Detecting Fake News Using Deep Learning

Avinash Chandrasekaran
University of California Berkeley

Abstract

The proliferation of misleading information in everyday access media outlets such as social media feeds, news blogs, and online newspapers have made it challenging to identify trustworthy news sources, thus increasing the need for computational tools able to provide insights into the reliability of online content [i]. In this project, we will focus on automatic identification of fake content in online news. We examine two different approaches. First, we use deep learning algorithms on word embeddings to classify news articles as this method allows deep learning scientists to focus on model architecture. Second, we extract linguistic features and use machine learning techniques to classify articles. This approach aids linguists and domain experts to focus on feature engineering. Finally, we compare results from these two experiments with the model employed by Perez-Rosas et al, the authors of the paper which was the source of our datasets [i].

1. Introduction and related work

Fake news detection has recently attracted a growing interest from the general public and researchers as the circulation of misinformation online increases, particularly in media outlets such as social media feeds, news blogs, and online newspapers. Since as many as 62% of U.S. adults consume news on social media [ii], being able to identify fake content in online sources is a pressing need. Currently there have been few ways for dealing with fake news that have shown good results. Fact-checking approaches rely on automated verification of propositions made in the news articles. Other works have used social network activity (e.g., tweets) on a specific news item to assess its credibility, for instance by identifying tweets voicing skepticism about the truthfulness of a claim made in a news article. However, this is not a straightforward task, as external sources might not be available, particularly for just-published news items.

The linguistic approach attempts to identify text properties, such as writing style and content, that can help to discriminate real from fake news articles. The underlying assumption for this approach is that linguistic behaviors such as punctuation usage, word type choices, part-of-speech tags, and emotional valence of a text are rather involuntary and therefore outside of the author's control, thus revealing important insights into the nature of the text. While there already exist tools and products to detect sources of fake news (e.g., whether a web site publishes misleading news) [iii], we will approach this problem as an instance of text classification, using only the content of the article as the source of features. Doing so affords us to focus on NLP-related algorithms.

NLP methods for fake news detection have focused on using linguistic-based features pertaining to the content of the news article. Some of the most important linguistic-based features fall into four categories: lexical, syntactic, psycholinguistic, and readability. Lexical features include character and word level properties, such as total words, characters per word, and frequency of large or unique words. Syntactic features include sentence-level properties, such as frequency of words (i.e. “bag of words”), frequency of phrases (i.e. “n-gram”), punctuation and parts of speech tagging. Sentiment falls into the category of psycholinguistic features. The final group includes things like sentence readability and complexity [i]. All of these features have shown the ability to be used to detect fake news.

NLP methods for fake news detection have also focused on using deep learning. Deep learning typically involves converting the words in a news article into vectors of numbers, and then using those numbers as inputs into neural network frameworks that learn patterns that are predictive of fake news. We will focus on implementing a few different versions of neural networks and compare their performance to the model that relies on linguistic based features.

2. Fake News Datasets

To conduct our experiments, we will rely on the data and models used in the paper titled “*Automatic Detection of Fake News*” written by researchers at the Universities of Michigan and Amsterdam, to establish a target accuracy for our algorithms. These researchers used two datasets that both contain news, that is labeled as fake or not, to train models that rely on linguistic features and can be used to predict the label of other pieces of news. They used the following features: ngrams, punctuation, psycholinguistic, readability and syntax. With these, they achieved an accuracy of 0.74 on the “FakeNewsAMT” data set, and 0.76 on the “Celebrity” data set.

Datasets used in previous work have relied either on satirical news which also have confounds such as humor or irony; or used fact-checking websites which are typically focused on only one domain. The researchers used two novel datasets covering six different domains (sports, business, entertainment, politics, technology and education). The dataset contains an even distribution between fake and real news items, therefore we use a random baseline of 50% as reference value. Through our experiments we hope to show that different classifiers and algorithms score well above the baseline and provide a comparison with original author’s scores.

Data Set 1: FakeNewsAMT

This data contains 240 records of excerpts, usually two or three paragraphs, of legitimate news from mainstream news websites including ABC News, CNN, USA Today, New York Times, Fox News, and Bloomberg. The content was manually fact-checked in order to verify its authenticity. 240 records of fake news were manually generated by Amazon Mechanical Turk (“AMT”) workers who were instructed to write one piece of fake news that imitates the topic, length, and journalistic style of each piece of legitimate news. The fake news had to have all proper nouns found in the real news. The resulting data set includes 480 records, consisting of an equal proportion of real and fake news

Dataset	Class	Entries	Average Words/Sent	Words
FakeNewsAMT	Fake	240	132/5	31,990
	Legitimate	240	139/5	33,378
Celebrity	Fake	250	399/17	39,440
	Legitimate	250	700/33	70,975

Fig 1: Class distribution and word statistics for datasets

Data Set 2: Celebrity

This data contains 500 news articles related to public figures, since they are frequently targeted by rumors and fake reports. The sources of information include Entertainment Weekly, People Magazine, and Radar Online. The data was evaluated using gossip-checking sites and collected in pairs of fake and corresponding legitimate stories. The resulting data set has an even distribution of each classification [i].

We split each of these data sets into three distinct sets, producing a 60:20:20 split for training, validation and testing.

3. Deep Learning Models

The researchers use a linear SVM classifier using five-fold cross validation, with accuracy, precision, recall, and F-score as performance metrics using machine learning algorithms implementation available in caret & e1071 package with their default parameters. In this project, we will explore the following different models: Long Short-Term Memory (LSTM): One layer and Two layers, Gated Recurrent Unit (GRU) and BERT Transformers

3.1 Long Short-Term Memory

The first deep learning network we applied is a Recurrent Neural Network (“RNN”) with Long Short-Term Memory (“LSTM”) units. RNNs have shown to be one of the most effective algorithms for classifying sequential data, because at each step in a sequence they are able to effectively remember important patterns from previous steps (iv). News has a temporal aspect to it, because a word used in a sentence depends heavily on words used before and after. In our RNN, each news article represents an input sequence, and each word in the article is associated with a specific time step. LSTM cells have input, output, and forget gates that are particularly good at remembering patterns over arbitrary time intervals, because they regulate the flow of information in and out of the cell. LSTM cells are typically better than traditional RNN cells at encapsulating information about long range dependencies, because traditional cells are more sensitive to the distance between steps, due to exploding and vanishing gradients $_{(x)}$.

3.2 Gated Recurrent Unit

A Gated Recurrent Unit ("GRU") is another unit that can alleviate the vanishing gradient problem inherent in a standard RNN. A GRU is like a LSTM in the way it works, as well as in its performance at classifying sequences of data $_{(xi)}$. A GRU is different because it uses update and reset gates. Our implementation of the GRU network hyperparameters is the same as the LSTM network.

3.3 BERT Transformers

Through the use of a transformer "attention mechanism", we can create better language representations since the longer relationships in the sequence can be captured. We leverage a powerful but easy to use library called SimpleTransformers to train BERT $_{(xii)}$. Simple Transformer is a wrapper around the Transformers library from Hugging Face whose main goal is to abstract away many of the implementation, technical details around Transformer models.
We only used the base hyperparameters from the library without much tuning

For LSTM we tested a few variations of hyperparameters and adopted ones that resulted in the best performance on the validation data. Our final networks capped the sequence length of the FakeNewsAMT data at 200 words, and the Celebrity data at 500 words, since most of the articles were under these lengths, apart from some outliers. We converted each word to a 50-dimensional vector by training word embeddings that minimized the loss function of the entire network. We used a hidden layer of 100 neurons. We believe smaller hidden layers performed worse because they could not detect certain data patterns, and larger layers performed worse due to overfitting the training data. We decided to use Adam Optimizer with a learning rate of 0.0001. We initially used a larger rate, which resulted in the undesirable behavior of divergence from the loss function minimum, meaning there was a quick drop, followed by an increase, in loss. Finally, we used a dropout rate of 0.5 which resulted in a network that is capable of better generalization and is less likely to overfit the training data. Dropout is a technique where randomly selected neurons are ignored during training

4. Linguistic Feature Engineering

We attempted to replicate Perez-Rosas et al's approach for feature engineering by building four sets of linguistic-based features (ngrams, punctuation, readability, and syntax).

Ngrams: We extracted unigrams and bigrams and weighted the extractions with TF-IDF to account for occasional differences in content features

Punctuation: We included punctuation as complete words in the text. Thus, the punctuation in the articles were counted and normalized in the n-gram portion. For example, the phase "bus!" would be counted as two units for the unigram and bigram extraction: "bus," "!," and "bus !."

Psycholinguistic Features: We skipped adding any psycholinguistic context to the articles. LIWC assigns words to 73 categories in a binary format. Some examples of concepts embodied in these categories include: "affiliation," "achieve," "power," "reward," and "risk." Due to the lack of freeware, we chose not to utilize this

Readability: We calculated the readability metrics$_{(xi)}$ "Flesch-Kincaid, Flesch Reading Ease, Gunning Fog, and the Automatic Readability Index (ARI)" as well as "The SMOG Index," "The Coleman-Liau Index," "Linsear Write Formula," "Dale-Chall Readability Score," and a readability consensus based on all of these listed metrics. We compute all 10 of these readability metrics using a single computing library, "textstat"

Syntax: Like our approach with punctuation-based feature building, we simply counted and normalized syntactic patterns seen in the text. We did this by tagging the part-of-speech ("POS") of each word, replacing the word with the POS, extracting unigrams, bigrams, and trigrams of the resulting "sentence," and finally weighting those extractions with TF-IDF.

We used a Random Forest classifier with optimized hyperparameters in both instances. We chose Random forest because of its flexibility and since it provides a pretty good indicator of feature importance. Another consideration was that Random forest does not suffer from overfitting problem as it takes the average of all predictions, which cancels out the biases.

5. Results

The best result was obtained from the classification performed by the machine learning algorithm using manually extracted linguistic features. The scores outperform the baseline SVM classification by as much as 3% in each of the datasets. This increase in performance can be primarily attributed to variations of model hyper-parameters since we used very similar features.

Table 1: Results with Linguistic Feature Engineering

FakeNewsAMT dataset	Celebrity Dataset
77%	80%

The best performing deep learning model was the BERT transformer with accuracy score of 65% for the FakeNewsAMT dataset and 67% for the celebrity dataset. We didn't tune any hyperparameters with BERT and believe we could have gotten better accuracy with attempting several different parameters. In all cases the deep learning models were less effective than the SVM classification baseline model. Further the FakeNewsAMT dataset fared poorer compared to Celebrity dataset. This is likely because the data from FakeNews was manually created by

MTurk workers who deliberately wrote fake articles that had the same topic, length and style as the corresponding news articles. On the other hand, Celebrity data contains a more accurate representation of fake news, since all these were collected from the web. Generators of fake news typically prioritize writing articles that incite emotional responses over writing articles that replicate real news content. Another potential reason why deep learning methods performed worse on FakeNewsAMT is because the articles were cut down to excerpts of around two to three paragraphs. We used a longer sequence length of 500 words on the Celebrity data, which allowed the models to detect more patterns.

Table 2: Parameters and Results for Dataset with deep learning

Data	Model	Dimensions	Sequence Length	Learning Rate	Optimizer	Hidden Memory size	Dropout Rate	Epochs	Accuracy
FakeNews	LSTM	50	200	0.0001	Adam	100	0.5	35	55% (2-layer)
	GRU	50	200	0.0001	Adam	100	0.5	45	54%
	BERT		128	4e-5	Adam			2	65%
Celebrity	LSTM	50	500	0.0001	Adam	100	0.5	35	63% (2-layer)
	GRU	50	500	0.0001	Adam	100	0.5	45	54%
	BERT		128	4e-5	Adam			2	67%

Finally , we believe our deep learning models were also restricted by the number of examples in each dataset. Each set contained around 500 total articles, with which we used around 300 for training. This is not a lot of data, and neural networks tend to learn better with more data. Larger data sets are available but are classified based on the source (i.e. source A is not reputable therefore every piece of news from source A is fake). This concerned us because non-reputable sites could produce real news, and reputable sites could produce fake news. The field of fake news detection could benefit from a larger corpus of individually labeled articles being made publicly available.

6. Conclusion

With this project we performed an exploratory analysis of 3 distinct deep learning models to perform the task of fake news detection. Further we compared the results with linguistic based feature engineering and machine learning classification of the same datasets. The need for data preprocessing and feature engineering to improve performance of deep learning is not uncommon. Even for image recognition, where the first deep learning success happened, data preprocessing can be useful. Finding the right color space to use can be very important for instance. Max pooling is also used a lot in image recognition networks. The conclusion is simple: many deep learning neural networks contain hard coded data processing, feature extraction, and feature engineering. They may require less of these than other machine learning algorithms, but they require some still.

This does not suggest deep learning should be ignored. It can detect patterns that may not have not been picked up by our engineered features. It also has the added benefit of not requiring domain expertise in linguistics, which was necessary to identify useful features. We believe the most robust models will combine both approaches and would like to explore this combined NLP approach in our future work. One of the key limitations in our project was the dataset size. We believe we may have achieved better results with deep learning models if the dataset was sufficiently large. One other additional tradeoff to consider will be the computational complexity between these approaches. We noticed that in order to run BERT transformers we had to substantially beef up our compute resources which can be a limiting factor in some cases.

The future of fake news detection will involve other important methods that can be used in addition to NLP. These methods consist of social context and fact-checking approaches [iii]. Social context revolves around user-driven engagements of news on social media platforms (e.g. credibility of users who post and interact with an article, content of comments on an article, etc.). Fact-checking is a knowledge-based approach that aims to verify the truthfulness of the major claims in an article, through vehicles such as experts, crowdsourcing and algorithms. The future of fake news detection will not be one-size fits all. The most effective fake news classifiers will enlist all these approaches.

7. References

(i) Veronica Perez-Rosas, Bennett Kleinberg, Alexandra Lefevre, Rada Mihalce. *Automatic Detection of Fake News*, 2018. http://aclweb.org/anthology/C18-1287

(ii) Samir Bajaj. *Fake News Detection Using Deep Learning*, 2017. stanford.edu/cs224n/2.pdf

(iii) Kai Shu, Amy Silva, Suhang Wang, Jiliang Tang, Huan Liu. *Fake News Detection on Social Media: A Data Mining Perspective*, 2018. https://arxiv.org/abs/1708.01967

(iv) Sepp Hochreiter and Jurgen Schmidhuber. *Long Short Term Memory*, 2604.pdf

(v) Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, Yoshua Bengio. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*, arxiv.org/1412.3555, 2014.

(vi) Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, https://arxiv.org/abs/1810.04805, 2018

(vii) Yoon Kim. *Convolutional Neural Networks for Sentence Classification*, arxiv.org/5882, 2014.

(viii) Sairamvinay Vijayaraghavan, Ye Wang, Zhiyuan Guo, John Voong, *Fake News Detection with Different Models*. https://arxiv.org/pdf/2003.04978.pdf, 2020

(ix) Kai Shu, Suhang Wang, Dongwon Lee, Huan Liu, *Mining Disinformation and Fake News: Concepts, Methods, and Recent Advancements*, https://arxiv.org/pdf/2001.00623.pdf, 2020

(x) Álvaro Ibrain Rodríguez, Lara Lloret Iglesias, *Fake news detection using Deep Learning,* **https://arxiv.org/pdf/1910.03496.pdf**, 2019

(xi) Shivam Bansal, Chaitanya Aggarwal. https://pypi.org/project/textstat/

(xii) Priya Dwivedi, SimpleTransformers, Classification_with_Simple_Transformers.ipynb