectional morphology related to the imperfect tense in auxiliaries (*aux_mood_dist_Imp*) was the only property related to higher prediction errors. However, high token counts (*n_tokens*) and long dependency links (*avg_links_len*, *max_links_len*) were shown to make the variability in complexity scores more predictable.

Overall, results suggest that incorporating syntactic information during the model's training process may further improve complexity and acceptability models.

---

[2] A description of produced annotations is omitted for brevity. Refer to Brunato et al. (2020a) for additional details.

6 Discussion and Conclusion

This work introduced a simple and effective data augmentation approach improving the fine-tuning performances of NLMs when only a modest amount of labeled data is available. The approach was first formalized and then empirically tested on the ACCEPT and COMPL shared tasks of the EVALITA 2020 campaign. Strong performances were reported for both acceptability and complexity prediction using a multi-task self-training approach, obtaining the top position in both subtasks. Finally, an error analysis highlighted the unpredictability of extreme scores and sentences having complex syntactic structures.

The suggested approach, although computationally refined and well-performing, is lacking in terms of complexity-driven biases that may prove useful in the context of complexity and acceptability prediction. A possible extension of this work may include a complementary syntactic task (e.g., biaffine parsing, as in Glavas and Vulic (2020)) during multi-task learning to see if forcing syntactically-competent representations in the top layers may prove beneficial in the context of syntax-heavy tasks like complexity and acceptability prediction. Moreover, it would be interesting to evaluate multi-task learning performances with complexity and acceptability parallel annotations given the conceptual similarity between the two tasks and estimate the effectiveness of a feed-forward network as the final aggregator f in the MTSA paradigm instead of merely averaging predictions. Finally, Du et al. (2020) findings suggest that using an unsupervised in-domain filtering approach may further improve the self-training procedure when large unlabeled corpora are available.

Acknowledgments

The author was supported by a scholarship for Data Science and Scientific Computing students from the International School of Advanced Studies (SISSA).

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Cristina Bosco, Simonetta Montemagni, and Maria Simi. 2013. Converting Italian treebanks: Towards an Italian Stanford dependency treebank. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 61–69, Sofia, Bulgaria. Association for Computational Linguistics.

Dominique Brunato, Andrea Cimino, Felice Dell'Orletta, Giulia Venturi, and Simonetta Montemagni. 2020a. Profiling-UD: a tool for linguistic profiling of texts. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 7147–7153, Marseille, France. European Language Resources Association.

Dominique Brunato, Chesi Cristiano, Felice Dell'Orletta, Simonetta Montemagni, Giulia Venturi, and Roberto Zamparelli. 2020b. AcCompl-it @ EVALITA2020: Overview of the acceptability complexity evaluation task for italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28:41–75.

Rodolfo Delmonte, Antonella Bristot, and Sara Tonelli. 2007. VIT–venice italian treebank: syntactic and quantitative features.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jingfei Du, E. Grave, Beliz Gunel, Vishrav Chaudhary, Onur Çelebi, M. Auli, Ves Stoyanov, and Alexis Conneau. 2020. Self-training improves pre-training for natural language understanding. *ArXiv*, abs/2010.02194.

Simone Francia, Loreto Parisi, and Magnani Paolo. 2020. UmBERTo: an italian language model trained with whole word maskings.

Goran Glavas and Ivan Vulic. 2020. Is supervised syntactic parsing beneficial for language understanding? an empirical investigation. *ArXiv*, abs/2008.06788.

Geoffrey E. Hinton, Oriol Vinyals, and J. Dean. 2015. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531.

Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online. Association for Computational Linguistics.

A. Radford, Jeffrey Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. OpenAI.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, W. Li, and P. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Manuela Sanguinetti and Cristina Bosco. 2015. *PartTUT: The Turin University Parallel Treebank*, pages 51–69. Springer International Publishing, Cham.

Manuela Sanguinetti, Cristina Bosco, Alberto Lavelli, Alessandro Mazzei, Oronzo Antonelli, and Fabio Tamburini. 2018. PoSTWITA-UD: an Italian Twitter treebank in Universal Dependencies. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Timo Schick and Hinrich Schutze. 2020a. Exploiting cloze questions for few-shot text classification and natural language inference. *ArXiv*, abs/2001.07676.

Timo Schick and Hinrich Schutze. 2020b. It's not just size that matters: Small language models are also few-shot learners. *ArXiv*, abs/2009.07118.

H Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Z. Yang, Zihang Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Dani Yogatama, Cyprien de Masson d'Autume, J. Connor, Tomás Kociský, M. Chrzanowski, Lingpeng Kong, A. Lazaridou, W. Ling, L. Yu, Chris Dyer, and P. Blunsom. 2019. Learning and evaluating general linguistic intelligence. *ArXiv*, abs/1901.11373.

Venses @ AcCompl-It: **Computing Complexity vs Acceptability with a Constituent Trigram Model and Semantics**

Rodolfo Delmonte
Dipartimento di Studi Linguistici e Culturali
Comparati
Ca' Bembo – Dorsoduro 1075 – Università
Ca' Foscari – 30131 Venezia
delmont@unive.it

Abstract

In this paper[1] we present work carried out for the Ac-ComplIt task. ItVENSES is a system for syntactic and semantic processing that is based on the parser for Italian called ItGetaruns to analyse each sentence. In previous EVALITA tasks we only used semantics to produce the results. In this year EVALITA, we used both a statistically based approach and the semantic one used previously. The statistic approach is characterized by the use of trigrams of constituents computed by the system and checked against a trigram model derived from the constituency version of VIT – Venice Italian Treebank. Results measured in term of a correlation, are not particularly high, below 50% the Acceptability task and slightly over 30% the Complexity one.

1 Introduction

In this paper we will present work carried out by the Venses Team in Evalita 2020 (Basile et. 2020). We will describe in detail in the following work carried out on the Ac-ComplIt task. We present the modules for automatic classification that uses two different approaches: a fully BOW and statistic one, a fully semantically based one. The trigram model is built on the basis of the analysis performed by ItVenses at different levels of linguistic complexity.

The procedure we organized for the semantically-based analysis is as follows.

At first we massaged the text in order to obtained a normalized version – wrong word accents like “nè” instead of “né” etc. The text is then turned into an xml file to suit the Prolog input requirements imposed by the system.

ItGetarun receives as input a string – the sentence(s) to be analysed - which is then tokenized into a list. The list is then sentence split, fully tagged, disambiguated and chunked. Sentence level chunks are then parsed together into a full sentence structure which is passed to the Island-Based predicate-argument structure (hence PAS) parser.

The output of the semantic parser is passed on to the module for classification called ItVenses. ItVenses inherits constituent labels from chunked sentences which have been first destructured, i.e. all embedded structures have been collapsed and linearized in order to construct a sequence of linear constituent labels.

In addition, ItVenses takes into account agreement, negation and non-factuality usually marked by unreal mood, information available at propositional level, used to modify previously assigned polarity from negative to positive, on the basis of PAS and their semantics. For this reason, we keep trace of hate and stereo words on a lexical basis, together with presence of negation. In particular, hate and stereo words and sentiment polarities (negative and positive), are checked together one by one, in order to verify whether polarity has to be attenuated, shifted or inverted (see Polanyi & Zaenen, 2006) as a result of the presence of intensifiers, maximizers, minimizers, diminishers, or simply negations at a higher than constituent level (see Ohana et al. 2016). All this information comes from the Deep

---

Island Parser (hence DIP) described in the section below.

2 The Deep Island Parser

Conceptually speaking, the deep island parser (hence DIP) is very simple to define, but hard to implement. A semantic island is made up by a set of A/As which are dependent on a verb complex (hence VCX). Arguments and Adjuncts may occur in any order and in any position: before or after the verb complex, or be simply empty or null. Their existence is determined by constituents surrounding the VCX. The VCX itself can be composed of all main and minor constituents occuring with the verb and contributing to characterize its semantics. We are here referring to: proclitcs, negation and other adverbials, modals, reconstruction verbs (lasciare/let, fare/make, etc.), and all auxiliaries. Tensed morphology can then appear on the main lexical verb or on the auxiliaries/modals/reconstruction verbs.

The DIP is preceded by an augmented context-free parser that works on top of a tagger and a chunker. Chunks are labeled with usual grammatical relations on the basis of syntactic subcategorization contained in our verb lexicon of Italian counting some 17,000 entries. There are some 270 different syntactic classes which differentiates also the most common preposition associated to oblique arguments. Position in the input string is assumed at first as a valid criterion for distinguishing SUBJects fro, OBJects. The semantic parser will then be responsible for a relabeling of the output.

The DIP receives a list of Referring Expressions and a list of VCX. Referring expressions are all nominal heads accompanied by semantic class information collected in a previous recursive run through the list of the now lemmatized and morphologically analyzed input sentence. It also receives the output of the context-free parser. The DIP searches for SUBJects at first and assumes it is positioned before the verb and close to it. In case there is none such chunk available the search is widened if intermediate chunks are detected: they can be Prepositional Phrases, Adverbials or simply Parentheticals. If this search fails, the DIP looks for OBJects close after the verb then and again possibly separated by some intermediate chunk. They will be relabeled as Subjects. Conditions on the A/As boundaries are formulated in these terms:

- between current VCX and prospective argument there cannot be any other VCX

Additional constraints regard presence of relative or complement clauses which are detected from the output chunked structure.

The prospective argument is deleted from the list of Referring Expressions and the same happens with the VCX. The same applies for the OBJect, OBJect1 and OBLique. When arguments are completed, the parser searches recursively for ADJuncts which are PPs, using the same boundary constraint formulation above.

Special provisions are given to copulative constructions which can often be reversed in Italian: the predicate coming first and then the subject NP. The choice is governed by looking at referring attributes, which include definiteness, quantification, distinction between proper/common noun. It assigns the most referring nominal to the SUBJect and the less referring nominal to the predicate. In this phase, whenever a SUBJect is not found from available referring expressions, it is created as little_pro and moprhological features are added from the ones belonging to the verb complex. The Predicate-Argument Structure (hence PAS) thus obtained, is then enriched by a second part of the algorithm which adds empty or null elements to untensed clauses.

3 The Classification Procedure

The classification and evaluation procedure is carried out on constituents and their corresponding semantics at propositional level in two steps.

The procedure is preceded by the creation of the model which is made up of the following three components:

- a dictionary of token trigrams, one for every occurrence in a sentence with associated frequency value and sentence id. We will use the following sentence no. AC-01-R0364 as example for the classification.

<sent>'AC-01-R0364'<lik_scl>'1.666666667'</lik_scl><st_err> '0.284267622'</st_err><text>Quando il dipartimento concedeva dei fondi lui spendevano tutti i soldi in trasferte.</text></sent>

The list below represents the sequence of constituents extracted from sentence reported above, with the final punctuation mark added.

The triple below is the first one extracted from the previous list.

tktr(1-[f,fs,f,sn,ibar,sq,sn,ibar,sn,sp,punto]-'AC-01-R0364_1').[2]
tktr(1- (f-fs-sn)-'AC-01-R0364_1').[3]

- a list of sentence constituent types corresponding to the training corpus made of an index, a list of trigrams with their local frequency of occurrence, an evaluation and classification value as derived from the training set: this is the list for the same sentence.

scst('AC-01-R0364'-[1-[f,fs,sn,ibar,sq,sn,ibar,sn,sp,punto],1- (f-fs-sn),1- (fs-sn-ibar),1- (sn-ibar-sq),1- (ibar-sq-sn),1- (sq-sn-ibar),1- (sn-ibar-sn),1- (ibar-sn-sp),1- (sn-sp-punto)]-['1.666666667','0.284267622']).

- a dictionary of type constituent trigrams or unique forms with frequency of occurrence in the whole corpus. For instance the following triple occurs 5 times in the training corpus:
tptr(5- (vcomp-savv-ibar)).

- a list of semantic parameters associated to each sentence, where since semantics is computed at propositional level, the list is constituted by a set of parameters preceded by a lemmatized predicate. Parameters considered are the following ones: agreement (may take on three values: false, true, null); negation (propositions – first slot - but also predicates may be lexically negatively marked! – second slot); speech act (8 different types); factivity (two values).

semp('AC-01-R0364'-[true-concedere-statement-factive-[pos,nil],false-spendere-statement-factive-[pos,neg]]-['1.666666667','0.284267622']).

Overall we collected from the training corpus 12309 token trigrams, 739 type trigrams, 2678 semantic feature sets. We then created the development corpus, by extracting 20% of sentences from the training corpus, which adds up to 414 sentences for the Complexity corpus and 252 sentences for the Acceptability corpus. The corresponding Development models were created by analysing the remaining sentences. We were then able to match the content of two models each for the two tasks: the new model of the reduced Training corpus that we obtained by extracting 20% of sentences which we matched against the corpus of the extracted sentences or DevSet. In order to evaluate the output we decided to consider as correct approximation a value whose difference from the target value was lower than 1. It is important to notice that results are to be referred to sentence level after splitting: this adds 3 more sentences to the Complexity DevSet which turns the total amount from 413 to 416. On the contrary, in the Acceptability DevSet the system didn't split any sentence. Here is the list of additional sentences processed: CO-01-R0317_2, CO-01-R0357_2, CO-01-R0637_2: they are caused by presence of dots which are interpreted by the parser as a possible sentence split.
We report here below Precision and Recall for the DevSet that we evaluated at first against the Training Corpus Model for coverage issues and then against the DevSet Corpus model. Results we obtained are as follows:

Coverage of the DevSet by the Training Corpus Model
- Acceptability
Total sentences processed 249 over 252 corresponding to 98.8%
207 over 249 Likert Scale (83.13%)
203 over 249 Standard Error (81.52%)
- Complexity
Total sentences processed 412 over 416 corresponding to 99.03%
398 over 416 Likert Scale (95.67%)
399 over 416 Standard Error (95.81%)

Results of the DevSet by the Development Corpus Model
- Acceptability
Total sentences processed 250 over 252 corresponding to 99.2%
151 over 252 Likert Scale (59.92%)
140 over 252 Standard Error (55.55%)
- Complexity
Total sentences processed 412 over 416 corresponding to 99.03%
263 over 416 Likert Scale (63.62%)
255 over 416 Standard Error (61.29%)

First step in the classification and evaluation procedure is the constituent trigram matching step. In this step trigrams are computed for the

---

[2] In more detail the sequence of constituents is as follows: [f-[fs-[fs-[Quando],f-[sn-[il dipartimento],ibar-[concedeva], sq-[dei fondi]]], sn-[lui],ibar-[spendevano],sn-[tutti i soldi],sp-[in trasferte]]]. As can be noted, we eliminate functional constituents like "fs" and "f" and keep only those containing a semantic head. We also keep the initial symbol.

[3] We use Italian constituent labels where F stands for S, SN for NP etc. and Phrase is turned into Sintagma.

input text and are matched against the token trigrams dictionary. The matching should produce a list of possible sentence types: we choose the sentence which has more than half of the trigrams matched. The sentence type trigram list is then used to check trigram sequences: here again more than half of the trigrams should be related in sequence. In case this process succeeds we take the associated classification and the evaluation stops. If the process fails, we search the trigram database derived from VIT, which is made of 273,000 (Delmonte et al., 2007) trigrams organized into four frequency related subclasses: rare trigrams with frequency of occurrence including all hapax, dis, trislegomena; frequent trigrams with frequency of occurrence from 4 to 20; very frequent trigrams with frequency of occurrence higher than 20. According to their placement, trigrams are regarded more or less easy to accept vs complex in case their frequency is rare.

VIT (Venice Italian Treebank) is a treebank consisting of 320.000 words created by the Laboratory of Computational Linguistics of the Department of Language Sciences of the University of Venice. The VIT Corpus consists of 57.000 words of spoken text and of 273.000 words of written text. Syntactic annotation was accomplished through a sequence of semi-automatic operations followed by manual validation. The first version of the Treebank was created in the years 1985-88 – manually parsing 40000 words of text with a constituent structure only representation. The resulting structure labels were collected and were used to build a context-free parser for a speech synthesizer (Delmonte R. and R. Dolci, 1991). The theoretical framework behind our syntactic representation was X-bar theory. One peculiarity of VIT is the intention to make it representative of the Italian linguistic syntactic and semantic variety: we thus introduced texts from five different genres – news, bureacratic genre, political genre, scientific genre, literary genre. This made the resulting structures a treebank with a high coverage but very sparse.

4 The Evaluation Module

We assigned rewards and penalties according to a scheme which was partly based on constituency and partly on semantics. In particular, we used agreement, negation, factivity from semantic processing and complex constituency structures from trigram model and a smal set of heuristically determined rules. To check agreement we took the main verb predicate and its morphology and matched this information with the one available on the lexically expressed subject. Here below some examples of semantic information used for agreement matching:

<sent>'AC-01-R0364'<lik_scl>'1.666666667'</lik_scl><st_err> '0.284267622'</st_err><text>
Quando il dipartimento concedeva dei fondi lui spendevano tutti i soldi in trasferte.</text></sent>

Sem = [concedere-statement-factive-[pos, nil], spendere-statement-factive-[pos, neg]]
Agrs = [false]
Negs = [neg]

In addition, we used lexical representations in order to verify the level of matching existing between two predicates. In particular we checked syntactic classes and conceptual classes[4] (Delmonte R., 1989; 1990; 1995).

Here are some verb lexical representation in our lexicon, where we list the root, the conjugation, the syntactic class, the aspectual class, the conceptual class, the list of arguments and their inherent semantic features preceded by constituent type and semantic role. Here below the example of "stonare"/clash

pv(ston,1,inerg,statv,exten,[np/subj1/theme_unaff/[-ani,+hum]]).

where "ston" = is the root, "1" = the conjugation (first implies the morpheme "are" to be adjoined), "intr" = the syntactic type, intransitive or unergative, "statv" = stative, the aspectual class, "exten" = extensional, the conceptual class. The list of possible arguments follows starting from

---

[4] Syntactic lexical classes include the following: tr=transitive; tr_cop=transitive+predicative argument; tr_perc=transitive_perceptive; ditr(+preps)=ditransitive; psych1=psychic 1; psych2=psychic 2; psych3=psychic 3; inac=unaccusative; inerg=unergative; rifl=reflexive; rifl_rec=reflexive reciprocal; rifl_in=reflexive inherent; erg_rifl=ergative reflexive; imp=impersonal; imp_atm=impersonal atmospheric; cop=copulative; mod=modal; C_mov=movement verb + another class; C_prop=propositional verb + another class;
Conceptual lexical classes include the following: ask_poss,at_posit,coerc,dir,dir_difclt,dir_tow,divid,eval,exten,exten_neg,factv, go_against,hold,hyper, inform, ingest, into_hole,let,manip,measu_maj,measu_min,ment_act, not_exten,not_let,not_react,over,percpt, perf,posit, possess,process,propr,react,rep_contr,subj,touch,unit

the "subj1" = subject, which is a "np" Noun-Phrase, and has "theme_unaff" = theme unaffected as semantic role. Semantic features are "-ani" = minus animate, "+hum" = plus human, i.e. only humans and not animate being are selected. In case a verb selects more argument types, the entry is repeated each one containing a different structural construction. This applies for instance to "scoppi"/burst,explode,break out.

pv(scoppi,1,inac,statv,exten,[np/subj1/theme_unaff/[-ani,+hum]]).
pv(scoppi,1,inac,statv,exten,[np/subj1/theme_unaff/[+hum],pp/obl/theme/di/[+abst]]).
pv(scoppi,1,inac,statv,exten,[np/subj1/theme_unaff/[+hum],vinf/vcomp/prop/a/[subj=subj1]]).

In the third entry, we have a quasi-idiomatic form "scoppiare a piangere"/burst into tears, where the infinitival has a subject bound to the higher governing verb's subject. This is done according to principles expressed in LFG theory (Bresnan, 1982; 2001).

Lack of agreement in lexical classes reduces the score associated to the similarity match between the two trigrams under evaluation for the current sentence. Other scoring functions are associated to speech act, grammatical agreement, presence/absence of negation at propositional/lexical level; factivity; complex constituency. Overall we have eight possible features.

Speech Act
Lexical classes:
    syntactic
    conceptual
Negation:
    lexical
    propositional
Agreement
Factivity
Complexity at constituent level

Table 1. Linguistic features used by ItVenses

Thus schematically we have:
Rewards:
0 no wrong agreements; 0 no negation; 0 no non-factive; same conceptual lexical features; similar syntactic lexical features; 0 no complex constituency structures
Else:
penalties (reducing acceptability vs increasing complexity)

Similarity in syntactic lexical classes tends to reduce the more detailed lexical classification into one single label, as for instance the label "transitive" will include: tr (transitive), tr_cop (transitive+predicative argument), tr_perc (transitive_perceptive), ditr(+preps) (ditransitive).

As to constituency complexity we count all constituent labels that are indicators of: sentential complement represented by FAC (Italian for SCOMP); subordinator for subordinate clause, CP; complementizer or interrogative pronoun represented by CP; relative clause, F2; coordinate clause, FC. According to the quantity of one or more of these constituent labels, we assign penalties or rewards. The decision is determined by heuristics but also by the length in number of constituents. For instance, 2 CP + 1 FAC will be computes as a penalty; 1 CP, 1 FAC, 1 F2 again penalty, however length in terms of constituents should be higher than 8. We also address specific constituent sequences which indicate complex or hard to understand structures as for instance the sequence:

[…, fc,sn,vcomp,sn,punto]

which classifies some 20 sentences in the Acceptability test set, one of which is sentence n. AC-OC-02-R0569:

"Ci dissero chi Maria aveva chiamato un uomo e Marco visitato l'anziano signore."

This sentence is ungrammatical due to presence of a lexical Object NP in the extraction place of the interrogative pronoun "chi". However this case of ungrammaticality is hard to detect solely on the base of constituent sequences because the NP containing "chi" is not lexically marked. On the contrary, the final participial clause is easily detectable.

The evaluation algorithm starts by searching trigrams collected in the current sentence analysis and by trying to match them with the ones memorized in the training set model. The search is successful if one or more matches have been obtained which have 3 or more trigrams. The following step is then collecting features as indicated in Table 1. from the syntactic and semantic output of the parser. These features are matched against the ones that are associated to each trigram sequence collected in the previous step. The matching algorithm receives a vector made of six slots:

match(Strct,Pred,Agrs,Negs,Fact,Spacs)

where, “Strct” stands for constituent structure; “Pred”, is the verbal predicate lemma; “Agrs”, is a binary value (true/false) for subject-verb agreement; “Negs” is a pair of binary values (neg/nil) for negation at lexical and propositional level; “Fact” is again a binary value (true/false) for factivity at propositional level; “Spacs” is one of the seven possible labels[5] used to classify speech act. For instance, in the case of sentence no. 'AC-01-R0364' above, the following counts are generated automatically:

Fact = ['AC-01-R0440_1'-factive, 'AC-01-R0440_1'-factive]
Spacs = [statement, statement]
N = N1 = Va = 0 [negation1, negation2]
N2 = N3 = 2 [agreement] *penalty
Sum = Val = 4 [final score] *penalty

5 Results and Discussion

As said above, results are not successful. In particular, results for the Complexity Task are well below the Baseline. Results for the Acceptability Task are higher and in one case they almost double the Baseline.

COMPL Task
RUN 1
Mean-Correlation: 0.312796825885, p value < 0.001
STD ERR-Correlation: 0.096751776, p value < 0.05
RUN 2
Mean-Correlation: 0.305504444563, p value < 0.001
STD ERR-Correlation: 0.0729839133, p value > 0.05

ACCEPT Task
RUN 1
Mean-Correlation: 0.441645891, p value < 0.001
STD ERR Correlation: 0.248478821, p value < 0.001
RUN 2
Mean-Correlation: 0.494713038815, p value < 0.001
STD ERR-Correlation: 0.405850132, p value < 0.001

As can be easily gathered, differences between Run-1 and Run-2 are not particularly high in the Complexity Task. Not so in the Acceptability task where Run-2 exceeds Run-1 by 0.053 points. Run-2 in both tasks is characterized by a different strategy determined by a policy of feature ablation. What we did, was trying to verify whether the presence of each of the eight features had an important impact on the final result and to what extent. Eventually, we found out that the use of lexical negation was not so relevant and so we deleted it from the final count. And that was the decision that determine the result for Run-2. The different behaviour of the system in the two tasks can be due to the length of the sentences which in the Complexity task is much longer. The system produces results for each proposition and not for the sentence as a whole – we don’t count relative and complement clauses as separate propositions. When generating the final document for the two runs we did not have a strategy in deciding in many cases, which proposition we had to choose as a representative of the whole sentence. We decided we could not make an average between the two or three propositions so we simply selected always the result obtained by the first proposition. This choice applied to 51 sentences, 41 with two propositions and 10 with three propositions. The Complexity text also suffered from failure of the parser in three sentences. We also have to consider the presence of 62 results determined heuristically, i.e. the system did not find the corresponding trigrams in the training set, so it used the VIT database and generated the final statistics by a set of heuristics. No such problems arose in the Acceptability Task, where all sentences where constituted by a single proposition. However, we had a higher number of heuristically determined statistics, 86. If we had the possibility to present more runs, then we could have achieved better results in the Complexity task.

6 Conclusion

We presented the results of our system for the two tasks Complexity and Acceptability. The system uses constituency-based trigrams associated to the semantics of each proposition. Evaluation is based on presence/absence of agreement/match between linguistic features, determined at a lexical, syntactic and semantic level. Worst results obtained for the Complexity Task may be due partly to the length of the sentences, which required a specific strategy in choosing the most relevant classification at propositional level. We concentrated our work on the use of constituent trigrams and did not consider the possibility to use ngrams based on words or lemmata which we had available from our deep analysis. In the future, we intend to use the same approach we produced for the other tasks of EVALITA which are all based on automatically

---

[5] We use the following: statement, question, exclamation, negated, unreal, opinionsubjective, conditional

generated fully supervised ngram models together with the one presented here.

References

Basile, Valerio and Croce, Danilo and Di Maro, Maria, and Passaro, Lucia C., 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian, in Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020), CEUR.org.

Bresnan Joan (ed.), 1982. The Mental Representation of Grammatical Relations, The MIT Press, Cambridge MA.

Bresnan Joan, 2001. Lexical function syntax. Oxford, Blackwell Publishers.

Delmonte, R., A. Bristot, S. Tonelli, 2007. VIT - Venice Italian Treebank: Syntactic and Quantitative Features, in K.De Smedt, Jan Hajic, Sandra Kübler(Eds.), Proc. Sixth International Workshop on TLT, Nealt Proc. Series Vol.1, 43-54.

Delmonte R., 1995. Lexical Representations: Syntax-Semantics interface and World Knowledge, in Notiziario AIIA (Associazione Italiana di Intelligenza Artificiale), Roma, pp.11-16.

Delmonte R., 1990. Semantic Parsing with an LFG-based Lexicon and Conceptual Representations, Computers & the Humanities, 5-6, 461-488.

Delmonte R., R.Dolci, 1991. Computing Linguistic Knowledge for a Text-To-Speech System With PROSO, in Proc. EUROSPEECH '91 – Second European Conference on Speech Communication and Technology, Genova, ISCA, Archive, pp. 1291-1294.downloadable at https://www.isca-speech.org/archive/ eurospeech_1991/e91_1291.html

Delmonte R., 2014. ITGETARUNS A Linguistic Rule-Based System for Pragmatic Text Processing, Proceedings of Fourth International Workshop EVALITA 2014, Pisa, Edizioni PLUS, Pisa University Press, vol. 2, pp. 64-69.

Ohana, B. and B. Tierney and S.J. Delany, 2016, Sentiment Classification Using Negation as a Proxy for Negative Sentiment, in Proceedings of 29th FLAIRS Conference, AAAI, 316-321.

Polanyi, Livia and Zaenen, Annie 2006. "Contextual valence shifters". In Janyce Wiebe, editor, Computing Attitude and Affect in Text: Theory and Applications. Springer, Dordrecht, 1–10.

Stingo M., & R. Delmonte, 2016. Annotating Satire in Italian Political Commentaries with Appraisal Theory, IN Larry Birnbaum, Octavian Popescu and Carlo Strapparava (eds.), Natural Language Processing meets Journalism - Proceedings of the Workshop, NLPMJ-2016, 74-79.

KIPoS: Part-of-speech Tagging on Spoken Language

KIPoS @ EVALITA2020:
Overview of the Task on KIParla Part of Speech Tagging

Cristina Bosco*, **Silvia Ballarè**$^{\diamond}$, **Massimo Cerruti**$^{\oplus}$, **Eugenio Goria**$^{\oplus}$, **Caterina Mauri**$^{\ominus}$
*Dipartimento di Informatica, Università degli Studi di Torino
$^{\diamond}$Dipartimento di Filologia Classica e Italianistica, Università degli Studi di Bologna
$^{\oplus}$Dipartimento di Studi Umanistici, Università degli Studi di Torino
$^{\ominus}$Dipartimento di Lingue, Letterature e Culture Moderne, Università degli Studi di Bologna
{`cristina.bosco,massimosimone.cerruti,eugenio.goria`}`@unito.it`,
{`silvia.ballare,caterina.mauri`}`@unibo.it`

Abstract

English. The paper describes the first task on Part of Speech tagging of spoken language held at the Evalita evaluation campaign, KIPoS. Benefiting from the availability of a resource of transcribed spoken Italian (i.e. the KIParla corpus), which has been newly annotated and released for KIPoS, the task includes three evaluation exercises focused on formal versus informal spoken texts. The datasets and the results achieved by participants are presented, and the insights gained from the experience are discussed.

Italiano. *L'articolo descrive il primo task sul Part of Speech tagging di lingua parlata tenutosi nella campagna di valutazione Evalita. Usufruendo di una risorsa che raccoglie trascrizioni di lingua italiana (il corpus KIParla), annotate appositamente per KIPoS, il task è stato focalizzato intorno a tre valutazioni con lo scopo di confrontare i risultati raggiunti sul parlato formale con quelli ottenuti sul parlato informale. Il corpus di dati ed i risultati raggiunti dai partecipanti sono presentati insieme alla discussione di quanto emerso dall'esperienza di questo task.*

1 Motivation

Even (Bosco et al., 2020) though in the last decades we have witnessed an increase in the resources available for the study of spoken Italian, a great unbalance can still be observed between spoken and written corpora, from different angles. Written corpora are generally larger, are able to provide a lot of information about the texts they include, and may count on a vast array of computational tools for morphological analysis and syntactic parsing. Conversely, spoken corpora of Italian are generally smaller, often give a minimum of information concerning the speakers and the context in which the interaction takes place and, finally, provide at most basic PoS-tagging and lemmatization tools. This, of course, poses considerable limitations on the searches that may be performed on these resources, eventually leading to a possible *written language bias* due to the different availability and richness of information of written vs. spoken corpora (Linell, 2005).

As a consequence of this unbalance, corpus-based sociolinguistic analyses of spoken Italian, which need a comprehensive set of metadata, have rarely been put to the test on publicly available speech corpora. In fact, most sociolinguistic studies have been conducted on ad hoc-collected datasets, see *inter al.* (Alfonzetti, 2002; Mereu, 2019).

The KIParla corpus (Mauri et al., 2019) (661k tokens approximately), which is available at the website `www.kiparla.it`, has been designed to overcome some shortcomings of previous resource tools. KIParla is a corpus of spoken Italian which encompasses various types of interactions between speakers of different origins and socioeconomic backgrounds. It consists of speech data collected in Bologna and Turin between 2016 and 2019, and contains two independent modules, i.e. KIP (cf. sec. 3) and ParlaTO. Among other things, KIParla provides a wide range of metadata, including situational characteristics (such as the symmetrical vs. asymmetrical relationship between the participants) and socio-demographic information for each speaker (such as age and level of education). Nevertheless, the lack of PoS-tagging and lemmatization currently places severe limits on its

---

application.

In order to enrich the scenario of investigation to be applied on the KIParla corpus, we proposed the KIPoS task. Following the experience of the Evalita 2016 PoSTWITA task on PoS tagging Italian Social Media Texts (Bosco et al., 2016) and the subsequent development of an Italian treebank for social media (Sanguinetti et al., 2017; Sanguinetti et al., 2018), where the issues related to a particularly challenging written text genre were addressed, KIPoS offers the opportunity of addressing the theoretical and methodological challenges related to PoS tagging of Italian spontaneous speech texts. Carrying out this task means processing a type of data that is known to be problematic for computational treatment, that is unplanned spoken language (as opposed to experimental speech data). PoS tagging of this corpus entails dealing with both a wide range of spontaneous speech phenomena and a great amount of sociolinguistic variation.

The most challenging aspects to be addressed in the unconstrained speech of KIParla are:

- To identify mode-specific phenomena, such as repetitions, reformulations, fillers, incomplete syntactic structures, etc.
- To trace a relevant set of non-standard alternatives back to the same linguistic phenomenon (e.g. the presence of socio-geographically marked forms like *annà* or *andà*, equal to standard Italian *andare* "to go"), either assigning them to the correct part-of-speech, or working out an ad-hoc solution.
- To deal with different types of interaction and registers (casual conversations, interviews, office hours, etc.) with a variable number of participants (1 to 5), each transcribed on a separate line and corresponding to an autonomous text string.

PoS-tagging of data from KIParla corpus is intended to bring an improvement to the current practices in use for tagging and parsing spoken Italian. Furthermore, this result is also significant for the purposes of (socio)linguistic research, in that the availability of annotated spoken corpora enables the researcher to validate previous assumptions based on smaller or less informative datasets, but also to collect knowledge to be meaningfully used in the development of automatic conversation systems and chatbots.

2 Definition of the task

Given the innovative features of KIParla, we proposed KIPoS as a task for EVALITA 2020 (Basile et al., 2020) to address the issues involved in the adaptation of a PoS tagger to the specific features of oral text, in order to systematically represent those features and to provide the mean to access to their specificities. We provided therefore data for training (i.e. Development Set, henceforth DEVSET) and testing (Test Set, henceforth TESTSET) systems organized in two ensembles which respectively represent formal (DEVSET–formal and TESTSET–formal) and informal texts (DEVSET–informal and TESTSET–informal). This allowed us to consider one main task and two subtasks, which are described as follows:

- Main task - general: training on all given data (both DEVSET–formal and DEVSET–informal) and testing on all test set data (both TESTSET–formal and TESTSET–informal)
- Subtask A - crossFormal: training on data from DEVSET–formal only, and testing separately on data from formal texts (TESTSET–formal) and from informal texts (TESTSET–informal)
- Subtask B - crossInformal: training on data from DEVSET–informal only, and testing separately on data from formal texts (TESTSET–formal) and from informal texts (TESTSET–informal).

While all tasks are oriented to investigate how challenging can it be to PoS-tag spontaneous speech data, the cross ones are especially useful for validating the hypothesis that some differences occur between the tagging of formal conversations and that of informal conversations. As we will see in section 5 and 6, this hypothesis is partially confirmed by results. Some example useful to draw the difference among the registers is provided in the next section.

3 Datasets

All the data provided for the KIPoS task are extracted from the KIP module (see Section 1),

Dataset	Register	Speakers	Turns	Tokens
DEVSET	Formal	5	1.998	13.864
	Informal	11	3.804	19.259
TESTSET	Formal	2	459	3.642
	Informal	2	582	3.532

Table 1: The sizes of the datset.

which includes various communicative situations occurring in the academic context. As explained in detail in (Mauri et al., 2019), the recordings involve five different types of interactions, each of which is assigned for the aims of KIPoS either to the section of formal texts or to the section of informal texts (mainly on the basis of the relationship between the participants, i.e. asymmetrical vs. symmetrical).

The KIP corpus structure can thus be outlined as follows:

- Formal dataset:
  - lessons
  - office hours
  - oral examinations
- Informal dataset:
  - semi-structured interviews
  - casual conversations.

Below are examples of formal (1) and informal (2) texts.

(1)[1]

```
BO088: una volta che carlo magno
conquisto' l'italia fu permesso ad
anselmo di tornare eh a mantova
BO088: nel settecentosettantaquattro
BO088: ehme cosi' po pote' riprendere
la sua attivita' prima eh di creazione
della biblioteca
BO088: perche' secondo appunto l'uso eh
delle biblioteche eh
BO088: medioev medievali diciamo prima
eh vi era
BO088: mh la insomma la raccolta di
libri dall'esterno
```

(2)[2]

```
BO003: povero cristo sono andata a
beccare questo
BO002: ma poi scusa il piu' carino di
tutti lo cornifichi
BO003: si' si' si' esa poi secondo me
lui e' il piu' carino di tutti
BO003: cioe' tra per i miei gusti tra il
gruppo
BO002: no eh
BO002: carino sia
BO002: di viso ma anche
BO003: poi e' anche il piu' si' si' si'
e' cornificatissimo non cornificato
```

Both excerpts feature spontaneous speech phenomena, such as fillers, repetitions and reformulations. However, example 1 shows several characteristics of formal styles, either cross-linguistically shared (e.g. clausal subordination, passive construction, abstract and specific terms) or language-specific (e.g. existential construction with *vi* as pre-copular proform); while example 2 displays various features which are typical of informal styles, such as simple sentence structure and pragmatically-marked word orders (e.g. *il più carino di tutti lo cornifichi*), multi-functional words (e.g. *carino*), colloquialisms (e.g. *povero cristo*, *beccare*, *cornifichi*, *cornificato*), elatives (e.g. *cornificatissimo*), deictics (e.g. *questo*, *lui*) and discourse markers (e.g. *cioè*, *scusa*).

All speakers were informed of the aims of the project, agreed to the recording and signed a consent form.

The set of data exploited for KIPoS precisely consists of around 200K tokens, corresponding to approximately one-third of the whole KIParla corpus, with an equal proportion of informal and formal speech data.

For the purposes of KIPoS, the UDpipe trained on all the treebanks available for Italian within the Universal Dependencies repository[3] has been applied on this 200K tokens portion of the KIParla corpus. Among these data, approximately 30K tokens have been submitted to a careful manual check and correction[4] and released as training sets of the KIPoS task (i.e. DEVSET–formal and

---

[1] KIP Corpus, BOC1001, oral examination

[2] KIP Corpus, BOA3001, casual conversation

[3] `https://universaldependencies.org/it/index.html`

[4] We thank three students for their precious help: Filippo Mulinacci, Martina Pittalis and Roberto Russo of the Department of Modern Languages, Literatures and Cultures of the University of Bologna.

Team	Affiliation
UniBO	FICLIT – University of Bologna
UniBA	University of Bari ”Aldo Moro”
KLUMSy	Friedrich Alexander Universität Erlangen-Nürnberg & Universität Stuttgart

Table 2: The teams which participated to KIPoS and their affiliation.

DEVSET–informal). From the remaining automatically annotated data, we extracted the formal-TESTSET and informal-TESTSET, and we also manually checked and validated them. Finally, we released as a silver standard (i.e. SILVERSET) the remaining data. They have been also made available together with the other data[5] to be used for training participants' systems.

3.1 Annotation

As far as the annotation is concerned, for the purpose of the task, the original orthographic transcriptions were provided in a tab-delimited .txt format. Three are the main identifiers we used in this format, respectively indicating the conversation (alphanumeric), the speaker's ID (alphanumeric) and the position of the turn (numeric) within the context of the conversation. For instance, the example below includes the first three turns of the conversation ”BOD2018”[6], in which three different speakers are involved (”1_MP_BO118”, ”2_MP_BO118” and ”3_AM_BO140”):

```
# conversation = BOD2018
# speaker = 1_MP_BO118
# turn = 1
# text = dovresti parlarmi della tua casa
1 dovresti AUX
2-3 parlarmi VERB_PRON
2 parlar VERB
3 mi PRON
4-5 della ADP_A
4 di ADP
5 la DET
6 tua DET
7 casa NOUN

# conversation = BOD2018
# speaker = 2_MP_BO118
# turn = 2
# text = attuale
1 attuale ADJ
```

```
# conversation = BOD2018
# speaker = 3_AM_BO140
# turn = 3
# text = mh sì
1 mh PARA
2 sì INTJ
```

The format and the labels for tagging the part of speech of the KIPoS data are compliant with that provided in the Universal Dependencies Italian treebanks. Data were indeed released in a CoNNL-U - like format, but which only includes the three first columns of it, separated by tab keys as usually. For a detailed list and description of the tagset used in KIPoS datasets, see the Appendix at the end of this paper.

3.2 Tokenization Issues

For what concerns words including multiple tokens, in the data released for the development and training of participant systems (DEVSET–formal and DEVSET–informal), we annotated their compound and splitting both. See for instance, in the first turn of the example above lines 2-3, 2 and 3: a verb with clitic suffix occurs and it is annotated as a compound in line 2-3, while its components, i.e. the verb and the clitic, are separately annotated on line 2 and 3 respectively.

In contrast, for the purpose of the evaluation, the format applied on the test set (TESTSET–formal and TESTSET–informal) only includes a word for each line, regardless of the fact that a word may be composed of more than one token. This makes the format of the test set slightly different from that used in the development data, but more compliant with the evaluation scripts and procedures. An example of this format follows, which consists in the first turn of the example above:

```
# conversation = BOD2018
# speaker = 1_MP_BO118
# turn = 1
# text = dovresti parlarmi della tua casa
1 dovresti AUX
2 parlarmi VERB_PRON
3 della ADP_A
4 tua DET
5 casa NOUN
```

---

[5] All the data annotate for KIPoS are available at `https://github.com/boscoc/kipos2020`, with the licence and the annotation guidelines.

[6] The alphanumeric code used to name the KIP's conversations provides information about the city in which the the data has been collected (BO= Bologna, TO=Turin) and the kind of interaction (A1=office hours, A3=free conversation, C1=exams, D1=lessons, D2=interviews). For example, BOD2018 is a semistructured interview recorded in Bologna.

Task	DEVSET	TESTSET	Team	Score
Baseline (from POSTWITA)				0.9319
Main	formal and informal	formal	UniBO	**0.934880**
			KLUMSy	0.875629
			UniBA	0.815819
		informal	UniBO	0.911316
			KLUMSy	0.882368
			UniBA	0.793684
Task A	formal	formal	KLUMSy	0.873672
			UniBA	0.787311
		informal	KLUMSy	**0.875789**
			UniBA	0.757895
Task B	informal	formal	KLUMSy	0.878144
			UniBA	0.771101
		informal	KLUMSy	**0.881053**
			UniBA	0.775000

Table 3: The official scores achieved by participants for the three subtasks (Main, Task A and Task B), by training systems on both or one of the datasets provided for development (DEVSET–formal and DEVSET–informal), on the TESTSET–formal and TESTSET–informal (best scores for each subtask in bold face).

In this example, the verb with clitic suffix ”parlarmi” (speak to me) has been annotated as a compound on a single line, i.e. line 2.

4 Evaluation measures

For the KIPoS task a single measure has been used for the evaluation of participants' runs, i.e. accuracy, which is defined as the number of correct Part-of-Speech tags assignment divided by the total number of tokens in the gold TESTSET. The evaluation metric will be based on a token-by token comparison and only a single tag is allowed for each token.

The evaluation is performed in a black box approach, where only the systems output is evaluated.

5 Participation and Results

As depicted in table 3, where the main task and the two subtasks results are presented at glance, three teams submitted their runs for KIPoS (see table 2 for their affiliation). Nevertheless, one team participated to the main task only, while the other two provided results for Task A and B too.

The three teams applied different approaches. UniBA team used a combination of two taggers implementing two different approaches, namely stochastic Hidden Markov Model and rule-based. UniBO applied a fine-tuning approach to Part of Speech tagging that is based on a pre-trained neural language BERT-derived model (UmBERTo) and an adapted fine-tuning script.

KLUMSy used a tagger based on the averaged structured perceptron, which supports domain adaptation and can incorporate external resources for dealing with the limited availability of in-domain data.

The overall higher accuracy has been achieved in the main task by the UniBO team on the TESTSET-formal. The availability of a larger training corpus for the main task, which includes the DEVSET–formal and the DEVSET–informal both, and the results calculated on both the portions of the TESTSET allowed, as expected, the achievement of the KIPoS overall best score. This is confirmed also by the fact that all teams provided their best runs in it, for formal and informal register both. Even if the official submission of UniBO did not include the runs for Task A and B, the results it provided in its report (Tamburini, 2020) show indeed that also this team has ranked worst in Task A and B than in the main one. More precisely, for Task A, it achieved 0.8647 accuracy on TESTSET–formal and 0.8316 on TESTSET–informal, while in Task B it achieved 0.8974 on TESTSET–formal and 0.8952 on TESTSET–informal.

As far as the other teams are concerned, UniBA

provided in its report (Izzi and Ferilli, 2020) also the results achieved using a version of the TESTSET where a few errors detected after the official evaluation has been fixed. This allowed a small improvement in their scores (e.g. in the main task, +0.0078 for formal and +0.0056 for informal register).
The KLUMSy team provided the best runs for both registers in Task A and B, but in its runs, because of a misunderstanding of the guidelines about the annotation of contractions in the TESTSET (which is slightly different with respect to the DEVSET), a certain amount of mis-tagged tokens occurred. After they were fixed, also the scores of this team were improved (with an increase that varies from 0.0456 to 0.0187) with respect to the official ones reported in table 3, as described in the report of this team (Proisl and Lapesa, 2020).

Considered that the PoS tagging is a task mostly solved, it is not surprising that the participants' scores are quite high and close for all the tracks. The larger difference observed between the best and the worst score is indeed 0.126, and it is referred to Task B on TESTSET–formal.
Given the peculiarity of oral text on which KIPoS is focused, it seems not especially meaningful a comparison of our results with state-of-the-art Pos taggers results for the written standard language. A more interesting comparison can be instead developed with respect to the scores achieved within the PoSTWITA task (Bosco et al., 2016) on written texts extracted from social media. This genre is indeed often considered in between written and oral, sharing some feature with the former and some with the latter. Using the best PoSTWITA task accuracy score (0.9319) as our *baseline* (see table 3), we can observe that the best scores achieved in KIPoS are in line with this result. This confirms the hypothesis that oral text can be considered as almost equally hard to be morphologically tagged than social media.

As far as the distinction between formal and informal conversation drawn in the KIPoS datasets is concerned, a general trend of better scoring in formal data tagging can be observed, but some meaningful difference among participant systems occurs. For all subtasks UniBO best scored in formal text, while KLUMSy did the same in informal data. UniBA achieved instead its best scores on TESTSET–formal with the exception of Task B where its score for the informal test set is a little bit (0.0038) higher than that for the formal one.
Focusing on the cross subtasks A and B, we can moreover notice that systems were not equally influenced by the type of data exploited for training: UniBO provided best scores against TESTSET–formal also when trained on DEVSET–informal (Task B), while KLUMSy provided best scores against TESTSET–informal also when trained on DEVSET–formal (Task A). UniBA seems instead slightly more influenced by the features of data used in training.

6 Discussion and Conclusion

The results described in this report can be only considered as preliminary. First of all, KIPoS is the first edition of a task about PoS tagging of spontaneous speech for Italian and there aren't other results about this kind of task for the same language to be compared with. Second, the corpus used for KIPoS has been newly released for the purpose of the task and never used before. Participants provided some useful feedback about errors occurring in the DEVSET and TESTSET, but some further check should be applied for improving the quality of data. Finally, only three participants submitted their runs (and only two provided official runs for cross-genre tasks). Even if PoS tagging is among the tasks which are considered as mostly solved in literature, only a larger participation may allow a meaningful comparison among different approaches and results.

Nevertheless, the KIPoS task produced the valuable result of making available a novel resource for the study of spoken Italian and for the advancement of NLP in this area. It can be of great relevance for the investigation of both spontaneous speech phenomena and sociolinguistic variation, but also e.g. in the development of chatbots and vocal recognition systems.
In particular, the insights gained within the context of this Evalita evaluation campaign for PoS tagging can pave the way for further investigating actual speech data. They provide a solid foundation for our future research also in the direction of more detailed morphological analysis and syntactic parsing, especially within the framework of Universal Dependencies where we would like to release the KIPoS dataset in the near future.

7 Acknowledgments

The construction of part of the corpus has been possible thanks to the financing of the *Fondazione CRT* under the *Erogazioni ordinarie 2018* program. The KIParla corpus has been made possibile thanks to SIR Project 'LEAdHOC' (n. RBSI14IIG0), funded by MIUR. We would like to thank also the students from our BA and MA courses at the Universities of Bologna and Torino, who participated in collecting and transcribing the data.

References

Giovanna Alfonzetti. 2002. *La relativa non-standard. Italiano popolare o italiano parlato?* Centro di Studi Filologici e Linguistici Siciliani, Palermo.

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Cristina Bosco, Fabio Tamburini, Andrea Bolioli, and Alessandro Mazzei. 2016. Overview of the EVALITA 2016 Part Of Speech on TWitter for ITAlian task. In *Proceedings of Evalita 2016*.

Cristina Bosco, Silvia Ballarè, Massimo Cerruti, Eugenio Goria, and Caterina Mauri. 2020. KIPoS @ EVALITA2020: Overview of the Task on KIParla Part of Speech tagging. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Giovanni Luca Izzi and Stefano Ferilli. 2020. UniBA@KIPoS: A Hybrid Approach for Part-of-Speech Tagging. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Per Linell. 2005. *The written language bias in linguistics: its nature, origins and transformations*. Routledge, London – New York.

Caterina Mauri, Silvia Ballarè, Eugenio Goria, Massimo Cerruti, and Francesco Suriano. 2019. KIParla Corpus: A New Resource for Spoken Italian. In Raffaella Bernardi, Roberto Navigli, and Giovanni Semeraro, editors, *Proceedings of the 6th Italian Conference on Computational Linguistics (CLiC-it 2019)*, Online. CEUR.org.

Daniela Mereu. 2019. *Il sardo parlato a Cagliari*. Franco Angeli, Milano.

Thomas Proisl and Gabriella Lapesa. 2020. KLUMSy@KIPoS: Experiments on Part-of-Speech Tagging of Spoken Italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Manuela Sanguinetti, Cristina Bosco, Alessandro Mazzei, Alberto Lavelli, and Fabio Tamburini. 2017. Annotating Italian social media texts in Universal Dependencies. In *Proceedings of the Fourth International Conference on Dependency Linguistics (Depling 2017)*, pages 229–239.

Manuela Sanguinetti, Cristina Bosco, Alberto Lavelli, Alessandro Mazzei, and Fabio Tamburini. 2018. PoSTWITA-UD: an Italian Twitter Treebank in Universal Dependencies. In *Proceedings of the 11th Language Resources and Evaluation Conference (LREC 2018)*, pages 1768–1775.

Fabio Tamburini. 2020. UniBO@KIPoS: Fine-tuning the Italian "BERTology" for PoS-tagging Spoken Data. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

APPENDIX: The KIPoS tagset

Tag	**Value(s)**	**Examples**
ADJ	• Qualifying, numeral, possessive adjectives • Interrogative adjectives • Adjectives used as pro-forms	*una **bella** casa* ***quanti** anni hai?* *-ci vediamo domani? -**esatto***
ADP	• Prepositions • Pospositions	*di, a, da, **senza** te, tranne, ...* *vent'anni **fa***
ADP_A	• Articled prepositions	*dalla, nella, sulla, ...*
ADV	• Adverbs • Interrogative adverbs	*lo metto **qui*** *non ricordo **come** si chiama*
AUX	• Auxiliaries • Modals • Periphrastic auxiliaries	*essere, avere* *potere, volere, dovere* ***sta** mangiando, **viene** visto, ...*
CCONJ	• Coordinating conjunctions • Discourse markers with predominantly connective function	*e, ma, o, però, anzi, quindi, dunque, ...*
DET	• Articles • Demonstratives • Numerals • Possessives • Quantifiers	*ho visto **un** film* *la senti **questa** voce?* *ho giocato **tre** numeri al lotto* *non nominare **mia**sorella* ***alcuni** studenti sono assenti*
DIA	• Italo-Romance dialects	*c'erano due **fiulin***
INTJ	• Interjections	*sì, no, ecco, ...*
LIN	• Languages other than Italian	*vi saluto **guys***
NEG	• Sentence negation	*non*
NOUN	• Nouns of any type except proper nouns	*ho visto un **re***
NUM	• Numbers (but not numeral adjectives)	*- quanti sono? -**tre***
PARA	• Paraverbal communication	*eh, mh, oh, bla bla, …*
PRON	• Personal and reflexive pronouns • Interrogative pronouns • Relative pronouns	*io, me, tu, te, sé, ...* *chi?, cosa?, quale?, che?* *il quale, dove, cui*
PROPN	• Proper nouns	*Gigi*
SCONJ	• Subordinating conjunctions	*dove, quando, perché* *ho detto **che**…* ***se** vuoi*
VERB	• Verbs	***aveva** vent'anni* ***era** molto stanco*
VERB_PRON	• Verb + clitic pronoun cluster	*mangiarlo, donarglielo, …*
X	• Other (e.g. truncated words)	*fior-*

UniBO @ KIPoS: Fine-tuning the Italian "BERTology" for PoS-tagging Spoken Data

Fabio Tamburini
FICLIT - University of Bologna, Italy
`fabio.tamburini@unibo.it`

Abstract

English. The use of contextualised word embeddings allowed for a relevant performance increase for almost all Natural Language Processing (NLP) applications. Recently some new models especially developed for Italian became available to scholars. This work aims at applying simple fine-tuning methods for producing high-performance solutions at the EVALITA KIPOS PoS-tagging task (Bosco et al., 2020).

Italian. *L'utilizzazione di* word embedding *contestuali ha consentito notevoli incrementi nelle performance dei sistemi automatici sviluppati per affrontare vari task nell'ambito dell'elaborazione del linguaggio naturale. Recentemente sono stati introdotti alcuni nuovi modelli sviluppati specificatamente per la lingua italiana. Lo scopo di questo lavoro è valutare se un semplice* fine-tuning *di questi modelli sia sufficiente per ottenere performance di alto livello nel task KIPOS di EVALITA 2020.*

1 Introduction

The introduction of contextualised word embeddings, starting with ELMo (Peters et al., 2018) and in particular with BERT (Devlin et al., 2019) and the subsequent BERT-inspired transformer models (Liu et al., 2019; Martin et al., 2020; Sanh et al., 2019), marked a strong revolution in Natural Language Processing (NLP), boosting the performance of almost all applications and especially those based on statistical analysis and Deep Neural Networks (DNN).

This work heavily refers to an upcoming work of the same author (Tamburini, 2020) experimenting various contextualised word embeddings for Italian to a number of different tasks and it is aimed at applying simple fine-tuning methods for producing high-performance solutions at the EVALITA KIPOS PoS-tagging task (Bosco et al., 2020; Basile et al., 2020).

2 Italian "BERTology"

The availability of various powerful computational solutions for the community allowed for the development of some BERT-derived models trained specifically on big Italian corpora of various textual types. All these models have been taken into account for our evaluation. In particular we considered those models that, at the time of writing, are the only one available for Italian:

- Multilingual BERT[1]: with the first BERT release Google developed also a multilingual model ('bert-base-multilingual-cased' – *bertMC*) that can be applied also for processing Italian texts.
- AlBERTo[2]: last year a research group from the University of Bari developed a brand new model for Italian especially devoted to Twitter texts and social media ('m-polignano-uniba/bert_uncased_L-12_H-768_A-12_italian_alb3rt0' – *alUC*) (Polignano et al., 2019). Only the uncased model is available to the community. Due to the specific training of alUC, it requires a particular pre-processing step for replacing hashtags, urls, etc. that alter the official tokenisation, rendering it not really applicable to word-based classification tasks in general texts; thus, it will be used only for

---

[1] https://github.com/google-research/bert

[2] https://github.com/marcopoli/AlBERTo-it

working on twitter or social media data. In any case we tested it in all considered tasks and, whenever results were reasonable, we reported them.

- GilBERTo[3]: it is a rather new CamemBERT Italian model ('idb-ita/gilberto-uncased-from-camembert' – *giUC*) trained by using the huge Italian Web corpus section of the OSCAR (Ortis Suárez et al., 2019) project. Also for GilBERTo it is available only the uncased model.

- UmBERTo[4]: the more recent model developed explicitly for Italian, as far as we know, is UmBERTo ('Musixmatch/umberto-commoncrawl-cased-v1' – *umC*). As well as GilBERTo, it has been trained by using OSCAR, but the produced model, differently from GilBERTo, is cased.

3 KIPOS 2020 PoS-tagging Task

Part-of-speech tagging is a very basic task in NLP and a lot of applications rely on precise PoS-tag assignments. Spoken data present further challenges for PoS-taggers: small datasets for system training, short training sentences, less constrained language, the massive presence of interjections, etc. are all examples of phenomena that increase the difficulties for building reliable automatic systems.

The PoS-tagging system used for our experiments is very simple and consist of a slight modification to the fine tuning script '`run_ner.py`' available with the version 2.7.0 of the Huggingface/Transformers package[5]. We did not employ any hyperparameter tuning, and, as the stopping criterion, we fixed the number of epoch to 10 and chose the UmBERTo model on the basis of the previous experience (Tamburini, 2020). After the challenge, we evaluated all the BERT-derived models in order to propose a complete overview of the available resources.

Table 1 shows the results obtained by fine tuning all the considered BERT-derived models for the Main Task. A very relevant increase in performance w.r.t. the other participants is evident looking at the results and UmBERTo is consistently the best system.

System	**Main Task Accuracy**		
	Form.	**Inform.**	**Both**
Fine-Tuning$_{\text{umC}}$	**93.49**	**91.13**	**92.26**
Fine-Tuning$_{\text{giUC}}$	92.96	89.92	91.38
Fine-Tuning$_{\text{alUC}}$	90.02	89.82	89.92
Fine-Tuning$_{\text{bertMC}}$	91.67	88.05	89.79
2nd ranked system	87.56	88.24	87.91
3rd ranked system	81.58	79.37	80.43

Table 1: PoS-tagging Accuracy for the EVALITA KIPOS 2020 benchmark for the Main Task. The Fine-Tuning$_{\text{umC}}$ has been submitted for the challenge as the system "UniBO".

We did not participate at the official challenge for the two subtasks, but we included the results of our best system also for these tasks into this report. Tables 2 and 3 show the results compared with the other two participating systems.

System	**Sub-Task A Accuracy**		
	Form.	**Inform.**	**Both**
Other Participant 1	**87.37**	**87.58**	**87.48**
Fine-Tuning$_{\text{umC}}$	86.47	83.16	84.75
Other participant 2	78.73	75.79	77.20

Table 2: PoS-tagging Accuracy for the EVALITA KIPOS 2020 benchmark for the Sub-Task A.

System	**Sub-Task B Accuracy**		
	Form.	**Inform.**	**Both**
Fine-Tuning$_{\text{umC}}$	**89.74**	**89.52**	**89.63**
Other participant 1	87.81	88.10	87.96
Other Participant 2	77.11	77.50	77.31

Table 3: PoS-tagging Accuracy for the EVALITA KIPOS 2020 benchmark for the Sub-Task B.

Again, the simple fine tuning of a BERT-derived model, namely UnBERTo, exhibits the best performance on Sub-task B. The small amount of data could probably affect the results on Sub-task A.

We collected the most frequent errors produced by the proposed system: Table 4 shows that, unexpectedly, the most frequent misclassifications involve grammatical words. The typical behaviour of the classical PoS-taggers tend to wrongly classify lexical words, namely nouns, verbs and adjectives, intermixing their classes. Apparently, on this dataset, grammatical words appear to be more complex to classify than lexical words. This be-

---

[3] https://github.com/idb-ita/GilBERTo

[4] https://github.com/musixmatchresearch/umberto

[5] https://github.com/huggingface/transformers

haviour should be investigated more appropriately by using bigger datasets and better consistency checks on the annotated data.

Formal		
#mistakes	**Gold tag**	**System tag**
19	ADP_A	ADP
16	CCONJ	ADV
12	PROPN	X
10	NOUN.LIN	X
10	ADJ	VERB
Informal		
#mistakes	**Gold tag**	**System tag**
59	PRON	SCONJ
38	ADP_A	ADP
22	ADV	CCONJ
15	NUM	DET
15	INTJ	PARA
15	CCONJ	ADV
12	NOUN	PROPN
10	VERB_PRON	VERB

Table 4: Error Analysis

4 Discussion and Conclusions

The starting idea of this work was to design the simplest DNN model for Italian PoS-tagging after the 'BERT-revolution' thanks to the recent availability of Italian BERT-derived models. Looking at the results presented in previous sections, we can certainly conclude that BERT-derived models, specifically trained on Italian texts, allow for a relevant increase in performance also when applied to spoken language by simple fine-tuning procedures. The multilingual BERT model developed by Google was not able to produce good results and should not be used when are available specific models for the studied language.

A side, and sad, consideration that emerges from this study regards the complexity of the models. All the DNN models used in this work involved very simple fine-tuning processes of some BERT-derived model. Machine learning and Deep learning changed completely the approaches to NLP solutions, but never before we were in a situation in which a single methodological approach can solve different NLP problems always establishing the state-of-the-art for that problem. Moreover, we did not apply any parameter tuning at all and fixed the early stopping criterion on 10 epochs without any optimisation. By tuning all the hyperparameters, it is reasonable we can further increase the overall performance.

Acknowledgments

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

C. Bosco, S. Ballarè, M. Cerruti, E. Goria, and C. Mauri. 2020. KIPoS@EVALITA2020: Overview of the Task on KIParla Part of Speech tagging. In *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *In Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

L. Martin, B. Muller, P.J. Ortiz Suárez, Y. Dupont, L. Romary, E. de la Clergerie, D. Seddah, and B. Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

P.J. Ortis Suárez, B. Sagot, and L. Romary. 2019. Asynchronous Pipeline for Processing Huge Corpora on Medium to Low Resource Infrastructures. In *7th Workshop on the Challenges in* *the Management of Large Corpora (CMLC-7)*, Cardiff, United Kingdom.

M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL-HLT 2018*, pages 2227–2237, New Orleans, Louisiana.

M. Polignano, P. Basile, M. de Gemmis, G. Semeraro, and V. Basile. 2019. ALBERTO: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*, Bari, Italy.

V. Sanh, L. Debut, J. Chaumond, and T. Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *Proc. 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019.*

F. Tamburini. 2020. How “BERTology” Changed the State-of-the-Art also for Italian NLP. In *Proceedings of the Seventh Italian Conference on Computational Linguistics (CLiC-it 2020)*, Bologna, Italy.

UniBA @ KIPoS: A Hybrid Approach for Part-of-Speech Tagging

Giovanni Luca Izzi
University of Bari Aldo Moro
Department of Computer Science
via E. Orabona 4, 70125 Bari, Italy
giovannilucaizzi@gmail.com

Stefano Ferilli
University of Bari Aldo Moro
Department of Computer Science
via E. Orabona 4, 70125 Bari, Italy
stefano.ferilli@uniba.it

Abstract

English. The Part of Speech tagging operation is becoming increasingly important as it represents the starting point for other high-level operations such as Speech Recognition, Machine Translation, Parsing and Information Retrieval. Although the accuracy of state-of-the-art POS-taggers reach a high level of accuracy (around 96-97%) it cannot yet be considered a solved problem because there are many variables to take into account. For example, most of these systems use lexical knowledge to assign a tag to unknown words. The task solution proposed in this work is based on a hybrid tagger, which doesn't use any prior lexical knowledge, consisting of two different types of POS-taggers used sequentially: HMM tagger and RDRPOSTagger [(Nguyen et al., 2014), (Nguyen et al., 2016)]. We trained the hybrid model using the Development set and the combination of Development and Silver sets. The results have shown an accuracy of 0,8114 and 0,8100 respectively for the main task.

Italiano. *L'operazione di Part of Speech tagging sta diventando sempre più importante in quanto rappresenta il punto di partenza per altre operazioni di alto livello come Speech Recognition, Machine Translation, Parsing e Information Retrieval. Sebbene l'accuratezza dei POS tagger allo stato dell'arte raggiunga un alto livello di accuratezza (intorno al 96-97%), esso non può ancora essere considerato un problema risolto perché ci sono molte variabili da tenere in considerazione. Ad esempio, la maggior parte di questi sistemi utilizza della conoscenza linguistica per assegnare un tag alle parole sconosciute. La soluzione proposta in questo lavoro si basa su un tagger ibrido, che non utilizza alcuna conoscenza linguistica pregressa, costituito da due diversi tipi di POS-tagger usati in sequenza: HMM tagger e RDRPOSTagger [(Nguyen et al., 2014), (Nguyen et al., 2016)]. Abbiamo addestrato il modello ibrido utilizzando il Development Set e la combinazione di Silver e Development Sets. I risultati hanno mostrato un'accuratezza pari a 0,8114 e 0,8100 rispettivamente per il task main.*

1 Introduction

Part-of-Speech tagging (which we will shorten from now on with POS-tagging), as its name implies, is the operation of tagging each word with the corresponding part of the speech (POS-tag, from now on simply tag). Usually these tags are also applied to punctuation marks, such as commas, question marks and so on. POS-tagging models are essentials to build models for higher level operations. For example, they have been used to build Parsing Trees, which are used by Named Entity Recognition and Named Entity Linking systems to extrapolate entities starting from a document or short sentences. In this regard, we can't ignore that every day through social media a large amount of textual data are produced, these data present different structures and even different variants of the same language. Therefore, in this scenario the main requirement becomes the availability of highly reliable POS-tagging models capable of adapting to the different forms that a language can exhibit. Most

---

POS-tagging algorithms can be grouped into two classes: rule-based taggers and stochastic taggers. Rule-based taggers generally involve a large database of handwritten disambiguation rules that specify, for example, that a word with the ambiguous tag is a noun rather than a verb if it is preceded by a word that has ”determiner” tag. While stochastic taggers generally solve the tagging ambiguities using a training set to calculate the probability that a given word has a given tag in a given context. There are also works that can be placed between these category like the Brill's works [(1992), (1994), (1995)]. However, most of these works include some lexical knowledge in order to tag word not learned during the training phase. It is a drawback we mustn't ignore because performance of these taggers may decrease, dramatically, for those languages where few or no lexical knowledge is available. Another important concern to think about are the necessary computational resources. For example (Mueller et al., 2013) reported that SVMTool tagger (Giménez et al., 2004) and CRFSuite tagger (Okazaki, 2007) require 2454 minutes (about 41 hours) and 9274 minutes (about 155 hours) respectively to complete the training phase on a dataset of 38727 sentences in the Czech language. The solution proposed in this work is a hybrid tagger whose philosophy is based on two simple factors: no use of lexical knowledge and no use of algorithms that require too high computational resources. For this reason we have decided to structure the hybrid tagger as a concatenation of a Hidden Markov Model (HMM) tagger and RDRPOSTagger [(Nguyen et al., 2014), (Nguyen et al., 2016)]. The proposed hybrid tagger has been evaluated during KIPoS task (Bosco et al., 2020) (KIParla Part of Speech) organized within Evalita 2020 (Basile et al., 2020), the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian, which will be held in Bologna (Italy) (December 16th – December 17th 2020).

KIPoS task consists of tagging a set of spoken sentences collected during some conversations held in Turin and Bologna. These conversations belong to different activity types: A1 (office hours), A3 (random conversation), C1 (exams), D1 (lessons) and D2 (interviews). A1, C1 and D1 are considered FORMAL conversations while A3 and D2 are considered INFORMAL conversations. Three different dataset were released: Development Set (DS), Silver Set (SS) and Test Set (TS). Each dataset is divided into Formal and Informal sentences, Table 1 show the details. The task is organized into three sub-tasks, based on the dataset used for training and testing the participants' systems:

- Main task - general: training on all given data (both DS-formal and DS-informal) and testing on all test set data (both TS-formal and TS-informal)
- Subtask A - crossFormal: training on data from DS-formal only and testing separately on data from formal register (TS-formal) and from informal register (TS-informal)
- Subtask B - crossInformal: training on data from DS-informal only and testing separately on data from formal register (TS-formal) and from informal register (TS-informal)

Dataset	**Conversation Turn**
DS-formal	1968
DS-informal	3383
DS+SS-formal	40768
DS+SS-informal	40817
TS-formal	455
TS-informal	571

Table 1: KIPoS datasets information

2 Description of the system

The proposed hybrid POS-tagger is a sequence of two POS-taggers, which don't use any prior lexical knowledge. We want to point out this sequence isn't fixed, anyone could create his one POS-tagger and replace one of the POS-tagger already used by the sequence. There is only one constraint that must be satisfied if you want to create a new tagger which will be the second tagger of the sequence, that is: the POS-tagger must be able to perform the learning starting from data tagged by the first tagger and perform the tagging operation on already tagged sentences. The first POS-tagger after receiving an untagged sentence (raw sentence) as input uses the information acquired during the training phase in order to transform this sentence into a tagged sentence, where each token is associated with a tag according to the following structure token/tag. This first version of the tagged sentence could contain errors that will be corrected by the subsequent POS-tagger. The sequence implemented consist of an HMM tagger and a rule-based tagger called RDRPOSTagger. We believe these two POS-taggers can complement each other. Furthermore, RDRPOSTagger, unlike the other rules systems, is very light and allows to carry out the learning phase even if there are limited computational resources. Since the proposed solution doesn't use lexical knowledge, it allows us to have a model applicable to any language with homogeneous performance. Below we proceed with a brief description of the two POS-taggers.

2.1 HMM Tagger

In relation to POS-tagging there are many things to keep in mind when building an HMM tagger:

1. How to handle words not seen during the training phase?
2. How many previous tags should we consider?
3. How to handle the probability $P(t_i|t_{i-1})$ of a tag sequence not observed during the training phase?

A suffix-based approach is used in the HMM tagger designed to manage unknown words. Indeed, the suffixes are highly specific for each language and also they help to deduce the category to which the unknown word belongs. For example, in English the words ending in "-ing" may be gerunds or nouns. So the best strategy is to extract suffixes for each POS tag learned during the training phase. It is a fairly natural solution because for an HMM tagger it is necessary to keep, for each word, all the tags to which it can be associated and the number of times it has been associated with each single tag. To this purpose we keep, for each tag, a list of words where each word has been observed, during the training, associated to this tag. Finally, we extract a list of suffixes for each tag using the list of words mentioned before and a suffixes extraction algorithm. We developed the suffixes extraction algorithm using the Apriori Algorithm. The algorithm works as follow: Given a set of words *W*, in order to extract the candidate suffixes, first each word *w* is inverted, that is the letters that form the word are conversely listed starting from the last one up to the first letter. After doing this the set of inverted words is used as input to obtain a suffixes list containing lists of candidate suffixes of increasing size. Finally, the obtained suffixes list will be further processed to obtain a tree representation.The obtained tree will be cut considering, at every node, three different thresholds: the support of this node, the number of distinct words which contain the suffix represented by this node, the percentage of *W* words which contain the current suffix and the suffix from which it is derived.

Regarding the second question, in the planned HMM tagger it was decided to consider the trigrams, that is for each tag the two previous tags are considered. Then the transition probability becomes: $P(t_i|t_{i-1}, t_{i-2})$. Considering trigram-based transition probabilities is the most commonly used method in state-of-the-art stochastic POS-taggers. At this point also the last question changes, since we are now interested in solving problems deriving from sequences of trigrams not observed during the training phase. The approach used to manage unknown tag sequences is a smoothing technique called linear interpolation described by the following formula:

$$P(t_i|t_{i-1}, t_{i-2}) = \lambda_3 P_{MLE}(t_i|t_{i-1}, t_{i-2}) + \lambda_2 P_{MLE}(t_i|t_{i-1}) + \lambda_1 P_{MLE}(t_i)$$

The main requirement of this formula is $\lambda_1 + \lambda_2 + \lambda_3 = 1$, thus ensuring that *P* is a probability distribution. The λ values are learned using the deleted interpolation (Jelinek et al., 1980), where we subsequently delete each trigram from the training dataset and choose the λ in order to maximize the probability of the rest of the dataset.

2.2 RDRPOSTagger

RDRPOSTagger [(Nguyen et al., 2014), (Nguyen et al., 2016)] is a rule-based tagger, this approach is also called transformation-based error-driven, able to automatically structure the rules in a particular tree structure called Single Classification Ripple Down Rules (SCRDR) [(Compton and Jansen, 1990), (Richards, 2009), (Nguyen et al., 2015)]. A SCRDR tree is a binary tree with two distinct

Training dataset	Formal (F)	num KF	KF	num UF	UF
DS	0.8236	2940	0.9180	638	0.3887
DS-formal	0.7954	2769	0.9176	809	0.3770
DS-informal	0.7778	2805	0.8709	773	0.4398
DS+SS	0.8085	3429	0.8293	149	0.3288
DS+SS-formal	0.8113	3406	0.8352	172	0.3372
DS+SS-informal	0.7758	3190	0.8128	388	0.4716

Table 2: Results obtained for Gold Test corrected Formal sentences

types of edges. These edges are usually called: except and if-not. Each tree node corresponds to a rule. Each rule has the form: if $\alpha \rightarrow \beta$, where α is the condition of the rule and β is the conclusion. Cases in a SCRDR tree are evaluated by passing a case to the root of the tree. In each node of the tree, if the condition of the rule in a node η is satisfied by the input case (so the node η is activated), the case is passed to the node except child of the node η using the except edge if it exists. Otherwise, the case is passed to the if-not node child of the node η. The conclusion of this process is given by the last activated node. A new node containing a new exception rule is added to a SCRDR tree when the evaluation process returns a wrong conclusion. The new node is connected to the last node in the evaluation path of a given case through an except edge if the last node of the path is the activated node, otherwise, it is connected to it with an if-not edge. To ensure that a conclusion is always provided, the root node (called the default node) generally contains a trivial condition that is always satisfied. The rule in the default node, called the default rule, is the only rule that is not the exception rule of any other rule. We decided to use RDRPOSTagger as a second tagger of our sequence because of its own abilities: It is a lightweight rule tagger; rules are learned in a controlled context, in this way they can't influence one another. Therefore, our hybrid model is very fast during training and tagging phase.

3 Results

We evaluated the performance of the hybrid tagger just described with just a single run as we did for the competition. For the competition we used only the DS dataset for the learning phase, but here we investigated experimental results using also the SS dataset. More precisely, we used Random Split to divide the dataset into 90% training set and 10% validation set, the latter has been used to learn the rules through RDRPOSTagger. We decided to use default configuration for RDRPOSTagger and for our suffixes extraction algorithm. More precisely we set the three different thresholds described before equals to 10, 3 and 0.4 respectively. Table 2 and Table 3 show the results obtained for Formal and Informal corrected Gold Test dataset, provided by the authors after the evaluation, which contains some improvements compared to the test dataset used during the competition. In these two tables we present the results listing: overall accuracy, number of known tokens, known tokens accuracy, number of unknown tokens and unknown tokens accuracy.

4 Discussion

The test dataset provided for the competition contains spoken sentences based on conversation turns, which make the competition quite challenging because these sentences have an irregular structure with misspelled words. Our evaluation will also have to take into account the number of conversation turns contained in the training dataset, fewer conversation turns in the overall dataset will imply fewer conversation turns in the validation set and therefore fewer rules learned by RDRPOSTagger. In fact, using only the DS it is able to learn about 5-6 rules while on the combination of DS-SS the rules learned are about 40. Moreover, these rules depend on the contexts contained in the validation set which, given the small number of data, can be very different from those encountered during the testing phase. Abstracting from the number of known words, which increase using the combination of the two datasets, the results show that the accuracy on these words remains around 90% when learning is performed using the Development Set (DS). While using the combination of Silver (SS) and Development Sets this percentage is closer to 80% and it is surprising if we consider that the DS contains far less data.

Training dataset	Informal (I)	num KI	KI	num UI	UI
DS	0.7992	3213	0.8954	587	0.2725
DS-formal	0.7631	3026	0.8853	774	0.2855
DS-informal	0.7802	3128	0.8772	672	0.3288
DS+SS	0.8425	3602	0.8425	198	0.2474
DS+SS-formal	0.7821	3436	0.8378	364	0.2554
DS+SS-informal	0.8050	3555	0.8447	245	0.2285

Table 3: Results obtained for Gold Test corrected Informal sentences

Such a difference can be explained if we consider that SS is an automatically tagged dataset and it isn't manually revised so it can be source of errors. Only for the subB task the accuracy on unknown words, considering a formal context, exceed, even if slightly, the results obtained using the DS. The results for the unknown words are quite low, these errors in turn propagate other errors on the known words. The errors concern words that are impossible to recognize without the use of lexical knowledge such as names, they are also written with a lowercase initial, date and numbers written in textual format. Other errors are related to polysemy words such as the word ”prego” used as both INTJ and VERB. However, in this case the word has been observed during training more often as VERB than INTJ and the particular contexts of the test sentences and those learned during training don't help us to tend towards the correct INTJ tag.

5 Conclusion

The KIPoS competition was the perfect situation to evaluate the solution we proposed because there are formal and informal sentences and they don't have a regular structure. In this work we presented a hybrid POS-tagger that tries to combine the advantages of a stochastic model and a rule model without using previous lexical knowledge while keeping learning and tagging times at a level suitable for real applications. Results showed that the percentage of known words tagged correctly is about 90% while for the unknown words the percentages vary in the range [27% - 44%], where the extremes of this interval represent the worst and best case respectively. The greatest difficulties occurred for unknown words in the informal context. The competition allowed us to get useful insights regarding which parts of the system need to be improved. For example, our suffixes extraction algorithm, which is still in a beta version. Future work directions will surely focus on improving the suffixes extraction algorithm and on the possible combination of suffixes and prefixes to identify the unknown words. Every future directions will always investigate solutions which will not require lexical knowledge. Therefore, they will be applicable to any language.

References

Bosco, Cristina and Ballarè, Silvia and Cerruti, Massimo and Goria, Eugenio and Mauri, Caterina. 2020. *KIPoS@EVALITA2020: Overview of the Task on KIParla Part of Speech tagging*. Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020).

Basile, Valerio and Croce, Danilo and Di Maro, Maria, and Passaro, Lucia C. 2020. *EVALITA 2020: Overview of the 7th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*. Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020).

Eric Brill. 1992. *A simple rule-based part of speech tagger. In Proceedings of the third conference on Applied natural language processing (ANLC '92)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 152-155. DOI: `https://doi.org/10.3115/974499.974526`

Eric Brill. 1994. *Some advances in transformation-based Part of Speech tagging*. Proceedings of the 12th National Conference on Artificial Intelligence (AAAI) vol. 1, pages 722–727.

Eric Brill. 1995. *Unsupervised learning of disambiguation rules for Part of Speech tagging*. In Natural Language Processing Using Very Large Corpora Workshop, pages 1–13. Kluwer.

T. Mueller, H. Schmid, and H. Schütze. 2013. *Efficient Higher-Order CRFs for Morphological Tagging*. In Proceedings of the 2013 Conference on Empirical Methods on Natural Language Processing, pages 322– 332.

J. Giménez, L. Màrquez, and L. Marquez. 2004. *SVMTool: A General POS Tagger Generator Based on Support Vector Machines*. In Proceedings of the 4th International Conference on Language Resources and Evaluation, pages 43–46.

N. Okazaki. 2007. *CRFsuite: A Fast Implementation of Conditional Random Fields (CRFs)*. `http://www.chokkan.org/software/crfsuite/`

Nguyen, Dat Quoc and Nguyen, Dai and Pham, Dang and Pham, Son. 2014. *RDRPOSTagger: A Ripple Down Rules-based Part-Of-Speech Tagger*. 17-20. 10.3115/v1/E14-2005.

Nguyen, Dat Quoc and Nguyen, Dai and Pham, Dang and Pham, Son. 2016. *A Robust Transformation-Based Learning Approach Using Ripple Down Rules for Part-of-Speech Tagging*. AI Communications. 29. 409-422. 10.3233/AIC-150698.

Jelinek, F. and Mercer, R. L. 1980. *Interpolated estimation of Markov source parameters from sparse data*. In Gelsema, E. S. and Kanal, L. N. (Eds.), Proceedings, Workshop on Pattern Recognition in Practice, pp. 381–397. North Holland.

P. Compton and R. Jansen. 1990. *A Philosophical Basis for Knowledge Acquisition*. Knowledge Acquisition, 2(3): 241–257.

D. Richards. 2009. *Two Decades of Ripple Down Rules Research*. Knowledge Engineering Review, 24(2):159–184.

D. Q. Nguyen, D. Q. Nguyen, and S. B. Pham. 2015. *Ripple Down Rules for Question Answering*. Semantic Web journal, to appear, 2015. URL: `http://www.semantic-web-journal.net/`

KLUMSy@KIPoS: Experiments on Part-of-Speech Tagging of Spoken Italian

Thomas Proisl
Computational Corpus Linguistics Group
Friedrich-Alexander-Universität Erlangen-Nürnberg
Bismarckstr. 6
91054 Erlangen, Germany
`thomas.proisl@fau.de`

Gabriella Lapesa
Institute for Natural Language Processing
Universität Stuttgart
Pfaffenwaldring 5 b
70569 Stuttgart, Germany
`gabriella.lapesa@ims.uni-stuttgart.de`

Abstract

In this paper, we describe experiments on part-of-speech tagging of spoken Italian that we conducted in the context of the EVALITA 2020 KIPoS shared task (Bosco et al., 2020). Our submission to the shared task is based on SoMeWeTa (Proisl, 2018), a tagger which supports domain adaptation and is designed to flexibly incorporate external resources. We document our approach and discuss our results in the shared task along with a statistical analysis of the factors which impact performance the most. Additionally, we report on a set of additional experiments involving the combination of neural language models with unsupervised HMMs, and compare its performance to that of our system.

1 Introduction

Part-of-speech taggers trained on standard newspaper texts usually perform relatively poorly on spoken language or on written communication that is “conceptually oral”, e. g. tweets or chat messages. The challenges of spoken language include non-standard lexis, e. g. the use of colloquial and dialectal forms, and non-standard syntax, e. g. false starts, repetitions, incomplete sentences and the use of fillers. To make things worse, the amount of training data available for spoken language – or non-standard varieties in general – is usually several orders of magnitude smaller than for the usual newspaper corpora. One strategy for coping with this is to incorporate additional resources, e. g. lexica or distributional information obtained from large amounts of unannotated text. Another strategy is to do domain adaptation, i. e. to leverage existing written standard corpora to pretrain an out-of-domain tagger model and to then adapt that model to the target domain using a small amount of in-domain data.

We experiment with these ideas in the context of the EVALITA 2020 shared task on part-of-speech tagging of spoken Italian (Bosco et al., 2020; Basile et al., 2020). The data of the shared task have been drawn from the KIParla corpus (Mauri et al., 2019) and consist of the manually annotated training and test datasets and a silver dataset that has been automatically tagged by the task organizers using a UDPipe[1] model trained on all Italian treebanks in the Universal Dependencies (UD) project.[2] While the silver dataset is annotated with the standard UD tagset (as are the corpora on which the tagger has been trained), the training and test sets use an extended version where tags can optionally be assigned one of two subcategories, .DIA for dialectal forms and .LIN for foreign words.

2 Additional resources

2.1 Corpora

We use a collection of plain text corpora to compute Brown clusters (Brown et al., 1992) that the tagger can use as additional resource.

Ideally, we would use large amounts of transcribed speech for the present task. Since there is no such dataset, we try to use corpora that come close. The closest to authentic speech is scripted speech, therefore we use the Italian movie subtitles from the OpenSubtitles corpus (Lison and Tiedemann, 2016).[3] Computer-mediated communication, e. g. in social media, sometimes exhibits features that are typical of spoken language use. Therefore, we also use a collection of roughly 11.7 million Italian tweets and

---

[1] `http://ufal.mff.cuni.cz/udpipe/1`

[2] `https://universaldependencies.org/`

[3] `http://opus.nlpl.eu/OpenSubtitles-v2018.php`

ca. 2.7 million Reddit posts (submissions and comments) from the years 2011–2018. We extracted the Reddit posts from Jason Baumgartner's collection of Reddit submissions and comments[4] using the processing pipeline by Blombach et al. (2020). Additionally, we also include all Italian corpora from the Universal Dependencies project and, to further increase the amount of data, a number of web corpora: The PAISÀ corpus of Italian texts from the web (Lyding et al., 2014),[5] the text of the Italian Wikimedia dumps,[6] i. e. Wiki(pedia|books|news|versity|voyage), as extracted by Wikipedia Extractor,[7] and the Italian subset of OSCAR, a huge multilingual Common Crawl corpus (Ortiz Suárez et al., 2019).[8]

We tokenize and sentence split all corpora using UDPipe trained on the union of all Italian UD corpora. We also remove all duplicate sentences. The sizes of the resulting corpora are given in Table 1. As final preprocessing steps, we lowercase all words and normalize numbers, user mentions, email addresses and URLs. Finally, we use the implementation by Liang (2005)[9] to compute 1,000 Brown clusters with a minimum frequency 5.

corpus	complete	deduplicated
oscar	–	13,787,307,218
opensubtitles	795,250,711	378,348,061
paisa	282,631,297	258,679,965
reddit	112,735,958	105,274,620
tweets	152,496,728	148,031,020
ud	672,929	615,057
wiki	578,425,024	560,863,691
wikibooks	12,106,499	11,825,870
wikinews	2,744,317	2,583,135
wikiversity	5,766,859	5,365,924
wikivoyage	3,911,881	3,825,872

Table 1: Sizes of the additional corpora in tokens. OSCAR is already deduplicated on the line level.

2.2 Morphological lexicon

We incorporate linguistic knowledge in the form of Morph-it! (Zanchetta and Baroni, 2005),[10] a morphological lexicon for Italian that contains morphological analyses of roughly 505,000 word forms that correspond to about 35,000 lemmata. In its analyses, Morph-it! distinguishes between derivational features and inflectional features. In total, there are 664 unique feature combinations. We simplify the analyses by stripping away all inflectional features and some of the derivational features, i. e. gender (for articles, nouns and pronouns) and person and number (for pronouns). This results in 39 coarse-grained categories that correspond to major word classes, with some finer distinctions for determiners and pronouns.

3 System description

For our submission to the shared task we use SoMeWeTa (Proisl, 2018), a tagger that is based on the averaged structured perceptron, supports domain adaptation and can incorporate external resources such as Brown clusters and lexica.[11] Its ability to make use of existing linguistic resources allows the tagger to achieve competitive results even with relatively small amounts of in-domain training data, which is particularly useful for non-standard varieties or under-resourced languages (Kabashi and Proisl, 2018; Proisl et al., 2019).

We participate in all three substasks: The main subtask where we use all the available silver and training data, subtask A where we only use the data from the formal register, and subtask B where we only use the informal data. The training scheme is the same for all three subtasks. First, we train preliminary models on the silver data provided by task organizers. Keep in mind that the silver dataset has been automatically tagged. Therefore, it is annotated with the standard version of the UD tagset and not with the extended one that is used in the shared task; in addition, there will be a certain amount of tagging errors in the data. Nevertheless, the dataset provides the tagger with (imperfect) domain-specific background knowledge. In the next step, we adapt the silver models to the union of the Italian UD treebanks, i. e. to high-quality but out-of-domain data. In the final step, we adapt the models to spoken Italian using the manually annotated training data. In every step we train for 12 iterations using a search beam size of 10 and provide the tagger with the Brown clusters and the Morph-it!-based lexicon (Section 2).

---

[4] `https://files.pushshift.io/reddit/`

[5] `http://www.corpusitaliano.it/`

[6] `https://dumps.wikimedia.org/`

[7] `http://medialab.di.unipi.it/wiki/Wikipedia_Extractor`

[8] `https://oscar-corpus.com/`

[9] `https://github.com/percyliang/brown-cluster/`

[10] `https://docs.sslmit.unibo.it/doku.php?id=resources:morph-it`

[11] `https://github.com/tsproisl/SoMeWeTa`

4 Evaluation

4.1 Data preparation and evaluation results

The silver data, training data and the data from the UD treebanks follow UD tokenization guidelines, i. e. contractions such as *parlarmi* (*parlar+mi*) ‘to talk+to me’ or *della* (*di+la*) ‘of+the’ are split into their constituents for annotation. This is not the case for the test data where contractions have to be assigned a joint tag, e. g. VERB_PRON or ADP_A. Therefore, we run the test data through the UDPipe tokenizer from Section 2.1, tag the resulting tokens and merge the tags for all tokens that have been split. Table 2 shows the results on the two testsets.[12] On the main task, SoMeWeTa performs reasonably well, only 1–1.4 points worse than the fine-tuned UmBERTo model by Tamburini (2020). On subtasks A and B, it even outperforms that system by a considerable margin.

task	system	formal	informal
main	corrected	92.12	90.11
	gold tokens	92.31	90.66
	Tamburini (2020)	**93.49**	**91.13**
subA	corrected	91.92	89.45
	gold tokens	**92.12**	**89.97**
	Tamburini (2020)	86.47	83.16
subB	corrected	92.37	89.97
	gold tokens	**92.54**	**90.53**
	Tamburini (2020)	89.74	89.52

Table 2: Accuracy scores for our submissions in two variants: (i) With ADP_DET corrected to ADP_A and (ii) based on the true token boundaries instead of on UDPipe tokens.

4.2 Mining tagging accuracy

To get a better insight into the impact of the different experimental variables involved in this study, we carried out feature ablation experiments which targeted the different components of our system, namely the different combinations of training and test data (formal vs. informal) and the different additional resources described in section 2 (use of Brown clusters, Morph-it!, silver data, and UD corpora). We then carried out a linear regression analysis with *tagging accuracy as a dependent variable* and the different *experimental parameters as independent variables (predictors)*. We follow the methodology outlined in Lapesa and Evert (2014) and quantify the impact of a specific predictor (e. g. the use of Brown clusters) as the amount of variance in the dependent variable (tagging accuracy) it accounts for. We considered the following experimental parameters as predictors.

- **setup**: Training/test setup; this predictor encodes the combination of training/test data and has the following values: *all_formal* (i. e. trained on the full set, tested on formal), *all_informal*, *formal_formal*, *formal_informal*, *informal_formal*, *informal_informal*
- **silver**: Use of silver data during training (*yes*, *no*)
- **ud**: Use of UD corpora during training (*yes*, *no*)
- **morph**: Use of Morph-it! (*yes*, *no*)
- **brown**: Use of Brown clusters (*yes*, *no*)

We tested all the possible configurations, i. e. all the combinations of the parameters described above, and, to account for random effects during training, ran each configuration 10 times. This resulted in 960 experimental runs, each corresponding to a single datapoint in our regression analysis. Given that it is reasonable to assume that specific parameter values will influence the performance of other parameters (e. g., use of Morph-it! could boost performance but only if larger corpora are employed), we also test all the 2-way interactions. As a sanity check, we also introduce the number of an experimental run as a predictor (1 to 10, as a categorical variable), in the hope, obviously, of finding no effect for it. Summing up, our regression equation looks as follows:

$$\texttt{accuracy} \sim \texttt{(setup + silver + ud + morph + brown + run)}\wedge\texttt{2}^{13}$$

Unsurprisingly, our model achieves an excellent fit to the data, quantified in an Adjusted R-squared of 95.2%. Table 3 lists all significant predictors and interactions, along with their explained variance. Explained variance quantifies the portion of the total R-squared that a specific parameter (or interaction) is responsible for and can be straightforwardly interpreted as the impact that the manipulation of a specific parameter has on the accuracy of our tagger. Reassuringly, we found no effect of experimental run. All other predictors, and

---

[12] Unfortunately, when preparing our submission, we did not notice that contractions of prepositions (ADP) and determiners (DET) have to be tagged as ADP_A. As a consequence, we mis-tagged all these contractions as ADP_DET. For reference, here are the evaluation results of our faulty submission on the formal/informal test sets: main 87.56/88.24, subA 87.37/87.58, subB 87.81/88.11.

[13] Given that we ran the regression analysis in R, and the equation follows the R syntax in which “∧2” denotes all pairwise interactions of the predictors between parentheses.

Predictor	Explained variance
setup	42.06 ***
silver	8.62 ***
ud	12.63 ***
brown	8.76 ***
morph	7.17 ***
setup:silver	1.21 ***
setup:ud	1.08 ***
setup:brown	0.42 ***
setup:morph	0.50 ***
silver:ud	6.00 ***
silver:brown	0.39 ***
silver:morph	1.98 ***
ud:brown	0.03 *
ud:morph	2.48 ***
brown:morph	2.44 ***

Table 3: Regression on tagging a accuracy: predictors and explained variance. Adj. R-squared: 95.2%. Sign. thresholds: ***: 0.001; *: 0.05.

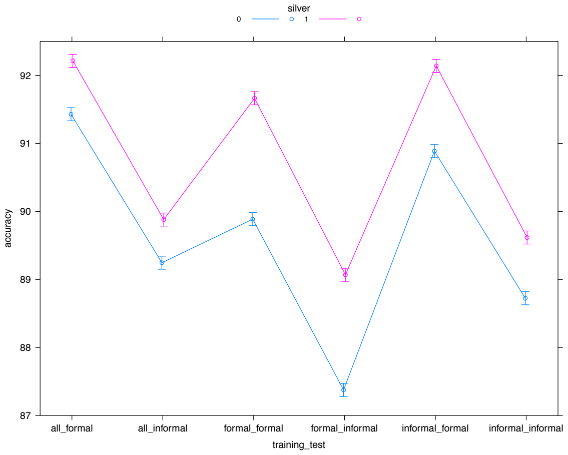


Figure 1: Interaction: setup and silver data

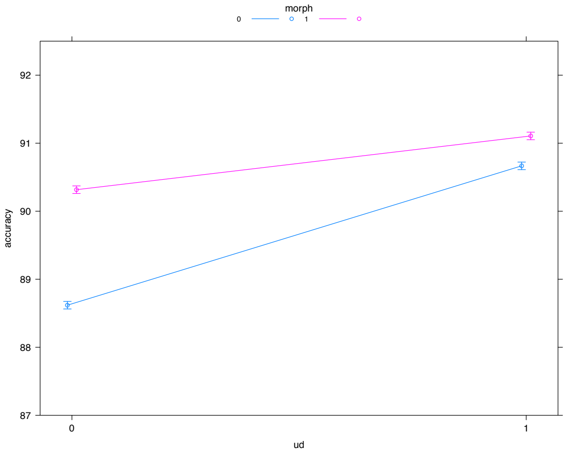


Figure 2: Interaction: UD corpora and Morph-it!

all the corresponding interactions, turned out to be highly significant (with one minor exception). The biggest role is played by the *setup* variable, which alone accounts for 42.06%. Using UD corpora in the training has also a strong impact, with a strong interaction involving the use of silver data (6.00% R-squared). Further strong interactions are found between brown and morph, and brown and UD – probably suggesting that introducing a 3-way interaction would be appropriate here. Given the increased complexity, however, this extension is left for future work.

Now that we have established which parameters or interactions have the strongest impact on model performance, it is time to ask which parameter values ensure the best performance. In our case, given that the system can be assembled incrementally (adding external resources and training data to a basic configuration), asking what the best parameter values are amounts to determining if, for example, the addition of Brown clusters improves performance or is detrimental. Note that the significance of the *brown* predictor in the regression analysis already tells us that the predictor affects performance, ruling out the possibility that it has no impact at all. To visualize the effects in the linear model, we follow Lapesa and Evert (2014) and employ effect displays which show the partial effect of one or two parameters by marginalizing over all other parameters. Unlike coefficient estimates, they allow an intuitive interpretation of the effect sizes of categorical variables irrespective of the dummy coding scheme used.

Let us start with the strongest predictor, *setup*, in its strongest interaction, the one with *silver*. Figure 1 displays the predicted accuracies resulting from the different parameter combinations of the two predictors. Note that, given the excellent fit of the regression model, we can assume predicted accuracy to be a reliable estimate of actual accuracy. Also, note that while we are visualizing the predicted accuracy of a 2-way interaction, we are actually displaying the effect of the individual terms (*setup* and *silver*) and of the interaction (*setup:silver*) jointly. We observe that, unsurprisingly, independently of the use of silver data, training on the whole dataset ensures the best performance on both the formal and informal test sets. The use of silver data (pink line) improves performance, but with differences in the different training/test setups. Interestingly, using the silver data makes the performance gap between the models trained on the whole dataset and those trained on just the informal dataset negligible. Surprisingly, we observe that the best performance is predicted for the formal test set when the informal set is used. Further experiments on the complementarity of the two subtasks are needed to further clarify this contradiction.

Figure 2 displays the interaction between the use of UD corpora and the integration of Morph-it! in SoMeWeTa. Note that the performance gaps are smaller here than in the previous interaction: this is no surprise, given the smaller explanatory power (explained variance) of the parameters and interactions involved. Morph-it! produces substantial improvements, but again, to a lesser extent if UD corpora are employed: this could either be due to a lower coverage of Morph-it! on the UD corpora, or to the boost in model robustness produced by the introduction of a larger training set. The steep slope of the blue line wrt. the pink one suggests that the presence of a morphological lexicon like Morph-it! can compensate the lack of training data. Let us conclude with the third strongest interaction, the one between the use of Brown clusters and the use of Morph-it!, not shown here for space constraints. It is strikingly similar to the one in Figure 2: Morph-it! improves performance overall, and the steeper improvement in absence of the Brown clusters suggests that the quality of the information encoded in Morph-it! can compensate for the lack of external resources.

In sum, our analysis supports the starting assumption that in a low-resource setting like the one of KIPoS, integrating additional, focussed resources always supports performance.

5 Additional experiments: RoBERTa with unsupervised HMM

Fine-tuned neural language models have been extremely successful in all areas of natural language processing (NLP). Not only can language models trained on huge amounts of plain text be fine-tuned to all NLP tasks, they have also been shown to learn certain linguistic abstractions (Tenney et al., 2019). At least that seems to be the case for English. Languages that are typologically different from English are both more difficult to model with current architectures (Mielke et al., 2019) and seem to be more challenging when it comes to learning linguistic abstractions (Ravfogel et al., 2018). In the experiment described in this section, we extend a state-of-the-art language model architecture to explicitly model part-of-speech information. To this end, we combine a RoBERTa language model (Liu et al., 2019) with an unsupervised neural hidden Markov model (HMM) for part-of-speech induction.

The architecture of the unsupervised HMM follows the LSTM-based variant described by Tran et al. (2016). We directly use the negative logarithm of the observation likelihood determined by the backward algorithm as additional loss for the language model. The embeddings of the best tag sequence (determined using the Viterbi algorithm) are added to the word embeddings before feeding them into the language model. Due to time and resource constraints, we opt for a small to medium-sized model[14] with a total of 45.5 million trainable parameters and train it on 1.9 billion tokens of text (the corpora described in Section 2.1 excluding OSCAR). The model variant with the unsupervised HMM totals 48.7 million trainable parameters. We pre-train and fine-tune both models with the same set of parameters.[15]

The results are summarized in Table 4. Due to the small model size and relatively little training data, the performance of both models is below SoMeWeTa's. (Keep in mind that state-of-the-art language models for Italian like UmBERTo or GilBERTo[16] are based on the same RoBERTa architecture but feature roughly three times as many parameters and have been trained on an order of magnitude more data.) However, the experiment is successful insofar as explicitly modelling part-of-speech information using an unsupervised HMM gives modest gains on both test sets. On the union of the two test sets, this corresponds to a statistically significant improvement from 89.84 to 90.42 (McNemar mid-p test: $p = 0.0133$).

model	formal	informal
RoBERTa	91.28	88.46
RoBERTa+HMM	**91.84**	**89.05**

Table 4: Results for RoBERTa and for RoBERTa with additional unsupervised HMM

---

[14] We use the RoBERTa implementation from the transformers library (`https://github.com/huggingface/transformers`) with 6 hidden layers, 8 attention heads, a hidden size of 512 and an intermediate size of 2048.

[15] Pretraining for 100,000 steps with a batch size of 500, peak learning rate of 5×10^{-4}, 6,000 warm-up steps and dropout set to 0.1. Fine-tuning to the KIPoS task using the entire training data for 4 epochs with a batch size of 32 and learning rate of 3×10^{-4}

[16] `https://github.com/musixmatchresearch/umberto`, `https://github.com/idb-ita/GilBERTo`

6 Conclusion

This paper started out with the assumption that in low-resource scenarios like the KIPoS shared task the integration of additional resources such as lexica (in our case, Morph-it!) and distributional information from larger corpora (in our case, the Brown clusters) can compensate for the lack of large amounts of training data. Moreover, our strategy also built on the assumption that in a low-resource scenario domain adaptation would be a winning strategy, as it would enable us to exploit larger training sets for written language (out of domain), and then fine-tune the tagger on the spoken language (in domain). The results of our experiments, and the insights gathered from the statistical analysis of our results indicate that both assumptions hold to be true, as far as our contribution to the KIPoS shared task is concerned. In subtasks A and B, where only half the amount of training data was available, this strategy even outperformed a fine-tuned state-of-the-art neural language model. Further work is needed to assess the complementarity of the error profiles of different configurations, taking into the picture also the neural architectures evaluated in Section 4.

References

Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro. 2020. Evalita 2020: Overview of the 7th evaluation campaign of natural language processing and speech tools for italian. In Valerio Basile, Danilo Croce, Maria Di Maro, and Lucia C. Passaro, editors, *Proceedings of Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*, Online. CEUR.org.

Andreas Blombach, Natalie Dykes, Philipp Heinrich, Besim Kabashi, and Thomas Proisl. 2020. A corpus of German Reddit exchanges (GeRedE). In *Proc. of LREC*, pages 6310–6316, Marseille. ELRA.

Cristina Bosco, Silvia Ballarè, Massimo Cerruti, Eugenio Goria, and Caterina Mauri. 2020. KIPoS@EVALITA2020: Overview of the task on KIParla part of speech tagging. In *Proc. of EVALITA*. CEUR.org.

Peter F. Brown, Vincent J. Della Pietra, Peter V. de Souza, Jennifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Besim Kabashi and Thomas Proisl. 2018. Albanian part-of-speech tagging: Gold standard and evaluation. In *Proc. of LREC*, pages 2593–2599, Miyazaki. ELRA.

Gabriella Lapesa and Stefan Evert. 2014. A large scale evaluation of distributional semantic models: Parameters, interactions and model selection. *TACL*, 2:531–546.

Percy Liang. 2005. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proc. of LREC*, pages 923–929, Portorož. ELRA.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach.

Verena Lyding, Egon Stemle, Claudia Borghetti, Marco Brunello, Sara Castagnoli, Felice Dell'Orletta, Henrik Dittmann, Alessandro Lenci, and Vito Pirrelli. 2014. The PAISÀ corpus of Italian web texts. In *Proc. of WaC-9*, pages 36–43, Gothenburg. ACL.

Caterina Mauri, Silvia Ballarè, Eugenio Goria, Massimo Cerruti, and Francesco Suriano. 2019. KIParla corpus: A new resource for spoken Italian. In *Proc. of CLiC-it*, Bari.

Sabrina J. Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. 2019. What kind of language is hard to language-model? In *Proc. of ACL*, pages 4975–4989, Florence. ACL.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. In *Proc. of CMLC-7*, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

Thomas Proisl, Peter Uhrig, Philipp Heinrich, Andreas Blombach, Sefora Mammerella, Natalie Dykes, and Besim Kabashi. 2019. The_Illiterati: Part-of-speech tagging for Magahi and Bhojpuri without even knowing the alphabet. In *Proc. of NSURL*, Trento.

Thomas Proisl. 2018. SoMeWeTa: A part-of-speech tagger for German social media and web texts. In *Proc. of LREC*, pages 665–670, Miyazaki. ELRA.

Shauli Ravfogel, Yoav Goldberg, and Francis Tyers. 2018. Can LSTM learn to capture agreement? The case of Basque. In *Proc. of BlackboxNLP*, pages 98–107, Brussels, November. ACL.

Fabio Tamburini. 2020. UniBO@KIPoS: Fine-tuning the Italian "BERTology" for the EVALITA 2020 KIPOS task. In *Proc. of EVALITA*. CEUR.org.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proc. of ACL*, pages 4593–4601, Florence. ACL.

Ke M. Tran, Yonatan Bisk, Ashish Vaswani, Daniel Marcu, and Kevin Knight. 2016. Unsupervised neural hidden Markov models. In *Proc. of the Workshop on Structured Prediction for NLP*, pages 63–71, Austin, TX. ACL.

Eros Zanchetta and Marco Baroni. 2005. Morph-it! A free corpus-based morphological resource for the Italian language. In *Proc. of Corpus Linguistics*, Birmingham.