A Kernel-based Approach for Irony and Sarcasm Detection in Italian

Andrea Santilli and **Danilo Croce** and **Roberto Basili**
Universitá degli Studi di Roma "Tor Vergata"
Via del Politecnico 1, Rome, 00133, Italy
andrea.santilli@live.it
{croce,basili}@info.uniroma2.it

Abstract

English. This paper describes the UNITOR system that participated to the Irony Detection in Italian Tweets task (IronITA) within the context of EvalIta 2018. The system corresponds to a cascade of Support Vector Machine classifiers. Specific features and kernel functions have been proposed to tackle the different subtasks: Irony Classification and Sarcasm Classification. The proposed system ranked first in the Sarcasm Detection subtask (out of 7 submissions), while it ranked sixth (out of 17 submissions) in the Irony Detection task.

Italiano. *Questo lavoro descrive il sistema UNITOR che è stato valutato nel corso dell' Irony Detection in Italian Tweets task IronITA ad EvalIta 2018. Il riconoscimento del sarcasmo e dell'ironia nei tweet corrisponde all'orchestrazione di diversi classificatori di tipo Support Vector Machine (SVM), studiata per risolvere i task legati alla competizione. Rappresentazioni specifiche sono state progettate per modellare i tweet attraverso la applicazione di funzioni kernel diverse utilizzate dai classificatori SVM. Il sistema ha ottenuto risultati promettenti risultando vincitore di 1 dei 2 task proposti.*

1 Introduction

Modern social networks allow users to express themselves, writing their opinions about facts, things and events. In social posting, people often adopt figurative languages, e.g. Irony and Sarcasm. These communication mechanism must be carefully considered in the automatic processing of texts in social media: as an example, they may be used to convey the opposite of literal meaning and thus just intentionally sketching a secondary or extended meaning (Grice, 1975). On Twitter, users can express themselves with very short messages. Given the short length, the information useful to detect figurative uses of natural language is very limited or missing. Irony and sarcasm detection represents challenging tasks within Sentiment Analysis and Opinion Mining often undermining the overall system accuracy. There is not a clear separation between irony and sarcasm, but the former is often considered to include the latter. In particular sarcasm is defined as sharp or cutting ironic expressions towards a particular target with the intention to offend (Joshi et al., 2016).

This paper presents and describes the UNITOR system participating in the *Irony Detection in Italian Tweets* (IronITA) task (Cignarella et al., 2018) within the EvalIta 2018 evaluation campaign. The system faces both the proposed subtasks within IronITA: *Irony Classification* and *Sarcasm Classification*. In a nutshell, the former subtask aims at evaluating the performance of a system in capturing whether a message is ironic or not. The second subtask is intended to verify if, given an ironic tweet, a system is able to detect sarcasm within the message.

The classification of each tweet is carried out by applying a cascade of kernel-based Support Vector Machines (Vapnik, 1998). In particular, two binary SVM classifiers (one per subtask) are designed to adopt specific combinations of different kernel functions, each operating over a task-specific tweet representation. This work extends the modeling proposed in (Castellucci et al., 2014) that was proved to be beneficial within the Irony Detection subtask within SENTIPOLC 2014. The UNITOR system here presented ranked 1^{st} and 2^{nd} in the Sarcasm Detection subtask, while it ranked 6^{th} and 7^{th} within the Irony Detection subtask.

In Section 2 the SVM classifiers, their features and the underlying kernels are described and the adopted workflow is presented. In Section 3 the performance measures of the system are reported, while Section 4 derives the conclusions.

2 System Description

The UNITOR system adopts a supervised learning setting where a multiple kernel-based approach is adopted to acquire two binary Support Vector Machine classifiers (Shawe-Taylor and Cristianini, 2004): a first classifier discriminates between ironic and non ironic tweets, while a second one decides whether an ironic tweet is sarcastic or not. In the rest of this section, we first summarize the pre-processing stage as well as the adopted linguistic resources (e.g. word embeddings or lexicons). Then, the feature modeling designed for the two steps is discussed.

2.1 Tweet processing and resources

Each tweet is linguistically processed through an adapted version of the Chaos parser (Basili and Zanzotto, 2002) in order to extract the information required for feature modeling, e.g. the Part-of-speech tags and lemmas of individual words. A normalization step is applied before the standard Natural Language Processing activity is carried out. A number of actions is performed: fully capitalized words are converted into their lowercase counterparts; hyperlinks are replaced by a special token, i.e. `LINK`; characters repeated more than three times are cleaned, as they increase lexical data sparseness (e.g. "*nooo!!!!!*" is converted into "*noo!!*"); all emoticons are replaced by special tokens[1].

In the feature modeling activities, we relied on several linguistic resources, hereafter reported.

First, we used a **Word Space model** (or Word Embedding) to generalize the lexical information of the (quite small) training material: this semantic space is obtained starting from a corpus of Italian tweets downloaded in July 2016 of about 10 millions of tweets (same used in Castellucci et al. (2016a)) and it is a 250-dimensional embedding generated according to a Skip-gram model (Mikolov et al., 2013)[2].

Moreover, we adopted a large scale sentiment specific lexicon, i.e., the **Distributional Polarity Lexicons** *(DPL)* (Castellucci et al., 2016b)[3]. Distributional Polarity Lexicon *(DPL)* is introduced to inject sentiment information of words in the learning process through a large-scale polarity lexicon that is automatically acquired according to the methodology proposed in (Castellucci et al., 2015). This method leverages on word embeddings to model lexical polarity by transferring it from entire sentences whose polarity is known. The process is based on the capability of word embeddings to represent both sentences and single words in the same space (Landauer and Dumais, 1997). First, sentences (here tweets) are labeled with some polarity classes: in (Castellucci et al., 2015) this labeling is achieved by applying simple heuristics, e.g. Distant Supervision (Go et al., 2009). 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 dimensions of the space that are 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. For more details, please refer to (Castellucci et al., 2015).

Finally, we also adopted an **Irony specific Corpus** to capture terms and patterns that are often used to express irony (e.g., "*non lo riconosceresti neanche se ti cascasse*" or "*... allora piove* "): it is a corpus composed by a set of Italian tweets automatically extracted using Distance Supervision (Go et al., 2009). In particular the Irony specific Corpus is composed by a set of 6,000 random tweets in Italian, freely available, assumed to be ironic, as they contain hashtags such as *#irony* or *#ironia*.

2.2 Modeling irony and sarcasm in kernel-based learning

UNITOR is based on kernel functions operating on vector representations of tweets, described hereafter. After the language processing stage, each tweet allows generating one of the follow-

[1] We normalized 113 well-known emoticons in 13 classes.

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

[3] The adopted lexicon has been downloaded from `http://sag.art.uniroma2.it/demo-software/distributional-polarity-lexicon/`

ing representations[4] , later exploited by the kernel-based SVM in the training/classification steps.

2.2.1 Irony-specific Features

The aim of this set of features is to capture irony by defining a set of irony-specific features inspired by the work of (Castellucci et al., 2014).

Word Space Vector (`WS`) is a 250-dimensional vector representation of the average semantic meaning of a tweet according to a Word space model. It is used to generalize the lexical information of tweets. We can summarize it as $\sum_{t \in T} We(t)/|T|$, where T is the set of nouns, verbs, adjectives, adverb and hashtag in a tweet t and $We(t)$ is a function that returns the 250-dimensional word embedding of the word t. Other words, such as articles and preposition are discarded as they do not convey useful information within a word space.

Irony Specific BOW (`ISBOW`) is a BoW vector representing the lexical information expressed in a message. The main difference with respect to a conventional BOW representation is the adopted weighting scheme. In fact, in this case we leverage on the Word Space previously described. For each dimension representing a lemma/part-of-speech pair, its weight is computed as the cosine similarity between the word embedding vector of the considered word and the vector obtained from the linear combination of all the other words in the message (*WS*)[5]. This vector aims at capturing how much odd is the occurrence of a given word in a sentence aiming at capturing its unconventional uses: it should be an indicator of potential ironic mechanisms, as suggested in (Castellucci et al., 2014).

Irony Specific BOW(Adjective, Noun, Verb) (`ISBOW-A`), (`ISBOW-S`), (`ISBOW-V`) are three BoW vectors that use the same weighting scheme specified in `ISBOW`. Each vector represents one individual part of speech (i.e. adjective, noun and verb), as words belonging to different POS-tag categories may be characterized by quite different distributions.

Irony Specific Mean and Variance (`ISMV`) is a 4-dimensional vector representation that summarized the information captured by the previous representations. It contains mean and variance of the cosine similarity, calculated between the words in a tweet in the `ISBOW` representation, and the maximum and minimum of the cosine similarity per tweet. This vector aims at summarizing the distribution and potential "spikes" of unusual patterns of use for words in a sentence.

Irony Specific Mean and Variance (Adjective, Noun, Verbs) (`ISMV-A`), (`ISMV-S`), (`ISMV-V`) are three distinct 4-dimensional vectors that are the same specified in `ISMV`, with the only difference that each representation works on one specific part of speech, respectively adjectives, nouns and verbs.

Char n-gram BOWs (`n-CHARS`) is a representation expressing the char n-grams contained in a message. We used 4 `n-CHARS` representations: `2-CHARS` BoW vector representing 2-char-ngrams contained in a message, `3-CHARS` BoW vector representing 3-char-ngrams, `4-CHARS` BoW vector representing 4-char-ngrams, `5-CHARS` BoW vector representing 5-char-ngrams. The aim of this representation is to capture the usage of specific textual patterns, e.g., *hihihihi* often used to express irony.

Synthetic Features (`SF`) is a 7-dimensional vector containing the following synthetic features, traditionally used in Sentiment Analysis: percentage of the number of uppercase letters in the tweet, number of exclamation marks, number of question marks, number of colons, number of semicolons, number of dots, number of commas. It has been inspired by works on irony detection of (Carvalho et al., 2009; Reyes et al., 2012).

2.2.2 Features based on Distribution Polarity Lexicons

The aim of this group of features is to exploit the negative evaluation towards a target typical of sarcasm mechanism (Joshi et al., 2016) using a polarity lexicon, here a Distribution Polarity Lexicon (DPL).

Distributional Polarity Lexicon Sum (`DSUM`) is a 15-dimensional vector representation made by the concatenation of 5 different representations, i.e. $\frac{1}{|N_T|}\sum_{w \in N_T} \underline{w}^p$, $\frac{1}{|V_T|}\sum_{w \in V_T} \underline{w}^p$, $\frac{1}{|Adj_T|}\sum_{w \in Adj_T} \underline{w}^p$, $\frac{1}{|Adv_T|}\sum_{w \in Adv_T} \underline{w}^p$, $\frac{1}{|T|}\sum_{w \in T} \underline{w}^p$, where N_T, V_T, Adj_T, Adv are the nouns, verbs, adjectives and adverbs occurring in the tweet,

[4]The code for the feature vector generation is available at: `https://github.com/andry9454/ironySarcasmDetection`

[5]If a word was not found in the word embedding, a smoothing weight, representing the mean cosine similarity between word and *WS* in the training set, is applied as cosine similarity measure.

$T = N_T \cup V_T \cup Adj_T \cup Adv_T$ and $\underline{w}^p$ expresses the 3-dimensional polarity lexicon entry[6] for the word w. This feature summarize the a-priori sentiment of words according to the different morphological categories. We speculate that the regularities or contrasts between these distributions may suggest the presence of irony or sarcasm.

Distributional Polarity Lexicon BoW (DBOW) is a BoW vector representing, for each word in a message, its polarity (positive, negative and neutral) as a three dimensional score derived from the DPL.

2.2.3 Irony Corpus Features

Generalizing linguistic information useful for Irony or Sarcasm detection is a very challenging tasks, as the adoption of these figurative languages mainly concern extra-linguistic phenomena. The idea underlying the following features is to define a tweet representation that is not directly connected to their (possibly limited) linguistic material, but that is connected with respect to a larger set of information derived from a Irony specific Corpus, i.e., a large scale collection of a Ironic tweets. This is used to extract an Irony specific Lexicon: a set of words and patterns occurring in such corpus with a high frequency.

Irony Corpus BOW (ICBOW) is a BoW vector representing lemmas of Nouns, Verbs, and Adjective in a message. Again, the main difference with respect to a conventional BoW representation is the adopted weighting scheme: a word is weighted 1.0 if that particular word was in the *Irony specific Corpus*, otherwise is weighted 0.

Irony Corpus weighted BOW (ICwBOW) is a BoW vector representing lemmas of Nouns, Verbs, and Adjective in a message. A word is weighted $log(f + 1)$ where f is the frequency of that particular word in the *Irony Corpus*.

Irony Corpus weighted Mean (ICM) is a 2-dimensional vector representation that summarize the mean words weight observed in a ICBOW representation and the mean over the ICwBOW. These scores indicate how a words or patterns in a tweet occur also in the Irony specific corpus. This information is very interesting as it is not tied to the lexical information from a tweet, so allowing a more robust generalization.

Irony Corpus BOW (bi-grams, three-grams) (IC2BOW), (IC3BOW) are two distinct BoW vector respectively representing bi-grams and three-grams of surface words in a message. The weighting scheme is the same explained in ICBOW.

Irony Corpus weighted BOW (bi-grams, three-grams) (IC2wBOW), (IC3wBOW) are two distinct BoW vectors respectively representing bi-grams and three-grams of terms in a message. The weighting scheme is the same explained in ICwBOW.

Irony Corpus weighted Mean (bi-grams, three-grams) (IC2M), (IC3M) are two distinct 2-dimensional vector representations that contain means that are the same specified in ICM, with the only difference that the first representation works on bi-grams (IC2BOW, IC2wBOW), while the second works on three-grams (IC3BOW, IC3wBOW).

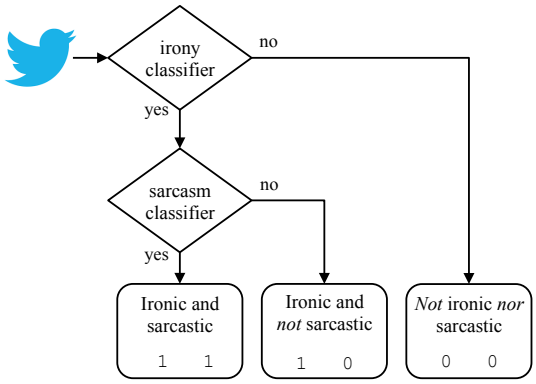

Figure 1: The UNITOR classifier workflow

3 Experimental evaluation and results

The cascade of SVM classifiers implemented in UNITOR is summarized in Figure 1. After the linguistic processing stage and the feature extraction stage, each tweet is classified by a binary classifier, the so-called *irony classifier*. If a message is judged as *not ironic*, we assume that it is also *not sarcastic* (according to the task guidelines) and a label 0 0 is assigned to it. Otherwise, if the tweet is judged as *ironic*, the second binary classifier, devoted to Sarcasm Detection, is invoked. If positive, the tweet is *sarcastic* and the message is labeled with 1 1, otherwise, 1 0.

Separated representations are considered in the *constrained* and *unconstrained* settings, according to the guidelines in (Cignarella et al., 2018). In the constrained setting only feature vectors using tweet information or public available lexicons are considered (*Irony-specific Features* and *Features derived from a DPL*). In the unconstrained

[6] If a word w is not present in the distributional polarity lexicon, $\underline{w}^p$ is set to the default $[0.33, 0.33, 0.33]$.

setting, feature vectors are derived also using the Irony specific Corpus.

In our experiments, we train the SVM classifiers using the same kernel combination for Irony Detection and Sarcasm Detection. Even if this is not a general solution (different tasks may require different representations) we adopted this greedy strategy, leaving the SVM to select the most discriminative information.

A normalized linear combination of specific kernel functions is used in both subtasks. In the linear combination, a specific linear kernel is applied to the following sparse representations: `ISBOW`, `ISBOW-A`, `ISBOW-S`, `ISBOW-V`, `DBOW`, `2BOW`, `3BOW`, `4BOW`, `5BOW`, `ICBOW`, `IC2BOW`, `IC3BOW`, `ICwBOW`, `IC2wBOW`, `IC3wBOW`; in the same combination a RBF kernel (Shawe-Taylor and Cristianini, 2004) is applied to the following dense representations `WS`, `SF`, `ICM`, `IC2M`, `IC3M`, `DSUM`, `ISMV`, `ISMV-A`, `ISMV-S`, `ISMV-V`[7].

Each SVM classifier is built by using the KeLP framework[8] (Filice et al., 2018).

Figure 1 reflects also the learning strategy that has been set up during the training phase: the Irony Classifier was trained on the complete training dataset composed by the entire training set (made of $3,977$ tweets) while the Sarcasm Classifier is trained only on the ironic tweets in the training dataset (made of $2,023$ tweets). A 10-fold cross validation strategy was applied to optimize the SVM parameters, while the linear combination of the kernel assigns the same weights to each kernel function.

In Table 1 the performances of the *Irony Classification* task are reported: in the constrained run the UNITOR system ranks 7^{th}, while in 6^{st} position in the unconstrained one. For this task the adopted representations were able to correctly determine whether a message is ironic with good precision. However, the winning system (about 3 points ahead) results more effective in the detection of non-ironic messages. In fact, according to the F1-score on the Ironic class, the system would have been ranked 2^{nd}. We also evaluated a slightly different modeling with two additional features vector, i.e., a classic BoW composed of lemmas derived from the input tweet, and a BoW of bigrams. These features have been excluded from our official submission to keep the model simple. However, these simple features would have been beneficial and the system would have ranked 2^{nd}. Performances on the *Sarcasm Classification* are in Table 2: UNITOR here ranks in 1^{st} or in 2^{nd} position, in the constrained and unconstrained run, respectively. Differences between the two results are not significant. Nevertheless the further features derived from the Irony specific corpus allow improving results (especially in terms of recall) in the Sarcasm Detection task. For this latter task, results achieved by UNITOR suggest that the proposed modeling, in particular the contribution of Polarity Features, seem to be beneficial. To prove it, we decided to evaluate a run with the same winning features, except Polarity Features. In this case the UNITOR system would have been ranked 4^{th}. These Polarity Features seem to exploit the negative bias typical of sarcasm (Joshi et al., 2016).

	Not Ironic			Ironic			Mean
	P	R	F1	P	R	F1	F1
1st	.785	.643	.707	.696	.823	.754	**.731**
2nd*	.771	.617	.686	.680	.816	.741	.714
6th(*u*)	.778	.577	.662	.662	.834	.739	.700
7th(*c*)	.764	.593	.668	.666	.816	.733	.700
BL	.501	1.00	.668	1.00	.000	.000	.334

Table 1: Constrained (*c*) and Unconstrained (*u*) UNITOR results in Irony Detection, i.e. scores 6th and 7th.

	Not Sarcastic			Sarcastic			Mean
	P	R	F1	P	R	F1	F1
1st(c)	.362	.584	.447	.492	.407	.446	**.520**
2nd(u)	.355	.553	.432	.469	.449	.459	.518
4th*	.344	.566	.428	.344	.566	.428	.508
BL	.296	.132	.183	1.00	.000	.000	.199

Table 2: Constrained (*c*) and Unconstrained (*u*) UNITOR results in Sarcasm Detection, i.e. 1st and 2nd scores

4 Conclusions

In this paper we described the UNITOR system participating to the IronITA task at EvalIta 2018. The system won 1 of the 2 evaluations carried out in the task, and in the worst case it ranked in the 6^{th} position. The good results in constrained and unconstrained settings suggest that the proposed irony and sarcasm specific features were beneficial to detect irony and sarcasm also in short messages. However, further work is needed to improve the *non ironic* F1 scores. The nature of the task seems to be non trivial also for a human reader, as some tweets extracted from

[7] A with $\gamma = 1$ was used in each RBF kernel

[8] `http://www.kelp-ml.org/`

the test set suggest: "*@beppe_grillo Beppe..tu sei un grande..questi si stanno finendo di mangiare l'Italia..*", "s*cusa hai ancora posti liberi nella app di braccialetti rossi*?"; here the interpretation of irony goes beyond the textual information and it is very difficult to state if these messages are ironic or not. Since tweets are very short, useful information for detecting irony is often out of the message, like this ironic tweet extracted from the test set may suggest: "*immagine perfetta ed esplicita che descrive la realtá della "buona scuola" a renzopoli*"; in this case the system may fail without a proper representation for the meaning of the neologism "*renzopoli*". So we think that the contextual approach suggested in (Vanzo et al., 2014) will be explored in future research.

References

Roberto Basili and Fabio Massimo Zanzotto. 2002. Parsing engineering and empirical robustness. *Nat. Lang. Eng.*, 8(3):97–120.

Paula Carvalho, Luís Sarmento, Mário J. Silva, and Eugénio de Oliveira. 2009. Clues for detecting irony in user-generated contents: Oh...!! it's "so easy" ;-). In *1st CIKM WS on Topic-sentiment Analysis for Mass Opinion*, pages 53–56. ACM.

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. 2015. Acquiring a large scale polarity lexicon through unsupervised distributional methods. In *Proc. of 20th NLDB*, volume 9103. Springer.

Giuseppe Castellucci, Danilo Croce, and Roberto Basili. 2016a. Context-aware convolutional neural networks for twitter sentiment analysis in italian. In *Proceedings of 3rd Italian Conference on Computational Linguistics (CLiC-it 2016) & Fifth EVALITA Workshop 2016), Napoli, Italy, December 5-7, 2016.*

Giuseppe Castellucci, Danilo Croce, and Roberto Basili. 2016b. A language independent method for generating large scale polarity lexicons. In *Proceedings of the 10th LREC Conference (LREC'16)*, Portoroz, Slovenia. European Language Resources Association (ELRA).

Alessandra Teresa Cignarella, Simona Frenda, Valerio Basile, Cristina Bosco, Viviana Patti, and Paolo Rosso. 2018. Overview of the evalita 2018 task on irony detection in italian tweets (ironita). In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*, Turin, Italy. CEUR.org.

Simone Filice, Giuseppe Castellucci, Giovanni Da San Martino, Alessandro Moschitti, Danilo Croce, and Roberto Basili. 2018. Kelp: a kernel-based learning platform. *Journal of Machine Learning Research*, 18(191):1–5.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *Processing*, pages 1–6.

H Paul Grice. 1975. Logic and conversation. *1975*, pages 41–58.

Aditya Joshi, Pushpak Bhattacharyya, and Mark James Carman. 2016. Automatic sarcasm detection: A survey. *CoRR*, abs/1602.03426.

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.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Antonio Reyes, Paolo Rosso, and Davide Buscaldi. 2012. From humor recognition to irony detection: The figurative language of social media. *Data and Knowledge Engineering*, 74(0):1 – 12.

John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Andrea Vanzo, Danilo Croce, and Roberto Basili. 2014. A context-based model for sentiment analysis in twitter. In *Proceedings of COLING*, pages 2345–2354. ACL and Dublin City University.

Vladimir N. Vapnik. 1998. *Statistical Learning Theory*. Wiley-Interscience.