Fake News Stance detection

Suraj Raju Guntimadugu

University of Waterloo

Abstract

Social networks and online news media are gaining popularity in recent years. Meanwhile, online fake news is becoming widespread. As a result, automating fake news detection is essential to maintaining robust online media and social networks. Fake news, defined by the New York Times as “a made-up story with an intention to deceive”, often for a secondary gain, is arguably one of the most serious challenges facing the news industry today. The primary goal of Fake News Challenge (FNC) [1] is to deploy Natural language processing (NLP) techniques to address the issues arising due to fake news. Even for skilled specialists, classifying news stories as a hoax or true tales can be challenging due to the sheer volume of information that must be taken into account. In this work, machine learning methods are employed to detect the stance of newspaper headlines on their bodies, which can serve as an important indication of content authenticity. If the newspaper headline is defined to be “unrelated” to their bodies, it indicates a high probability of the news to be “fake”. Multiple techniques were employed to extract the relevant features for stance detection from data collection of headlines and news articles with different stances. These features are then used to train machine learning models. The primary objective of this experimentation is to out-perform the baseline model which is implemented using the Gradient Tree boosting technique.

1 Introduction

The growth of fake news is a massive problem because it steers the readers in the wrong path. Identifying such fake news is found to be very difficult even for experts in the domain. Recent improvements in the field of NLP provides a possible solution to automate this process. However, accurately identifying fake news is still proven to be difficult due to complex nature of human language. This complex task can be broken down into multiple stages, the first one being identifying the relevance of the article with the headline of the news, called as Stance detection.

We can consider FNC as a classification problem, because we are trying to predict the stance which is one of the 4 possible stances from the given dataset. There are multiple proven approaches that work for text data classification problems. We started with implementing a basic Convolution Neural Network (CNN) model. Results of CNN were decent but nowhere comparable to the baseline model performance numbers. Since it is text data and it requires semantics of the sentence, we chose to work with Recurrent Neural Networks (RNN). Long Short-Term Memory is one such RNN which can process single data points and entire sequences of data. We found that the LSTM was not performing well as the classification accuracy was way less than the desired metrics. Then we implemented a Bi-directional LSTM, an efficient extension to LSTM. As expected, the Bi-Directional LSTM yielded better metrics compared to a regular LSTM model.

We then explored the capabilities of BERT (Bidirectional Encoder Representation from Transformers) which is a recent paper published by researchers at Google AI Language [2]. BERT makes use of a transformer, an attention mechanism that learns contextual relations between words in a text. We then also explored an optimized version of BERT called RoBERTa (Robustly Optimized BERT), developed by Facebook researchers [3]. With the RoBERTa model we were able to beat the baseline accuracy numbers with FNC score of 10411.25.

2 Related Works

2.1 Baseline Model

The FNC implementation shown in [1] serves as base test accuracy for future implementations. This approach is a three step process. Firstly, the data set is pre-processed which includes, converting the text to lower case, removing numeric values, removing punctuations and stopwords. Then in second phase, the sentences are converted into list of words and then the corpus is lemmatized. Lastly, some important features are extracted from headings and article bodies. Features like the overlap between heading and body, refuting words in headlines, n-grams and cosine similarity are considered. Then, a Gradient Tree Boosting algorithm is used to train the classification model and that is sued for predictions on competition test data. This baseline implementation achieved a score of 8748.75.

2.2 Convolution Neural Network

Yang Shao [4] developed a CNN system for semantic text similarity detection. The semantic vector is generated by max pooling over all word vectors and the similarity score between two sentences is calculated using the semantic vectors of these sentences.

2.3 BoW MLP

Davis and Proctor [5] developed the model Bag of Word Multi Layer Perceptron for fake news detection. This model first converts the corpus into bag-of-word vectors, and then uses two softmax layers: one for relevance and one for attitudes and entropy-based cost function to make classification.

2.4 Stance Detection with Bidirectional Conditional Encoding

This Related work [6] considers the usage of a conditional encoding scheme with two bidirectional LSTM models for stance detection for tweets towards some target. The tweet and the target are processed by separate LSTM models and the first hidden state of the LSTM that processes the target is initialized with the final hidden state of LSTM that processes the tweet.

2.5 CNN-LSTM model with dimensionality reduction approaches

The authors in this work [7] employed dimensionality reduction techniques to reduce the dimensionality of feature vectors before inputting them into the classifier. The non-linear feature vectors are fed to PCA and chi-square which provides more contextual features for the classification task. This proposed model improved the accuracy by ~4%. Their experimental results have shown that PCA outperforms chi-square and state-of-the-art models with better accuracy.

2.6 Transfer Learning from Transformers

In this paper [8] the authors made efficient use of the generalization power of large language models based on transformer architecture, invented, and trained publicly released. They fine-tuned BERT, XLNet and RoBERTa transformers on FNC-1 dataset to obtain better accuracy metrics.

3 Dataset

In the given data, each instance consists of a headline, body and corresponding stance. Stance is one of the {'unrelated, 'agree', 'disagree', 'discuss'}. The whole set consists of 1648 distinct headlines, 1683 distinct articles and 49972 headline-article pairings. Headlines had various lengths ranging from 10 to 220 words, while articles had lengths ranging from 25 to 5000 words. This dataset is heavily biased towards unrelated pairs. The true class breakdown is around 73% 'unrelated', 18% 'discuss', 7% 'agree' and 2% 'disagree'. We have also preprocessed our dataset in order to split sentences, normalize casing, handle punctuations, and non-alphabetic symbols to improve token consistency. We also used vocabulary lists for terms occurring in either the headline or article texts and extracted corresponding pre-trained word embeddings. We have used tokens from the Stanford GloVe corpus(50d vectors from 6B twitter corpus) [9].

4 Methodology

The goal in approaching the stance detection problem was to experiment with a wide range of deep learning and NLP techniques that we have learned in the msci641 course, which includes:

- Word embeddings (Word2Vec, GloVe)
- Neural networks (CNN, RNN)
- Attention Mechanisms
- Transformer models

The following sub-sections will give an overview of the techniques and models that were experimented.

4.1 Word Embeddings

Any neural network should be fed with numerical data. So it is very important to represent the documents (sentences) in an appropriate numerical format without losing the semantic context. We have experimented with Word2Vec embeddings to generate the embedding vectors of the data, but found that the results were not promising because of the relatively less training data. So we made use of a pre-trained embedding model to get the vector representations of our data. We found that the results were better with GloVe [9] model which creates 50d vectors of the texts, developed by Stanford. We can also experiment with other higher dimension vector representations, but kept it simple at 50d to reduce computational complexity.

4.2 Convolution Neural Networks

CNNs are a type of artificial neural networks which uses perceptrons for cognitive tasks. The layers of CNN consist of an input layer, output layer and hidden layers that includes convolution layers, pooling layers, fully connected layers, and normalization layers. The filters in the convolution layers can establish the relationship between two words. In each step, the filter is multiplied with the word embedding matrix where the filter with values 1 results in word embedding values, while the filter values 0, results in 0. This way, filters are used to deduce the relationship between words. Stacking of convolution layers helps to extract more complex information from the dataset.

There are a few parameters which can be tuned according to the dataset we have for CNN models. We have used the categorical cross-entropy loss function since we have a multi-class output. The activation functions in the hidden layers are 'ReLu' functions which are known to perform well for sparse datasets. 'Softmax' function is used as an activation function for the output layer since the output is a multi-class categorical value. The optimizer used for CNN is 'adam' which is a state-of-the-art optimizer provided by Keras in TensorFlow that overcomes the drawbacks of its predecessors.

4.3 RNN (LSTM, Bidirectional-LSTM)

In addition to the CNN model, we consider an additional baseline in the form of a standard LSTM [10] model that processes a concatenation of tokens in the headline and the article body to produce a classification of the stance. No special token is used to delineate the transition between the header and the body in the concatenated input. However, the results were not very impressive. So we implemented a Bi-directional LSTM model. It has two networks, the first one gets the text data in the forward direction and the other network gets the data in the reverse direction. We can say that the network has access to past and future data. Basically LSTM model is a RNN competent in learning long-term semantic dependencies. Usually, LSTM models consume a lot of memory because of their sequential processing. So we trained the model by running 20 epochs, where we observed 94% of validation accuracy and a 19% validation loss.

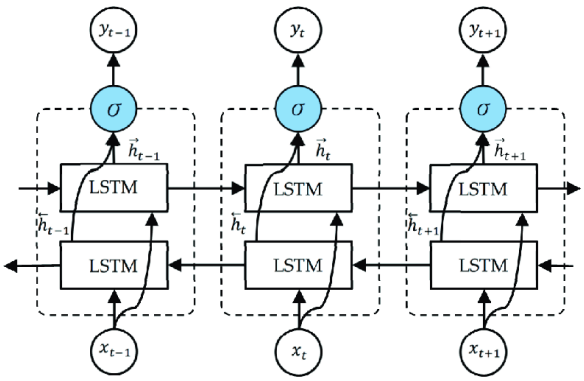

Fig 1. Bidirectional LSTM architecture

The concatenated sequence of words (headline and article body) is used as input to the network. The embedding layer in the network contains the weight matrix with word embeddings obtained using pre-trained GloVe50d model. We have used stacked activation layers after the bi-directional LSTM layer. We have used 'adam' optimizer, same as that of CNN model. The accuracy metrics were a little better compared to the regular LSTM model.

4.4 Transformer models

We made use of Simple Transformer [11] implementation of transformer based models. This library provides out-of-the-box implementations of models for text classification problems. Bidirectional Encoder Representations from Transformers (BERT) architecture is composed of several transformer encoders stacked together. Each transformer encoder is composed of two sub-layers: a feed-forward layer and a self-attention layer. BERT makes use of transformer that learns contextual relations between words in a text sequence. In contrast to the state-of-the-art models, transformer encoder reads the entire sentence at once as it is bidirectional and thus yields more accurate results.

For BERT, base uncased model was used with 12 layers and 12 attention heads and a total of 110M parameters. We have trained the transformer model until only 3 epochs since it is very computational heavy and takes a long time to train.

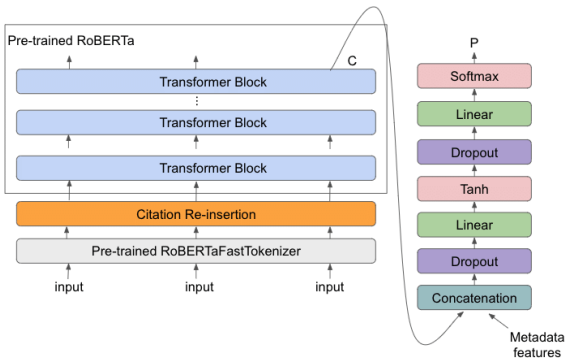

Fig 2. RoBERTa model architecture

RoBERTa builds on BERT's language masking strategy wherein the system learns to predict intentionally hidden sections of text within otherwise unannotated language examples. We have an already implemented version of this model from simpletransformers [11] library. We have set the sequence length of 512 and a learning rate of 1e-3 and trained over 3 epochs, considering the time taken to train the model.

5 Results

5.1 Scoring

To address the issue of class imbalance, the Fake News competition [1] will be scored on the basis of a weighted accuracy measure where 0.25 points are awarded correctly classifying pair as “unrelated”, 0.25 points are awarded for an incorrect classification if true label is one of “agrees”, “disagrees”, and “discusses”. 1 point is given if the correct classification is made for pairs with true labels of “agrees”, “disagrees”, and “discusses”. A script is provided to calculate F1 scores for our predictions. Below image (Fig 3) explains the algorithm used to calculate scores.

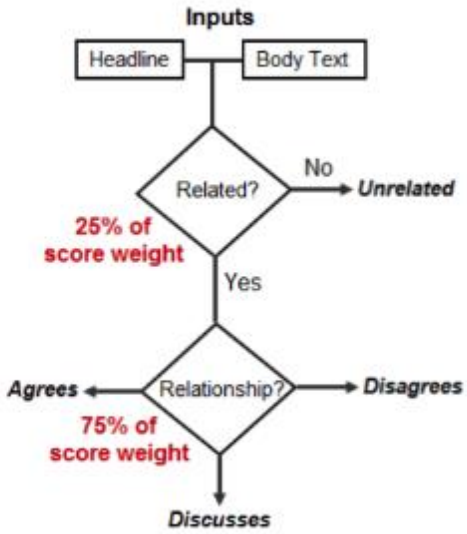

Fig 3. Scoring mechanism for FNC-1

5.2 Result Analysis

Amongst all the deep learning models we experimented with, CNN was the one train quickly as we can see from Table 1. Training time of RNN models were significantly high because they take sequential data as input. It ixs important to note that all the timings provided are dependent on the hardware used. We trained out models on Google Colab, with no additional computing power.

Model	Average training time per epoch (sec)
CNN	106
LSTM	190
Bi-directional LSTM	280

Table 1. Training times for deep learning models

Both CNN and RNN models failed to outperform the baseline implementation model. It is evident from the scores we achieved. This might be because of the fact that we are extracting only basic features from the training data, whereas baseline implementation used many features like overlaps, refuting words and polarity which provide more information. This can also because of the fact that we used GloVe embeddings which use shallow language models that pit a limit on extracted features.

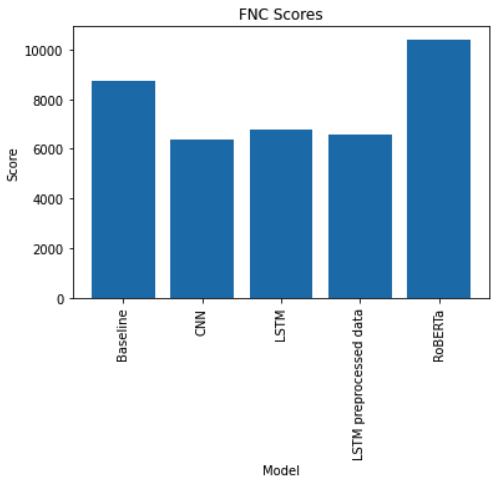

Fig 4. FNC scores histogram

Transformer models clearly outperformed the other neural network models. RoBERTa model performed the best amongst all the models we experimented with the calculated score of 10411.25.

6 Conclusion and Future work

In this project, we have investigated the performance of different classification algorithms on stance detection task proposed by FNC-1. We observed that transformer based RoBERT model performed best with an accuracy of 92.5%.

We have experimented with a basic version of models which are available out of the box on excessively preprocessed data. Further research would be conducted on using some dimensionality reduction techniques as proposed in [7] for deep learning models and evaluate the results. To counter the class imbalance, we can apply down sampling to the dataset. A custom loss function weighting heavily losses on minor classes may be a good solution for it. We could also explore further on Bidirectional LSTMs by considering the whole sequence of words since there might be important information at the end for a the articles which might change our predictions. We can also explore a little more on the BERT and RoBERTa models as they keep improving.

7 References

[1] FakeNewsChallenge, "Fakenewschallenge/fnc-1- baseline."

[2] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

[3] Liu, Yinhan, et al. "Roberta: A robustly optimized bert pretraining approach." arXiv preprint arXiv:1907.11692 (2019).

[4] Shao, Yang. "Hcti at semeval-2017 task 1: Use convolutional neural network to evaluate semantic textual similarity." Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). 2017.

[5] Davis, Richard, and Chris Proctor. "Fake news, real consequences: Recruiting neural networks for the fight against fake news." (2017).

[6] Tim Rocktaschel, Edward Grefenstette, Karl Moritz Hermann, Tom Ko ¨ cisk ˇ y, and Phil Blun- ´som. Reasoning about Entailment with Neural Attention. 9 2015.

[7] Umer, Muhammad, et al. "Fake news stance detection using deep learning architecture (CNN-LSTM)." IEEE Access 8 (2020): 156695-156706.

[8] Slovikovskaya, Valeriya. "Transfer learning from transformers to fake news challenge stance detection (FNC-1) task." arXiv preprint arXiv:1910.14353 (2019).

[9] Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.

[10] Luan, Yi, Yangfeng Ji, and Mari Ostendorf. "LSTM based conversation models." arXiv preprint arXiv:1603.09457 (2016).

[11] T. Rajapakse, "https://simpletransformers.ai/."

[12] Mahesh, R., et al. "Identification of Fake News Using Deep Learning Architecture." 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA). IEEE, 2021.

[13] Dulhanty, Chris, et al. "Taking a stance on fake news: Towards automatic disinformation assessment via deep bidirectional transformer language models

for stance detection." arXiv preprint arXiv:1911.11951 (2019).

[14] Antoun, Wissam, et al. "State of the art models for fake news detection tasks." 2020 IEEE international conference on informatics, IoT, and enabling technologies (ICIoT). IEEE, 2020.

[15] Küçük, Dilek, and Fazli Can. "Stance detection: A survey." ACM Computing Surveys (CSUR) 53.1 (2020): 1-37.

[16] Mrowca, Damian, Elias Wang, and Atli Kosson. "Stance detection for fake news identification." Eliaswang. Com (2017).