Fake News Detection Using LSTMs
Fundamentals of Data Science 20-21 Prof. Galasso

Shayan Alipour - Alessio Orlando - Michele Miranda
Sapienza Università di Roma
orlando.1792394@studenti.uniroma1.it
miranda.1802232@studenti.uniroma1.it
alipour.1889595@studenti.uniroma1.it

Abstract

This project addresses the issue of fake news detection using Long Short-term Memory architecture. Our specific goal was to identify the fake news using text and title of the articles. We used NLTK to extract tokens, lemmas, POS and GloVe for lemmas embedding in order to feed data with more features and improve the results. Our best F1-score was **99.97**% with precision **99.96**% and recall **99.98**%.

1 Introduction

Nowadays we're bombed from each and every side with information and it's really tough to distinguish between true and false. People don't always have time or will to verify news and debunkers are not enough to handle the amount of fake news that is spreaded on a daily basis all over the world. In most cases this fake news address some political agenda and are used by politicians as a tool to gain power and people's support. Some analysts even say that fake news spreading helped the election of Donald Trump in 2016.

For this reasons, fake news spreading is one of the biggest issues of modern democracies. There already are debunkers which work to verify news and help people to distinguish the truth, but too often they're not quick enough. This is the main reason why debunkers are not enough to solve this issue: fake news spread hugely quicker and wider than the debunking news, making the debunking as it never existed. So we need a quick way to detect fake news and prevent them from spreading and that's where Machine Learning comes to our aid.

Nowadays Neural Networks are successfully used for many text processing tasks, so we thought we could use a NN-based model to solve this problem. The aim of this project is to use a Neural Network with a Long Short Term Memory architecture to classify the articles in fake and true.

2 Dataset

We have chosen the Kaggle Fake and real news dataset which contains 22851 fake news and 21192 true news collected between 31/03/15 and 19/02/18.

As we can see in the table, in the csv, each article has four columns, which are title, text, subject and date. Of course there isn't any correlation between the date a news comes out and its truthfulness, so date is irrelevant to us. As for the date, the category is not useful to establish if a news is true or fake. As we can see, there's no intersection between the true news categories and the fake news categories, so in our case the feature would actually be useful to detect fake news, but generally speaking, category is not a very good feature to use and can cause a lot of noise, therefore we decided to not use it.

So from the csv we only took title and text of the article and concatenated them to have only one text for each article and we added a label to flag them as true or false.

3 Data Preprocessing

At this point we had one only unsegmented text for each article and an associated label. Before being able to feed the neural network with the texts we needed to clean the text, tokenize it and extract the lemma for each token. A raw text contains a lot of punctuation, which is basically noise for us, so we removed punctuation first. Then we used NLTK (Loper and Bird, 2002) a powerful library for text processing, to tokenize the text, extract lemmas and, in a second version of the project, to extract

Title	Text	Category	Date
As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017
U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017
Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017
FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017
Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017

Table 1: True News

Title	Text	Category	Date
Donald Trump Sends Out Embarrassing New Year'...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017
Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017
Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017
Trump Is So Obsessed He Even Has Obama's Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017
Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017

Table 2: Fake News

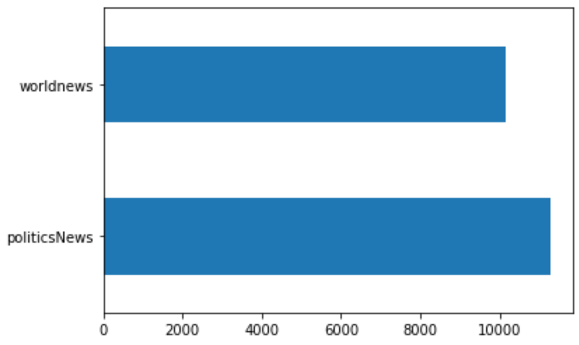

Figure 1: True News

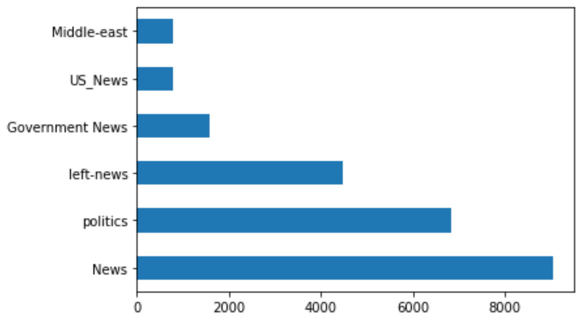

Figure 2: Fake News

the POS (Part Of Speech) tags. Tokenize a text means dividing it in tokens (the single words), then extracting the lemma means take the base form of the token and the POS is a tag which identifies the grammatical part of speech represented by the word, for example "running" would be tagged as "VERB". At this point we extracted the features: lemma and POS.

Since we couldn't feed the Neural Network with strings, we needed to embed lemmas and pos into vectors. For words, we used GloVe pretrained vectors. GloVe (Pennington et al., 2014) is a word representation technique that helped us assign vectors to each word in which the distance between vectors is related to the words' semantic similarity. For parts of speech instead, we associated a random initialized vector to each POS.

Original	RUBIO Sides With...
Tokens	['RUBIO', 'Sides', 'With'...]
Lemmas	[rubio, side, with...]
POS	['NNP', 'NNP', 'IN'...]

Table 3: Dataset Features

3.1 Batching

In order to feed data into the model, we need to create batches. That being said, we had to figure out different article lengths in the dataset because, in each batch, the length of the sentences should be the same.

Considering the fact that most of the articles have a length of over 1000 words, we defined a max length and truncated articles longer than that, and add padding to shorter ones.

4 LSTM Model

Since we're dealing with sequential data, in which the order matters, we used an LSTM (Hochreiter and Schmidhuber, 1997), which stands for Long short-term memory. A particular recursive neural network architecture composed of a cell that stores the memory and three gates that regulate the flow of information through the cell, for the purpose of removing some of the vanishing gradients problems in long sequences. In order to improve our results, we used a bidirectional LSTM that goes through a sequence both in a forward and backward way.

4.1 Shallow LSTM

As for our base model, we considered a single-layer LSTM, following by a projection layer to map LSTM output to our desire number of classes which are two labels true of fake. It goes without saying that we had an embedding layer in the beginning in order to use generated weights based on GloVe. The base model LSTM's layer has the size of 64 hidden units, and it was a simple LSTM, not bidirectional. The performance and more detail about it is provided in the results section.

4.2 Deep LSTM

The Deep LSTM architecture has two bidirectional LSTM layers with 128 and 64 hidden units followed by a sequential layer of three fully connected layers. We had an embedding layer consist of part of speech tags and lemmas. ReLu activation used for the first two dense layers. We tried several hidden units in order to achieve better results. Figure 1 shows the architecture of the Deep LSTM model.

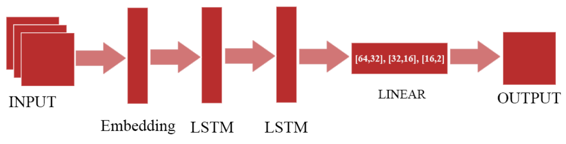

Figure 3: Deep LSTM Architecture

5 Experiments/Results/Discussion

For hyperparameters, we conducted several hand-tuning such as dropout rate, learning rate, batch size, max length of article and etc. We use Cross-entropy loss which is commonly applied as a loss function for classification problems.

We observe that after 20 epochs models started to overfit, however training them for 15-20 epochs made the results better. So after few tests we started training models for at least 15 epochs. We observe also that by decreasing the batch size we can achieve a more stable version of our models, so we set the batch dimension at 64 instead of the initial 256. Other tuning has been tried and particular observation can be done regards of the maximal length of the article. We initially tried a modest length of 300 and that gave honest results. We then tried to increase and decrease that value. We observed that by increasing the value the recall was dropping really quickly, one of the most probable cause is the vanishing gradient due to the really large text given to the model (400 words). When decreased, instead, we didn't really saw a change in the ratio between precision and recall, but we did saw a decrease of the general accuracy of the model, we hypothesized that was a cause of the lack of information of that short part of the article (200 words).

We used ReLu as an activation function for our projection layer in the DeepLSTM model. Last but not least, Adam Optimizer is used to minimize the loss function during back-propagation(Kingma and Adam, 2014). Adam proved to be one of the best choices for deep learning and has benefited from previous optimizers.

We also tried different version of our models,

Model	Features	Epochs	Batch	Max. Len.	Bi-dir	LR	Precision/Recall	F1 Score
Shallow	1	10	256	300	False	0.001	95.38, 96.14	95.76
Shallow	1	10	256	300	True	0.001	98.72, 95.88	97.28
Shallow	1	10	256	200	True	0.001	94.57, 93.27	93.91
Shallow	1	10	256	400	True	0.001	96.38, 55.324	70.29
Shallow	2	15	64	300	True	0.001	99.63, 99.44	99.54
Deep	1	10	64	300	True	0.001	99.89, 99.96	99.92
Deep	2	20	64	300	True	0.0005	**99.96, 99.98**	**99.97**

Table 4: Hyperparameter Tuning Results

bidirectional/unidirectional and multi-features. We can see how, as predicted, the usage of a bidirectional LSTM helped us to achieve better results (2% in the f1 score). The multi-feature version also added performance to our model. Our best model, in fact, is both bidirectional and multi-feature. In particular the model uses the lemmatisation as first feature and add the POS tagging as a second feature in the multi-feature version.

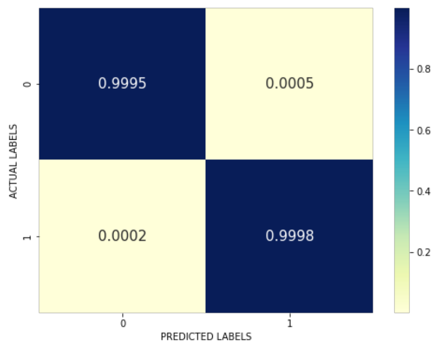

Figure 4: Best model confusion matrix heat map

5.1 Models Result

In the tests table, ”Model” refers to the type of the model used: shallow or Deep LSTM, ”Bi-dir” refers to models with bidirectional LSTM, ”Max Len” refers to the maximal length of the article used in that particular model.

6 Conclusion

In this project, we investigate out the performance of LSTM models in Fake news detection task by constructing multiple models with different variants. We concluded that the usage of the bidirectional version of the models combined with the lemmatisation of the words and little more information (like POS tagging in our case) embedded with word vectors improved the result. According to our results, using article too long or too short hasn’t been an appropriate strategy in fake article classification problems with LSTM. More other strategies can be tried and studied. Splitting the articles instead of cutting them, trying slightly different models or even changing to a transformer model can increase the performance. Since the score is so high, it would be really difficult to see the effectiveness of other models, although probably with another dataset, less categorizable, the score wouldn’t be so high and there would be more room for improvements.

References

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

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

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.