Word Embeddings in Sentiment Analysis

Ruggero Petrolito•, Felice Dell'Orletta◇
• Università di Pisa
ruggero.petrolito@gmail.com
◇Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC–CNR)
ItaliaNLP Lab - *www.italianlp.it*
felice.dellorletta@ilc.cnr.it

Abstract

English. In the late years sentiment analysis and its applications have reached growing popularity. Concerning this field of research, in the very late years machine learning and word representation learning derived from distributional semantics field (i.e. word embeddings) have proven to be very successful in performing sentiment analysis tasks. In this paper we describe a set of experiments, with the aim of evaluating the impact of word embedding-based features in sentiment analysis tasks.

Italiano. *Recentemente la Sentiment Analysis e le sue applicazioni hanno acquisito sempre maggiore popolarità. In tale ambito di ricerca, negli ultimi anni il machine learning e i metodi di rappresentazione delle parole che derivano dalla semantica distribuzionale (nello specifico i word embedding) si sono dimostrati molto efficaci nello svolgimento dei vari compiti collegati con la sentiment analysis. In questo articolo descriviamo una serie di esperimenti condotti con l'obiettivo di valutare l'impatto dell'uso di feature basate sui word embedding nei vari compiti della sentiment analysis.*

1 Introduction

In the late years sentiment analysis has reached great popularity among NLP tasks. As reported by Mäntylä et al. (2016) the number of papers on this subject has increased significantly in the first two decades of 21st century, as well as the extent of its applications. A wide variety of technologies has been used to assess sentiment analysis tasks during this period. In the latter years, machine learning techniques proved to be very effective; in particular, in recent years systems based on deep learning techniques represent the state of the art. In this field, **word embeddings** have been widely used as a way of representing words in sentiment analysis tasks, and proved to be very effective.

A relevant mirror of the state of the art in sentiment analysis field can be found in the *SemEval* workshops. In the 2015 edition (Rosenthal et al., 2015), most participants used machine learning techniques; in many of the subtasks, the top ranking systems used deep learning methods and word embeddings, like the system submitted by Severyn and Moschitti (2015), which was ranked 1st in subtask A and 2nd in subtask B. In 2016 edition (Nakov et al., 2016), deep learning based techniques, such as convolutional neural networks and recurrent neural networks, were the most popular approach. In 2017 edition (Rosenthal et al., 2017), machine learning methods were very popular, especially support vector machines and deep neural networks like convolutional neural networks and long short-term neural networks.

Concerning Italian language, **EVALITA** conference well represents the state of the art in the natural language processing field. In 2016 edition (Barbieri et al., 2016), the top ranking systems used machine learning and deep learning techniques (Castellucci et al. (2016), Attardi et al. (2016), Di Rosa and Durante (2016)).

The purpose of this study is to explore ways of using word embeddings to build meaningful representations of documents in sentiment analysis tasks performed on Italian tweets.

2 Our Contribution

In this paper we aimed to evaluate the effect of exploiting word embeddings in sentiment analysis tasks. In particular, we explore the effect of five factors on the performance of a sentiment analysis classification system, to answer five research questions:

1. What is the effect of the size of the corpus used to train the embeddings?
2. Which text domain allows us to train better embeddings (in-domain vs out-of-domain data)?
3. Which type of learning method produces better embeddings (word vs character-based word embeddings)?
4. Which method to combine the word vectors produces a better document vector representation?
5. What are the most important words (in terms of part-of-speech) to produce a better document vector representation?

To answer such questions, we performed several classification experiments testing our system on the three sentiment analysis tasks proposed in the 2016 EVALITA SENTIPOLC campaign (Barbieri et al., 2016): **Subjectivity Classification**, **Polarity Classification** and **Irony Detection**. In the first of these tasks, the highest accuracy was achieved by the system of Castellucci et al. (2016). Concerning the 2nd task, the most accurate system was the one submitted by Attardi et al. (2016). Regarding the 3rd task, the highest accuracy value was reached by the system of Di Rosa and Durante (2016). Among these systems, Castellucci et al. (2016) and Attardi et al. (2016) use deep learning techniques (convolutional neural networks), while Di Rosa and Durante (2016) use an ensemble of many supervised learning classifiers.

3 Datasets

We tested our system on the three sentiment analysis tasks proposed in 2016 EVALITA SENTIPOLC campaign. These tasks and the related datasets have been described by Barbieri et al. (2016). We conducted our experiments on the training set provided by the organizers of the evaluation campaign, which is composed of 7921 tweets.

We train our word embeddings on two corpora: in-domain and out-domain. The in-domain dataset is a collection of tweets that we collected for this work, named **Tweets**. It is composed by almost 80 millions of tweets, resulting in around 1.2 billions of tokens. The out-of-domain dataset is the **Paisà** corpus, a collection of Italian web texts described by Lyding et al. (Lyding et al., 2013).

4 Experimental Setup

For our experiments, we used a classifier based on SVM using LIBLINEAR (Rong-En et al., 2013) as machine learning library. As features, the classifier uses only information extracted combining the word-embeddings of the words of the analyzed tweet.

In all the experiments described in this paper, our system addresses the classification tasks by performing **5-fold cross-validation** on the training set provided for the SENTIPOLC 2016 evaluation campaign. The final score is the average score. We evaluate each fold using the Average F-score described by Barbieri et al. (2016).

For what concerns the word embeddings, we trained two types of word embedding representations: i) the first one using the **word2vec**[1] toolkit (Mikolov et al., 2013). This tool learns lower-dimensional word embeddings, which are represented by a set of latent (hidden) variables, and each word is associated to a multidimensional vector that represents a specific instantiation of these variables; ii) the second one using **fastText** (Bojanowski et al., 2016), a library for efficient learning of word representations and sentence classification. This library allows to overcome the problem of out-of-vocabulary words which affects the methodology of word2vec. Generating out-of-vocabulary word embeddings is a typical issue for morphologically rich languages with large vocabularies and many rare words. FastText overcomes this limitation by representing each word as a bag of character n-grams. A vector representation is associated to each character n-gram and the word is represented as the sum of these character n-gram representations.

In both cases, each word is represented by a 100 dimensions vector, computed using the CBOW algorithm – that learns to predict the word in the middle of a symmetric window based on the sum of the vector representations of the words in the window – and considering a context window of 5 words.

5 Experiments and Results

To answer the questions listed in Section 2, we conducted a great amount of experiments, testing many ways of representing the tweets by exploiting in different manners the word embeddings of

[1] http://code.google.com/p/word2vec/

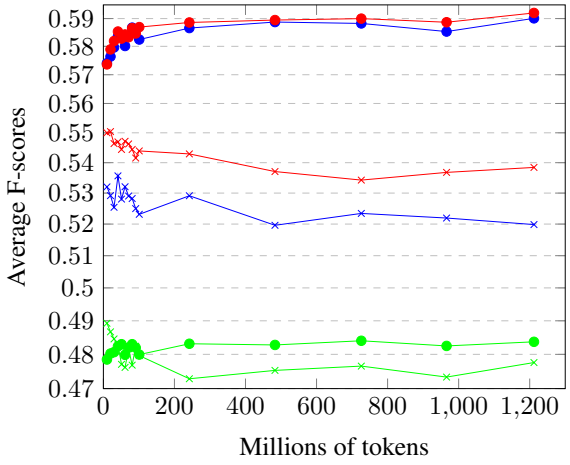

Figure 1: Average F-scores obtained by using embeddings trained on increasing amounts of token, using word2vec (circles) and fastText (crosses). Blue is assigned to *Subj. Classification*, red to *Pol. Classification* and green to *Irony Detection*.

the words extracted from the tweets.

To evaluate the impact (in terms of classification accuracy) of the variations of each studied parameter, we report the accuracy for each variation of the parameter calculated as the average accuracy across all the classification experiments that we conducted by varying all the other parameters (in a 5-fold cross-validation scenario).

In all the experiments, we used only features based on word embeddings.

5.1 Size of the Embeddings Training Corpus

To answer the question n. 1, we trained several word embedding models on different partitions of *Tweets* corpus of increasing sizes, using both *word2vec* and *fastText*. Ten smaller partitions were obtained starting with just ten millions of tokens (for the smaller one) and adding other ten millions for each new partition, reaching the amount of 100 millions. We created other four bigger partitions, which contain respectively 240, 480, 720 and 960 millions of tokens; the size of the smaller of this four partitions is comparable to the size of *Paisà*.

Figure 1 reports the results. When we use embeddings trained with word2vec on increasing amounts of data, the average value of F-score grows for all the three subtasks. The amount of this growth is similar for the subtasks *Subjectivity Classification* (0.016) and *Polarity Classification* (0.019), while it's smaller for the subtask *Irony Detection*, which is the most challenging among the three. In all cases the increase is significantly faster in the first 80 to 100 millions of tokens, particularly as regards the *Irony Detection* task: in this case, the average F-score basically stops growing after around 80 millions of tokens.

When we use embeddings trained with fastText, the outcome is the opposite: the average F-score values decrease as bigger amounts of data are used to train the embeddings. The decrease of the values is faster when using the first hundreds of millions of tokens.

Lesson learned: these results suggest that, regarding word-based word embeddings, as the training corpus grows the accuracy rises, but it becomes stable quickly. On the other hand, the increase of the size of the training corpus apparently doesn't influence the accuracy values when the embedding have been produced using fastText (or it even causes a lowering of the accuracy values).

5.2 Domain of the Embeddings Training Corpus

To answer the question n. 2, we ran a set of experiments using the four models obtained using word2vec and fastText on **Paisà** and **Tweet** corpora. Table 1 reports the results of the experiments. As we can see, the embeddings trained with word2vec on the in-domain dataset (*Tweets*) provide features that allow to achieve a higher average accuracy compared to the features extracted from the out-domain corpus. Differently, there isn't any variation in terms of accuracy when the embeddings are trained with fastText.

Lesson learned: the in-domain word embeddings are very important in a semantic classification scenario. Apparently, this is not true when character-based word embedding are used.

	Subj.		Pol.		Iro.	
	w2v	ft	w2v	ft	w2v	ft
tw	**0.5901**	0.5198	**0.592**	0.5384	**0.4837**	0.4776
pa	0.572	0.5206	0.5693	0.5312	0.4793	0.4759

Table 1: Average F-scores obtained by using word embeddings trained on Twitter (tw) and Paisà (pa) corpora.

5.3 Type of Embeddings Learning Model

For what regards the question n. 3, the type of embeddings learning model (words vs character n-grams) influences considerably the performance of the classifier. Using embeddings trained with word2vec leads to F-score values that are significantly higher in comparison to the accuracy ob-

tained using embeddings trained with fastText (see Table 1).

Lesson learned: this outcome suggests that embeddings learned by methods that treat words as atomic entities provide features that are more useful in a semantic task such as sentiment classification, in comparison with character-based embeddings.

5.4 Methods to Combine Word Embeddings

To answer the question n. 4, we tested many methods to combine the embeddings of the words of each document into a document-level vector representation.

We experimented five combining methods: *Sum*, *Mean*, *Maximum-pooling*, *Minimum-pooling*, *Product*. Each of this methods returns a single vector $\vec{t}$, such that each t_n is obtained by combining the nth components $w_{1n}, w_{2n} \dots w_{mn}$ of the embedding of each tweet word. Figure 2 shows a graphical representation of this process.

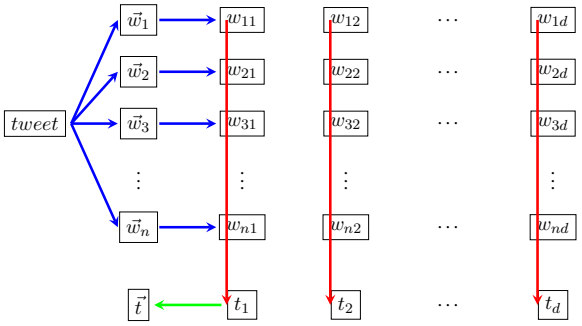

Figure 2: Embeddings combination process

We tested these methods separately, and all of them jointly as well. When using all methods, the document representation is obtained concatenating the vectors returned by each method.

As we can see in Table 2, the *Sum* method proved to be the best method for all the tasks, when using embeddings obtained by word2vec. The best results overall are obtained using the concatenation of each of the vectors returned by the used methods (row *All* in the Table). When using embeddings trained with fastText, the best results are obtained with *mean* for *Subjectivity* and *Polarity Classification*, and with *sum* for *Irony Detection*. In this case, the combination of all vector leads to poor results.

Lesson learned: these outcomes suggest that the best combination methods are *sum* for word vectors obtained by using word-based word embeddings and *mean* for character-based ones.

	Subj.		Pol.		Iro.	
	w2v	ft	w2v	ft	w2v	ft
Sum	**0.6054**	0.534	**0.6085**	0.5532	**0.4887**	**0.5033**
Mean	0.6017	**0.5951**	0.5954	**0.5916**	0.4709	0.4811
Max	0.5957	0.5012	0.5964	0.507	0.4736	0.4698
Min	0.593	0.5012	0.5951	0.5011	0.4754	0.4707
Prod	0.4415	0.4759	0.4384	0.5012	0.4693	0.4628
All	**0.6236**	0.4846	**0.6246**	0.51	**0.5202**	0.4715

Table 2: Average F-scores obtained by using different strategies of combination of word embeddings. Bold black values are the best F-scores overall; blue bold values are the best F-scores obtained by using a single combination method in the word-based word embeddings scenario (w2v); red bold values are the best F-scores in the character-based word embeddings scenario (ft).

Meanwhile, the worst approach is the *Product* combination. Interestingly, while the concatenation of all the combined word-based word embeddings is surely the best approach to produce the document-level vector representation, this is not true for the character-based ones.

5.5 Selection of Morpho-syntactic Categories of Combined Word Embeddings

To answer the question n. 5, we ran a set of experiments using only a subset of the word embeddings of each document to produce the document vector representation. The word selection is guided by the morpho-syntactic categories of the words. We tested four categories: *noun*, *verb*, *adjective*, *adverb*. The embeddings of the words belonging to each of these categories were combined in a pos-based vector representation document. In addition, we tested the document representation vector obtained through the concatenation of the different pos-based vectors (*N, V, Adj, Adv*) with and without the all-word document vector *All words*, which is the only one taking into account emoticons and hash tags.

Table 3 reports the results of the experiments. In the word-based word embedding scenario, regarding the contribution of single morpho-syntactic categories, *noun* shows the highest performance. Overall, the highest score is yielded by the combination of all the selected categories concatenated with the combined vector of all the word embeddings (*All words* rows in the table). For what regards the character-based word embeddings, we can see that the *noun* is the individually best performing category only for the Subjectivity Classification task, while the *adjective* and the *verb* are the best performing category for the other two tasks.

	Subj.		Pol.		Iro.	
	w2v	ft	w2v	ft	w2v	ft
N	**0.553**	**0.5171**	**0.5417**	0.5091	**0.4725**	0.4749
V	0.4755	0.4778	0.5091	0.5136	0.469	**0.4897**
Adj	0.4406	0.4534	0.5184	**0.5335**	0.4705	0.4826
Adv	0.4397	0.4504	0.4971	0.5033	0.4702	0.485
N, V, Adj, Adv	0.6266	0.5578	0.6141	0.5667	0.4948	0.5041
All words	0.6251	0.5363	0.5941	0.515	0.4773	0.4521
All words, N	0.6287	0.5221	0.6032	0.5343	0.4887	0.4646
All words, V	0.6326	0.5276	0.6035	0.5339	0.4841	0.4634
All words, Adj	0.6374	0.5328	0.6185	0.5184	0.4867	0.4693
All words, Adv	0.6337	0.5243	0.6087	0.5187	0.4856	0.4674
All words, N, V, Adj, Adv	**0.6521**	0.5691	**0.6319**	0.5546	**0.5139**	0.4886

Table 3: Average F-scores obtained using embedding of words belonging to different morpho-syntactic classes. Bold black values are the best F-scores overall; blue bold values are the best F-scores obtained using a single grammar class in the word-based word embeddings scenario (w2v); red bold values are the best F-scores obtained using a single grammar class in the character-based word embeddings scenario (ft).

Lesson learned: these results show that *noun* class is the most important grammatical category only in the word-based word embedding scenario; meanwhile the concatenation of all the pos-based vectors and the *All words* vector yields the best accuracy in both scenarios.

6 Conclusions

In this work we study the impact of word embedding-based features in the sentiment analysis tasks. We performed several classification experiments to investigate the effects on classification performances of five dimensions related to the word embeddings. We tested several different ways of selecting and combining the embeddings and we studied how the performance of a sentiment classifier changes.

Despite the lessons learned from this work, several aspects remain to investigate, such as, for example, the tuning of the parameters used to train the embeddings, and new vector combining strategies.

References

Giuseppe Attardi, Daniele Sartiano, Chiara Alzetta and Federica Semplici. 2016. *Convolutional Neural Networks for Sentiment Analysis on Italian Tweets*. CLiC-it/EVALITA.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli and Viviana Patti. 2016. *Overview of the evalita 2016 sentiment polarity classification task*. 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).

Yoshua Bengio, Réjean Ducharme, Pascal Vincent and Christian Jauvin. 2003. *A Neural Probabilistic Language Model*. Journal of Machine Learning Research 3 (2003) 1137–1155.

Piotr Bojanowski, Edouard Grave, Armand Joulin and Tomas Mikolov. 2016. *Efficient Estimation of Word Representations in Vector Space*. CoRR abs/1607.04606, 2016.

Giuseppe Castellucci, Danilo Croce and Roberto Basili. 2016. *Context-aware Convolutional Neural Networks for Twitter Sentiment Analysis in Italian*. CLiC-it/EVALITA.

Emanuele Di Rosa and Alberto Durante. 2016. *Tweet2Check evaluation at Evalita Sentipolc 2016*. CLiC-it/EVALITA.

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*. Institute for Applied Linguistics, Eurac Research.

Mika V. Mäntylä, Daniel Graziotin and Miikka Kuutila. 2016. *The Evolution of Sentiment Analysis - A Review of Research Topics, Venues, and Top Cited Papers*. Computer Science Review, Volume 27, February 2016, Pages 16-32.

Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. 2013. *Efficient Estimation of Word Representations in Vector Space*. CoRR abs/1301.3781, 2013.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani and Veselin Stoyanov. 2016. *SemEval-2016 task 4: Sentiment analysis in Twitter*. Proceedings of the 10th international workshop on semantic evaluation (semeval-2016).

Fan Rong-En, Chang Kai-Wei, Hsieh Cho-Jui, Wang Xiang-Ruind Lin Chih-Jen. 2008. *LIBLINEAR: A Library for Large Linear Classification*. Journal of Machine Learning Research, 9:1871-1874.

Sara Rosenthal, Preslav Nakov, Svetlana Kiritchenko, Saif Mohammad, Alan Ritter and Veselin Stoyanov. 2015. *Semeval-2015 task 10: Sentiment analysis in twitter*. Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015), 451–463.

Sara Rosenthal, Noura Farra and Preslav Nakov. 2017. *SemEval-2017 task 4: Sentiment analysis in Twitter*. Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), 502–518.

Aliaksei Severyn and Alessandro Moschitti. 2015. *Unitn: Training deep convolutional neural network for twitter sentiment classification*. Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015), 464–469.