10.12871/clicit2014214

Linguistically–motivated and Lexicon Features for Sentiment Analysis of Italian Tweets

Andrea Cimino[◇], **Stefano Cresci**[•], **Felice Dell'Orletta**[◇], **Maurizio Tesconi**[•]
[◇]Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC–CNR)
ItaliaNLP Lab - *www.italianlp.it*
[•]Institute for Informatics and Telematics (IIT–CNR)
{andrea.cimino,felice.dellorletta}@ilc.cnr.it
{stefano.cresci,maurizio.tesconi}@iit.cnr.it

Abstract

English. In this paper we describe our approach to EVALITA 2014 SENTIment POLarity Classification (SENTIPOLC) task. We participated only in the Polarity Classification sub–task. By resorting to a wide set of general–purpose features qualifying the lexical and grammatical structure of a text, automatically created ad–hoc lexicons and existing free available resources, we achieved the second best accuracy[1].

Italiano. *In questo articolo descriviamo il nostro sistema utilizzato per affrontare il compito di Polarity Classification del task SENTIPOLC della conferenza Evalita 2014. Sfruttando un gran numero di caratteristiche generiche che descrivono la struttura lessicale e sintattica del testo, la creazione automatica di lessici ad–hoc e l'uso di risorse disponibili esistenti, il sistema ha ottenuto il secondo miglior punteggio della competizione.*

1 Description of the system

Our approach to the Twitter Sentiment polarity detection task was implemented in a software prototype, i.e. a classifier operating on morpho-syntactically tagged and dependency parsed texts which assigns to each document a score expressing its probability of belonging to a given polarity class. The highest score represents the most probable class. Given a set of features and a training corpus, the classifier creates a statistical model using the feature statistics extracted from the training corpus. This model is used in the classification of unseen documents. The set of features and the machine learning algorithm can be parameterized through a configuration file. For this work, we used linear Support Vector Machines (SVM) using LIBSVM (Chang et al., 2001) as machine learning algorithm.

Since our approach relies on multi–level linguistic analysis, both training and test data were automatically morpho-syntactically tagged by the POS tagger described in (Dell'Orletta, 2009) and dependency-parsed by the DeSR parser using Multi-Layer Perceptron as learning algorithm (Attardi et al., 2009), a state-of-the-art linear-time Shift-Reduce dependency parser.

1.1 Lexicons

In order to improve the overall accuracy of our system, we developed and used sentiment polarity and similarity lexicons. All the created lexicons are made freely available at the following website: *http://www.italianlp.it/software/.*

1.1.1 Sentiment Polarity Lexicons

Sentiment polarity lexicons provide mappings between a word and its sentiment polarity (positive, negative, neutral). For our experiments, we used a publicly available lexicons for Italian and two English lexicons that we automatically translated. In addition, we adopted an unsupervised method to automatically create a lexicon specific for the Italian twitter language.

Existing Sentiment Polarity Lexicons

We used the Italian sentiment polarity lexicon (hereafter referred to as $OPENER$) (Maks et al., 2013) developed within the OpeNER European project[2]. This is a freely available lexicon for the Italian language[3] and includes 24,000 Italian word

[1]Because of an error of the conversion script from our internal format (of the output system) to the official one, we submitted the correct output after the task deadline, as soon as we noticed the error.

[2]http://www.opener-project.eu/

[3]https://github.com/opener-project/public-sentiment-lexicons

entries. It was automatically created using a propagation algorithm and manually reviewed for the most frequent words.

Automatically translated Sentiment Polarity Lexicons

- The Multi–Perspective Question Answering (hereafter referred to as $MPQA$) Subjectivity Lexicon (Wilson et al., 2005). This lexicon consists of approximately 8,200 English words with their associated polarity. In order to use this resource for the Italian language, we translated all the entries through the Yandex translation service[4].

- The Bing Lui Lexicon (hereafter referred to as BL) (Hu et al., 2004). This lexicon includes approximately 6,000 English words with their associated polarity. Like in the former case, this resource was automatically translated by the Yandex translation service.

Automatically created Sentiment Polarity Lexicons

We built a corpus of positive and negative tweets following the Mohammad et al. (2013) approach adopted in the Semeval 2013 sentiment polarity detection task. For this purpose we queried the Twitter API with a set of hashtag seeds that indicate positive and negative sentiment polarity. We selected 200 positive word seeds (e.g. "vincere" *to win*, "splendido" *splendid*, "affascinante" *fascinating*), and 200 negative word seeds (e.g., "tradire" *betray*, "morire" *die*). These terms were chosen from the OPENER lexicon. The resulting corpus is made up of 683,811 tweets extracted with positive seeds and 1,079,070 tweets extracted with negative seeds.

The main purpose of this procedure was to assign a polarity score to each n-gram occurring in the corpus. For each n-gram (we considered up to five n-grams) we calculated the corresponding sentiment polarity score with the following scoring function: $score(ng) = PMI(ng, pos) - PMI(ng, neg)$, where PMI stands for pointwise mutual information. A positive or negative score indicates that the n-gram is relevant for the identification of positive or negative tweets.

1.1.2 Word Similarity Lexicons

Since the lexical information in tweets can be very sparse, to overcame this problem we built two similarity lexicons.

For this purpose, we trained two predict models using the word2vec[5] toolkit (Mikolov et al., 2013). As recommended in (Mikolov et al., 2013), we used the CBOW model 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. For our experiments, we considered a context window of 5 words. These models learn lower-dimensional word embeddings. Embeddings are represented by a set of latent (hidden) variables, and each word is a multidimensional vector that represent a specific instantiation of these variables. We built the word similarity lexicons by applying the cosine similarity function between the embedded words.

Starting from two corpora, we developed two different similarity lexicons:

- The first lexicon was built using the lemmatized version of the PAISÀ[6] corpus (Lyding et al., 2014). PAISÀ is a freely available large corpus of authentic contemporary Italian texts from the web, and contains approximately 388,000 documents for a total of about 250 millions of tokens.

- The second lexicon was built from a lemmatized corpus of tweets. This corpus was collected starting from 30 generic seed keywords used to query Twitter APIs. The resulting corpus is made up of 1,200,000 tweets. These tweets were automatically morpho-syntactically tagged and lemmatized by the POS tagger described in (Dell'Orletta, 2009).

1.2 Features

In this study, we focused on a wide set of features ranging across different levels of linguistic description. The whole set of features we started with is described below, organised into four main categories: namely, *raw and lexical text features*, *morpho-syntactic features*, *syntactic features* and *lexicon features*. This proposed four–fold partition closely follows the different levels of linguistic analysis automatically carried out on the text being evaluated, (i.e. tokenization, lemmatization, morpho-syntactic tagging and dependency parsing) and the use of external lexical resources.

In the descriptions below, in brackets are reported the names of the features listed in Table 1.

[4]http://api.yandex.com/translate/

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

[6]http://www.corpusitaliano.it/

The second column of the table reports for each features the sizes of the used n–grams (for the n–gram features) or it marks whether the considered feature has been used in the final experiment (for the non n–gram features).

1.2.1 Raw and Lexical Text Features

Number of tokens: number of blocks consisting of 5 tokens occurring in the analyzed tweet. *(AVERAGE_TWEET_LENGTH)*

Character n-grams: presence or absence of contiguous sequences of characters in the analyzed tweet. *(NGRAMS_CHARS)*

Word n-grams: presence or absence of contiguous sequences of tokens in the analyzed tweet. *(NGRAMS_WORDS)*

Lemma n-grams: presence or absence of contiguous sequences of lemma occurring in the analyzed tweet. *(NGRAMS_LEMMAS)*

Repetition of n-grams chars: this feature checks the presence or absence of contiguous repetition of characters in the analyzed tweet. *(HAS_NGRAMS_CHARS_REPETITIONS)*

@ Number: number of @ occurring in the analyzed tweet. *(NUM_AT)*

Hashtags number: number of hashtags occurring in the analyzed tweet. *(NUM_HASHTAGS)*

Punctuation: checks whether the analyzed tweet finishes with one of the following punctuation characters: "?", "!". *(FINISHES_WITH_PUNCTUATION)*

1.2.2 Morpho-syntactic Features

Coarse grained Part-Of-Speech n-grams: presence or absence of contiguous sequences of coarse–grained PoS, corresponding to the main grammatical categories (e.g. noun, verb, adjective). *(NGRAMS_CPOS)*

Fine grained Part-Of-Speech n-grams: presence or absence of contiguous sequences of fine-grained PoS, which represent subdivisions of the coarse-grained tags (e.g. the class of nouns is subdivided into proper vs common nouns, verbs into main verbs, gerund forms, past particles). *(NGRAMS_POS)*

Coarse grained Part-Of-Speech distribution: the distribution of nouns, adjectives, adverbs, numbers in the tweet. *(CPOS_DISTR_PERC)*

1.2.3 Syntactic Features

Dependency types n-grams: presence or absence of sequences of dependency types in the analyzed tweet. The dependencies are calculated with respect to *i)* the hierarchical parse tree structure and *ii)* the surface linear ordering of words. *(NGRAMS_DEPTREE, NGRAMS_DEP)*

Lexical Dependency n-grams: presence or absence of sequences of lemmas calculated with respect to the hierarchical parse tree. *(NGRAMS_LEMMATREE)*

Lexical Dependency Triplet n-grams: distribution of lexical dependency triplets, where a triplet represents a dependency relation as (ld, lh, t), where ld is the lemma of the dependent, lh is the lemma of the syntactic head and t is the relation type linking the two. *(NGRAMS_LEMMA_DEP_TREE)*

Coarse Grained Part-Of-Speech Dependency n-grams: presence or absence of sequences of coarse-grained part–of–speech calculated with respect to the hierarchical parse tree. *(NGRAMS_CPOSTREE)*

Coarse Grained Part-Of-Speech Dependency Triplet n-grams: distribution of coarse-grained part–of–speech dependency triplets, where a triplet represents a dependency relation as (cd, ch, t), where cd is the coarse-grained part–of–speech of the dependent, h is the coarse-grained part–of–speech of the syntactic head and t is the relation type linking the two. *(NGRAMS_CPOS_DEP_TREE)*

1.2.4 Lexicon features

Emoticons: presence or absence of positive or negative emoticons in the analyzed tweet. The lexicon of emoticons was extracted from the site *http://it.wikipedia.org/wiki/Emoticon* and manually classified. *(SNT_EMOTICONS)*

Lemma sentiment polarity n-grams: for each lemma n-grams extracted from the analyzed tweet, the feature checks the polarity of each component lemma in the existing sentiment polarity lexicons. Lemma that are not present are marked with the *ABSENT* tag. This is for example the case of the trigram "tutto molto bello" (*all very nice*) that is marked as "*ABSENT-POS-POS*" because *molto* and *bello* are marked as positive in the considered polarity lexicon and *tutto* is absent. The feature is computed for each existing sentiment polarity lexicons. *(NGRAMS_SNT_OPENER, NGRAMS_SNT_MPQA, NGRAMS_SNT_BL)*.

Polarity modifier: for each lemma in the tweet occurring in the existing sentiment polarity lexicons, the feature checks the presence of adjectives or adverbs in a left context window of size 2.

If this is the case, the polarity of the lemma is assigned to the modifier. This is for example the case of the bigram "non interessante" (*not interesting*), where "interessante" is a positive word, and "non" is an adverb. Accordingly, the feature "non_POS" is created. The feature is computed 3 times, checking all the existing sentiment polarity lexicons. *(SNT_WITH_MODIFIER_OPENER, SNT_WITH_MODIFIER_MPQA, SNT_WITH_MODIFIER_BL)*

PMI score: for each set of unigrams, bigrams, trigrams, four-grams and five-grams that occur in the analyzed tweet, the feature computes the score given by $\sum_{\text{i}-gram \in tweet} score(\text{i}-gram)$ and returns the minimum and the maximum values of the five values (approximated to the nearest integer). *(PMI_SCORE)*

Distribution of sentiment polarity: this feature computes the percentage of positive, negative and neutral lemmas that occur in the tweet. To overcome the sparsity problem, the percentages are rounded to the nearest multiple of 5. The feature is computed for each existing lexicon. *(SNT_DISTRIBUTION_OPENER, SNT_DISTRIBUTION_MPQA, SNT_DISTRIBUTION_BL)*

Most frequent sentiment polarity: the feature returns the most frequent sentiment polarity of the lemmas in the analyzed tweet. The feature is computed for each existing lexicon. *(SNT_MAJORITY_OPENER, SNT_MAJORITY_MPQA, SNT_MAJORITY_BL)*

Word similarity: for each lemma of the analyzed tweet, the feature extracts the first 15 similar words occurring in the similarity lexicons. For each similar lemma, the feature checks the presence of negative or positive polarity. In addition, the feature calculates the most frequent polarity. Since we have two different similarity lexicons and three different sentiment lexicons, the feature is computed 6 times. *(COS_EXPLOSION_OPENER_PAISA, COS_EXPLOSION_OPENER_TWITTER, COS_EXPLOSION_MPQA_PAISA, COS_EXPLOSION_MPQA_TWITTER, COS_EXPLOSION_BL_PAISA, COS_EXPLOSION_BL_TWITTER)*

Sentiment polarity in tweet sections: the feature first splits the tweet in three equal sections. For each section the most frequent polarity is computed using the available sentiment polarity lexicons. The purpose of this feature is aimed at identifying change of polarity within the same tweet. *(SNT_POSITION_PRESENCE_OPENER, SNT_POSITION_PRESENCE_MPQA, SNT_POSITION_PRESENCE_BL)*

1.3 Feature Selection Process

Since our approach to Twitter Sentiment polarity detection task relies on a wide number of general-purpose features, a feature selection process was necessary in order to prune irrelevant and redundant features which could negatively affect the classification results. This feature selection process is a variant of the selection method described in (Cimino et al., 2013) used for the Native Language Identification shared task. This new approach has shown better results in terms of the accuracy of the resulting system.

The selection process starts taking into account all the n features described in Section 1.2 and listed in Table 1. The feature selection algorithm drops and adds features until a termination condition is satisfied.

Let F_e be a set containing all the features, and F_d another set of features, initially empty. Let $F_{we} = F_e$ and $F_{wd} = F_d$ two auxiliary sets. In the drop–feature stage, for each feature $f_i \in F_{we}$ we generate a configuration c_i such that the features in $\{f_i\} \cup F_{wd}$ are disabled and all the other features are enabled. When an iteration finishes, we obtain for each c_i a corresponding accuracy score $score(c_i)$ which is computed as as the average of the accuracy obtained by the classifier on five non overlapping test-sets, each one corresponding to the 20% of the training set. We used this five cross fold validation in order to reduce overfitting.

Being c_b the best configuration among all the c_i configurations, and c_B the best configuration found in the previous iterations, if

$$score(c_b) \geq score(c_B) \qquad (1)$$

- Move f_b from F_{we} to F_{wd};
- set $F_d := F_{wd}$ and $F_e := F_{we}$;
- set $c_B := c_b$.

If the condition (1) is not satisfied and:

$$score(c_b) + k \geq score(c_B) : \qquad (2)$$

- Move f_b from F_{we} to F_{wd}.

For our experiments we set the k initial value to 1.

If the condition (1) or (2) is satisfied, the feature selection process continues with another drop–iteration, otherwise set $k = \frac{k}{2}$.

If $k \leq \alpha$ the feature selection process stops and the configuration c_B is the result of our feature selection process[7]. Otherwise:

- set $F_{wd} := F_d$ and $F_{we} := F_e$,

and the feature selection process continues with the add–feature stage.

In the add–stage we add to the currently best model (c_B) the features previously pruned. For each feature $f_i \in F_{wd}$ we generate a configuration c_i such that the features in $\{f_i\} \cup F_{we}$ are enabled and all the other features are disabled.

For each add–iteration, the process checks the conditions (1) and (2). If the condition (2) is verified and $k \geq \alpha$, another drop–feature stage starts.

In spite of the fact that the described selection process does not guarantee to obtain the global optimum, it however permitted us to obtain an improvement of 2 percentage points (on the five cross validation set) with respect to the starting model indiscriminately using all features.

Table 1 lists the features resulting from the feature selection process.

2 Results and Discussion

Table 2 reports the overall accuracies achieved by our classifier using different feature configuration models in the Polarity Classification task on the official test set. The accuracy is calculated as the average F–score of our system obtained using the evaluation tool provided by the organizers (Basile et al., 2014). Since the official scoring function assigns a bonus also for partial matching (e.g. a Positive or Negative assignment instead of Positive–Negative class), we also report the F–score for each considered polarity class considering only the correct assignments. The first row of the Table shows the results for the *FeatSelLexicons* model resulting from the feature selection process described in section 1.3. This is our official result submitted for the competition. The second row reports the results for the model that uses the same features of the *FeatSelLexicons* classifier where all the lexicon features are disabled. The last row shows the results for the model that contains all the features listed in Table 1. Table 3 reports the accuracy over the training data before and after the feature selection process. In both cases, we performed a five-fold cross validation evaluation.

For what concerns the results on the official test set, the *AllFeat* model performs slightly better than the *FeatSelLexicons* model, even if the difference in terms of accuracy is not statistically significant. This demonstrates that the lexical, morpho-syntactic, syntactic and lexicon features excluded

Lexical features	
Feature name	n-grams
HAS_NGRAMS_CHARS_REPETITIONS	1 2 **3** **4**
NGRAMS_CHARS	**1** **2** **3** **4**
NGRAMS_WORDS	**1** **2** **3** **4**
NGRAMS_LEMMAS	**1** **2** **3** 4
Feature name	boolean
FINISHES_WITH_PUNCTUATION	True
NUM_AT	True
NUM_HASHTAGS	False
AVERAGE_TWEET_LENGTH	True
SNT_EMOTICONS	True
Morpho–syntactic features	
Feature name	n-grams
NGRAMS_CPOS	1 2 **3**
NGRAMS_POS	1 2 **3**
Feature name	boolean
CPOS_DISTR_PERC	True
Syntactic features	
Feature name	n-grams
NGRAMS_DEP	**1** 2 3
NGRAMS_DEPTREE	**1** **2** 3 4
NGRAMS_LEMMATREE	1 **2** **3** 4
NGRAMS_LEMMA_DEP_TREE	**1** 2 3 4
NGRAMS_CPOSTREE	**1** 2 **3** **4**
NGRAMS_CPOS_DEP_TREE	**1** 2 3 4
Lexicon features	
Feature name	n-grams
NGRAMS_SNT_OPENER	**1** 2 3 **4**
NGRAMS_SNT_MPQA	1 2 3 4
NGRAMS_SNT_BL	**1** **2** **3** 4
NGRAMS_SNT_WITH_MODIFIER_MPQA	**1** **2** 3 4
NGRAMS_SNT_WITH_MODIFIER_BL	**1** 2 **3** 4
Feature name	boolean
COS_EXPLOSION_OPENER_PAISA	True
COS_EXPLOSION_OPENER_TWITTER	True
COS_EXPLOSION_MPQA_PAISA	True
COS_EXPLOSION_MPQA_TWITTER	True
COS_EXPLOSION_BL_PAISA	True
COS_EXPLOSION_BL_TWITTER	False
PMI_SCORE	True
SNT_DISTRIBUTION_OPENER	True
SNT_DISTRIBUTION_MPQA	True
SNT_MAJORITY_OPENER	False
SNT_MAJORITY_MPQA	True
SNT_MAJORITY_BL	False
SNT_POSITION_PRESENCE_OPENER	True
SNT_POSITION_PRESENCE_MPQA	True
SNT_POSITION_PRESENCE_BL	False

Table 1: All the features used for the global model. The features resulting from the features selection process are marked in bold or with the *True* label.

[7]For our experiments we set α to 0.25

Model	Avg. F–score	NEU	POS	NEG	POS_NEG
FeatSelLexicons	0.663	57.1	55.0	62.5	15.3
FeatSelNoLexicons	0.647	56.9	51.0	61.7	11.8
AllFeat	**0.667**	**58.4**	**56.3**	**63.4**	**16.4**

Table 2: Classification results of different feature models on official test data with respect to the four considered classes: Neutral (NEU), Positive (POS), Negative (NEG) and Positive-Negative (POS_NEG).

Model	Avg. F–score
FeatSelLexicons	**0.698**
AllFeat	0.678

Table 3: Classification results obtained by the five-fold cross validation evaluation before and after the feature selection (over the training set).

by the features selection process are not so relevant for this task. The results obtained by the *FeatSelLexicons* classifier show that lexicon features contribute (+1.6 points) to significantly improve the accuracy of our classifier.

3 Conclusion

In this paper, we reported the results of our participation to the Polarity Classification shared task. By resorting to a wide set of general–purpose features qualifying the lexical and grammatical structure of a text and ad hoc created lexicons, we achieved the second best score in the competition.

Current directions of research include adding to our models contextual features derived from contextual information of tweets (e.g. the user attitude, the overall set of recent tweets about a topic), successfully tested by (Croce et al., 2014).

References

Giuseppe Attardi, Felice Dell'Orletta, Maria Simi and Joseph Turian. 2009. Accurate dependency parsing with a stacked multilayer perceptron. In *Proceedings of Evalita '09, Evaluation of NLP and Speech Tools for Italian*. December, Reggio Emilia.

Valerio Basile, Andrea Bolioli, Malvina Nissim, Viviana Patti and Paolo Rosso. 2014. Overview of the Evalita 2014 SENTIment POLarity Classification Task. In *Proceedings of the 4th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'14)*. December, Pisa.

Chih-Chung Chang and Chih-Jen Lin. 2001. LIBSVM: a library for support vector machines *Software available at http://www.csie.ntu.edu.tw/cjlin/libsvm*.

Andrea Cimino, Felice Dell'Orletta, Giulia Venturi and Simonetta Montemagni. 2013. Linguistic Profiling based on General–purpose Features and Native Language Identification. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Application*, 207–215. Atlanta, Georgia. ACL.

Felice Dell'Orletta. 2009. Ensemble system for Part-of-Speech tagging. In *Proceedings of Evalita '09, Evaluation of NLP and Speech Tools for Italian*. December, Reggio Emilia.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '04*. 368-177, New York, NY, USA. ACM.

Verena Lyding, Egon Stemle, Claudia Borghetti, Macro Brunello, Sara Castagnoli, Felice Dell'Orletta, Henrik Dittmann, Alessandro Lenci and Vito Pirrelli. 2013. The *PAISÀ* Corpus of Italian Web Texts. In *Proceedings of 9th workshop on Web as Corpus (WAC-9)*. 26 April, Gothenburg, Sweden.

Isa Maks, Ruben Izquierdo, Francesca Frontini, Montse Cuadros, Rodrigo Agerri and Piek Vossen. 2014. Generating Polarity Lexicons with WordNet propagation in 5 languages. *9th LREC, Language Resources and Evaluation Conference*. Reykjavik, Iceland.

Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv1:1301.3781.

Saif Mohammad, Svetlana Kiritchenko and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the Seventh international workshop on Semantic Evaluation Exercises, SemEval-2013*. 321-327, Atlanta, Georgia, USA.

Andrea Vanzo, Danilo Croce and Roberto Basili. 2014. A context-based model for Sentiment Analysis in Twitter. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. August, Dublin, Ireland.

Theresa Wilson, Zornitsa Kozareva, Preslav Nakov, Sara Rosenthal, Veselin Stoyanov and Alan Ritter. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT '05*. 347-354, Stroudsburg, PA, USA. ACL.