Brazilian Presidential Elections in the Era of Misinformation: A Machine Learning Approach to Analyse Fake News

Jairo L. Alves[1], Leila Weitzel[1](✉), Paulo Quaresma[2],
Carlos E. Cardoso[1], and Luan Cunha[1]

[1] Fluminense Federal University, Rio de Janeiro, Brazil
jairo.luciano@gmail.com, {leila_weitzel, carloseac, luanpereiracunha}@id.uff.br
[2] Universidade de Évora, Évora 17, Portugal
pq@uevora.pt

Abstract. As Brazil faced one of its most important elections in recent times, the fact-checking agencies handled the same kind of misinformation that has attacked voting in the US. However, stopping fake content before it goes viral remains an intense challenge. This paper examines a sample database of the 2018 Brazilian election articles shared by Brazilians over social media platforms. We evaluated three different configuration of Long Short-Term Memory. Experiment results indicate that the 3-layer Deep BiLSTMs with trainable word embeddings configuration was the best structure for fake news detection. We noticed that the developments in deep learning could potentially benefit fake news research.

Keywords: Fake news · Machine learning · Long Short-Term Memory · Word embeddings · Deep learning · Recurrent neural network

1 Introduction

Recent political events, notably the Brexit referendum in the U.K., the presidential election of 2016 in the U.S. and the Brazilian's economic and political crisis in 2016 have led to a wave of interest in the phenomenon of fake news. There is already strong concern about the impact that the spread of fake news could cause to the 2018 Brazilian elections, the most important elections in recent times.

Fact-checking is a journalistic method by which it is possible to ascertain whether accurate information has been obtained from reliable sources and then assess whether it is true or false, whether it is sustainable or not. Fact-checking is a costly process, given the large volume of news produced every minute in our post-truth era, making it difficult to check contents in the real time. The rate and the volumes at which false news are produced overturn the possibility to fact-check and verify all items in a rigorous way, i.e. by sending articles to human experts for verification. The process of fact checking requires researching and identifying evidence, understanding the context of information and reasoning about what can be inferred from the evidence. Besides,

I. Nyström et al. (Eds.): CIARP 2019, LNCS 11896, pp. 72–84, 2019.
https://doi.org/10.1007/978-3-030-33904-3_7

not all journalists have the knowledge to investigate the databases that would allow a rigorous verification, and access to the real data is not always possible. The goal of automated fact checking is to reduce the human burden in assessing the veracity of a claim [1, 2].

The Brazilian fact-checking agencies are handling the same kind of misinformation that has attacked voting in the US. It is essential to cite the concern of the fact-checking agencies in order to try to resolve the impact caused by fake news before it goes viral. However, stopping fake content before it goes viral remains an intense challenge. Fake news are deliberately been created to mislead the readers, resulting in an adversarial scenario where it is very hard to distinguish real facts from fakes. While fake news is not a new phenomenon, questions such as why it has emerged as a world topic and why it is attracting increasingly more public attention are particularly relevant at this time. The leading cause is that fake news can be created and published online faster and cheaper when compared to traditional news media such as newspapers and television [3].

Hence, based on the scenario described herein, the main goal of this research is to detect fake news, which is a classic text classification problem with the applications of NLP (Natural Language Processing). We have crawled the labeled articles published by one independent fact-checking agency, "Aos Fatos[1]", for our purpose. The present analysis focuses on the period from May 2018 to the end of September 2018. During this time, we collected 2,996 articles. It must be highlighted that the language analyzed is Brazilian Portuguese. Brazilian Portuguese can be considered different from European Portuguese. There are grammatical peculiarities of Brazilian Portuguese, such as, the syntactic position of sentence subjects; the preferred position of clitic pronouns; the pronominal paradigm and other characteristics, which make it different. To the best of our knowledge, it is the first experiment that uses a dataset in the Brazilian Portuguese language for fake news detection.

2 Fake News Characterization

The term fake news is used in a variety of (sometimes-conflicting) ways, thereby making conceptual analysis more difficult. It must be stressed the difference between satire and fake news. Satire is meant to be comedic in nature. Satire presents stories as news that are factually incorrect, but the intent is not to deceive but rather to call out, ridicule, or expose behavior that is shameful, corrupt, or otherwise "bad". Satirical news as designed specifically to entertain the reader, usually with irony or wit, to critique society or a social figure. Fake news are defined as a news story that is factually incorrect and designed to deceive the consumer into believing it is true [4].

Shu, Sliva, Wang, Tang and Liu [5] argue that there is no agreed definition of the term fake news. They draw attention to the fact that fake news is intentionally written to mislead readers, which makes it nontrivial to detect simply based on news content. Fake news tries to distort truth with diverse linguistic styles. For example, fake news may cite true evidence within the incorrect context to support a non-factual claim.

[1] https://aosfatos.org/.

The concepts of fake news have existed since the beginning of civilization. There are several historical examples of its use over the centuries as a strategic resource for winning wars, gaining political support, manipulating public opinion, defame peoples and religions. Years have passed and the Web has turned into the ultimate manifestation of User-Generated Content (UGC) systems [6]. The UGC can be virtually about anything including politics, products, people, events, etc. Hence, the popularization of social networks, the use of fake news to serve the most different purposes has grown dramatically, allowing the free creation and large-scale dissemination of any type of content. Words in news media and political discourse have a considerable power in shaping people's beliefs and opinions [7].

Facebook has begun to mark each news story depending on whether the news is truthful or not. In 2016, "Google News" began to mark news about the USA. Since then, they expanded this practice to the United Kingdom, Germany, France, Mexico, Brazil and Argentina. In 2018, the International Federation of Library Associations and Institutions[2] (IFLA) issued a statement that contains recommendations to governments and libraries regarding fake news marking, and it is accompanied by a toolkit of resources. IFLA is concerned by the risk that this can pose to access to information, where people do not have the skills to spot it, but also by the way it is used by governments to justify potentially repressive policies. The agency posted an infographic on Facebook and Twitter to help combat fake news, which contains eight rules. The set of rules include; (i) Consider the source, (ii) Read beyond the headline, (iii) Check the author, (iv) What is the support? (v) Check the date, (vi) Is this some kind of joke? (vii) Check your biases, and (viii) Consult the experts.

3 LSTM Main Characteristic

In the mid-90s, a variation of RNN called LSTMs was proposed by the German researchers Sepp Hochreiter and Juergen Schmidhuber [8] as a solution to the vanishing gradient problem. RNNs are a family of neural networks well suited to model sequential data like time series, speech, text, financial data, audio, language, and other similar types of data. In general, the recurrence aspect allows RNNs to form a much deeper understanding of a sequence and its context, compared to other algorithms, such as static neural networks or purely statistical approaches. The core concept of LSTM are the memory cell state and its various gates. The gate can regulate the flow of information and it can automatically learn which data in a sequence is important to keep track of and witch data can be thrown away. By doing that, it can pass relevant information down the line in order to make better predictions for long sequences. As there is a direct path to pass the past context along, it suffers much less from vanishing gradients than vanilla RNNs. The gates typically are implemented with sigmoid activation functions and can learn what information is relevant to keep or forget during training with BPTT algorithm.

[2] https://www.ifla.org/.

To avoid overfitting, we employ standard regularization techniques, such as, early stopping and dropout. Early stopping monitors and may stop the training process once the validation loss metric starts to increase or when validation accuracy starts to decrease. The main idea of dropout is to randomly remove computing units in a neural network during training to reduce hard memorization of training data. In this case, we want to trade training performance for better generalization on unseen data. Both regularization methods are used when indicated to prevent overfitting [9].

4 Related Work

Fake news is a concern, because they can affect the minds of millions of people every day, these have led to the term post-truth. Fake news detection has attracted the interest of researchers in recent years with several approaches being proposed [4–6, 10–15]. Several previous studies have relied on feature engineering and standard machine learning methods, such as Support Vector Machines (SVM), Stochastic Gradient Descent, Gradient Boosting, Bounded Decision Trees, Random Forests and Naïve Bayes. Several other studies have proposed *Deep Learning Methods* [16–19].

Pfohl [18] organized an analytic study on the language of news media in the context of political fact-checking and fake news detection. Two approaches were applied: lexicon and neural network based. The lexicon approach shows that first-person and second person pronouns are used more in less reliable or deceptive news types. In the second methodology, the models were trained with Max-Entropy classifier with L2 regularization on n-gram TF-IDF feature vectors (up to trigrams). The output layer includes four labels: trusted, satire, hoax and propaganda. The model achieved F1 scores of 65%. The author also trained a Long Short Term Memory (LSTM) network and Naïve Bayes, both used lexicon measurements in order to concatenate to the TF-IDF vectors. Ajao, Bhowmik and Zargari [19] proposed a framework to detect and classify fake news from Twitter posts. The authors use a hybrid approach consisting of CNN - Convolutional Neural Networks and LSTM with dropout regularization and Word Embedding layer. Wang [16] first introduced the Liar dataset. The Liar dataset contains 12.8K manually labeled short statements in various contexts sampled from one decade of content in politifact.com, which provides detailed analysis reports and links to source documents for each case. Wang [16] has evaluated several popular machine-learning methods on this dataset. The baselines include logistic regression classifiers, SVMs, LSTMs and CNN. A model that combines three characteristics for a more accurate and automated prediction was proposed by Ruchansky, Seo and Liu [17]. They incorporated the behavior of both parties, users and articles, and the group behavior of users who propagate fake news. They proposed a model, which is composed of three modules. The first module named *Capture* captures the abstract temporal behavior of user encounters with articles, as well as temporal textual and user features, to measure response as well as the text. To extract temporal representations of articles they use a Recurrent Neural Network (RNN). The second module, *Score*, estimates a source suspiciousness score for every user, which is then combined with the first module by integration to produce a predicted label.

5 Methodology

5.1 Dataset Construction

The data was acquired from the website of the fact-checking agency “Aos Fatos”. Fact checking is a task that is normally performed by trained professionals. Depending on the complexity of the specific claim, this process may take from less than one hour to a few days. Daily, journalists of this agency check the statements of politicians and authorities of national expression in order to check if they are speaking the truth. The labeling methodology is based on the following rules: The use of the ***True*** badge is simple, the statement or the information is consistent with the facts and does not lack contextualization to prove correct. The ***Unsustainable*** label represents the statements that cannot be refuted or confirmed. i.e., there are no facts, data or any consistent information to support the claim. When the statement receives the ***Inaccurate*** label, it means that it needs context to be true and sometimes lack contextualization to prove to be true. The label ***Exaggerated*** means that the fact was exaggerated, for example, “more than 150 million Brazilians live below poverty line”. In reality, only 50 million live like that. The ***Contradictory*** label is when content of the statement is exactly the opposite of the real fact that it happened. Suppose the statement “I have nothing against homosexuals or women, I am not xenophobic”. This can be contradictory, if a person has already said homophobic, sexist and xenophobic statements at other times. The ***Distorted*** label is used only for rumors and news with misleading content. It serves for those texts, images and audios samples that bring information factually correct, but applied with the intention of confusing. If a statement or news or rumor has information without any factual support, they receive the ***False*** label.

The labeling methodology and other features are available on the website https://aosfatos.org/. As can be observed, some labels have high semantic similarity, which makes it difficult, even for a human, to perform this analysis.

We performed our data collection by means of an in-house procedure, using Python and some other libraries, such as Beautifulsoup and Regex. Colleting news was not an easy task, since the website “AosFatos” was not in standard formatting, making it difficult to scrape. The dataset has the following attributes: *body-text, *date, *label and *text. The column ***body-text*** is the original sentence that someone thought or published in a social network. The column ***date*** is the day of the publication. The column ***label*** identifies in one of seven labels discussed herein. Finally, column ***text*** explains why the news was categorized with a certain label. The columns of interest to this research are ***label*** (as output) and the ***body-text*** (as input). The data set is available for download at <*blind review*>.

Following the recommendation of Shu, Sliva, Wang, Tang and Liu [5], due to minimal semantic differences between the many of the labels and in accordance with our goal, we decide to use only the ***true*** and ***fake*** labels as binary classification. Hence, after that, only 1,187 news were retained. The following preprocessing phases were performed (not necessarily in the succeeding order): we remove Hashtags, emoticons, punctuation and special characters ($, @, etc.); the text was tokenized (strip white space) and normalized (converting all letters to lower case).

5.2 Model Configuration

We conducted several experiments with different combinations of feature sets. Table 1 shows all the model configurations. The models are implemented using the Keras framework [20] with the TensorFlow backend engine [21]. In the following paragraphs, we describe our models of Regular LSTM (Naïve), Bidirectional LSTM (BiLSTM) and Deep BiLSTM. For all models, a Dense (Fully Connected) 3-neuron output layer is present at the end to generate the final output prediction. Table 1 shows the network configurations.

Table 1. Model configurations.

Model	Model configuration	LSTM inputs	Trainable embeddings?	wmDropout	Embeddings dimension
M20	3 layers of Bidirectional CuDNNLSTMs, 64 internal states on each CuDNNLSTM block	Random embeddings	Yes	Yes (rate = 0.3)	200
M21	3 layers of Bidirectional LSTMs, 32/16/16 internal states	Random embeddings	Yes	Yes (rate = 0.5, recurrent dropout)	50
M22	3 layers of Bidirectional LSTMs, 32/16/16 internal states + 1 additional Dense(8) layer after the LSTMs	Random embeddings	Yes	Yes (rate = 0.5, recurrent dropout)	50
M23	3 layers of Bidirectional CuDNNLSTMs, 32/16/16 internal states + 1 additional Dense(8) layer after the LSTMs	Random embeddings	Yes	Yes (rate = 0.5)	500
M24	**3 layers of Bidirectional CuDNNLSTMs, 64 internal states on each CuDNNLSTM block**	**Random embeddings**	**Yes**	**Yes (rate = 0.5)**	**80**
M25	3 layers of Bidirectional CuDNNLSTMs, 8 internal states on each CuDNNLSTM block	Random embeddings	Yes		50
M26	3 layers of Bidirectional CuDNNLSTMs, 8 internal states on each CuDNNLSTM block	Pre-trained embeddings	Yes	No	50 (embeddings multiplied by 2)

(*continued*)

Table 1. (*continued*)

Model	Model configuration	LSTM inputs	Trainable embeddings?	wmDropout	Embeddings dimension
M27	3 layers of Bidirectional CuDNNLSTMs, 8 internal states on each CuDNNLSTM block	Pre-trained embeddings	Yes	No	50 (embeddings multiplied by 0.25)
M28	3 layers of Bidirectional CuDNNLSTMs, 8 internal states on each CuDNNLSTM block	Pre-trained embeddings	Yes	No	50 (embeddings multiplied by 0.10)
M29	**3 layers of Bidirectional CuDNNLSTMs, 8 internal states on each CuDNNLSTM block**	**Pre-trained embeddings**	**Yes**	**No**	**50 (embeddings multiplied by 0.05)**
M30	3 layers of Bidirectional CuDNNLSTMs, 8 internal states on each CuDNNLSTM block	Pre-trained embeddings	Yes	No	50 (embeddings multiplied by 0.03)
M31	3 layers of Bidirectional CuDNNLSTMs, 8 internal states on each CuDNNLSTM block	Random embeddings	Yes		–
M32	3 layers of Bidirectional CuDNNLSTMs, 8 internal states on each CuDNNLSTM block	Pre-trained embeddings	Fixed	No	50 (embeddings multiplied by 0.03)
M33	1 layer of Bidirectional CuDNNLSTM, 128 internal states	Pre-trained embeddings	Yes	No	50
M34	3 layers of Bidirectional CuDNNLSTMs, 8 internal states on each CuDNNLSTM block	Pre-trained embeddings	Yes		50
M35	3 layers of Bidirectional CuDNNLSTMs, 64/32/32 internal states, +1 additional Dense(8) layer after the LSTMs + maximum input length increased to 250 words	Random embeddings	Yes	Yes (0.5 Recurrent)	50
M36	3 layers of CuDNNLSTMs, 6 internal states on each CuDNNLSTM block + maximum input length increased to 300 words	Pre-trained embeddings	Yes		50 (embeddings multiplied by 0.05)

6 Results

As discussed herein, we consider fact checking as an ordinal classification task. Hence, in theory it would be possible to frame it as a supervised classification task using algorithms that learn from annotated samples with the ground truth labels. We tested four main architectures with different parameters, as seen on Table 1. We tested three-layers of Bidirectional LSTM and Bidirectional CuDNN-LSTM[3] with different internal state sizes. We also tested pre-trained and random embedding as input methods. Word embeddings can be learned from text data and reused among projects. They can also be learned as part of fitting a neural network on the specific textual data. Keras offers an Embedding layer that can be used as part of a deep learning model where the embedding is learned along with the rest of the model (trainable embeddings). Also, this layer can be used to load a pre-trained word-embedding model, a type of transfer learning. We used pre-trained Portuguese word embedding (FastText, 50 dimension) available at http://nilc.icmc.usp.br/embedding.

When indicated, dropout was used and recurrent dropout when the blocks were regular LSTMs (in that case, the dropout varied between 0.3 and 0.5). We also tested models with different batch size (varying between 100 and 50). In most cases, batch size did not affect the result, so most scenarios were run with batch size equal to 100. We noticed that the text processing affected positively the performance of the models, albeit marginally. We also observed that fixed, pre-trained word embedding hurts performance when compared to random, trainable embeddings. The best performance is achieved by pre-trained trainable embeddings, especially when attenuated by a previous 0.05 factor multiplier on the magnitude of the pre-trained embedding vectors.

Naive LSTM – BiLSTM: The first model was a simple layer LSTM - (Regular-Naïve, see Fig. 1). The accuracy achieved was about 60%, which can be considered low. We first tested encoding the words as one-hot vectors and, then, as embeddings in various configurations (refer to Table 1). A one hot encoding is a representation of categorical variables as binary vectors where only one dimension is set to 1, while all others are zero. Traditional approaches to NLP, such as bag-of-words models whilst useful for various machine learning (ML) tasks, tend to not capture enough information about a word's meaning or context. Such models often provide sufficient baselines for simple NLP tasks. Also, it is well known that one-hot encodings, however simpler, do not capture syntactic (structure) and semantic (meaning) relationships across collections of words and, therefore, represent language in a rather limited way.

[3] It is high-level deep learning library and it can only run on GPU.

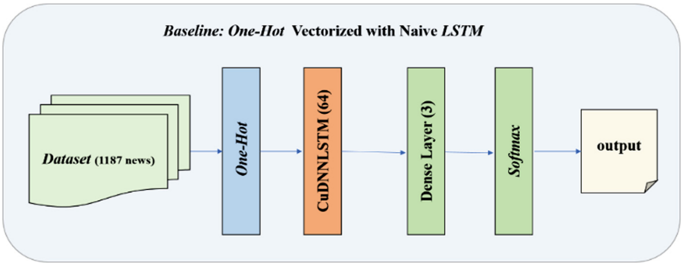

Fig. 1. Baseline model – Naïve LSTM

The input was a single LSTM layer, followed by a Dense (fully-connected) layer and the activation function was Softmax. The non-regularized model is our baseline model. This model had computational issues. It would take more than 500 s to run each epoch of training. In order to tackle this issue, we adopted the cuDNN-LSTM[4] (CUDA is Deep Neural Network library) [22–24] functions on the Google Colab[5] platform instead of the regular TensorFlow LSTM functions. This allows our models to run on Google's Collaboratory GPU hardware. Despite the lack of some parameters (recurrent dropout for example), we noticed that cuDNN-LSTMs were much faster than the "regular" version. Thus, unless indicated otherwise, all remaining references to LSTMs layers are implemented with cuDNN-LSTMs APIs. We also noticed that, the number of trainable parameters in the LSTM layer is strongly affected by its input sentence length and since our samples sequence lengths were rather long (>200 words for some sentences). The number of words in the sentence allows us to set a fixed sequence length, zero pad shorter sentences, and truncate longer sentences to that length as appropriate. Henceforth, we limited the input length to a maximum of 100 words, also padding to zero the shorter sentences. The pad_sequences() function in the Keras library was used. The default padding value is 0.0, which is suitable for most applications, although this can be changed if needed. Thus, every sequence length was exactly 100 words long. The dimensions for data are [Sequence Length (100), Input Dimension (2887)] with 50 epochs, early stopping and 128 hidden states. The batch size limits the number of samples to be shown to the network before a weight update can be performed. The accuracy was about 60%.

LSTM Bidirectional (BiLSTM): The next model was Bidirectional LSTMs (BiLSTMs). Using BiLSTMs is advisable because we can better capture dependencies in both ends of the input. The performance at training phase was about 70% (Fig. 2). Conversely, a validation accuracy started decreasing. Hence, we decided to set 64 the number of hidden states. Over again, we cannot acquire an improvement in performance.

[4] cuDNN provides highly tuned implementations for standard routines such as forward and backward convolution, pooling, normalization, and activation layers.

[5] Colab is a research tool for machine learning based on Jupyter notebook.

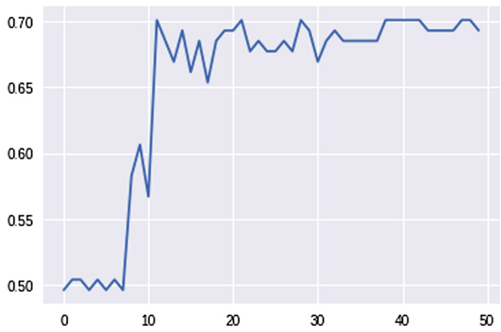

Fig. 2. Visualizing the training process. Accuracy (y-axis) and epoch (x-axis).

Deep BiLSTM: In the next model, we increase the depth of BiLSTM layers, by adding two more layers. We use Dropout regularization between each BiLSTM layer. As more capacity was added when adding layers, we reduced the number of hidden states of each individual BiLSTM to 32. This configuration achieved an average of 75% testing accuracy. We also used in this stage the callback module ModelCheckpoint from Keras. The metric used for triggering the checkpointing callback was the validation accuracy. During training, the best model weights are saved whenever an improvement is observed. The best validation accuracy achieved was about 80% (see Fig. 3).

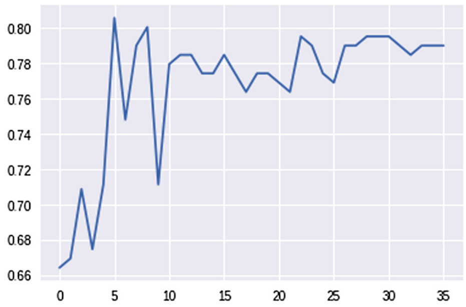

Fig. 3. Visualizing the training process. Accuracy (y-axis) and epoch (x-axis).

Deep BiLSTM and Embedding Layers: The last experiment aimed to take advantage of the similar meanings between correlate words, instead of relying exclusively on the LSTM to infer these relationships. We added an Embedding layer prior to the BiLSTMs. Word embeddings provide a dense representation of words and their relative meanings. The Embedding layer is initialized with random weights and is allowed to learn during the training. The average validation accuracy improved to just above 80%. Figure 4 shows the evolution of validation accuracy with this configuration.

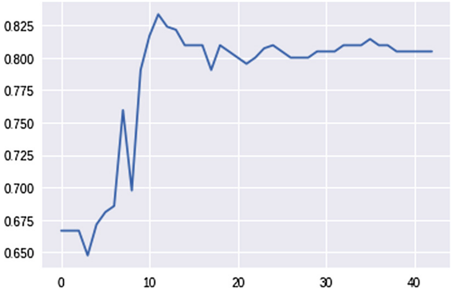

Fig. 4. Visualizing the training process. Accuracy (y-axis) and epoch (x-axis)

Figure 5 shows the evaluation of classification accuracy on the test set of 1,187 news. The accuracy performance ranged between 71% and 81%. The models M24 and M29 achieved the best performance (the accuracy was about 81%). The configurations of these networks are described in Table 1.

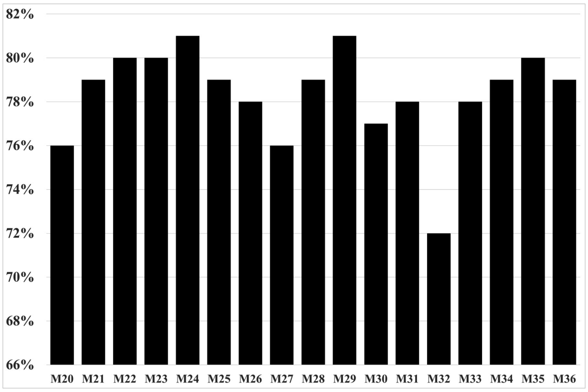

Fig. 5. The performance of all models. The y-axis is the accuracy and the x-axis are the models (the settings are in Table 1).

7 Conclusions

The increasing attention in fake news is motivated by the fact that people are typically not suited to distinguish between good information and fake news, in particular when the source of information is the Internet (and especially social media). Fake news is viewed as one of the greatest threats to journalism and it has weakened public trust in governments. The paper studied various recurrent neural network configurations based on LSTM and embeddings as binary classifiers for detecting Brazilian fake news. We have also built (and exploited) a new dataset collected from the website of Brazilian fact-checking agency AosFatos. Experimental results indicate that, from the models tested, the 3-layer Deep BiLSTMs with trainable word embeddings configuration was the best structure for the task of fake news detection. We noticed that fake news research could potentially benefit from recent developments in deep learning techniques. Several works in NLP tackle this problem in the English language. Nevertheless, in contrast, very little has been done in the (Brazilian) Portuguese language. This research is a small step toward filling this void. Despite being a simple approach (basic LSTMs), it proved rather efficient in classifying fake news with a reasonable performance. For future work, we intend to explore more advanced models for embeddings, such as, BERT or ELMO as well as using more sophisticated structures such as attentional mechanisms.

References

1. Shao, C., Ciampaglia, G.L., Varol, O., Yang, K.-C., Flammini, A., Menczer, F.: The spread of low-credibility content by social bots. Nat. Commun. **9**(1), 4787 (2018)
2. Vlachos, A., Riedel, S.: Fact checking: task definition and dataset construction. In: Fact Checking: Task Definition and Dataset Construction (2014)
3. Allcott, H., Gentzkow, M.: Social media and fake news in the 2016 election. J. Econ. Perspect. **31**(2), 211–236 (2017)
4. Golbeck, J., et al.: Fake news vs satire: a dataset and analysis. In: Proceedings of the 10th ACM Conference on Web Science, Amsterdam, Netherlands (2018)
5. Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: a data mining perspective. SIGKDD Explor. Newsl. **19**(1), 22–36 (2017)
6. Tandoc, E.C., Lim, Z.W., Ling, R.: Defining “Fake News”. Digital J. **6**(2), 137–153 (2018)
7. Fallis, D.: A functional analysis of disinformation. In: A Functional Analysis of Disinformation, iSchools edn (2014)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
9. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press (2016)
10. Ferrara, E.: Disinformation and social bot operations in the run up to the 2017 French presidential election (2017)
11. Rashkin, H., Choi, E., Jang, J.Y., Volkova, S., Choi, Y.: Truth of varying shades: analyzing language in fake news and political fact-checking. In: Truth of Varying Shades: Analyzing Language in Fake News and Political Fact-Checking (2017)
12. Hanselowski, A., et al.: A retrospective analysis of the fake news challenge stance detection task. arXiv preprint arXiv:1806.05180 (2018)
13. Lazer, D.M., et al.: The science of fake news. Science **359**(6380), 1094–1096 (2018)
14. Vosoughi, S., Roy, D., Ara, S.: The spread of true and false news online. Science **359**(6380), 1146–1151 (2018)
15. Zannettou, S., Caulfield, T., De Cristofaro, E., Sirivianos, M., Stringhini, G., Blackburn, J.: Disinformation warfare: understanding state-sponsored trolls on twitter and their influence on the web. arXiv preprint arXiv:1801.09288 (2018)
16. Wang, W.Y.: “Liar, Liar Pants on Fire”: a new benchmark dataset for fake news detection. arXiv preprint arXiv:1705.00648 (2017)
17. Ruchansky, N., Seo, S., Liu, Y.: CSI: a hybrid deep model for fake news detection. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, Singapore, Singapore (2017)

18. Pfohl, S.R.: Stance detection for the fake news challenge with attention and conditional encoding. In: Stance Detection for the Fake News Challenge with Attention and Conditional Encoding (2017)
19. Ajao, O., Bhowmik, D., Zargari, S.: Fake news identification on twitter with hybrid CNN and RNN models. Proceedings of the 9th International Conference on Social Media and Society, Copenhagen, Denmark (2018)
20. Chollet, F.: 'Keras'. In: Book Keras (2015)
21. Abadi, M., et al.: TensorFlow: large-scale machine learning on heterogeneous distributed systems. In: TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems (2016)
22. Appleyard, J., Kocisky, T., Blunsom, P.: Optimizing performance of recurrent neural networks on GPUs. In: Optimizing Performance of Recurrent Neural Networks on GPUs (2016)
23. Chetlur, S., et al.: cuDNN: efficient primitives for deep learning. arXiv preprint arXiv:1410.0759 (2014)
24. Lei, T., Zhang, Y., Wang, S.I., Dai, H., Artzi, Y.: Simple recurrent units for highly parallelizable recurrence. In: Simple Recurrent Units for Highly Parallelizable Recurrence, pp. 4470–4481 (2018)