UO_IRO: Linguistic informed deep-learning model for irony detection

Reynier Ortega-Bueno
Center for Pattern Recognition and
Data Mining, Santiago de Cuba, Cuba
reynier.ortega@cerpamid.co.cu
Computer Science Department,
University of Oriente
reynier@uo.edu.cu

José E. Medina Pagola
University of Informatics Sciences
Havana, Cuba
jmedinap@uci.cu

Abstract

English. This paper describes our UO_IRO system developed for participating in the shared task IronITA, organized within EVALITA: 2018 Workshop. Our approach is based on a deep learning model informed with linguistic knowledge. Specifically, a Convolutional (CNN) and Long Short Term Memory (LSTM) neural network are ensembled, also, the model is informed with linguistics information incorporated through its second to last hidden layer. Results achieved by our system are encouraged, however a more fine-tuned hyper-parameters setting is required for improving the model's effectiveness.

Italiano. *Questo articolo descrive il nostro sistema UO_IRO, sviluppato per la partecipazione allo shared task IronITA, presso EVALITA 2018. Il nostro approccio si basa su un modello di deep learning con conoscenza linguistica. In particolare: una Convolutional Neural Network (CNN) e una Long Short Term Memory Neural Network (LSTM). Inoltre, il modello è arricchito da conoscenza linguistica, incorporata nel penultimo hidden layer del modello. Sebbene sia necessario un miglioramento a grana fine dei parametri per migliorare le prestazioni del modello, i risultati ottenuti sono incoraggianti.*

1 Introduction

Computers interacting with humans through language, in natural way, continues to be one of the most salient challenge for Artificial Intelligent researchers and practitioners. Nowadays, several basic tasks related to natural language comprehension have been effectively resolved. Notwithstanding, slight advances have been archived by the machines when figurative devices and creativity are used in language with communicative purposes. Irony is a peculiar case of figurative devices frequently used in real life communication. As human beings, we appeal to irony for expressing in implicit way a meaning opposite to the literal sense of the utterance (Attardo, 2000; Wilson and Sperber, 1992). Thus, understanding irony requires a more complex set of cognitive and linguistics abilities than literal meaning. Due to its nature, irony has important implications in sentiment analysis and other related tasks, which aim at recognizing feelings and emotions from texts. Considering that, detecting irony automatically from textual messages is an important issue to enhance sentiment analysis and it is still an open research problem (Gupta and Yang, 2017; Maynard and Greenwood, 2014; Reyes et al., 2013).

In this work we address the fascinating problem of automatic irony detection in tweets written in Italian language. Particularly, we describe our irony detection system (UO_IRO) developed for participating in IronITA 2018: Irony Detection in Italian Tweets (Cignarella et al., 2018a). Our proposed model is based on a deep learning model informed with linguistic information. Specifically, a CNN and an attention based LSTM neural network are ensembled, moreover, the model is informed with linguistic information incorporated through its second to last hidden layer. We only participated in Task A (irony detection). For that, two constrained runs and two unconstrained runs were submitted. The official results shown that our system obtains interesting results. Our best run was ranked in 12^{th} position out of 17 submissions. The paper is organized as follows. In Section 2, we introduce our UO_IRO system for irony detection. Experimental results are subsequently

discussed in Section 3. Finally, in Section 4, we present our conclusions and attractive directions for future work.

2 UO_IRO system for irony detection

The motivation for this work comes from two directions. In a first place, the recent and promising results found by some authors (Deriu and Cieliebak, 2016; Cimino and Dell'Orletta, 2016; Gonález et al., 2018; Rangwani et al., 2018; Wu et al., 2018; Peng et al., 2018) in the use of convolutional networks and recursive networks, also the hybridization of them for dealing with figurative language. The second direction is motivated by the wide use of linguistic features manually encoded which have showed to be good indicators for discriminating among ironic and non ironic content (Reyes et al., 2012; Reyes and Rosso, 2014; Barbieri et al., 2014; Farías et al., 2016; Farías et al., 2018).

Our proposal learns a representation of the tweets in three ways. In this sense, we propose to learn a representation based on a recursive network with the purpose of capturing long dependencies among terms in the tweets. Moreover, a representation based on convolutional network is considered, it tries to encode local and partial relation between words which are near among themselves. The last representation is based on linguistic features which are calculated for the tweets. After that, all linguistic features previously computed are concatenated in a one-dimensional vector and it is passed through a dense hidden layer which encodes the linguistic knowledge and includes this information to the model.

Finally, the three neural network based outputs are combined in a merge layer. The integrated representations is passed to a dense hidden layer and the final classification is performed by the output layer, which use a softmax as activation function for predicting ironic or not ironic labels. For training the complete model we use categorical cross-entropy as loss function and the Adam method (Kingma and Ba, 2014) as the optimizer, also, we use a batch size of 64 and training the model for 20 epochs. Our proposal was implemented using the Keras Framework[1]. The architecture of the UO_IRO is shown in Figure 1 and described below.

[1]https://keras.io/

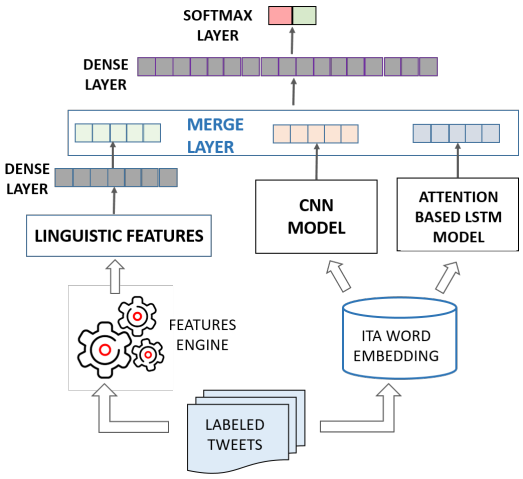

Figure 1: Overall Architecture of UO_IRO: Irony Detection System.

2.1 Preprocessing

In the preprocessing step, the tweets are cleaned. Firstly, the emoticons, urls, hashtags, mentions and twitter-specific tokens (RT for retweet and FAV for favorite) are recognized and replaced by a corresponding wild-card which encodes the meaning of these special words. Afterwards, tweets are morphologically analyzed by FreeLing (Padró and Stanilovsky, 2012). In this way, for each resulting token, its lemma is assigned. Then, the words in the tweets are represented as vectors using a word embedding model. In this work we use the Italian pre-trained vectors[2] public available (Bojanowski et al., 2017).

2.2 Attention Based LSTM

We use a model that consists in a Bidirectional LSTM neural network (Bi-LSTM) at the word level. Each time step t, the Bi-LSTM gets as input a word vector w_t with syntactic and semantic information known as word embedding. The idea behind this Bi-LSTM is to capture long-range and backwards dependencies in the tweets. Afterward, an attention layer is applied over each hidden state h_t. The attention weights are learned using the concatenation of the current hidden state h_t of the Bi-LSTM and the past hidden state s_{t-1}. The goal of this layer is then to derive a context vector c_t that captures relevant information for feeding it as input to the next level. Finally, a LSTM layer is stacked at the top. This network at each time step receives the context vector c_t which is propagated

[2]https://s3-us-west-1.amazonaws.com/fasttext-vectors/wiki.it.zip

until the final hidden state s_{T_x}. This vector (s_{T_x}) can be considered as a high level representation of the tweet. For more details, please see (Ortega-Bueno et al., 2018).

2.3 Convolutional Neural Network

We use a CNN model that consists in 3 pairs of convolutional layers and pooling layers in this architecture. Filters of size three, four and five were defined for the convolutional layers. In case of pooling layer, the maxpooling strategy was used. We also use the Rectified Linear Unit (ReLU), Normalization and Dropout methods to improve the accuracy and generalizability of the model.

2.4 Linguistic Features

In our work, we explored some linguistic features useful for irony detection in texts which can be grouped in three main categories: Stylistic, Structural and Content, and Polarity Contrast. We define a set of features distributed as follows:

Stylistic Features

- *Length*: Three different features were considered: number of words, number of characters, and the means of the length of the words in the tweet.
- *Hashtags*: The amount of hashtags.
- *Urls*: The number of url.
- *Emoticons*: The number of emoticons.
- *Exclamations*: Occurrences of exclamation marks.
- *Emphasized Words*: Four different features were considered: word emphasized through repetition, capitalization, character flooding and exclamation marks.
- *Punctuation Marks*: The frequency of dots, commas, semicolons, and question marks.
- *Quotations*: The number of expressions between quotation marks.

Structural and Content Features

- *Antonyms*: This feature considers the number of pairs of antonyms existing in the tweet. WordNet (Miller, 1995) antonym relation was used for that.
- *Lexical Ambiguity*: Three different features were considered using WordNet: the first one is the mean of the number of synsets of each word. The second one is the greatest number of synsets that has a single word. The last is the difference between the number of synsets of the word with major number of synsets and the average number of synsets.
- *Domain Ambiguity*: Three different features were considered using WordNet: the first one is the mean of the number of domains of each word. The second one is the greatest number of domains that a single word has in the tweet. The last one is the difference between the number of domains of the word with major number of domains and the average number of domains. It is important to clarify that the resources WordNet Domains[3] and SUMO[4] were separately used.
- *Persons*: This feature tries to capture verbs conjugated in the first, second, third person and nouns and adjectives which agree with such conjugations.
- *Tenses*: This feature tries to capture the different verbal tenses used in the tweet.
- *Questions-answers*: Occurrences of questions and answers pattern in the tweet.
- *Part of Speech*: The number of nouns, verbs, adverbs and adjectives in the tweet are quantified.
- *Negation*: The amount of negation words.

Polarity Contrast Features

With the purpose of capturing some types of explicit polarity contrast we consider the set of features proposed in (Peña et al., 2018). The Italian polarity lexicon (Basile and Nissim, 2013) was used to determine the contrast between different parts of the tweet.

- *WordPolarityContrast*: It is the polarity difference between the most positive and the most negative word in the tweet. This feature, also consider the distance, in terms of tokens, between the words.
- *EmotiTextPolarityContrast*: It is the polarity contrast between the emoticons and the words in the tweet.
- *AntecedentConsequentPolarityContrast*: This considers the polarity contrast between two parts of the tweet, when it is split by a delimiter. In this case, adverbs and punctuation marks were used as delimiters.
- *MeanPolarityPhrase*: It is the mean of the polarities of the words that belong to quotes.
- *PolarityStandardDeviation*: It is the standard deviation of the polarities of the words that

[3] http://wndomains.fbk.eu/hierarchy.html
[4] http://www.adampease.org/OP/

belong to quotes.

- *PresentPastPolarityContrast*: It computes the polarity contrast between the parts of the tweet written in present and past tense.
- *SkipGramPolarityRate*:It computes the rate among skip-grams with polarity contrast and all valid skip-grams. The valid skip-grams are those composed by two words (nouns, adjectives, verbs, adverbs) with skip=1. The skip-grams with polarity opposition are those that match with the patterns *positive-negative*, *positive-neutral*, *negative-neutral*, and vise versa.
- *CapitalLetterTextPolarityContrast*: It computes the polarity contrast between capitalized words and the rest of the words in the tweets.

3 Experiments and Results

In this section we show the results of the proposed model in the shared task of "Irony Detection" and discuss them. In a first experiment we analyze the performance of four variants of our model using 10 fold cross-validation strategy on the training set. Also, each variant was running in unconstrained and constrained setting, respectively. In Table 1, we summarize the obtained results in terms of F1 measure macro averaged (F1-AVG). Specifically, we rely on the macro for preventing systems biased towards the most populated classes.

Table 1: Results obtained by UO_IRO on the training set by using 10-fold cross-validation.

Run	**Model**	**AVG-F_1**
	Constrained	
run1-c	CNN-LSTM	0.7019
run2-c	CNN-LSTM-SVM	0.6927
run3-c	*CNN-LSTM-LING*	**0.7124**
run4-c	CNN-LSTM-LING-SVM	0.7040
	Unconstrained	
run1-u	CNN-LSTM	0.7860
run2-u	CNN-LSTM-SVM	0.7900
run3-u	*CNN-LSTM-LING*	**0.8226**
run4-u	CNN-LSTM-LING-SVM	0.8207

For the run1-c and run1-u (CNN-LSTM) we only combine the representation obtained by the attention based LSTM model with the CNN model, in these runs, no linguistic knowledge was considered. Run2-c and run2-u (CNN-LSTM-SVM) are a modification of the CNN-LSTM model, in this case we change the softmax layer at the output of the model and use a Linear Support Vector Machine (SVM) with default parameters as final classifier. Run3-c and run3-u (CNN-LSTM-LING) represent the original introduced model without any variations. Finally, for run4-c and run4-u (CNN-LSTM-LING-SVM) we change the softmax layer by a linear SVM as final classifier. For unconstrained runs, we include the ironic tweets provided by the corpus Twittirò (Cignarella et al., 2018b), to the official training set releases by the IronITA organizers.

Analyzing Table 1, several observations can be made. Firstly, unconstrained runs achieved better results than constrained ones. These results reveal that introducing more ironic examples improves the performance of the UO_IRO. Secondly, the results achieved with the variants that consider the linguistic knowledge (run3-c, run4-c, run3-u and run4-u) obtain an increase in the effectiveness. With respect to the strategy used for the final classification of the tweets, generally, those variants that use SVM obtain a slight drop in the AVG-F_1.

Regarding the official results, we submitted four runs, two for constrained setting (RUN1-c and RUN2-c) and two for unconstrained setting (RUN3-u and RUN4-u). For the unconstrained variants of the UO_IRO, the tweets provided by the corpus Twittirò were also used with the training set. Taking into account the results of the Table 1 we select to CNN-LSTM-LING (RUN1-c and RUN3-u) and CNN-LSTM-LING-SVM (RUN2-c and RUN4-u) as the most promising variants of the model for evaluating in the official test set.

As can be observed in Table 2, our four runs were ranked 12^{th}, 13^{th}, 14^{th} and 15^{th} from a total of 17 submissions. The unconstrained variants of the UO_IRO achieved better results than constrained ones. Contrary to the results shown in the Table 1, the runs that use SVM as final classification strategy (RUN2-c and RUN4-u) were better ranked than the other ones. We think that this behavior may be caused by softmax classifiers (last layer of the UO_IRO), those are more sensitive to the over-fitting problem than Support Vector Machines. Notice that, in all cases our model surpass the two baseline methods established by the organizers.

Table 2: Official results for the Irony Detection subtask.

Rank	Runs	F_1-I	F_1-noI	Avg-F_1
12/17	*RUN4-u*	**0.700**	**0.603**	**0.651**
13/17	RUN3-u	0.665	0.626	0.646
14/17	RUN2-c	0.678	0.579	0.629
15/17	RUN1-c	0.577	0.652	0.614

4 Conclusions

In this paper we presented the UO_IRO system for the task of Irony Detection in Italian Tweets (IronITA) at EVALITA 2018. We participated in the "Irony classification' subtask and our best submission ranked 12^{nd} out of 17. Our proposal combines attention-based Long Short-Term Memory Network, Convolutional Neural Network, and linguistics information which is incorporated through the second to last hidden layer of the model. The results shown that the consideration of linguistic features in combination with the deep representation learned by the neural network models obtain better effectiveness based on F1-measure. Results achieved by our system are interesting, however a more fine-tuned hyper-parameters setting is required for improving the model's effectiveness. We think that including the linguistic features of irony into the firsts layers of the model could be a way to increase the effectiveness. We would like to explore this approach in the future work. Also, we plan to analyze how affective information flows through the tweets, and how it impacts on the irony realization.

References

Salvatore Attardo. 2000. Irony as relevant inappropriateness. *Journal of Pragmatics*, 32(6):793–826.

Francesco Barbieri, Horacio Saggion, and Francesco Ronzano. 2014. Modelling Sarcasm in Twitter, a Novel Approach. In *Proceedings ofthe 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 136–141.

Valerio Basile and Malvina Nissim. 2013. Sentiment analysis on Italian tweets. In *4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the ACL.*, 5:135–146.

Alessandra Cignarella, Frenda Simona, Basile Valerio, Bosco Cristina, Patti Viviana, and Rosso Paolo. 2018a. 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 Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, Turin, Italy. CEUR.org.

Alessandra Teresa Cignarella, Cristina Bosco, Viviana Patti, and Mirko Lai. 2018b. Application and Analysis of a Multi-layered Scheme for Irony on the Italian Twitter Corpus TWITTIRÓ. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 4204–4211.

Andrea Cimino and Felice Dell'Orletta. 2016. Tandem LSTM-SVM Approach for Sentiment Analysis. In *CLiC-it/EVALITA. 2016*, pages 1–6. CEUR-WS.org.

Jan Deriu and Mark Cieliebak. 2016. Sentiment Analysis using Convolutional Neural Networks with Multi-Task Training and Distant Supervision on Italian Tweets. In *CLiC-it/EVALITA. 2016*, pages 1–5. CEUR-WS.org.

Delia Irazú Hernańdez Farías, Viviana Patti, and Paolo Rosso. 2016. Irony Detection in Twitter. *ACM Transactions on Internet Technology*, 16(3):1–24.

Delia-Irazú Hernádez Farías, Viviana Patti, and Paolo Rosso. 2018. ValenTO at SemEval-2018 Task 3 : Exploring the Role of Affective Content for Detecting Irony in English Tweets. In *Proceedings ofthe 12th International Workshop on Semantic Evaluation (SemEval-2018)*, pages 643–648. Association for Computational Linguistics.

José Angel Gonález, Lluís-F. Hurtado, and Ferran Pla. 2018. ELiRF-UPV at SemEval-2018 Tasks 1 and 3 : Affect and Irony Detection in Tweets. In *Proceedings ofthe 12th International Workshop on Semantic Evaluation (SemEval-2018)*, pages 565–569.

Raj Kumar Gupta and Yinping Yang. 2017. CrystalNest at SemEval-2017 Task 4 : Using Sarcasm Detection for Enhancing Sentiment Classification and Quantification. In *Proceedings ofthe 11th International Workshop on Semantic Evaluations (SemEval-2017)*, pages 626–633, Vancouver, Canada. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Diana Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets ? Investigating the impact of sarcasm on sentiment analysis. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. European Language Resources Association.

George A Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Reynier Ortega-Bueno, Carlos E Mu, and Paolo Rosso. 2018. UO_UPV : Deep Linguistic Humor Detection in Spanish Social Media. In *Proceedings of the Third Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018)*, pages 1–11.

Lluís Padró and Evgeny Stanilovsky. 2012. FreeLing 3.0: Towards Wider Multilinguality. In *Proceedings of the (LREC 2012)*.

Anakarla Sotolongo Peña, Leticia Arco García, and Adrián Rodríguez Dosina. 2018. Detección de ironía en textos cortos enfocada a la minería de opinión. In *IV Conferencia Internacional en Ciencias Computacionales e Informáticas (CICCI' 2018)*, number 1-10, Havana, Cuba.

Bo Peng, Jin Wang, and Xuejie Zhang. 2018. YNU-HPCC at SemEval-2018 Task 3 : Ensemble Neural Network Models for Irony Detection on Twitter. In *622 Proceedings ofthe 12th International Workshop on Semantic Evaluation (SemEval-2018)*, pages 622–627. Association for Computational Linguistics.

Harsh Rangwani, Devang Kulshreshtha, and Anil Kumar Singh. 2018. NLPRL-IITBHU at SemEval-2018 Task 3 : Combining Linguistic Features and Emoji Pre-trained CNN for Irony Detection in Tweets. In *Proceedings ofthe 12th International Workshop on Semantic Evaluation (SemEval-2018)*, pages 638–642. Association for Computational Linguistics.

Antonio Reyes and Paolo Rosso. 2014. On the difficulty of automatically detecting irony: beyond a simple case of negation. *Knowledge and Information Systems*, 40(3):595–614.

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

Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in Twitter. *Language Resources and Evaluation*, 47(1):239–268.

Deirdre Wilson and Dan Sperber. 1992. On verbal irony. *Lingua*, 87(1):53–76.

Chuhan Wu, Fangzhao Wu, Sixing Wu, Junxin Liu, Zhigang Yuan, and Yongfeng Huang. 2018. THU NGN at SemEval-2018 Task 3 : Tweet Irony Detection with Densely Connected LSTM and Multi-task Learning. In *Proceedings ofthe 12th International Workshop on Semantic Evaluation (SemEval-2018)*, pages 51–56. Association for Computational Linguistics.