Deep Learning for Fake News Classification

Miguel Molina-Solana
Data Science Institute
Imperial College London
London, UK
m.molina-solana@imperial.ac.uk

Julio Amador
Business School
Imperial College London
London, UK
j.amador@imperial.ac.uk

Juan Gómez Romero
Dept Computer Science and AI
Universidad de Granada
Granada, Spain
jgomez@decsai.ugr.es

***Abstract*—This work presents the application of several Deep Learning techniques for Natural Language Processing to the classification of tweets into containing fake news or not. To validate our approach, we use an open-access dataset containing annotated tweets related to the 2016 US elections. From our experiments, we can confirm that Deep Learning techniques are indeed able to identify tweets containing fake news, and that LSTMs with pre-computed embeddings is the best performing among the tested techniques (validation AUC = 0.70), particularly in avoiding misclassification of the minority class.**

***Index Terms*—deep learning, fake news, 2016 US elections**

I. Introduction

Since the 2016 US elections, the term 'fake news' has become mainstream and is nowadays commonly used to refer to pieces of information that are misleading, controversial or plainly inaccurate. This explosion has brought the attention of academics and practitioners from several fields, in an attempt to better understand the phenomenon and its causes.

Although the concept of fake news —as deliberately misleading pieces of information— is nothing new, the wide availability of social networking, publishing platforms, and general access to communication tools, has enabled their ultimate wide-spreading, overtaking the usual process of editorial curating.

Surprisingly enough, the acute characterization of fake news and its proper definition is elusive, remaining a fundamental question to be answered [11]. Cultural backgrounds, previous knowledge and ultimate interpretation of motives behind fake news play a prominent role on the interpretation of what it is and what it isn't a fake news, with plenty of academics sidelining the difficulty by focusing on the simpler issue of *false news* (to refer to those that have been fact-checked).

While diverse, the reasons for promoting fake news get often reduced to two [3]: pecuniary and ideological. Their impact, if widely embraced, believed and shared, can indeed be quite high as suggested by several pundits and academics in relation to the US presidential election [10].

With fake news becoming commonplace and being cheap to be generated, tools to flag controversial pieces of information are most welcome. Some approaches to identify fake news and their effects on behaviour have been suggested [8]. However, the question of how viral fake news effectively differ from other type of viral content remains unanswered.

Although focused on news stories and their mentions in tweets, a recent study from Vosoughi et al. [16] offers some insights as to how false news (as opposed to fact-checked verified news) might get spread. In particular, they report that falsehood diffused farther and faster than the truth despite structural elements of the network, not because of them.

On the other hand, Deep Learning models and techniques have demonstrated great performance and a very high potential in the recognition of complex patterns in several fields, especially in Computer Vision and Natural Language Processing. These models, which closely resemble the organization of neurons in the brain, mark the evolution of Neural Networks in an era of large data and very high computational power.

Neural Networks perform a non-linear transformation of the input values to the output values by means of several layers of interconnected computing units —i.e. the neurons. Their key approach is achieving learning by example: they take a (large) set of samples as training data, usually with already known labels, and automatically extract the relevant features that can be used to distinguish among classes, thus yielding a model able to classify new unknown samples. To do so, the training process applies an optimization algorithm to adjust the model parameters in order to minimize the network error.

In this context, it looks sensible to apply Deep Learning techniques to the task of classifying unseen pieces of information into containing fake news or not. The work reported here precisely looks into this, trying to offer some insights into the validity of this idea and its ultimate performance.

The paper is organized as follows. Next section introduces the dataset and algorithms we have used for the study. Section III presents the different experiments we carried out and their results and implications. The work finalizes with some pointers for further work.

II. Materials and methods

To test our hypothesis, we used a dataset containing tweets collected during 4 months just after the 2016 US presidential election [5], starting on November 8th. This dataset includes tweets that got re-tweeted more than 1000 times, and offer two sets of manual annotations for each tweet into fake news or not (Table I), attending to the categories established by [8]. For simplicity, the rest of this paper will consider a tweet as *fake* if it has been labeled as such by at least one annotator of the first or second team.

		2nd Label		
		Other Tweets	Fake News	Unknown
1st Label	Other Tweets	6482	1444	330
	Fake News	213	133	7
	Unknown	250	98	44

TABLE I
CONTINGENCY TABLE REPORTING THE DIFFERENCES AND THE SIMILARITIES BETWEEN THE LABELLING PERFORMED BY THE TWO TEAMS IN THE USED DATASET.

The complete dataset includes 9000 tweets and 20 variables, including the text of the tweet itself. After removing registers with empty values, we have 8336 tweets, with 6445 belonging to the negative class (*not fake*) and 1891 to the positive class (*fake*). Note that this procedure differs from the one reported by [4], which only takes into account the labelling by the second team.

We focus on the task of classifying tweets into fake news or not using tweet contents. The aim of this paper is to explore different approaches to build a classifier capable of tagging unseen viral tweets into fake news or not, and particular, those involving deep learning techniques. To do so, we will also limit ourselves to solely work with the actual textual content of the tweet.

Feed-forward Networks are the most basic and common type of neural networks. In these networks, computation moves in one direction, from the input to the output. The typical model is the perceptron, which defines several layers of fully-connected neurons. Model training is performed by backpropagation, a supervised learning algorithm in which the training loss, obtained as the difference between the expected result and the model output, is minimized after iterative adjustment of the model weight values. How the weights are adjusted is determined by an optimization algorithm such as the gradient descendent algorithm —and its variants for large data processing in Deep Learning; e.g. Stochastic Gradient Descendent (SGD), RMSProp, Adam, AdaGrad, etc.

Recurrent Neural Networks (RNNs) are a type of networks aimed at processing sequential data. Essentially, this type of networks compute a result not only from an input sample, but also by considering its previous state. Therefore, the model output for two equal samples can be different depending on the state of the network. Typically, RNNs process an input sequence x_{t-1}, x_t, ... to produce an output sequence o_{t-1}, o_t, ... Among them, long short-term memory (LSTM) networks [15] are particularly appropriate to deal with data sequences and time series with time lags of unknown size and duration between important events.

Formally, LSTM units are composed of a cell, an input gate, an output gate and a forget gate (see diagram in Figure 1). The cell is responsible for 'remembering' values over arbitrary time intervals —hence the word 'memory' in LSTM—, which are also passed towards the next units. Internally, the LSTM unit combines the input values, the values received through the input gate and the memory values to calculate the output values and the output gate values, taking as well into consideration the forget gate values.

Finally, and in order to validate our deep learning models built by following the LSTM approach, we compare their performance against traditional classification algorithms using only meta-data about each tweet and author; i.e. number of retweets, number of friends, etc. Specifically, we will use Random Forests [6], which are known to generally perform well in instance classification problems.

The software used for the experimentation is the R framework. We have used the caret package for creation and validation of the Random Forest classifiers [9] and the randomForest package for the underlying implementation of the algorithms [12]. For the deep learning techniques, we have used keras [1] with the tensorflow backend [2].

Due to our dataset being highly unbalanced (as shown in Table I), accuracy is not a proper metric to measure performance of the models. Instead we turn our attention to the ROC (receiver operating characteristic) curve and the AUC (area under the curve) measure, because they give a better overview of how true positive rate and false positive rate trade off.

III. Experiments and Discussion

A. Classification with metadata and Random Forest

In our first experiment, we have used the meta-data columns —all but the tweet text— to classify the tweets by the Random Forest algorithm. To do so, we have removed columns describing tweet id, user id, tweet creation date, and software used to publish the tweet. The ratio of training and validation datasets was set to 80-20%. After exploring a parameter grid through several experiments, parameter mtry, representing the number of variables randomly selected at each split of the forest generation process, was set to 4.

The results of this experiment can be seen in Figure 2, which shows the ROC curves of the classifiers trained with: (a) original data; (b) data augmented with the SMOTE algorithm [7] —using 200 and 100 as percentages for upsampling and downsampling, respectively. The threshold value for the class probability is automatically calculated by the randomForest package.

The AUC values can be seen in Table II. Results with the original data are unsatisfactory because of the strong bias of the classifier towards the majority class, and hence, the low sensitivity values. For this reason, although the AUC value of the classifier without augmentation is slightly better, we would select the classifier with augmentation as the baseline result to improve (AUC = 0.66, sensitivity = 0.74, specificity = 0.51).

The variable importance of the Random Forest classifier according to Gini measure the is shown in Figure 3. The results are consistent with the observations in [4] and [16]: the most descriptive variables are the number of retweets (i.e. the *virality* of the tweet) and the number of followers of the author. Note that other parameters, such as whether the user is verified or not, are not as relevant as they might be expected.

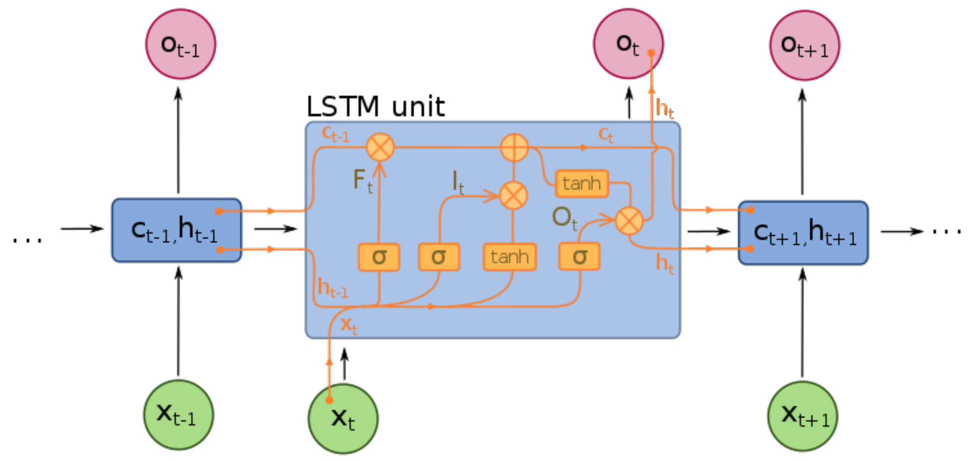

Fig. 1. Diagram of a LSTM unit

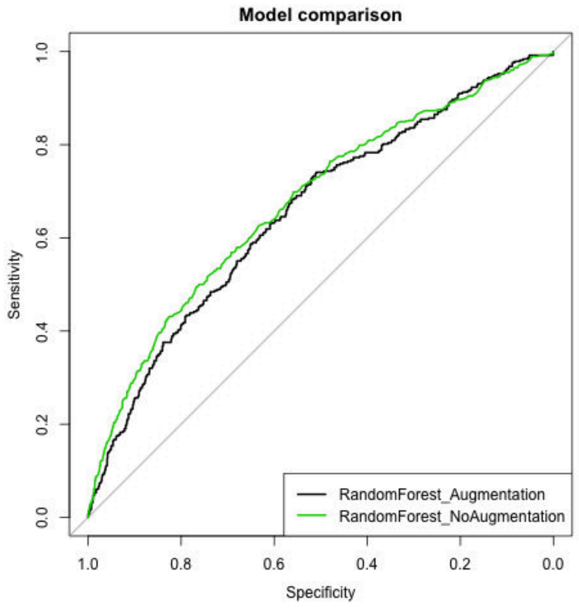

Fig. 2. ROC curves for the Random Forest classifiers

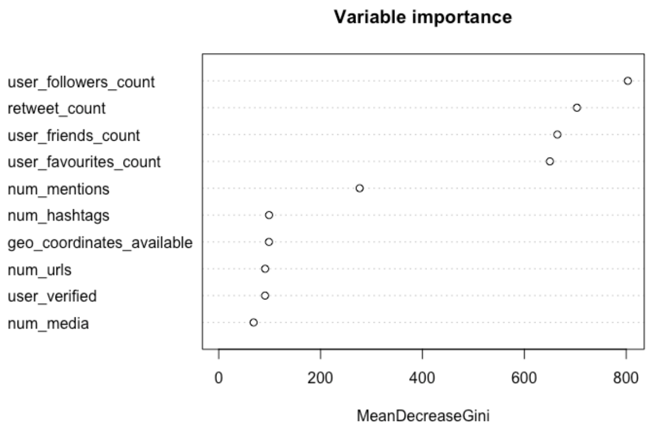

Fig. 3. Variable importance according to the Random Forest model with data augmentation

B. Classification with text and Deep Learning

In the second set of experiments, we have firstly used a feed-forward network to classify tweets from their contents. Secondly, we have implemented a LSTM network. In both cases, it is required to encode the input text into a numeric input, and then train the network to recognize if a tweet is a fake news or not.

There are several alternative approaches to perform the text-to-number encoding. A straightforward approach is one-hot encoding, in which texts are transformed into number vectors after the steps below:

1) Tokenize the tweets to extract all the words used in the texts.
2) Select the top-k used words ($k = 10,000$ in our case, from a total number of $21,421$ tokens).
3) Create a binary matrix where each column represents a top-k word and each row represents a tweet. Set the corresponding value $m_{i,j}$ to 1 if the word j appears in tweet i; otherwise, set it to 0.

This approach has several limitations, in particular because the sparse nature of the encodings resulting after step 3, which makes it difficult to train the classification model. Recently, it has been shown that it is better to encode texts as word embeddings. With this technique, we associate an n-dimension vector with each word, instead of a binary flag. Accordingly, a piece of text is a sequence of n-dimension vectors, being n typically 256, 512 or 1024. The translation from a word to a n-dimension vector is done in a way that the encoding somehow preserves the semantics of the original words; e.g. two encoding vectors corresponding to semantically related words will be close in the embeddings space.

This implies that embeddings must be also learnt in some way; e.g. by using algorithms such as the popular *word2vec* by Mikolov et al. [13]. While learnt word embeddings would be specific of the problem domain, it is common to use pre-computed embeddings from large text corpora, thus avoiding the problem of reduced input data (as it happens in our problem) and long execution times (which require costly computational resources).

In this paper, we have used the Global Vectors for Word Representation (GloVe) version 1.2 [14]. From the publicly available editions, we have selected glove.6B, which is trained with Wikipedia 2014 and Gigaword 5 news dataset, and the 100-dimension embedding space.

After these considerations, we have performed the following experiments:

- Networks
  - Feed-forward networks (one embedding layer, two hidden layers and an output layer)
    - with self-trained embeddings
    - with GloVe word embeddings
  - LSTM networks (one embedding layer, a LSTM layer of 32 units and an output layer)
    - with self-trained embeddings
    - with GloVe word embeddings
- Tokenizer: default Keras text_tokenizer function
- Embeddings space size: 100
- Loss function: binary crossentropy
- Optimizer: RMSprop
- Epochs: < 10
- Batch size: 32
- Activation function for hidden layers: ReLU
- Activation function for output layer: sigmoid
- Training / validation: 80%, 20%

Note that, despite the simplicity of the network topologies, overfitting has frequently appeared in the experiments. In such cases, we have used the most accurate network obtained before detecting a significant decrease of the validation performance.

Figure 4 shows the ROC curves for validation of each one of the four models. It can be seen that the best model is the LSTM trained with GloVe word embeddings, which yields an AUC = 0.70 with sensitivity = 0.62 and specificity = 0.68 (see Table II for details).

C. Discussion

The main result from our work is that LSTM with pre-computed embeddings is the best performer among the techniques we have tested. It is particularly interesting that the ratio of false negatives, the main problem in an unbalanced dataset like ours, obtained with this technique is lower than in the other cases.

However, the overall improvement compared to a Random Forest using tweets' meta-data is not extremely high, as we can see in Figure 5. This poses the question as to how much is worth the extra computation incurred by LSTMs.

More interestingly, a more detailed study of the errors incurred by each model should be performed, in order to identify if both classifiers are orthogonal to some degree. If this is the case, the next step should be to build an ensemble method using these two classifiers, and to determine a proper aggregation function. From our initial studies, we anticipate that double-counting the positive detections (tweets including fake news) could improve the current results.

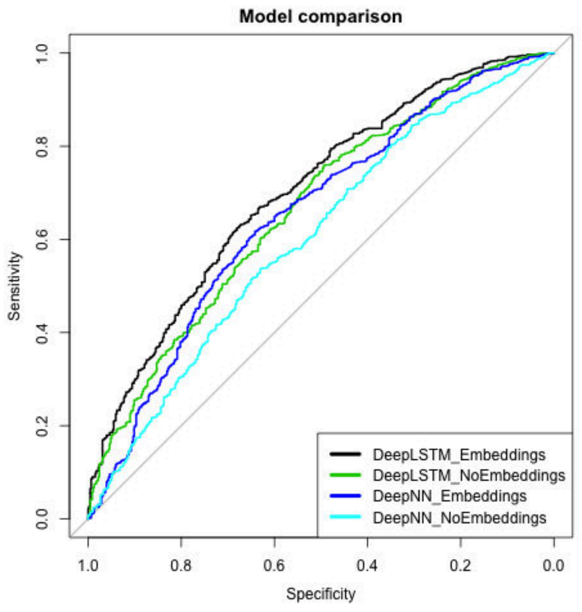

Fig. 4. ROC curves for the Deep Learning classifiers

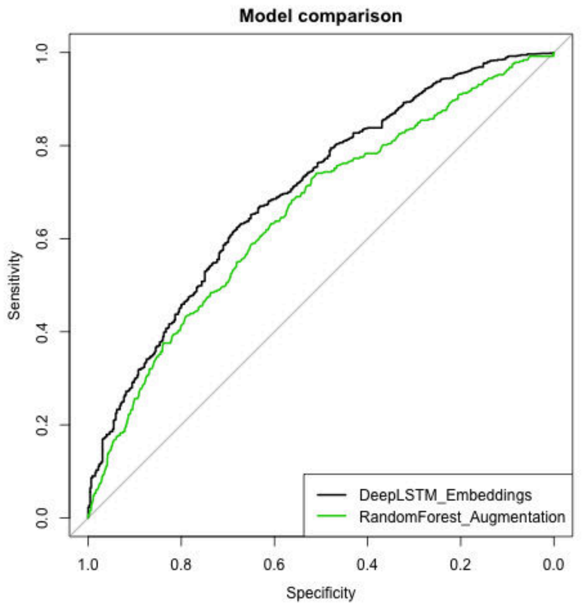

Fig. 5. ROC curves for the two best classifiers

IV. Future work

In this work, we have shown some preliminary results with off-the-shelf deep learning techniques applied to tweets' text in order to identify fake news. Even without a fine-grained adjustment of hyper-parameters nor a long training phase, results are quite promising over the tested dataset.

TABLE II
EXPERIMENT RESULTS

		Training				Validation			
Algorithm	Version	Accuracy	Sensitivity	Specificity	AUC	Accuracy	Sensitivity	Specificity	AUC
RandomForest	NoAugmentation	1.00	1.00	1.00	1.00	0.71	0.49	0.77	0.68
RandomForest	Augmentation	1.00	1.00	1.00	1.00	0.56	0.74	0.51	0.66
DeepNN	NoEmbeddings	0.99	1.00	0.99	1.00	0.56	0.54	0.63	0.60
DeepNN	Embeddings	0.96	0.98	0.95	1.00	0.62	0.62	0.64	0.65
DeepLSTM	NoEmbeddings	0.95	0.95	0.94	0.99	0.70	0.76	0.49	0.67
DeepLSTM	Embeddings	0.76	0.77	0.76	0.83	0.64	0.62	0.68	0.70

We plan applying more complex Deep Learning models (including a more extensive adjustment of parameters), and couple them together with a text pre-processing stage in order to better clean the text (e.g. currently, we are not considering text elements such as emojis and hyperlinks). It can be also helpful to tune the embeddings representation, either by exploring other pre-calculated transformation or by training on large corpora of only Twitter data.

Furthermore, there seems to be opportunity for ensemble methods that take into account both the meta-data of tweets and their actual text. Adding together the classification capabilities of Random Forests on the meta-data and LSTMs on the text appears as a very promising line of action, which we are currently exploring.

Finally, this study (and hence its results) is tightly coupled to and heavily dependent on the employed dataset. A replication of the experiments on a complementary one is necessary to further validate the current findings strengthening the conclusions and eventually leading to further insights.

All in all, the application of Deep Learning techniques to the task of identifying fake news in social media looks like a very relevant and timely application.

Acknowledgements

The authors want to thank the support of the Data Science Institute, Imperial College London. M. Molina-Solana is receiving funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 743623. Juan Gómez-Romero is supported by University of Granada under the Young Researchers Fellowship, and received funding from the Spanish Ministry of Education, Culture and Sport under the José Castillejo Research Stays Programme.

References

[1] J. Allaire and F. Chollet. *keras: R Interface to 'Keras'*. R package version 2.1.6.9001.

[2] J. Allaire and Y. Tang. *tensorflow: R Interface to 'TensorFlow'*. R package version 1.5.0.9001.

[3] H. Allcott and M. Gentzkow. Social media and fake news in the 2016 election. Technical Report 23089, National Bureau of Economic Research, 2017.

[4] J. Amador, A. Oehmichen, and M. Molina-Solana. Characterizing Political Fake News in Twitter by its Meta-Data. *arXiv*, 2017.

[5] J. Amador, A. Oehmichen, and M. Molina-Solana. Fakenews on 2016 US elections viral tweets (November 2016 - March 2017). http://dx.doi.org/10.5281/zenodo.1048826, 2017.

[6] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[8] N. J. Conroy, Y. Chen, and V. L. Rubin. Automatic deception detection: Methods for finding fake news. In *Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community*, pages 82:1–82:4, 2015.

[9] M. K. C. from Jed Wing, S. Weston, A. Williams, C. Keefer, A. Engelhardt, T. Cooper, Z. Mayer, B. Kenkel, the R Core Team, M. Benesty, R. Lescarbeau, A. Ziem, L. Scrucca, Y. Tang, C. Candan, and T. Hunt. *caret: Classification and Regression Training*, 2018. R package version 6.0-79.

[10] R. Gunther, E. C. Nisbet, and P. Beck. Fake news may have contributed to trump's 2016 victory. Technical report, Ohio State University, 2018.

[11] D. M. J. Lazer, M. A. Baum, Y. Benkler, A. J. Berinsky, K. M. Greenhill, F. Menczer, M. J. Metzger, B. Nyhan, G. Pennycook, D. Rothschild, M. Schudson, S. A. Sloman, C. R. Sunstein, E. A. Thorson, D. J. Watts, and J. L. Zittrain. The science of fake news. *Science*, 359(6380):1094–1096, 2018.

[12] A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[14] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[15] S. H. J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[16] S. Vosoughi, D. Roy, and S. Aral. The spread of true and false news online. *Science*, 359(6380):1146–1151, 2018.