Towards A Deep-Learning-Based Methodology for Supporting Satire Detection

Alfredo Cuzzocrea*
IDEA Lab, University of Calabria, Rende, Italy & LORIA, Nancy, France
alfredo.cuzzocrea@unical.it

Giosuè Lo Bosco
Dept of Computer Science, University of Palermo & IEMEST, Palermo, Italy
giosue.lobosco@unipa.it

Mariano Maiorana
Dept of Computer Science, University of Palermo, Palermo, Italy
mariano.maiorana@community.unipa.it

Giovanni Pilato
ICAR-CNR, Palermo, Italy
giovanni.pilato@cnr.it

Daniele Schicchi
Istituto Euro-Mediterraneo di Scienza e Tecnologia, Palermo, Italy
danieleschicchi@iemest.eu

Abstract

This paper describes an approach for supporting automatic satire detection through effective deep learning (DL) architecture that has been shown to be useful for addressing sarcasm/irony detection problems. We both trained and tested the system exploiting articles derived from two important satiric blogs, Lercio and IlFattoQuotidiano, and significant Italian newspapers.

1 Introduction

Satire is a way of criticizing people (or ideas) by ridiculing them on political, social, and morals topics (e.g., [18]). Most of the time, such a language form is utilized to influence people’s opinions. It is a figurative form of language that leverages comedic devices such as parody (i.e. to imitate techniques and style of some person, place or thing), exaggeration (i.e. to represent something beyond normality make it ridiculous), incongruity (i.e. to present things that are absurd concerning the context), reversal (i.e. to present the opposite of normal order), irony/sarcasm (i.e. to say something that is the opposite of what a person mean). Moreover, satire masks emotions like irritation and disappointment by using ironic content.

The easy way of denouncing political and societal problems exploiting humor has brought consensus to satire that has been widely accepted. It leads people to constructive social criticism, to participate actively in the socio-political life, representing a sign of democracy. Unfortunately, the ironic nature of satire tends to mislead subjects that can believe the humorous news as they were real; therefore, satirical news can be deceptive and harmful.

Detecting satire is one of the most challenging computational linguistics tasks, natural language processing, and social multimedia sentiment analysis. It differs from irony detection since satire mocks something or someone, while irony is intended to be a way for causing laughter. Tackling such a task means both to pinpoint linguistic entities that characterize satire and look at how they are used to express a more complex meaning.

As satirical texts include figurative communication for expressing ideas/opinions concerning people, sentiment analysis systems may be negatively affected. In this case, satire should be adequately addressed to avoid performances degradation of such systems, mainly if sar-
casm/irony is used [1]. Moreover, reliably detecting satire can benefit many other research areas where figurative language usage can be a problem, such as Affective Computing [15]. An autonomous way of detecting satire might help computers interpret human interaction and notice its emotional state, improving the human-computer experience.

In this paper, we tackle automatic satire detection through effective deep learning (DL) architecture that has been shown to be effective for addressing the sarcasm/irony detection problem. The Neural Network (NN) exploits articles derived from two important satiric blogs, Lercio and Il-FattoQuotidiano, and major Italian newspapers. The dataset has been specifically created for the task, and it includes news concerning similar topics. Experiments show an optimal performance achieved by the network that is capable of performing well on satire recognition. The network demonstrates the ability to detect satire in a context where it is not marked as in IlFattoQuotidiano. In fact, in this special case, news are so realistic that they seem to be true. [15]. An autonomous way of detecting satire might help computers interpret human interaction and notice its emotional state, improving the human-computer experience.

On the other hand, studying these techniques as combined with the emerging big data trend (e.g., [11, 10, 6, 8, 9]) is an interesting challenge.

The extended version of this paper appears in [7].

2 The Overall Proposed Methodology

Recognizing satire can be modeled as a classification task subdividing satiric and non-satiric articles in two different classes. Such a task has been widely tackled by using machine learning algorithms, and it has been shown that it is important to consider various aspects related to the application domain. For what concerns the subject problem, many factors should be taken into account: the way the text is represented and how it is structured (sec. 2.1), the model’s architecture for tackling the task and its tuning (sec 2.2 and 2.3). Le Hoang Son et al. [13] have introduced a deep learning model that promises optimal performances for detecting sarcasm/irony. We believe that such a network can also help recognizing the main aspects of the satire; a detailed description is given in sec. 2.2.

2.1 Preprocessing

The preprocessing phase deals with the input arrangement to make it analyzable to the model as best as possible. Most of the time, the text is changed by removing punctuation marks, stop-words, etc. In this case, since the articles have been harvested from online resources we focused on the removal of the author’s name, HTML tags, hyperlinks, and hashtags. Subsequently, the input text is split into tokens (i.e., words and punctuation marks) using NLTK\(^1\). To level out the lengths of the articles, we have analyzed the cumulative frequency of the length of the texts, and then we have selected a value \(L = 4500\) words such that we considered 95% of the entire set of articles. Finally, each token is mapped to a 300-dimensional space by a pre-trained embedding tool that relies on FastText [3, 12]. Therefore, each article is represented by a matrix of real values of size \((L, 300)\). We crop texts longer than \(L\), and we pad with 0s texts that are shorter.

2.2 Architecture

The network’s architecture is inspired from the one presented by Le Hoang Son et al [13], that exploits Bidirectional Long Short Term Memory (BiLSTM), Soft Attention Mechanism, Convolutional NNs, and Fully Connected NNs. Moreover, such a model consider five different auxiliary characteristics that have been shown to be relevant to sarcasm/irony detection: number of exclamation marks (!), number of question marks (?), number of periods (.), number of capital letters, number of uses of or. A complete model representation is given in figure 1.

2.2.1 Input Layer

The first network’s layer is the Input layer which manage the pre-processed text in order to allow the analysis by the BiLSTM.

2.2.2 BiLSTM Layer

BiLSTM is composed of two LSTM layers which examine respectively the input sequence in forward (from the first token \(x_0\) to the last one \(x_T\)) and backward (from the last token \(x_T\) to the first one \(x_0\)) ways. LSTM cell, is a neural unit created specifically for overcoming the vanish/exploding gradient problem [2] that affects the training phase by using the backpropagation through time algorithm. The cell is composed of a set of gates (i.e input, forget, and output gate) which control the flow of information. The forget gate deals with choosing the information part should be kept and what should be gotten rid, the input gate proposes new information that is worth to be considered, and the output gate mix the contributes given by both the input and forget gates for creating the final cell’s output. LSTM cell leverages two feedback loops (i.e internal and external) which allow to track the sequence of elements the cell has already analyzed through a sequence of internal states \(h_1, \ldots, h_T\). The final output of the LSTM cell is its final internal state that is strictly dependent of the previous ones. The formulation of

\(^1\)www.nltk.org
2.2.3 Soft Attention Layer

The Soft Attention is a mechanism that weight the input sequence elements on the basis of their relevance for the classification task, allowing the network to capture the salient words considering the input text totally.

The analysis of the input text in these two opposite directions creates two representation of the input sequence: straight and reversed. BiLSTM layer merges the output of the two LSTM layers into a single output by concatenating them. The final vector, if examined through the soft attention, allows the network to capture the salient words considering the input text totally.

2.2.4 Convolutional Layer

We stacked three convolutional layers for the feature learning. Each convolving filter of size \(s\) slides over the input vector to compute a localized feature vector \(v_j\) for each possible word through a nonlinear activation function. For each filter, a transition matrix \(T\) is generated. Such a matrix is iteratively applied to a part of the input vector to compute the features as following:

\[
v_j = f \left( \langle T, F_{g,l} \rangle + b_a \right)
\]

where \(\langle \cdot, \cdot \rangle\) is the inner product, \(F_{g,l}\) is the part of the input vector which includes elements from position \(g\) to position \(l\), \(b_a\) is a bias related to the specific filter, and \(f\) is a non-linear function.

The output of the convolutional layers is a vector of features \(v = v_1, v_2, \ldots, v_{n-s+1}\) where \(n\) is the length of the input vector.

A max-pooling layer then processes the convolutional layer’s output. Such a layer extracts the largest computed feature for each filter, considering only the most relevant ones. The output layer then analyzes the output vector that included the selected features.

2.2.5 Output Layer

The output layer is a Fully Connected NN activated by Softmax. Such a layer takes as input the features extracted by the max-pooling layer. Employing the Softmax activation function computes the probability that the input text belongs to the either satiric or non-satiric class.

2.3 Parameters

Hyperparameters have been chosen empirically and taking inspiration from [13, 1]. Different tries have shown that taking a small learning rate and using a small minibatch coupled with Dropout regularization factors helps the network improve its performance by diminishing the loss. A complete list of them can be found in table 1.

3 Conclusion and Future Works

Satire aims at criticizing either something or someone leveraging on comedic devices. Its automatically detec-
A novel approach for supporting Italian satire detection through deep learning.

In this paper, we have introduced a powerful DL model that tackles the satire detection problem by examining lexical, syntactical, and auxiliary features. To support the analysis by the system, we exploited an effective pre-trained embedding tool based on FastText.

Future work will further analyze the network’s behavior by exploiting incremental data [5] and clustering [4]. Moreover, we are going to study how satire might affect the text comprehension [16] and if it might be reproduced through automatic creative processes [17].

References


