Context-aware Convolutional Neural Networks for Twitter Sentiment Analysis in Italian

Giuseppe Castellucci, Danilo Croce, Roberto Basili
Department of Enterprise Engineering
University of Roma, Tor Vergata
Via del Politecnico 1, 00133 Roma, Italy
castellucci@ing.uniroma2.it, {croce,basili}@info.uniroma2.it

Abstract

English. This paper describes the `Unitor` system that participated to the *SENTIment POLarity Classification* task proposed in Evalita 2016. The system implements a classification workflow made of several Convolutional Neural Network classifiers, that generalize the linguistic information observed in the training tweets by considering also their context. Moreover, sentiment specific information is injected in the training process by using Polarity Lexicons automatically acquired through the automatic analysis of unlabeled collection of tweets. `Unitor` achieved the best results in the Subjectivity Classification sub-task, and it scored 2^{nd} in the Polarity Classification sub-task, among about 25 different submissions.

Italiano. *Questo lavoro descrive il sistema `Unitor` valutato nel task di SENTIment POLarity Classification proposto all'interno di Evalita 2016. Il sistema é basato su un workflow di classificazione implementato usando Convolutional Neural Network, che generalizzano le evidenze osservabili all'interno dei dati di addestramento analizzando i loro contesti e sfruttando lessici specifici per la analisi del sentimento, generati automaticamente. Il sistema ha ottenuto ottimi risultati, ottenendo la miglior performance nel task di Subjectivity Classification e la seconda nel task di Polarity Classification.*

1 Introduction

In this paper, the `Unitor` system participating in the *Sentiment Polarity Classification* (SENTIPOLC) task (Barbieri et al., 2016) within the Evalita 2016 evaluation campaign is described. The system is based on a cascade of three classifiers based on Deep Learning methods and it has been applied to all the three sub-tasks of SENTIPOLC: *Subjectivity Classification*, *Polarity Classification* and the pilot task called *Irony Detection*. Each classifier is implemented with a Convolutional Neural Network (CNN) (LeCun et al., 1998) according the modeling proposed in (Croce et al., 2016). The adopted solution extends the CNN architecture proposed in (Kim, 2014) with (i) sentiment specific information derived from an automatically derived polarity lexicon (Castellucci et al., 2015a), and (ii) with the contextual information associated with each tweet (see (Castellucci et al., 2015b) for more information about the contextual modeling in SA in Twitter). The `Unitor` system ranked 1^{st} in the Subjectivity Classification task and 2^{nd} in the Polarity Detection task among the unconstrained systems, resulting as one of the best solution in the challenge. It is a remarkable result as the CNNs have been trained without any complex feature engineering but adopting almost the same modeling in each sub-task. The proposed solution allows to achieve state-of-the-art results in *Subjectivity Classification* and *Polarity Classification* task by applying unsupervised analysis of unlabeled data that can be easily gathered by Twitter.

In Section 2 the deep learning architecture adopted in `Unitor` is presented, while the classification workflow is presented in 3. In Section 4 the experimental results are reported and discussed, while Section 5 derives the conclusions.

2 A Sentiment and Context aware Convolutional Neural Networks

The `Unitor` system is based on the Convolutional Neural Network (CNN) architecture for text classification proposed in (Kim, 2014), and further extended in (Croce et al., 2016). This deep net-

work is characterized by 4 layers (see Figure 1).

The *first layer* represents the input through word embedding: it is a low-dimensional representation of words, which is derived by the unsupervised analysis of large-scale corpora, with approaches similar to (Mikolov et al., 2013). The embedding of a vocabulary V is a look-up table $\mathbf{E}$, where each element is the $d-$dimensional representation of a word. Details about this representation will be discussed in the next sections. Let $\mathbf{x}_i \in \mathbb{R}^d$ be the d-dimensional representation of the i-th word. A sentence of length n is represented through the concatenation of the word vectors composing it, i.e., a matrix $\mathbf{I}$ whose dimension is $n \times d$.

The *second layer* represents the convolutional features that are learned during the training stage. A *filter*, or *feature detector*, $\mathbf{W} \in \mathbb{R}^{f \times d}$, is applied over the input layer matrix producing the learned representations. In particular, a new feature c_i is learned according to: $c_i = g(\mathbf{W} \cdot \mathbf{I}_{i:i+f-1} + b)$, where g is a non-linear function, such as the rectifier function, $b \in \mathbb{R}$ is a bias term and $\mathbf{I}_{i:i+f-1}$ is a portion of the input matrix along the first dimension. In particular, the filter slides over the input matrix producing a feature map $\mathbf{c} = [c_1, \ldots, c_{n-h+1}]$. The filter is applied over the whole input matrix by assuming two key aspects: *local invariance* and *compositionality*. The former specifies that the filter should learn to detect patterns in texts without considering their exact position in the input. The latter specifies that each local patch of height f, i.e., a f-gram, of the input should be considered in the learned feature representations. Ideally, a f-gram is composed through $\mathbf{W}$ into a higher level representation.

In practice, multiple filters of different heights can be applied resulting in a set of learned representations, which are combined in a *third layer* through the *max-over-time* operation, i.e., $\tilde{c} = max\{\mathbf{c}\}$. It is expected to select the most important features, which are the ones with the highest value, for each feature map. The *max-over-time* pooling operation serves also to make the learned features of a fixed size: it allows to deal with variable sentence lengths and to adopt the learned features in fully connected layers.

This representation is finally used in the *fourth layer*, that is a fully connected softmax layer. It classifies the example into one of the categories of the task. In particular, this layer is characterized by a parameter matrix $\mathbf{S}$ and a bias term $\mathbf{b_c}$ that is used to classify a message, given the learned representations $\tilde{c}$. In particular, the final classification y is obtained through $\text{argmax}_{y \in Y}(softmax(\mathbf{S} \cdot \tilde{c} + \mathbf{b_c}))$, where Y is the set of classes of interest.

In order to reduce the risk of over-fitting, two forms of regularization are applied, as in (Kim, 2014). First, a *dropout* operation over the penultimate layer (Hinton et al., 2012) is adopted to prevent co-adaptation of hidden units by randomly dropping out, i.e., setting to zero, a portion of the hidden units during forward-backpropagation. The second regularization is obtained by constraining the l_2 norm of $\mathbf{S}$ and $\mathbf{b_c}$.

2.1 Injecting Sentiment Information through Polarity Lexicons

In (Kim, 2014), the use of word embeddings is advised to generalize lexical information. These word representations can capture paradigmatic relationships between lexical items. They are best suited to help the generalization of learning algorithms in natural language tasks. However, paradigmatic relationships do not always reflect the relative sentiment between words. In Deep Learning, it is a common practice to make the input representations trainable in the final learning stages. This is a valid strategy, but it makes the learning process more complex. In fact, the number of learnable parameters increases significantly, resulting in the need of more annotated examples in order to adequately estimate them.

We advocate the adoption of a multi-channel input representation, which is typical of CNNs in image processing. A first channel is dedicated to host representations derived from a word embedding. A second channel is introduced to *inject* sentiment information of words through a large-scale polarity lexicon, which is acquired according to the methodology proposed in (Castellucci et al., 2015a). This method leverages on word embedding representations to assign polarity information to words by transferring it from sentences whose polarity is known. The resultant lexicons are called Distributional Polarity Lexicons (DPLs). The process is based on the capability of word embedding to represent both sentences and words in the same space (Landauer and Dumais, 1997). First, sentences (here tweets) are labeled with some polarity classes: in (Castellucci et al., 2015a) this labeling is achieved by apply-

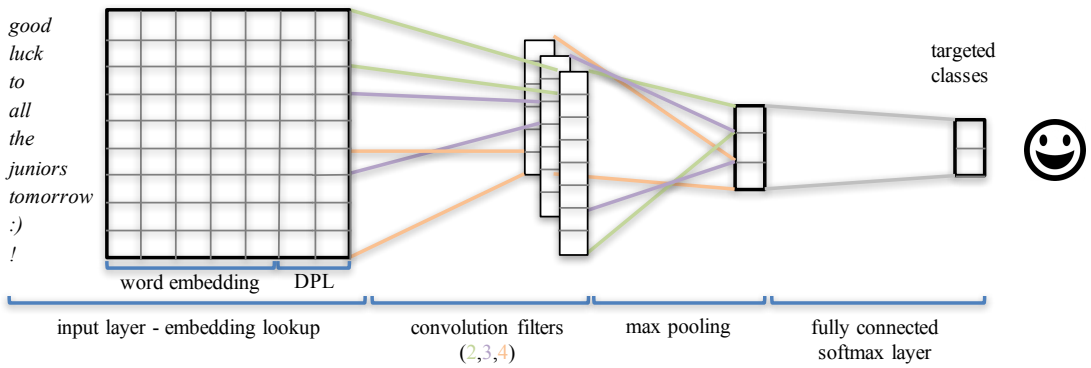

Figure 1: The Convolutional Neural Network architecture adopted for the `Unitor` system.

ing a Distant Supervision (Go et al., 2009) heuristic. The labeled dataset is projected in the embedding space by applying a simple but effective linear combination of the word vectors composing each sentence. Then, a polarity classifier is trained over these sentences in order to emphasize those dimensions of the space more related to the polarity classes. The DPL is generated by classifying each word (represented in the embedding through a vector) with respect to each targeted class, using the confidence level of the classification to derive a word polarity signature. For example, in a DPL the word *ottimo* is 0.89 positive, 0.04 negative and 0.07 neutral (see Table 1). For more details, please refer to (Castellucci et al., 2015a).

Term	w/o DPL	w/ DPL
ottimo (0.89,0.04,0.07)	pessimo eccellente ottima	ottima eccellente fantastico
peggiore (0.17,0.57,0.26)	peggior peggio migliore	peggior peggio peggiori
triste (0.04,0.82,0.14)	deprimente tristissima felice	deprimente tristissima depressa

Table 1: Similar words in the embedding without ($2^{nd}column$) and with ($3^{rd}column$) DPL, whose scores (*positivity, negativity, neutrality*) are in the first column.

This method has two main advantages: first, it allows deriving a signature for each word in the embedding to be used in the CNN; second, this method allows assigning sentiment information to words by observing their usage. This represents an interesting setting to observe sentiment related phenomena, as often a word does not carry a sentiment if not immersed in a context (i.e., a sentence).

As proposed in (Croce et al., 2016), in order to keep limited the computational complexity of the training phase of CNN, we augment each vector from the embedding with the polarity scores derived from the DPL[1]. In Table 1, a comparison of the most similar words of polarity carriers is compared when the polarity lexicon is not adopted (second column) and when the multi-channel schema is adopted (third column). Notice that, the DPL positively affects the vector representations for SA. For example, the word *pessimo* is no longer in set of the 3-most similar words of the word *ottimo*. The polarity information captured in the DPL making words that are semantically related and whose polarity agrees nearer in the space.

2.2 Context-aware model for SA in Twitter

In (Severyn and Moschitti, 2015) a pre-training strategy is suggested for the Sentiment Analysis task. The adoption of heuristically classified tweet messages is advised to initialize the network parameters. The selection of messages is based on the presence of emoticons (Go et al., 2009) that can be related to polarities, e.g. :) and :(. However, selecting messages only with emoticons could potentially introduce many topically unrelated messages that use out-of-domain linguistic expressions and limiting the contribution of the pre-training. We instead suggest to adopt another strategy for the selection of pre-training data. We draw on the work in (Vanzo et al., 2014), where topically related messages of the target domain are selected by considering the *reply-to* or *hashtag contexts* of each message. The former (*conversational context*) is made of the stream of messages belonging to the same conversation in Twitter, while the latter (*hashtag context*) is composed by tweets preceding a target message and sharing at least one hashtag with it. In (Vanzo et al., 2014), these messages are first classified through a

[1] We normalize the embedding and the DPL vectors before the juxtaposition.

context-unaware SVM classifier. Here, we are going to leverage on contextual information for the selection of pre-training material for the CNN. We select the messages both in the *conversation* context, and we classify them with a context-unaware classifier to produce the pre-training dataset.

3 The `Unitor` Classification Workflow

The SENTIPOLC challenge is made of three subtasks aiming at investigating different aspects of the subjectivity of short messages. The first subtask is the Subjectivity Classification that consists in deciding whether a message expresses subjectivity or it is objective. The second task is the Polarity Classification: given a subjective tweet a system should decide whether a tweet is expressing a neutral, positive, negative or conflict position. Finally, the Irony Detection sub-task aims at finding whether a message is expressing ironic content or not. The `Unitor` system tackles each sub-task with a different CNN classifier, resulting in a classification workflow that is summarized in the Algorithm 1: a message is first classified with the *Subjectivity* CNN-based classifier `S`; in the case the message is classified as *subjective* (`subjective=True`), it is also processed with the other two classifiers, the *Polarity* classifier `P` and the *Irony* classifier `I`. In the case the message is first classified as *objective* (`subjective=False`), the remaining classifiers are not invoked.

Algorithm 1 `Unitor` classification workflow.

```
1: function TAG(tweet T, cnn S, cnn P, cnn I)
2:     subjective = S(T)
3:     if subjective==True then
4:         polarity = P(T), irony = I(T)
5:     else
6:         polarity = none, irony = none
7:     end if
       return subjective, polarity, irony
8: end function
```

The same CNN architecture is adopted to implement all the three classifiers and tweets are modeled in the same way for the three sub-tasks. Each classifier has been specialized to the corresponding sub-task by adopting different selection policies of the training material and adapting the output layer of the CNN to the sub-task specific classes. In detail, the *Subjectivity* CNN is trained over the whole training dataset with respect to the classes `subjective` and `objective`. The *Polarity* CNN is trained over the subset of subjective tweets, with respect to the classes `neutral`, `positive`, `negative` and `conflict`. The *Irony* CNN is trained over the subset of subjective tweets, with respect to the classes `ironic` and `not-ironic`.

Each CNN classifier has been trained in the two settings specified in the SENTIPOLC guidelines: *constrained* and *unconstrained*. The constrained setting refers to a system that adopted only the provided training data. For example, in the constrained setting it is forbidden the use of a word embedding generated starting from other tweets. The unconstrained systems, instead, can adopt also other tweets in the training stage. In our work, the constrained CNNs are trained without using a pre-computed word embedding in the input layer. In order to provide input data to the neural network, we randomly initialized the word embedding, adding them to the parameters to be estimated in the training process: in the following, we will refer to the constrained classification workflow as `Unitor`. The unconstrained CNNs are instead initialized with pre-computed word embedding and DPL. Notice that in this setting we do not back-propagate over the input layer. The word embedding is obtained from a corpus downloaded in July 2016 of about 10 millions of tweets. A 250-dimensional embedding is generated according to a Skip-gram model (Mikolov et al., 2013)[2]. Starting from this corpus and the generated embedding, we acquired the DPL according to the methodology described in Section 2.1. The final embedding is obtained by juxtaposing the Skip-gram vectors and the DPL[3], resulting in a 253-dimensional representation for about $290,000$ words, as shown in Figure 1. The resulting classification workflow made of unconstrained classifier is called `Unitor-U1`. Notice that these word representations represent a richer feature set for the CNN, however the cost of obtaining them is negligible, as no manual activity is needed.

As suggested in (Croce et al., 2016), the contextual pre-training (see Section 2.2) is obtained by considering the conversational contexts of the provided training data. This dataset is made of about $2,200$ new messages, that have been classified with the `Unitor-U1` system. This set of

[2]The following settings are adopted: window 5 and min-count 10 with hierarchical softmax

[3]Measures adopting only the Skip-gram vectors have been pursued in the classifier tuning stage; these have highlighted the positive contribution of the DPL.

messages is adopted to initialize the network parameters. In the following, the system adopting the pre-trained CNNs is called `Unitor-U2`.

The CNNs have a number of hyper-parameters that should be fine-tuned. The parameters we investigated are: *size of filters*, i.e., capturing $2/3/4/5$-grams. We combined together multiple filter sizes in the same run. The *number of filters* for each size: we selected this parameter among 50, 100 and 200. The *dropout keep probability* has been selected among 0.5, 0.8 and 1.0. The final parameters has been determined over a development dataset, made of the 20% of the training material. Other parameters have been kept fixed: *batch size* (100), *learning rate* (0.001), *number of epochs* (15) and *L2 regularization* (0.0). The CNNs are implemented in Tensorflow[4] and they have been optimized with the Adam optimizer.

4 Experimental Results

In Tables 2, 3 and 4 the performances of the `Unitor` systems are reported, respectively for the task of Subjectivity Classification, Polarity Classification and Irony Detection. In the first Table (2) the `F-0` measure refers to the F1 measure of the *objective* class, while *F-1* refers to the F1 measure of the *subjective* class. In the Table 3 the `F-0` measure refers to the F1 measure of the *negative* class, while *F-1* refers to the F1 measure of the *positive* class. Notice that in this case, the *neutral* class is mapped to a "not negative" and "not positive" classification and the *conflict* class is mapped to a "negative" and "positive" classification. The *F-0* and *F-1* measures capture also these configurations. In Table 4 the `F-0` measure refers to the F1 measure of the *not ironic* class, while *F-1* refers to the F1 measure of the *ironic* class. Finally, *F-Mean* is the mean between these *F-0* and *F-1* values, and is the score used by the organizers for producing the final ranks.

System	F-0	F-1	F-Mean	Rank
`Unitor-C`	.6733	.7535	.7134	4
`Unitor-U1`	.6784	.8105	.7444	1
`Unitor-U2`	.6723	.7979	.7351	2

Table 2: Subjectivity Classification results

Notice that our unconstrained system (`Unitor-U1`) is the best performing system in recognizing when a message is expressing a subjective position or not, with a final *F-mean* of .7444 (Table 2). Moreover, also the `Unitor-U2` system is capable of adequately classify whether a message is subjective or not. The fact that the pre-trained system is not performing as well as `Unitor-U1`, can be ascribed to the fact that the pre-training material size is actually small. During the classifier tuning phases we adopted also the *hashtag* contexts (about $20,000$ messages) (Vanzo et al., 2014) to pre-train our networks: the measures over the development set indicated that probably the *hashtag* contexts were introducing too many unrelated messages. Moreover, the pre-training material has been classified with the `Unitor-U1` system. It could be the case that the adoption of such added material was not so effective, as instead demonstrated in (Croce et al., 2016). In fact, in that work the pre-training material was classified with a totally different algorithm (Support Vector Machine) and a totally different representation (kernel-based). In this setting, the different algorithm and representation produced a better and substantially different dataset, in terms of covered linguistic phenomena and their relationships with the target classes. Finally, the constrained version of our system, obtained a remarkable score of .7134, demonstrating that the random initialization of the input vectors can be also adopted for the classification of the subjectivity of a message.

System	F-0	F-1	F-Mean	Rank
`Unitor-C`	.6486	.6279	.6382	11
`Unitor-U1`	.6885	.6354	.6620	2
`Unitor-U2`	.6838	.6312	.6575	3

Table 3: Polarity Classification results

In Table 3 the Polarity Classification results are reported. Also in this task, the performances of the unconstrained systems are higher with respect to the constrained one (.662 against .6382). It demonstrates the usefulness of acquiring lexical representations and use them as inputs for the CNNs. Notice that the performances of the `Unitor` classifiers are remarkable, as the two unconstrained systems rank in 2^{nd} and 3^{rd} position. The contribution of the pre-training is not positive, as instead measured in (Croce et al., 2016). Again, we believe that the problem resides in the size and quality of the pre-training dataset.

In Table 4 the Irony Detection results are reported. Our systems do not perform well, as all the submitted systems reported a very low recall

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

System	F-0	F-1	F-Mean	Rank
Unitor-C	.9358	.016	.4761	10
Unitor-U1	.9373	.008	.4728	11
Unitor-U2	.9372	.025	.4810	9

Table 4: Irony Detection results

for the *ironic* class: for example, the Unitor-U2 recall is only .0013, while its precision is .4286. It can be due mainly to two factors. First, the CNN devoted to the classification of the irony of a message has been trained with a dataset very skewed towards the *not-ironic* class: in the original dataset only 868 over 7409 messages are ironic. Second, a CNN observes local features (bi-grams, tri-grams, ...) without ever considering global constraints. Irony, is not a word-level phenomenon but, instead, it is related to sentence or even social aspects. For example, the best performing system in Irony Detection in SENTIPOLC 2014 (Castellucci et al., 2014) adopted a specific feature, which estimates the violation of paradigmatic coherence of a word with respect to the entire sentence, i.e., a global information about a tweet. This is not accounted for in the CNN here discussed, and ironic sub-phrases are likely to be neglected.

5 Conclusions

The results obtained by the Unitor system at SENTIPOLC 2016 are promising, as the system won the Subjectivity Classification sub-task and placed in $2^n d$ position in the Polarity Classification. While in the Irony Detection the results are not satisfactory, the proposed architecture is straightforward as its setup cost is very low. In fact, the human effort in producing data for the CNNs, i.e., the pre-training material and the acquisition of the Distributional Polarity Lexicon is very limited. In fact, the former can be easily acquired with the Twitter Developer API; the latter is realized through an unsupervised process (Castellucci et al., 2015a). In the future, we need to better model the irony detection problem, as probably the CNN here adopted is not best suited for such task. In fact, irony is a more global linguistic phenomenon than the ones captured by the (local) convolutions operated by a CNN.

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 Pierpaolo Basile, Anna Corazza, Franco Cutugno, Simonetta Montemagni, Malvina Nissim, Viviana Patti, Giovanni Semeraro, and Rachele Sprugnoli, editors, *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)*. Associazione Italiana di Linguistica Computazionale (AILC).

Giuseppe Castellucci, Danilo Croce, Diego De Cao, and Roberto Basili. 2014. A multiple kernel approach for twitter sentiment analysis in italian. In *Fourth International Workshop EVALITA 2014*.

Giuseppe Castellucci, Danilo Croce, and Roberto Basili. 2015a. Acquiring a large scale polarity lexicon through unsupervised distributional methods. In *Proc. of 20th NLDB*, volume 9103. Springer.

Giuseppe Castellucci, Andrea Vanzo, Danilo Croce, and Roberto Basili. 2015b. Context-aware models for twitter sentiment analysis. *IJCoL vol. 1, n. 1: Emerging Topics at the 1st CLiC-It Conf.*, page 69.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2016. Injecting sentiment information in context-aware convolutional neural networks. *Proceedings of SocialNLP@ IJCAI*, 2016.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*.

Geoffrey Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings EMNLP 2014*, pages 1746–1751, Doha, Qatar, October. Association for Computational Linguistics.

Tom Landauer and Sue Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. of the IEEE*, 86(11), Nov.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

Aliaksei Severyn and Alessandro Moschitti. 2015. Twitter sentiment analysis with deep convolutional neural networks. In *Proc. of the SIGIR 2015*, pages 959–962, New York, NY, USA. ACM.

Andrea Vanzo, Danilo Croce, and Roberto Basili. 2014. A context-based model for sentiment analysis in twitter. In *Proc. of 25th COLING*, pages 2345–2354.