Deep Learning for Automatic Image Captioning in Poor Training Conditions

Caterina Masotti*
Università di Roma, Tor Vergata

Danilo Croce**
Università di Roma, Tor Vergata

Roberto Basili†
Università di Roma, Tor Vergata

Recent advancements in Deep Learning have proved that an architecture that combines Convolutional Neural Networks and Recurrent Neural Networks enables the definition of very effective methods for the automatic captioning of images. The disadvantage that comes with this straightforward result is that this approach requires the existence of large-scale corpora, which are not available for many languages.

This paper introduces a simple methodology to automatically acquire a large-scale corpus of 600 thousand image/sentences pairs in Italian. At the best of our knowledge, this corpus has been used to train one of the first neural captioning systems for the same language. The experimental evaluation over a subset of validated image/captions pairs suggests that the achieved results are comparable with the English counterpart, despite a reduced amount of training examples.

1. Introduction

The image captioning task consists of generating a brief description in natural language of a given image that is able to capture the depicted objects and the relations between them, as discussed in (Bernardi et al. 2016). More precisely, given an image I as input, an *image captioner* should be able to generate a well-formed sentence $S(I) = (s_1, ..., s_m)$, where every s_i is a word from a vocabulary $V = \{w_1, ..., w_n\}$ in a given natural language. Some examples of images and corresponding captions are reported in Figure 1. This task is rather complex as it involves non-trivial subtasks to solve, such as object detection, mapping visual features to text and generating text sequences.

Recently, neural methods based on deep neural networks have reached impressive state-of-the-art results in solving this task (Karpathy and Li 2015; Mao et al. 2014; Xu et al. 2015). One of the most successful architectures implements the so-called *encoder-decoder* end-to-end structure (Goldberg 2016).
Differently by most of the existing encoder-decoder structures, in (Vinyals et al. 2014) the encoding of the input image is performed by a convolutional neural network which transforms it in a dense feature vector; then, this vector is "translated" to a descriptive sentence by a Long Short-Term Memory (LSTM) architecture, which takes the vector

* Dept. of Enterprise Engineering - Via del Politecnico 1, 00133 Rome, Italy.
E-mail: caterinamasotti@yahoo.it
** Dept. of Enterprise Engineering - Via del Politecnico 1, 00133 Rome, Italy.
E-mail: croce@info.uniroma2.it
† Dept. of Enterprise Engineering - Via del Politecnico 1, 00133 Rome, Italy.
E-mail: basili@info.uniroma2.it

(a) English: *A yellow school bus parked in a handicap spot*, Italian: *Uno scuolabus giallo parcheggiato in un posto per disabili.*

(b) English: *A cowboy rides a bucking horse at a rodeo*, Italian: *Un cowboy cavalca un cavallo da corsa a un rodeo.*

(c) English: *The workers are trying to pry up the damaged traffic light*, Italian: *I lavoratori stanno cercando di tirare su il semaforo danneggiato.*

Figure 1
Three images from the MSCOCO dataset, along with two human-validated descriptions, one for English and one for Italian.

as the first input and generates a textual sequence starting from it. This neural model is very effective, but also very expensive to train in terms of time and hardware resources[1], because there are many parameters to be learned; not to mention that the model is overfitting-prone, thus it needs to be trained on a training set of annotated images that is as large and heterogeneous as possible, in order to achieve a good generalization capability.

Hardware and time constraints do not always allow to train a model in an optimal setting, and, for example, cutting down on the dataset size could be necessary: in this case we have *poor training conditions*. Of course, this reduces the model's ability to generalize on new images at captioning time.
Another cause of poor training conditions is the lack of a good quality dataset, for example in terms of annotations: the manual captioning of large collections of images requires a lot of effort and, as of now, human-annotated datasets only exist for a restricted set of languages, such as English. As a consequence, training such a neural model to produce captions in another language (e.g. in Italian) is an interesting problem to explore, but also challenging due to the lack of data resources.

A viable approach is building a resource by *automatically translating the annotations from an existing dataset*: this is much less expensive than manually annotating images, but of course it leads to a loss of human-like quality in the language model. This approach has been adopted in this work to perform one of the first neural-based image captioning in Italian: more precisely, the annotations of the images from the MSCOCO dataset, one of the largest datasets in English of image/caption pairs, have been automatically translated to Italian in order to obtain a first resource for this language. This has been exploited to train a neural captioner, whose quality can be improved over time (e.g., by manually validating the translations). Then, a subset of this Italian dataset has been used as training data for the neural captioning system defined in (Vinyals et al. 2014), while a subset of the test set has been manually validated for evaluation purposes.

1 As of now, training a neural encoder-decoder model such as the one presented at `http://github.com/tensorflow/models/tree/master/im2txt` on a dataset of over $580,000$ image-caption examples takes about two weeks even with a very performing GPU.

In particular, prior to the experimentations in Italian, some early experiments have been performed with the same training data originally annotated in English, to get a reference benchmark about convergence time and evaluation metrics on a dataset of smaller size. These results in English will suggest if the Italian image captioner shows similar performance when trained over a reduced set of examples or the noise induced in the automatic translation process compromises the neural training phase. Moreover, these experiments have also been performed with the introduction of a pre-trained word embedding (derived using the method presented in (Mikolov et al. 2013)), in order to measure how it affects the quality of the language model learned by the captioner, with respect to a randomly initialized word embedding that is learned together with the other model parameters.

Overall the contributions of this work are threefold: (*i*) the investigation of a simple, automatized way to acquire (possibly noisy) large-scale corpora for the training of neural image captioning methods in poor training conditions; (*ii*) the manual validation of a first set of human-annotated resources in Italian; (*iii*) the implementation of one of the first automatic neural-based Italian image captioners.

In the rest of the paper, the adopted neural architecture is outlined in Section 2. The description of a brand new resource for Italian is presented in Section 3. Section 4 reports the results of the early preparatory experimentations for the English language and then the ones for Italian. Finally, Section 5 derives the conclusions.

2. The Show and Tell Architecture

The Deep Architecture considered in this paper is the *Show and Tell* architecture, described in (Vinyals et al. 2014) and sketched in Figure 2. It follows an encoder-decoder structure where the image is encoded in a dense vector by a state-of-the-art deep CNN, in this case *InceptionV3* (Szegedy et al. 2016), followed by a fully connected layer; the resulting feature vector is fed to a LSTM, used to generate a text sequence, i.e. the caption.

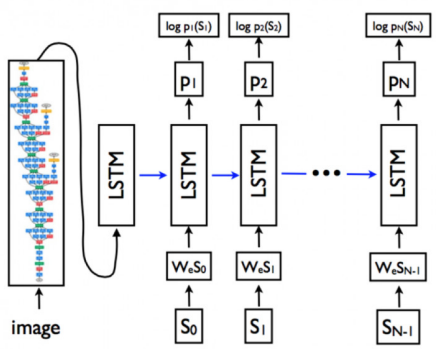

Figure 2
The Deep Architecture presented in (Vinyals et al. 2014). LSTM model combined with a CNN image embedder and word embeddings. The unrolled connections between the LSTM memories are in blue.

As the CNN encoder has been trained over an object recognition task, it allows encoding the image in a dense vector that is strictly connected to the entities observed in the image.

At the same time, the LSTM implements a language model, in line with the idea introduced in (Mikolov et al. 2010): it captures the probability of generating a given word in a string, given the words generated so far. In the overall training process, the main objective is to train a LSTM to generate the next word given not only the string produced so far, but also a set of image features.

Since the CNN encoder is (mostly) language independent, due to the fact that it is trained to recognize visual features that are not related to natural language, it can be totally re-used even in the captioning of images in other languages, such as Italian. On the contrary, the language model underlying the LSTM needs new examples to be trained.

In the following subsections, further details will be provided on the CNN and LSTM components of the *Show and Tell* architecture.

2.1 CNN image encoder: InceptionV3

As said before, the image-encoding CNN is the *InceptionV3* network, whose details can be found in (Szegedy et al. 2016): it is a modified version of GoogLeNet introduced in (Szegedy et al. 2015), whose architecture consists in repeated *Inception* modules: they are a combination of *convolutional layers*, that analyze adjacent groups of features coming as output from the previous Inception modules in the network, and *pooling layers*, which output a function f (in this case, max) of incoming inputs from the convolutional phase. Their structure is shown in Figure 3 and Figure 4.

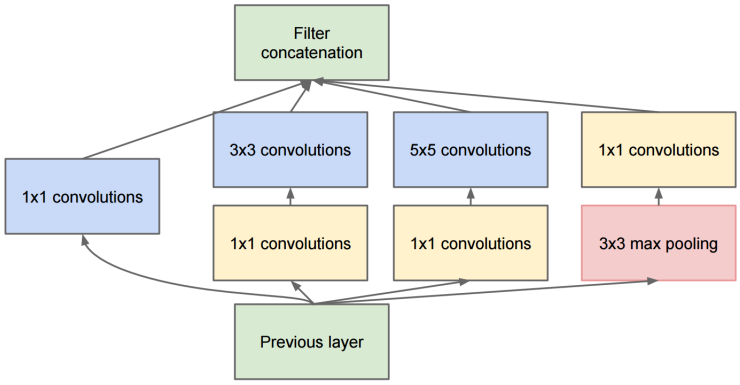

Figure 3
The original Inception module in GoogLenet.

InceptionV3 has shown state-of-the-art results in various visual recognition tasks: before being integrated in this architecture its weights are pre-trained on the image classification task.

The last layer of the network is a fully connected one and outputs a fixed-length vector representation of the image: this vector will be the first input fed to the LSTM.

2.2 LSTM decoder

The LSTM is the part of the architecture that takes as input the visual feature vector that comes as output from the CNN, and translates it to a descriptive sentence. At every time step, it outputs a softmax vector of probabilities over all the words in the vocabulary:

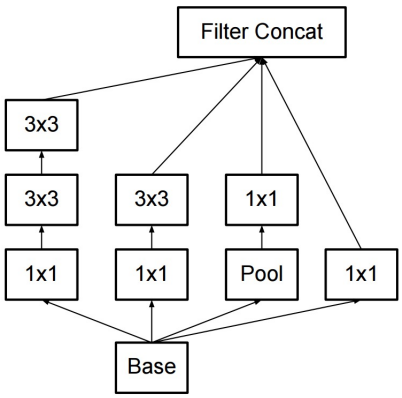

Figure 4
The new Inception module after refactoring 5×5 convolutions.

its elements represent the conditional probabilities $P(S_t|I, S_0, ..., S_{t-1})$ of the t-th word of the sequence. The input at the first time step is the output of the CNN, then it is the word vector $\mathbf{v}_{s_t}$ of the last word predicted s_t. The input, forget and output layers are sigmoidals functions: they are used as filters in order to choose the relevant parts of the sequence generated until that moment. All the definitions can be found below, where the notation and the figure are the same used in the papers (Vinyals et al. 2014) and (Vinyals et al. 2017):

$$
\begin{aligned}
i_t &= \sigma(W_{ix}x_t + W_{im}m_{t-1}) \\
f_t &= \sigma(W_{fx}x_t + W_{fm}m_{t-1}) \\
o_t &= \sigma(W_{ox}x_t + W_{om}m_{t-1}) \\
c_t &= f_t \odot c_{t-1} + i_t \odot h(W_{cx}x_t + W_{cm}m_{t-1}) \\
m_t &= o_t \odot c_t \\
p_{t+1} &= Softmax(m_t)
\end{aligned}
$$

As in (Vinyals et al. 2014), the core of the LSTM model is a memory cell *c* encoding knowledge at every time step of what inputs have been observed up to this step (as reported in Figure 5. The cell is controlled by *gates* ,Äì layers which are applied multiplicatively and thus can either keep a value from the gated layer if the gate is 1 or zero such value if the gate is equal to 0. In particular, three gates are being used which control whether to forget the current cell value (forget gate *f*): the memory block contains a cell *c* which is read its input (input gate *i*) and whether to output the new cell value (output gate *o*). Here $\odot$ represents the product with a gate value, and the various W matrices are trained parameters. The nonlinearities are sigmoid $\sigma(\cdot)$ and hyperbolic tangent $h(\cdot)$. Finally m_t is used to feed to a $Softmax$, which will produce a probability distribution p_t over all words.

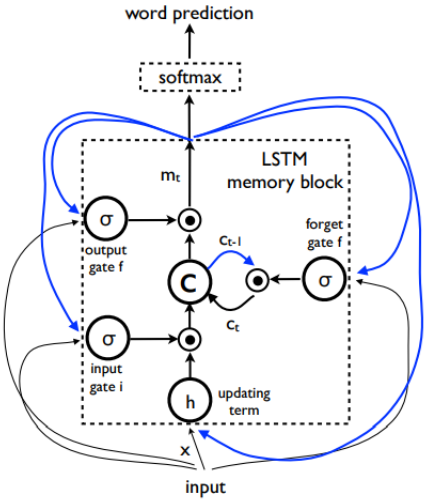

Figure 5
The gates and connections of the LSTM.

At training time, the loss function is the sum of the negative log-likelihoods for the generated words at every time step t:

$$L(I, S) = -\sum_{t=1}^{N} \log P_t(S_t). \quad (1)$$

The loss is minimized with respect to the LSTM parameters, the weights of the last fully connected layer of the CNN and the weights of the embedding matrix W.

At inference time, the used strategy to develop the best sequence starting from the output probability distribution is called **beam search**: at every time step t the k sentences with highest probabilities are kept and their $t+1$-th word is predicted. Finally, among the $k \times k$ generated sentences of length $t+1$, only the k best are chosen and expanded in the next step, and so on. In (Vinyals et al. 2017), the best results have been achieved with beam size $k = 3$, and this parameter has been kept in the experimental setting.

In this work, we will train this architecture over a corpus that has been automatically translated from the MSCOCO dataset. We thus speculate that the LSTM will learn a sort of simplified language model, more inherent to the automatic translator than to an Italian speaker. However, we are also convinced that the quality achievable by modern translation systems (Luong, Pham, and Manning 2015), combined with the generalization that can be obtained by a LSTM trained over thousands of (possibly noisy) translations will be able to generate reasonable and intelligible captions.

3. Automatic acquisition of a Corpus of Captions in Italian

In this section we present the first release of the MSCOCO-it, a new resource for the training of data-driven image captioning systems in Italian. It has been built starting from the MSCOCO dataset for English (Lin et al. 2014): in particular we considered the training and validation subsets, made respectively of $82,783$ and $40,504$ images, where every image has 5 human-written annotations in English.

The Italian version of the dataset has been acquired with an approach that automatizes the translation task: for each image, all its five annotations have been translated by using an automatic translator[2]. The result is a big amount of visual data annotated with multiple sentences: the annotations in English are fully translated to Italian, but not of the best quality with respect to the Italian fluent language. This automatically translated data can be used to train a model, but for the evaluation a test set of human-validated examples is needed: so, the translations of a subset of the MSCOCO-it have been manually validated.

In (Vinyals et al. 2014), two subsets of $2,024$ and $4,051$ images from the MSCOCO validation set have been held out from the rest of the data and have been used for development and testing of the model, respectively. A subset of these images has been manually validated: 308 images from the development set and 596 from the test set. In Table 1, statistics about this brand new corpus are reported, where the specific amount of unvalidated (*u.*) and validated (*v.*) data is made explicit[3].

Table 1
Statistics about the MSCOCO-it corpus. *p.* stands for *partially validated*, since some images have only some validated captions out of five. The partially validated images are between parentheses because they are already counted in the validated ones.

		#images	#captions	#words
training	*u.*	116,195	581,286	6,900,546
development	*v.*	308	1,516	17,913
	u.	1,696	8,486	101,448
	p.	(14)	25	304
test	*v.*	596	2,941	34,657
	u.	3,422	17,120	202,533
	p.	(23)	41	479
total		**122,217**	**611,415**	**7,257,880**

4. Experimental Evaluation

In order to be consistent with a scenario characterized by *poor training conditions* (limited hardware resources and time constraints) all the experimentations in this paper have been made by training the model on significantly smaller samples of data with respect to the whole MSCOCO dataset (made of more than $583,000$ image-caption examples).

First of all, some early experimentations have been performed on smaller samples of data from MSCOCO in English, in order to measure the loss of performance caused by the reduced size of the training set[4]. Each training example is a image-caption pair and these have been grouped in data *shards* during the training phase: each shard contains about $2,300$ image-caption examples. The model has been trained on datasets of $23,000$, $34,500$ and $46,000$ image-caption pairs (less than 10% of the entire dataset). In order to

2 Sentences have been translated by using Bing Translator (`https://www.bing.com/translator`) between December 2016 and January 2017.
3 Although Italian annotations are available for all the images of the original dataset, in the table some images were not counted because they are corrupted and therefore have not been used.
4 A proper tuning phase was too expensive so we adopted the parameters provided in `https://github.com/tensorflow/models/tree/master/im2txt`

balance the reduced size of the training material and provide some kind of linguistic generalization, we evaluated the adoption of pre-trained word embedding in the training/tagging process. In fact, in (Vinyals et al. 2014) the LSTM architecture initializes randomly all vectors representing input words; these are later trained together with the other parameters of the network.

We wondered if a word embedding already pre-trained on a large corpus could help the model to generalize better on brand new images at test time. The neural model chosen for this experiment, in order to learn a word embedding which could effectively represent a proper natural language model, is the Skip-gram model. The Skip-gram model is one of the variations of the *word2vec* language model, introduced by Mikolov et al. in (Mikolov et al. 2013) and (Mikolov et al. 2013) in 2013, which consists in training a shallow feed-forward network in order to learn word embedding vectors: its structure is made of an embedding layer and an output layer. The embedding layer maps the input to d-embedding vectors accordingly to the weights matrix C, while the output computes a probability distribution over words. *word2vec*'s success comes from being able to produce word vectors that, linearly combined, catch meaningful semantic properties between words. The Skip-gram model, applied to *word2vec*, given a word w_i, predicts its context words (the words that could possibly surround w_i) $\{w_{1,i}, ..., w_{C,i}\}$.

The input word w_i is mapped to a vector in the weights matrix W of the embedding layer and then a vector h is computed from it; the output are C probability distribution softmax vectors for the context words $\{w_{1,i}, ..., w_{C,i}\}$. The architecture of the Skip-gram neural model is reported in Figure 6.

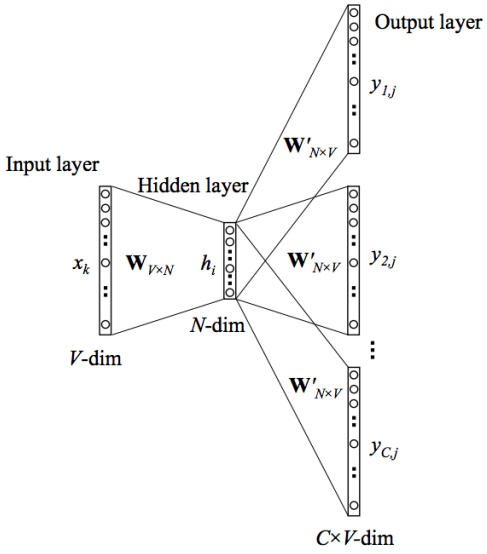

Figure 6
The Skip-gram model.

Due to this embedding model's ability to capture many different semantic shades of the natural language's words, we speculate that it could help the Show and Tell captioner when the amount of training data is too small to generalize, and therefore introduce in the model a word embedding learned through a Skip-gram model from an English dump of Wikipedia. The LSTM architecture has been trained on the same shards but initializing the word vectors with this pre-trained word embedding.

Table 2
Results on `im2txt` for the English language with a training set of reduced size, without / with the use of a pre-trained word embedding. Moreover benchmark results are reported.

# Shards	BLEU-4	METEOR	CIDEr
1	10,1 / 11,5	13,4 / 13,1	18,8 / 24,4
2	15,7 / 18,9	18,2 / 16,3	36,1 / 51,9
5	22,0 / 22,7	20,2 / 20,4	64,1 / 65,0
10	22,4 / 24,7	22,0 / 21,7	73,2 / 73,7
20	26,5 / 26,2	21,9 / 22,3	79,3 / 79,1
NIC (Vinyals et al. 2014)	27,7	23,7	85,5
NICv2 (Vinyals et al. 2014)	32,1	25,7	99,8
`im2txt`	31,2	25,5	98,1

Table 2 reports results on the English dataset in terms of BLEU-4, CIDEr and METEOR, the same used in (Vinyals et al. 2014): they are commonly used in machine translation evaluation to evaluate the quality of an automatic translation in comparison to other human-written reference sentences. Since the sentences generated by *Show and Tell* are *"translations"* of an image, it makes sense to apply these metrics to evaluate how good the generated captions are. Some details about them:

- **BLEU-4:** a geometric mean of the modified n-gram precisions on the whole corpus, penalized by a *brevity penalty* for too short sentences.
- **CIDEr**: an average, on n-grams, of average cosine similarities between the sentences to evaluate and their human-validated reference sentences.
- **METEOR**: a metric which creates a *matching* between unigrams of the sentence to evaluate and the validated reference sentence. The score is computed by combining the unigram precision, the unigram recall and a fragmentation penalty.

In the first five rows of Table 2, results are reported both in the case of randomly initialized word embedding and pre-trained ones. We compare these results with the ones achieved by the original NIC and NICv2 networks presented in (Vinyals et al. 2014), and the ones measured by testing a model available in the web[5], trained on the original whole training set (the last row, referred as `im2txt`).

Results obtained by the network when trained on a reduced dataset are clearly lower w.r.t. the NIC results, but it is straightforward that similar result are obtained, especially considering the reduced size of the training material. The contribution of pre-trained word embeddings is not significant, in line with the findings from (Vinyals et al. 2014). However, it is still interesting noting that the lexical generalization of this unsupervised word embeddings is beneficial, especially when the size of the training material is minimal (e.g. when one shard is used, especially if considering the CIDEr metrics). As the amount of training data grows, its impact on the model decreases, until it is not significant anymore.

5 `http://github.com/tensorflow/models/issues/466`

Table 3
Metrics for the experimentations on `im2txt` for the Italian language with a training set of reduced size, without / with and the use of a pre-trained word embedding.

# Shards	BLEU-4	METEOR	CIDEr
1	11.7 / 12.9	16.4 / 16.9	27.4 / 29.4
2	16.9 / 17.1	18.8 / 18.7	45.7 / 45.6
5	22.0 / 21.4	21.2 / 20.9	62.5 / 60.8
10	22.4 / 22.9	22.0 / 21.5	71.9 / 68.8
20	23.7 / 23.8	22.2 / 22.0	73.0 / 73.2

For what concerns the results on Italian, the experiments have been performed by training the model on samples of $23,000$, $34,500$ and $46,000$ examples that are the counterpart of the ones used for English, where the captions are automatically translated with Bing. The model has been evaluated against the validated sentences, and results are reported in Table 3. Results are impressive as they are in line with the English counterpart. It supports the robustness of the adopted architecture, as it seems to learn even from a noisy dataset of automatically translated material. Most importantly, it confirms the applicability of the proposed simple methodology for the acquisition of datasets for image captioning.

When trained with 20 shards, the Italian captioner generates the following description of the images shown in Figure 1: Image 1a: "*Un autobus a due piani guida lungo una strada.*", Image 1b: "*Un uomo che cavalca una carrozza trainata da cavalli.*", Image 1c: "*Una persona che cammina lungo una strada con un segnale di stop.*"

An attempt to use a word embedding that has been pre-trained on a large corpus (more precisely, on a dump of Wikipedia in Italian) has also been made, but the empirical results reported in Table 3 show that its contribution is not relevant but still significant when fewer examples are adopted. This confirm the beneficial impact of word embedding in neural training when the size of the labeled material is reduced, while it seems neglected when this amount grows.

4.1 Automatic translation: before or after sentence generation?

After demonstrating the viability of the proposed approach, we argued the following research question: *is it necessary to translate the sentences of the MSCOCO training set in order to let the model learn a more accurate Italian, or does the translation of the sentences generated from a model trained on English provide the same language quality?*

In order to answer to such question, the validated portion of the MSCOCO-it test set has been captioned by the `im2txt` model (see Table 2) trained on the *whole* English dataset (about $580,000$ training examples). Then, the generated captions have been automatically translated to Italian. Finally, these captions have been compared with the ones produced by the best model trained on Italian (training set of $46,000$ examples, see Table 3), in terms of quality, by comparing their BLEU-4, METEOR and CIDER metrics. The results are reported in the table below.

Results shown in table 4 suggest that training a model on a dataset already translated in Italian (last row) seems to achieve a better sentence quality w.r.t. performing the automatic translation task after the captions have already been generated from a model in English (row referred as `im2txt` + automatic translation: it is worth noting

Table 4
Results for the Italian language, by translating the training set and feeding it to the model and by translating the captions after being generated from the `im2txt` English model.

# Model	BLEU-4	METEOR	CIDEr
`im2txt` + automatic translation	21,6	21,8	70,9
Best model trained on Italian	23,8	22,0	73,2

(a) `im2txt`+translation: *Un giocatore di baseball che oscilla una mazza ad una sfera*, Italian model: *Un giocatore di baseball che tiene una mazza da baseball su un campo.*

(b) `im2txt`+translation: *Una grande torre dell'orologio che sovrasta una cittá*, Italian model: *Un grande edificio con un orologio sulla parte superiore.*

(c) `im2txt`+translation: *Un gruppo di persone che cavalcano sulle spalle dei cavalli*, Italian model: *un uomo che cavalca un cavallo in un campo.*

(d) `im2txt`+translation: *Una persona che salta una tavola skate in aria*, Italian model: *Un uomo che cavalca uno skateboard su una strada.*

Figure 7
Some images from the MSCOCO dataset, along with their descriptions generated from both the model trained on the full English dataset, with a subsequent translation to Italian, and the model trained directly on Italian.

the advantage of the model trained directly on Italian, although the number of training examples is significantly smaller than the other model ($580,000$ examples in English vs $46,000$ in automatically translated Italian).

We report a comparative analysis of some of the sentences generated from both models. Some images from the MSCOCO test set, along with their captions produced by both models, are reported in Figure set 7. Some of the differences that emerge from the captions are that the sentenced obtained by translating the caption of the English

`im2txt` model better captures *actions*, probably due to seeing many more training examples, but this ability is then heavily penalized by the noise introduced by the automatic translation (often raw and literal) as we can see in Figure 7a: the action of moving the baseball bat towards the ball (*[...] oscilla una mazza a una sfera*) has been described in the English sentence, but has been translated poorly in Italian. Meanwhile, the caption produced from the model trained on automatically translated Italian, even if the *swinging* action has not been captured, produces a less exhaustive but acceptable (from a linguistic perspective) description.

A similar phenomenon can be observed in Figure 7b: a verb is inserted in the English caption by `im2txt`, but even if this verb is properly translated, the automatic sentence translation is less accurate somewhere else (*torre dell'orologio*), while the Italian model produces a simpler but clearer description.

In Figure 7c, the model trained on Italian surprisingly captures more accurately the visual features (*Un uomo che cavalca*, a single person, not more than one), while the other model confuses the walking action of the group of persons ahead with the act of riding horses, probably due to their proximity: once more, the English model tries to provide more details about the action of riding horses (*[...]che cavalcano sulle spalle dei cavalli*), but the automatic translation makes it sound a bit rough.

What actually emerges is that the model trained directly on a dataset in Italian tends to learn simpler descriptions, with few action-describing verbs, that are clearer to read and less error-prone; the model trained on the whole English dataset has certainly seen more examples and is able to capture more actions from a scene, but the more syntactically complex are the sentences, the more error-prone they become when translated automatically.

5. Conclusions

In this paper a simple methodology for the training of neural models for the automatic captioning of images is presented, along with a large-scale dataset of about $600,000$ image captions in Italian produced by using an automatic machine translator. Although the noise introduced in this step, it allows to train one of the first neural-based image captioning systems for Italian.

The quality of this system seems comparable with the English counterpart, if trained over a comparable set of data: these results are impressive and confirm the robustness of the adopted neural architecture. Moreover, empirical evidence tell us that the approach of training a system on an Italian translated dataset produces sentences with a higher accuracy, with respect to producing captions with an English system, even if trained on many more visual and language features, and then translating them automatically.

We believe that the obtained resource paves the way to the definition and evaluation of Neural Models for Image captioning in Italian, and we hope to contribute to the Italian Community, hopefully using the validated dataset in a future Evalita[6] campaign.

References

Bernardi, Raffaella, Ruket Cakici, Desmond Elliott, Aykut Erdem, Erkut Erdem, Nazli Ikizler-Cinbis, Frank Keller, Adrian Muscat, and Barbara Plank. 2016. Automatic description generation from images: A survey of models, datasets, and evaluation measures. *Journal of Artificial Intelligence Research*, 55(1):409–442, January.

6 `http://www.evalita.it/`

Goldberg, Yoav. 2016. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57(1):345–420, September.

Karpathy, Andrej and Fei-Fei Li. 2015. Deep visual-semantic alignments for generating image descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pages 3128–3137, Boston, MA, USA, June 7-12. IEEE Computer Society.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 13th European Conference*, pages 740–755, Zürich, Switzerland, September 6-12. Springer International Publishing.

Luong, Thang, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, pages 1412–1421, Lisbon, Portugal, September 17-21. The Association for Computational Linguistics.

Mao, Junhua, Wei Xu, Yi Yang, Jiang Wang, and Alan L. Yuille. 2014. Deep captioning with multimodal recurrent neural networks (m-rnn). *CoRR*, abs/1412.6632.

Mikolov, Tomas, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association*, pages 1045–1048, Makuhari, Chiba, Japan, September 26-30, 2010.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Leon Bottou, Max Welling, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Proceedings of the 26th International Conference on Neural Information Processing Systems NIPS'13*. Curran Associates, Inc., Lake Tahoe, Nevada (USA), December 05 - 10, 2013, pages 3111–3119.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pages 1–9, June 7-12.

Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pages 2818–2826, Las Vegas, NV, USA, June 27-30.

Vinyals, Oriol, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2014. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555.

Vinyals, Oriol, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2017. Show and tell: Lessons learned from the 2015 MSCOCO image captioning challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):652–663.

Xu, Kelvin, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2048–2057, Lille, France, 07–09 Jul. PMLR.