Computational Detection of Irony in Textual Messages

Filipe Silva
Instituto Superior Técnico, Universidade de Lisboa
Filipe.Pires.Silva@tecnico.ulisboa.pt

Abstract

Analyzing user generated content in social media, using natural language processing methods, involves facing problematic rhetorical devices, such as sarcasm and irony. These devices can cause wrong responses from review summarization systems, sentiment classifiers, review ranking systems, or any kind of application dealing with the semantics and the pragmatics of text. Studies to improve the task of detecting textual irony have mostly focused mostly on simple linguistic cues, intrinsic to the text itself, although these have been insufficient in detecting ironic sentences that have no intrinsic clues. However, a new approach for classifying text, which can be made to explore external information (e.g. in the form of pre-trained word embeddings) has been emerging, making use of neural networks with multiple layers of data processing. This dissertation focuses on experimenting with different neural network architectures for addressing the problem of detecting irony in text, reporting on experiments with different datasets, from different domains. The results from the experiments show that neural networks are able to outperform standard machine learning classifiers on the task of irony detection, on different domains and particularly when using different combinations of word embeddings.

1 Introduction

The field of Natural Language Processing (NLP) has been addressing the development of methods for analyzing social media data, for instance in order to support review summarization systems, sentiment classifiers, and review ranking systems.

One of the problems that has been recurring in these NLP applications is the existence of sarcasm and irony (i.e., the use of words to convey a meaning that is the opposite of its literal meaning) in user comments. For example, ironic utterances might cause the NLP applications to misinterpret a given comment as a positive review when, in actuality, it should have been classified as a negative review, since the user comment was intended to be interpreted ironically. With this in mind, the work developed in the context of my MSc thesis relates to the computational detection of irony and sarcasm in text.

The most common approach on dealing with detecting irony in text involves using feature-based machine learning classifiers, which leverage from a large set of features that might indicate if a given text contains irony. However, this approach requires the design of dozens of features (Reyes et al., 2013; Rajadesingan et al., 2015) which have been becoming more and more complex (specially classifiers that leverage from external contextual information (Karoui et al., 2015; Bamman and Smith, 2015)). However, a new trend on the classification of textual documents consists of using classifiers based on deep neural networks (e.g., Amir et al. (2016) or Ghosh and Veale (2016)). These algorithms, that contrast with feature-based machine learning algorithms that require a heavy use of feature engineering, have been achieving similar or better results without the demanding task of implementing a large feature set. Instead, they rely on word embeddings to give semantic or/and contextual information of the documents being classified.

The work described in this dissertation consists of multiple experiments using deep learning algorithms, with different model architectures, which are then compared with machine learning algorithms based of feature engineering, similar to those that have been used on previous studies on the task of irony detection in textual messages.

The experiments performed in this research include datasets from different domains (discussion forums, a social media service and news headlines) and uses different architectures of neural networks leveraging from multiple embeddings to classify the texts. On this work we use several different deep learning architectures, from convolutional to recurrent neural networks, and also an hybrid of the two architectures.

The contributions that can be derived from the

results of the research performed on this dissertation include: (*i*) new approaches for classifying texts based on the presence of irony, together with their experimental evaluation; (*ii*) a neural network architecture, combining convolutional and recurrent layers, that performs well on multiple datasets from different domains; (*iii*) a way to increase the performance of neural networks on the task of irony detection using multiple word embeddings in combination.

2 Related Work

Some previous studies have already addressed the automated detection of sarcasm and irony. However, most have focused on semantic or lexical cues directly available from the source texts, often within relatively simple models, without properly exploring external context. Examples of previous work include: studies concerned with the identification of ironic cues by matching text with expressions usually associated with ironic speech, such as *yeah right* (Tepperman et al., 2006; Carvalho et al., 2009), as well as through the presence of profanity and slang (Burfoot and Baldwin, 2009); studies concerned with identifying words and phrases that contrast positive sentiments and negative situations, which usually denote irony (Riloff et al., 2013); studies concerned with finding simple linguistic clues associated to specific pragmatic factors often associated with irony (González-Ibáñez et al., 2011);

A classification algorithm that also has the information about the utterer of the ironic speech, and information about the subject of the utterance, will also help detect the nuances of an ironic expression. Studies that try to benefit from these types of information have also been experimented with, for example, using relevant newspapers and Wikipedia (Karoui et al., 2015), or past user texts (Bamman and Smith, 2015).

A new trend on the classification of textual documents consists of using classifiers based on deep neural networks (i.e., a feedforward neural network with *many* layers) and pre-trained word embeddings (e.g., Amir et al. (2016) or Ghosh and Veale (2016)). These algorithms, which contrast with machine learning techniques requiring a heavy use of feature engineering (i.e., the process of transforming data into features that help the performance of the classification approach being used), have been achieving similar or better results without the demanding task of implementing a large feature set. Instead, they rely on word embeddings (e.g., representations for words based on dense vectors, which are effective at capturing semantic similarity) to give semantic or/and contextual information about the documents being classified (Mikolov et al., 2013a).

3 Design of the Neural Networks

The neural networks architectures used in the experiments included two variants of Recurrent Neural Networks (RNNs), namely a stack of two Long Short-Term Memory (LSTM) layers and a bidirectional LSTM. A Convolutional Neural Network (CNN) architecture and a hybrid CNN-LSTM architecture were also used. These architectures were chosen due to the good results from studies that use them in similar tasks (Kim, 2014; Zhou et al., 2015; Ghosh and Veale, 2016).

All of the architectures start with an input layer where the texts are represented as vectors of the word identifiers, followed by an embedding layer. The architectures also had multiple dropout layers with increasingly smaller dropout rates, to try addressing overfitting problems with the different datasets. However, the tests revealed that the dropouts had almost no effect on the results.

The LSTM layers and the final dense layers used a *sigmoid* activation function, which is a function defined for all real input values that has a positive derivative at each point. The convolutional layers used a rectified linear activation function (Nair and Hinton, 2010).

3.1 Stack of Two LSTMs

The stack of two LSTM layers is a simple sequential model where, following the input and embedding layers, we have a pair of two LSTM layers which processes the input sequence normally (i.e., in the order it is received), and ending with a dense layer which sets the output into the target output shape (i.e., a single value where 0 represents a not-ironic prediction and 1 an ironic prediction).

3.2 Bidirectional LSTM Layers

The bidirectional LSTM is an architecture running multiple layers in parallel. The model starts with the input layer and an embedding layer, which is then split into two pairs of LSTM layers running in parallel, where one of the pairs processes the input from left-to-right and the other pair processes the input from right-to-left. The output of the LSTM layers is then merged into a dense layer to obtain the correct output shape.

3.3 Convolutional Neural Network

The CNN architecture also runs in parallel, considering multiple parallel convolutional layers with different filter windows. Several combination of filter windows were tested but the results did not change enough for this to be relevant. Because of this, the experiments reported on this dissertation were made with the combination of filters with length 3, 5, and 7. After each convolutional layer the data goes through a max pooling layer and then had the output of the max pool is flattened into a

shape that can be accepted by the final dense layer (i.e., we go from a three dimensional matrix to a two dimensional matrix).

3.4 CNN-LSTM

Finally, the hybrid CNN-LSTM architecture corresponds to a combination of an LSTM layer, and the CNN architecture described previously. After the initial layers, we have a CNN that is followed an LSTM Layer, with a dropout layer in between, before having the dense layer that prepares the output. This architecture was mostly based on the work done by Ghosh and Veale (2016).

4 Word Embeddings

A specific experiment used different word representations for each of the domains. Each dataset, from a different domain, had specific word embeddings trained on documents from that domain, to help the neural networks achieve the best results possible on the task of irony detection over the different domains (i.e., Reddit, Twitter, and news headlines).

However, even if no pre-trained word embedding exist, one can initialize the embeddings randomly and as the neural network is being trained, those embeddings will be adjusted according to the labeled training data. The disadvantage of this approach is that, since we are dealing with relatively small datasets, it is very difficult for the neural network to obtain good embeddings from a random initialization. Consequently, these models will not be able to achieve as good results as those using pre-trained embeddings. With this in mind, for this experiment, the neural networks leveraged from random word embeddings as the baseline in order to verify the performance gain of using different pre-trained embeddings.

For the domain of the discussion forum named Reddit[1], two different word embeddings were used. One of these resources was a publicly available word embedding dataset extracted from Wikipedia[2] documents and created by Mikolov et al. (2013a). The other was a Reddit word embedding created using the word2vec tool (Mikolov et al., 2013b) on a large dataset of Reddit comments which were from the same subreddits (i.e., a specific forum for a given topic) of the labeled dataset being classified.

For the Twitter[3] domain, two different word embedding datasets, specific to Twitter texts, were used. One of these datasets was optimized for named-entity recognition (Godin et al., 2013), and the other was created for the study of Amir et al. (2016) on irony detection, and has already achieved good results on one of the Twitter datasets that has also been used on the work being described on this dissertation.

Finally, for the news headlines domain, a word embedding dataset with Portuguese words was created from a dataset of news articles belonging to the online newspaper *Público*[4].

Each of the word embeddings used on the dataset Reddit and Twitter datasets were also combined, using both representations concatenated on a single architecture.

5 Experimental Setup

In this section it is presented the experimental setup used in the experiments performed on the research project.

5.1 Datasets

Four different datasets were used in the experiments, namely, two collected from Twitter, one from the discussion forum named Reddit and, lastly a Portuguese news headline dataset.

5.2 Reddit Dataset

The first dataset is a labeled Reddit dataset described in the work of Wallace et al. (2014). It consists of comments from several subreddits (i.e., forums of different topics) based on politics and religion. It has 1949 labeled comments and, of these, 537 were labeled as sarcastic and 1412 as not sarcastic.

5.3 Twitter Datasets

The Twitter datasets used in the experiments were described in the studies of Riloff et al. (2013) and Bamman and Smith (2015). for The original datasets used in those studies could not be collected in full. For instance, the dataset of Riloff et al. (2013) consisted of 3200 labeled tweets, but by the time my research project started it was only possible to extract 2118 of the tweets from the original dataset, since some of the data could not be extracted with the Twitter API for various reasons (e.g., the tweet was deleted or the user changed permissions). From those 2118 labeled tweets, 453 were labeled as sarcastic and 1665 were not sarcastic.

In the dataset from Bamman and Smith (2015) the same issue occurred. From the original 19,534 tweets, only 11,457 were possible to extract using the Twitter API. The labels ended being distributed into 6,478 ironic tweets and 4,907 that were not ironic.

5.4 Portuguese News Headline Dataset

The Portuguese News headline dataset was collected from two news sources. The first was a fac-

[1] `https://www.reddit.com/`
[2] `https://www.wikipedia.org/`
[3] `https://www.twitter.com/`
[4] `https://www.publico.pt/`

Classifier	Reddit		Twitter Riloff		Twitter Bamman		News Headlines	
	F1-score	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score	Accuracy
SVM	65%	64.4%	79%	80.2%	68%	67.8%	83%	83.1%
Logistic Regression	66%	68.9%	79%	80.4%	70%	69.4%	85%	84.7%
CNN-LSTM	**67%**	**69.5%**	**80%**	**82.0%**	**70%**	**70.5%**	**91%**	**90.7%**

Table 1: Results of the baseline involving the feature-based classifiers and the CNN-LSTM neural network.

tual news website called *Público* from which we got the data labeled as not ironic. The source for the ironic news headlines was a satirical news website called *Inimigo Público*. The overall dataset is balanced, consisting of 4,335 ironic headlines from *Inimigo Público* and 4,313 not ironic headlines from *Público*.

5.5 Baseline

Baseline methods were based on common classification approaches (machine learning algorithms leveraging simple features), namely naïve Bayes, logistic regression and Support Vector Machine (SVM) classifiers. An SVM that leverages from the results yielded by a Naïve Bayes classifier (i.e., a model inspired on the work of Wang and Manning (2012)), was also used on the experiments.

These classifiers relied on simple features like unigrams and bigrams, term frequency-inverse document frequency, or simply the number of term occurrences when building the representations (depending on the datasets). English stop words were removed, but the representations kept punctuation and emoticons.

Irony has been observed to be related to the emotions expressed on an utterance (Riloff et al., 2013; Reyes et al., 2013), another baseline feature used on the Riloff dataset was based on the contrast of affective norm (Warriner et al., 2013) associated to words within a sentence. For the featured-based classifiers, this feature corresponds to the average of the arousal, valence and dominance of all the words within sentence where the average belongs to one of three groups, low ($average < 3$) or medium ($3 \leq average \leq 7$) or high ($average > 7$), for each affective norm. For example, considering that the words *hate*, *being*, and *emotional* have each a respective valance rating of, 3, 7, and 2 and an arousal rating of 9, 7, and 8. Since the average of the valance and arousal ratings for those words are 4 and 8, respectively, the phrase *hate being emotional* belongs to the medium valence group and the high arousal group. This allows for the classifier to capture existing contrasts of low and high values of the different affective norms.

Due to the fact that the Portuguese news headline dataset consisted of Portuguese texts and contains a formality (e.g., it does not make use of extra punctuation to imply irony) that is uncommon on the other social media datasets, only the bag-of-words with unigrams and bigrams was considered for the classification task over this particular dataset.

5.6 Experiments

The experiments on this dissertation consisted of running the classification task over the different neural network architectures mentioned previously.

The first set of experiments consisted of running each of the algorithms/classifiers over each of the datasets described in the previous section, comparing the results of the feature-based and neural network classifiers. For this experiment, the neural networks leveraged the publicly available word embeddings described in Section 4.

The second set of experiments consisted of running the neural networks leveraging from different combination of embeddings, to compare the performance of using different word embeddings on the task of irony detection. Three tests were performed for each of the Reddit and Twitter datasets. The first test consisted on using a randomly ini-

Dataset	Wikipedia Embeddings		Reddit Embeddings		Wikipedia+Reddit Embeddings	
	F1-score	Accuracy	F1-score	Accuracy	F1-score	Accuracy
Reddit	67%	69.5%	68%	70.5%	69%	70.8%

Table 2: The results of the CNN-LSTM, over the Reddit dataset, using different embeddings.

Dataset	Frederic's Embeddings		Amir's Embeddings		Frederic+Amir Embeddings	
	F1-score	Accuracy	F1-score	Accuracy	F1-score	Accuracy
Riloff's Twitter	80%	82.0%	80%	81.6%	81%	82.7%
Amir's Twitter	70%	70.5%	70%	69.8%	71%	71.3%

Table 3: The results yielded by the CNN-LSTM, over the Riloff and Bamman Twitter datasets, using different embeddings.

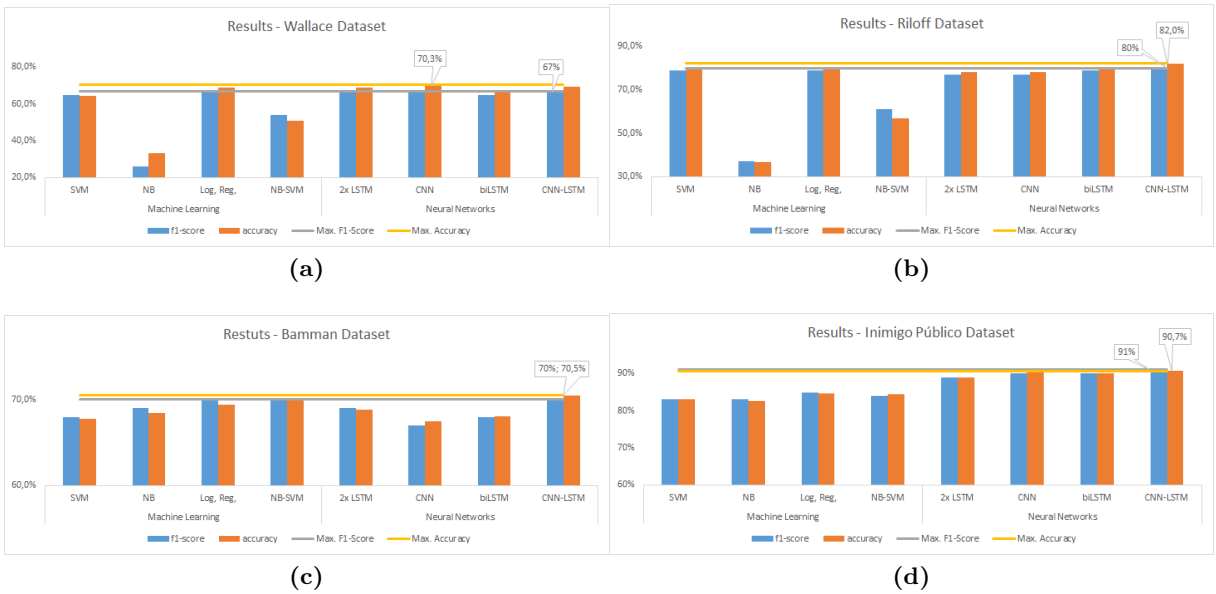

Figure 1: Results of the different classifiers on the (a) Reddit (b) Riloff's Twitter (c) Bamman's Twitter and (d) Portuguese news headline datasets.

tialized word embedding. On the second test, the pre-trained word embeddings described in Section 4 were used individually. The third test involved using a neural network that used two word embeddings in a single architecture. For the Portuguese news headline dataset, only random embeddings and a single word embedding matrix were used.

6 Results

The results from each of the different baseline classifiers leveraging with the features mentioned on the previous sections varied substantially depending on the classifier being used, however the SVM and the logistic regression model yielded the best results. The results of these two classifiers can be seen on Table 1.

The logistic regression yielded the best results, having produced 66% F1-score on the Reddit dataset (against 65% produced by the SVM), 79% on Riloff's Twitter dataset (on both the logisitic regression and SVM classifier), 70% on Bamman's Twitter dataset (against 68% from the SVM classifier) and finally 85% F1-score on the news headlines dataset (against 83% yielded by the SVM). However, it is worth mentioning that the logistic regression model did not suffer any relevant improvement with the emoticons and punctuation features while the SVM classifier substantially improved when the features were added. It is also worth noting that the SVM classifier leveraging the affective norm feature increased the f1-score by 7 p.p. on the Riloff's Twitter dataset.

With the neural networks, each of the classifiers yielded quite similar results between each other. On Figure 1, you can observe the results from the CNN-LSTM architecture. All the architectures produced similar results, having the CNN-LSTM consistently yielded the best results, with F1-scores of 67% on the reddit dataset, 80% and 70% on the Twitter datasets (Riloff's and Bamman's dataset, respectively) and 91% over the Portuguese news headline datasets. You can observe that the CNN-LSTM performed equally well or better than the classic machine learning classifiers with the added lexical features.

Over the Reddit dataset, the random initialized embeddings yielded the worst results, with 66% over the F1-score. With using the publicly available word embedding from Wikipedia and the word embeddings created using word2vec on a large Reddit dataset, the neural network yielded 67% and 68%, and finally the combination of the previous two embeddings yielded the best results on the reddit dataset, 69% F1-score. When compared to the logistic regression, the improvement was quite significant having resulted in an improvement of 3 p.p. on the F1-score.

Over the Riloff and Bamman Twitter datasets, the difference between using the different embeddings yielded similar results. When using each of the word embeddings individually each of them had the same performance, 80% F1-score over the Riloff dataset and 70% over the Bamman dataset. Combining the two, the result was a slightly better performance of 1 p.p. on the F1-score. On both the cases the performance was better than the baseline classifiers.

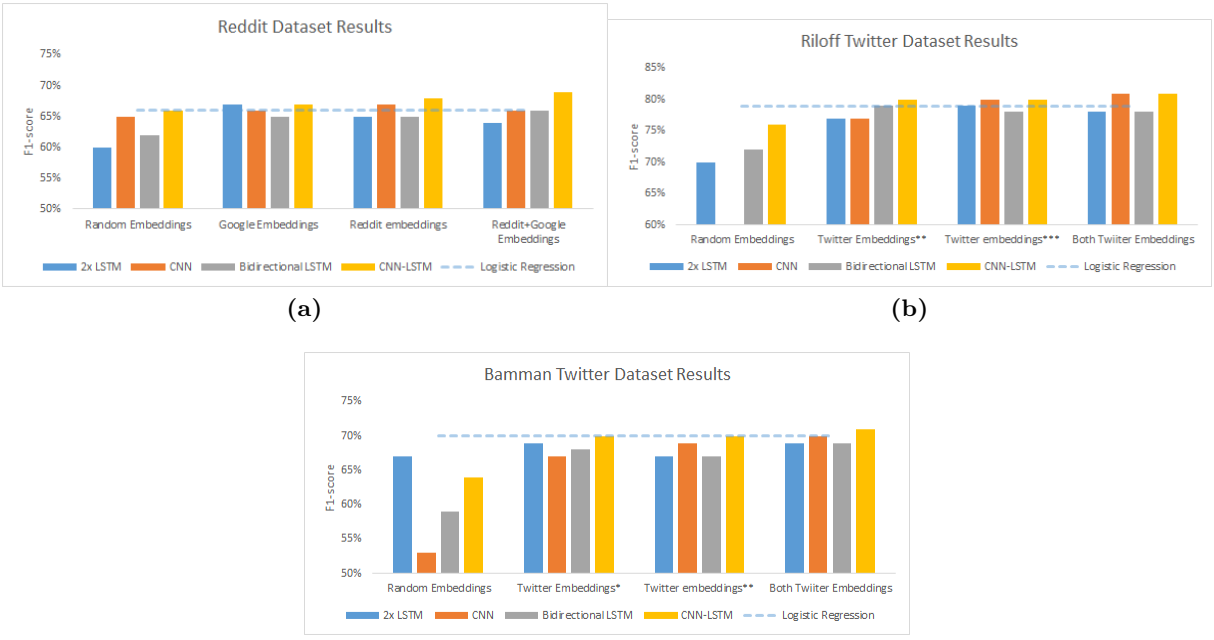

Figure 2: Results of the neural networks leveraging different embeddings compared to the logistic regression classifier over the different datasets.

7 Discussion

he main motivation behind the work reported on this dissertation was to make a comparison between the performance of feature-based machine learning algorithms, like logistic regression models and SVMs, and the deep learning algorithms that have been appearing on tasks involving semantic interpretation of texts. To be able to make a broader comparison, datasets from slightly different domains were used, one from a social media discussion forum named Reddit, two datasets from the social media networking service named Twitter, and a dataset consisting of satirical and factual news headlines.

The first point to be made is that in terms of the classification approaches using feature-based machine learning classifiers, logistic regression models consistently yielded the best results, while SVMs yielded the second best results. However, when adding more lexical features, the logistic regression showed little or no improvement, and observing the results from the SVM, the inclusion of those same features allowed the SVM to perform much better, making it able to compete with the performance of the logistic regression classifier. This shows that when handling a large set of features, as most of the previous studies mentioned in Section 2, an SVM classifier will be more likely to outperform the other feature-based classifiers on this particular task.

The second point that can be made, this time from the results of the second set of experiments, is that neural networks are able to perform as well as any of the classic machine learning algorithms, as can be seen in Figure 1. Even though these neural networks only used word embeddings that were not created specifically for the particular task of irony detection or even for the domain of the dataset being used on this work, these models are able to, outperform the best feature-based classifier used on the first set of tests.

The main feature that neural networks require to perform decently on any text-based classification task are appropriate embeddings. The better the word representations from the embeddings, the better the performance of the classifier. A simple word embedding matrix can be created using word2vec (Mikolov et al., 2013b) on a large dataset. With some tuning of the available parameters (e.g., context window and number of dimensions of the embeddings), it is possible to improve the performance of deep learning classifiers (Figure 2). Even though these embeddings can take some time to create and optimize in order to obtain good results, with the tools that are publicly available today, this does not require much of an implementation effort which is one of the major issues when engineering features for the classic machine learning classifiers.

With this in mind, the third set of experiments consisted of verifying the impact of using different word embeddings on neural networks. Using random embeddings with the neural networks performed quite poorly. However, this was to be ex-

pected since we are dealing with datasets that are too small for the neural networks to adjust the weights well enough to obtain a good representation. Using individual pre-trained word embeddings, it was possible to outperform the feature-based machine learning classifiers, specifically using the CNN-LSTM architecture. As anticipated, with the word embeddings that are more specific to the domain, the classifiers performed better than with word embeddings that were created for another domain or task.

Combining multiple word embeddings is another way to improve the results of the neural networks, as can be observed from the results of the experiments performed on this dissertation. Even though the embeddings used in these experiments only considered the words from their datasets, they still complemented each other and consequently improved the results from the classifiers.

8 Conclusions and Future Work

This dissertation described three set of experiments involving neural networks, with the objective of verifying the performance of this approach on the task of irony detection. It was expected that neural networks would outperform the previously used feature-based machine learning classifiers, as it has been happening on other NLP tasks of NLP.

The first conclusion that can be made from the results of the experiments is that, from the different machine learning algorithms, both feature-based (i.e., naïve Bayes, logistic regression, SVM and naïve Bayes-SVM) and neural network approaches (i.e., two stacks of LSTM layers, bidirectional LSTM layers, a CNN, and a CNN-LSTM), the CNN-LSTM is the best classifier on the task of irony detection over datasets with different domains, outperforming any of the other architectures on all the different datasets, including the feature-based classifiers that are most commonly used on the task at hand.

From the second set of experiments using different word embeddings, it was possible to conclude that even though, as expected, word embeddings more specific to the domain of the dataset achieve better results, it is possible to improve the performance of the neural networks by combining different word embeddings, even if not created using a dataset from the same domain as the one of the datasets being classified.

The third experiment, related to using affective norms to detect irony, showed that using a 2-dimensional vector space representation of affective norms is insufficient to allow for an improvement on the performance on neural networks.

The findings of this research study show that there are two viable approaches that correspond to different paradigms for the task of detecting irony in text. The first approach has been commonly used over the last years and is based on the design of features, where each feature requires a lot of manual labor. The second approach, based on neural networks, is a more recent approach that leverages on word embeddings to classify documents.

Neural networks have the advantage of not requiring the same amount of manual labor to obtain the same results. However, these models do require the use of other resources, they need time (needed to train a neural network) and also better hardware (to be able to train the neural networks over shorter training times). Nonetheless, feature-based machine learning algorithms, that have been used on previous studies, have been requiring larger and more complex features to obtain better results on the classification of text according to irony, consequently requiring more design and implementation time, to only slightly improve the results.

Using neural networks seems to require less time to implement and design than methods based on features that would still be outperformed by a simple neural network leveraging pre-trained word embeddings that can be found publicly available. This makes deep learning an interesting new approach, although there are still improvements that can be made for the task of irony detection.

The use of neural networks on the task of irony detection is very recent. There are many ideas that can be used to improve the performance of deep neural classifiers on this task, specifically on what regards the creation of word representations (i.e., embeddings).

A possibility to give continuity to this research project is to use other algorithms to create embeddings besides word2vec, such as the algorithm from the work of Pennington et al. (2014), and combine the resulting embeddings into a single architecture (Zhang et al., 2016).

Another possible work that can be performed is the creation of embeddings specific to a user (i.e., user embeddings) instead of words. Information of the user has been shown to improve the performance of classifiers over the task of irony detection. Because of this, a user embeddings would follow the logic of Bamman and Smith (2015) where the features took into consideration the users historical data and his audience. This can also be done to create embeddings specific to a user, as it has been shown on the work of Amir et al. (2016).

References

Amir, S., Wallace, B. C., Lyu, H., Carvalho, P., and Silva, M. J. (2016). Modelling context with user embeddings for sarcasm detection in social media.

Bamman, D. and Smith, N. A. (2015). Contextualized sarcasm detection on Twitter. In *Pro-*

ceedings of the International AAAI Conference on Web and Social Media.

Burfoot, C. and Baldwin, T. (2009). Automatic satire detection: Are you having a laugh? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and of the International Joint Conference on Natural Language Processing*.

Carvalho, P., Sarmento, L., Silva, M. J., and de Oliveira, E. (2009). Clues for detecting irony in user-generated contents: Oh...!! it's "so easy" ;-). In *Proceedings of the International CIKM Workshop on Topic-sentiment Analysis for Mass Opinion*.

Ghosh, A. and Veale, T. (2016). Fracking sarcasm using neural network. In *Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*.

Godin, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Multimedia lab @ acl wnut ner shared task: Named entity recognition for twitter microposts using distributed word representations.

González-Ibáñez, R., Muresan, S., and Wacholder, N. (2011). Identifying sarcasm in twitter: A closer look. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Karoui, J., Farah, B., Moriceau, V., Aussenac-Gilles, N., and Hadrich-Belguith, L. (2015). Towards a contextual pragmatic model to detect irony in tweets. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and of the International Joint Conference on Natural Language Processing*.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Mikolov, F., Vandersmissen, B., De Neve, W., Walle, R. V. d., and Dean, J. (2013a). Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013b). Efficient estimation of word representations in vector space. *Computing Research Repository*, abs/1301.3781.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the International Conference on Machine Learning*.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing*.

Rajadesingan, A., Zafarani, R., and Liu, H. (2015). Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the Association for Computing Machinery International Conference on Web Search and Data Mining*.

Reyes, A., Rosso, P., and Veale, T. (2013). A multidimensional approach for detecting irony in twitter. *Language Resources and Evaluation*, 47(1).

Riloff, E., Qadir, A., Surve, P., Silva, L. D., Gilbert, N., and Huang, R. (2013). Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing*.

Tepperman, J., Traum, D. R., and Narayanan, S. (2006). "yeah right": sarcasm recognition for spoken dialogue systems. In *Proceedings of Interspeech*.

Wallace, B. C., Choe, D. K., Kertz, L., and Charniak, E. (2014). Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Wang, S. and Manning, C. D. (2012). Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Warriner, A. B., Kuperman, V., and Brysbaert, M. (2013). Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior Research Methods*, 45(4).

Zhang, Y., Roller, S., and Wallace, B. C. (2016). Mgnc-cnn: A simple approach to exploiting multiple word embeddings for sentence classification. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Zhou, C., Sun, C., Liu, Z., and Lau, F. C. M. (2015). A c-lstm neural network for text classification. *Computing Research Repository*, abs/1511.08630.