# Fake News Detection Analysis

Adithiyya Kishoore, Gaurav Madarkal

University of Colorado Boulder

**Abstract.** Fake news is a major issue in the current time, it has caused serious problems on a global scale in the recent past. This paper talks about some of the existing solutions, issues with such solutions and improvements that can be made for a better outcome. It also compares the different architectural patterns used in text classification such as RNNs and Transformers and the superiority of one over the other. It also aims to provide an opinion based on the results observed after performing these experiments. While it doesn't provide a conclusive result to whether any of these methods might solve the problem as a whole, it does shed light as to why the issue of fake news is more complicated than it seems to be at first glance.

**Keywords:** Fake News, Deep Learning, Text Summarization, Artificial Intelligence

## 1 Introduction

### 1.1 Motivation

What is fake news? It is the intentional and calculated presentation of false information or misinformation for the express purpose of misleading the reader. Even though this seems like a small issue at first, it can have catastrophic effects when deployed on a grand scale. One need not look further than the recent elections, where both parties were negatively affected by this, and more importantly the common man.

Fake news became a major cause for concern with the advent of social media,[1] when every single platform started to become a news provider, with it's own set of sources, whose veracity could not be objectively judged. As each and every news article couldn't be checked for truth fullness on a large scale, the idea of coming up with an automated solution came into picture using the technique of deep learning and AI.

If we are to successfully overcome the predicament, the average consumer would be able to make the right decisions that could benefit more than themselves.

### 1.2 Issues with Existing Solutions

The state of the art solutions that have been deployed and tested till now have all either categorized a news articles in a binary fashion. However, we cannot label news as fake or otherwise with little knowledge and context of the truth. Also, most of the fake news that gets proliferated has a semblance of truth to it, and requires common sense to judge correctly.

Existing solutions are mostly keyword based, due to which AI can easily be fooled. In fact, AI have been deployed to generate fake news with an agenda and is quite problematic.

### 1.3 The Proposal

The main aim of the project was to compare different techniques that could be used for classification. We decided to test LSTM, BERT and RoBERTa based approaches on the dataset and to test the models on the testdata as well as current real world data.

We would also parse the articles cited in the document by extracting the links and performing a simple fetch to compare the similitude between them. This would combat the issue of spurious websites citing popular articles whose content barely resembles the content in the article being judged.

### 1.4 Deep-Dive of the Data-Set

The dataset that is being primarily used is the *Fake and Real News Dataset* from Kaggle.[1] It consists of 44898 unique articles out of which 21417 were true articles and 23481 were false. This was the largest and most comprehensive of the datasets on Kaggle for fake news, and hence provided us the range of data we were aiming for. Therefore, we decided to use this dataset.

The dataset columns are distributed into *title*, *text*, *subject*, and *date of publication*. Some insights into the data appear to reveal the most frequent trigrams present in the dataset.

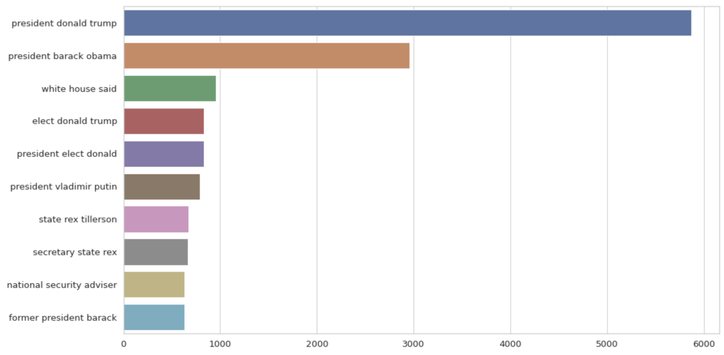

Fig. 1: Trigrams in Real Text

[1] `https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset`

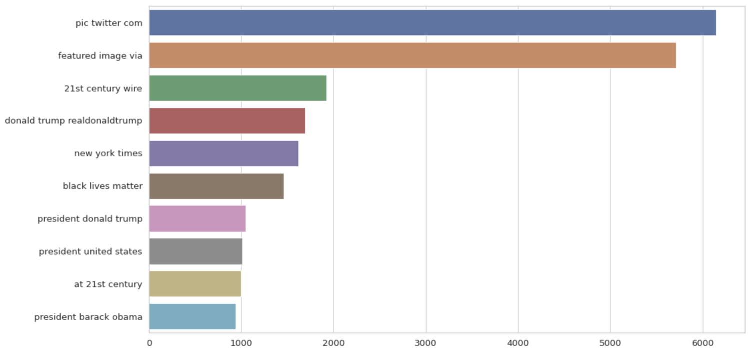

Fig. 2: Trigrams in Fake Text

## 2 Related Work

The core idea leveraged in this project comes from the paper *Attention is all you Need* which is one of the breakthrough papers detailing the method of self-attention to generate context while creating the embeddings that are passed to the deep-learning model.

### 2.1 Contributors of Fake News

To keep users engaged and maintain retention rate a number of fake news writers use provocative language for their own nefarious purposes, under this assumption a preliminary analysis can be done on the initial content of the article to determine if a further analysis is required. Text summarizing[2] techniques can be used to generate a gist of the article and run an initial analysis.

Though most of the Social Media users are genuine, there is a significant percentage of accounts which are focused with the sole purpose of proliferating fake information to the public. These accounts can mostly be broadly categorized as social bots, trolls and make-believe bot accounts which temporarily are taken control over by a human, which is defined by the special term cyborg users. As the barrier to creating fake social media accounts is not substantial, it is quite easy to do so.

A computer controlled account, which runs on the basis of an algorithm generates content and also engages in a conversation with real users without any human intervention. While social bots are not always created with the intention of causing harm, they can be leveraged by malevolent parties to contribute bountifully towards the creation of fake news.

While the ratio of trolls to genuine users is skewed in the favor of real users, trolls strategically place themselves, and act as an initiator or a stimulus that triggers the emotions of the general populace. This, is the cause for the fake news to go viral. The primary purpose of most of them is to spring up negative emotions such as fear and anger, that are suppressed in genuine users. This might change the thought processes of the users giving rise to doubt and distrust.

The most treacherous of these are the cyborgs, who are an amalgamation of automated activities with human input. These accounts can switch their operating mode between human and bot, according to their needs, which provides them with immense potential to spread false information.

As we are now aware of the reasons for fake news generation, let us delve into the next step, which is to combat it. There are primarily two methods to identify misinformation: linguistic methods, and network analysis.

## 2.2 Methods of Combat

Linguistic approaches involve Data Representation, Deep Syntax, Semantic Analysis. In this approach, subterfuge is studied with the help of various communicative manners. It is believed that method of communication by deceivers and truth-tellers is distinct. In written forms of communication, deceivers are inclined to have a higher character count as opposed to straight arrows. Further, the pronouns used by liars are more oriented towards other parties, rather than themselves. The writing style of the information manipulators is used against them in this approach.

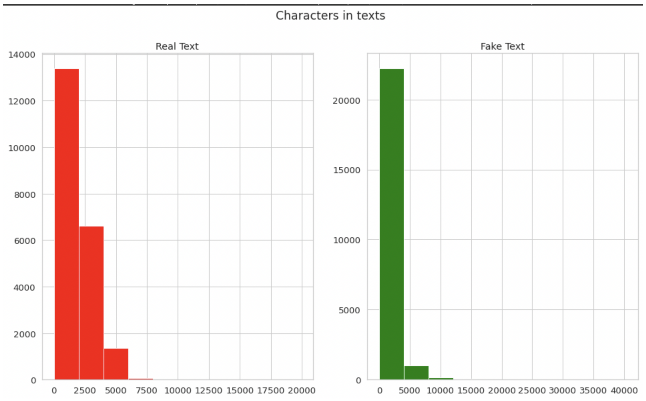

Fig. 3: Fake v Real Text - Article Lengths

Data Representation[3] methods use tokenization, which treats a word as a individual unit of significance and converts it into a number. Extending this, it utilizes parts of speech tagging and location-based approaches.

Semantic Analysis[4] is another approach where the veracity of the author is put to test by juxtaposing, what is written with the kind of information that would be present, if it were to come from a personal encounter. It can be assumed that the deceiver has little personal experience apropos to the matter at hand, it is highly likely that they would end up confuting their prior claims in the article. They might also omit crucial information that are present in related topics.

The infamous approach of sentiment analysis[5] uses opinion mining as a bedrock, that involves makes use of analytical techniques to study the text for patterns and sentiments. But this approach is not flawless as the problems of credibility and verification are not in the picture.

Network Analysis uses the interconnection between articles and links on the internet among the related topics, to arrive at a conclusion. Examples of this would be the page-rank algorithm which was the founding idea on which Google was built.

This project leverages a combination of Data Representation, Semantic Analysis and Network Analysis to judge the article under question.

### 2.3 Related Data-sets

For the purpose of this project a very popular Kaggle data set will be used, this data set is quite comprehensive in terms of the variety of the data and quantity for training. Most of the related work in this field has been around a binary classification as the end result, but this project aims to turn that into a confidence prediction algorithm, meaning, the end result would give an insight about how fake (evaluation metric) an article under question is.

## 3 Methods

In order to solve the issue of fake news, we broke the problem into its subparts. These sub-parts included an initial analysis of the data-set, general methods of cleansing the data-set, and specific methods of cleaning to be employed based on the initial data-set analysis.

### 3.1 Data Preprocessing

The dataset consisted of three important columns, 'Title', 'Text' and 'Subject'. The text field was concatenated with a standard keywords in News or articles such as "Breaking", "Just out", "Breaking news" and "Watch" etc... Such repeated words were removed because they caused the model to get biased and learn things unrelated to the actual content of the article. Further processing was performed by removing stopwords and HTML elements. A final cleaned text was obtained which was purely the important content of the article. The same process was performed on the Title field because it was appended to the main text to help the model learn better. Each of the below architectures were used to built three different models trained with just the Text field, just the Title field and Text, Title concatenated. The best performing model is discussed in further in the Results section below.

## 3.2 LSTM

One good reason to use LSTM[6] is that it is effective in memorizing important information. If we look at other non-neural network classification techniques they are trained on multiple word as separate inputs that are just words having no actual meaning as a sentence, and while predicting the class it will give the output according to statistics and not according to meaning. That means, every single word is classified into one of the categories. This is not the same in case of LSTM. In LSTM we can use multiple word strings to find out the class to which it belongs. This is very helpful while working with Natural language processing. If we use appropriate layers of embedding and encoding in LSTM, the model will be able to find out the actual meaning in input string and will give the most accurate output class.

The LSTM model built for analysis comprised of an Embedding layer with a maximum feature to be 10000 which is also the corpus size of the Tokenizer used. Glove 6B 50D embedding was used to create the embedding matrix. The first LSTM layers had a recurrent dropout of 0.1 and dropout value of 0.2. The second LSTM layer had a recurrent dropout value of 0.2 and dropout value of 0.15. These were the best dropout and recurrent dropout values observed after trying a few variations. Following the LSTM layers was a Dense layer with ReLU activation function, the choice of ReLU was made because of the gradient it offers, the output expected from LSTM model is a binary classification hence an Dense layer with 1 output unit, this layer is used with a Sigmoid Activation function. The binary classification model was build with a Binary Cross Entropy loss function with Adam optimizer. The model was built with the method of Reducing Learning Rate if the model learning stagnates by monitoring the validation accuracy.

The model was trained with a train/val/test split of 70/20/10 and training duration of 10 epochs. The model accuracy and loss curves are discussed in the following sections. The testing of this model was performed by gathering articles from content providers such as News websites, Bloggers and Social Media. To briefly discuss the results of this testing, the model performed poorly, and the basic conclusion that was made was that the model does not understand the article as a whole instead looks at parts of the document like individual sentences and tries to understand it.

This problem requires a more sophisticated and specific solution that is able to generate a gist of the document and then make a judgement. After extensive research and exploration of existing techniques in Natural Language processing the Transformers architecture seemed to be appropriate for this problem.

## 3.3 Transformers

Keeping in mind the above challenges of using LSTM, and the advantages of using self-attention,[7] Transformer based approaches were chosen. The most popular among these were BERT, which also had a few variant models that were built on top of it like RoBERTa.

## BERT

Generally, language models read the input sequence in one direction: either left to right or right to left. This kind of one-directional training works well when the aim is to predict/generate the next word. But in order to have a deeper sense of language context, BERT uses bidirectional training. Sometimes, it's also referred to as "non-directional". So, it takes both the previous and next tokens into account simultaneously.

BERT[8] applies the bidirectional training of Transformer to language modeling, learns the text representations. LSTM architecture lacked this feature which is one of the reason why it performs exceedingly well during training and poorly with unseen test data. Diving into the details of the model used, the bert-base-uncased model was used with the BERT Tokenizer to encode the data, a max length of 120 tokens was used while tokenizing the data. Text inputs were transformed to numeric token ids and arranged in several Tensors before being input to BERT.

A similar 70/20/10 split was used for train/validation/test datasets. The BERT model was trained for 10 epochs over GPUs. The pretrained BERTforSequenceClassification model was used. For fine-tuning the model was given the same optimizer that BERT was originally trained with: the "Adaptive Moments" (Adam). This optimizer minimizes the prediction loss and does regularization by weight decay (not using moments), which is also known as AdamW. In line with the BERT paper,[8] the initial learning rate was set smaller for fine-tuning to 5e-5.

The BERT model seemed to overfit during the training process which is a possibility based on the observed accuracy metric, the results of testing BERT model were better than LSTM but the observation made about model overfitting seemed to be true because of the behaviour of the model on unseen data, further analysis of the results is described in the following sections.

## RoBERTa

A robustly optimized[9] method for pretraining natural language processing (NLP) systems developed by Facebook that improves on Bidirectional Encoder Representations from Transformers, or BERT. This model was heavily trained in comparison with BERT and a better masking language modelling objective. Before the choice of RoBERTa was made the models such as DistillBERT and XLNet were explored, these two performed similar to BERT with slight variations in the training accuracy, for the scope of this paper the comparison is only between LSTM, BERT and RoBERTa.

Roberta for sequence classification with "roberta-base" pre-trained model was used. As discussed earlier RoBERTa model was trained with a higher magnitude data-set when compared to the BERT model, hence the tokenizer provided by RoBERTa was preferred. When compared to BERT the model parameters used for this model were the same. The results seen during training in this case were marginally better, but the model performed exceedingly better with the test data, but these results were better when compared with the other models discussed earlier, but in general these models cannot be used as a conclusive factor while deciding the truthfulness of an article.

## 4 Experimental Design

The models were trained and run on Google Collab with Tesla K80 GPU and all of the models were run for 10 epochs. The three methods that were discussed earlier were trained with variation in the model and as well as variations in the training data. The models were trained with different number of layers, activation functions (ReLU and Sigmoid) and number of epochs (5, 8, 10) in case of model parameters.

With respect to the training data, the text passed to the model was varied by giving it just the text field, the title and text concatenated together, and just the title field.

The best results were obtained when they were trained on title and text which had an accuracy of 0.82 on LSTM, 0.91 on BERT, and 0.97 on RoBERTa.

An experiment was conducted by parsing the text for hyperlinks in the dataset. Using these links an HTTP GET was performed to fetch the contents of the articles being cited which was then cleaned and later the fetched content was appended to the text as contextual data.

The reasoning behind this approach, was keeping in mind the potential behaviours that are usually exhibited by fake news authors, which would be to refer popular articles (having no connection to the content in the original article) that would result in a higher page rank to better reach the users. We hoped that this method used by fake news authors could be used against them. But this experiment was not fruitful as the model trained with additional contextual information achieved similar results as the model that was trained without the contextual information. This observation showed that the model did not learn anything new by performing this experiment.

### 4.1 LSTM

For LSTM based approach, the model trained with one, two and four *LSTM* layers of varying umber of units. The two LSTM layers were tested had a recurrent dropout an dropout values ranging from 0.1 to 0.35. The LSTM models were trained on 2, 5 and 10 epochs and the number of LSTM layers were varied from 3 to 5, but finally we settled on two LSTM layers as that was providing the best results.

The model also had two dense layers in all these cases, the first using ReLU and the second dense layer using Sigmoid. The size of the first dense layer was 32 units and the second dense layers had a single output value reflecting the binary classification output for this problem.

It was trained on the following hyper-parameters. *Binary_CrossEntropy* was used as the loss function, with *Sigmoid* as the activation function. *Adam* was used as the optimizer. The learning rate was 0.01.

### 4.2 BERT and RoBERTa

For BERT and RoBERTa, the uncased version of the models were utilized. Both the models used a learning rate of $5 * e^{-5}$ and decay value of $(\epsilon)$ of $1 * e^{-8}$. The max length for tokenization was 120. The optimizer was $AdamW$.

With the base BERT and RoBERTa models, we used the a wrapper called *BERTForSequenceClassification*[2] from the transformers library to train the model which added a dropout layer with a value of 0.1 and a linear layer that had 768 inputs, with a bias term which made the total number of inputs to be 1538 as we have two classification labels - fake and real.

### 4.3 Outputs of the models

To analyse our implemented model which is entirely trained on past instances of fake or real news, the model was tested on currently trending topics which are labelled as fake from trusted sources of media. An F1 score was be used as the target metric along with accuracy because it valued both precision and recall with equal weightage.

## 5 Experimental Results

The progression of the loss curves for training and validation the three models are displayed below over the ten epochs in figures 4, 5, and 6.

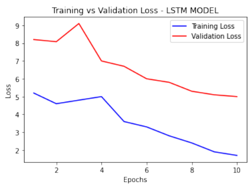

Fig. 4: LSTM: Training and Validation Loss

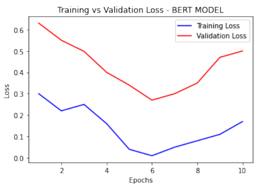

Fig. 5: BERT: Training and Validation Loss

[2] https://huggingface.co/docs/transformers/model_doc/bert

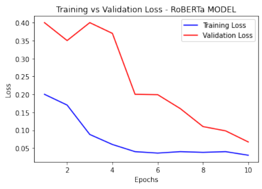

Fig. 6: RoBERTa: Training and Validation Loss

The confusion matrices for the three models are represented below in Figures 7, 8, and 9.

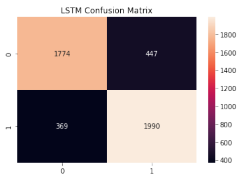

Fig. 7: LSTM: Confusion Matrix

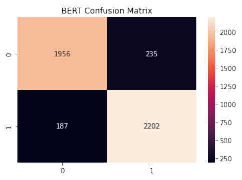

Fig. 8: BERT: Confusion Matrix

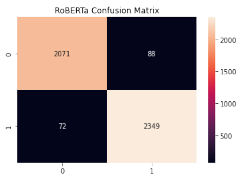

Fig. 9: RoBERTa: Confusion Matrix

Below are displayed the classification reports of the three models trained on both title and text when it was scored with the test data-set.

As can be seen from the tables below, LSTM had the worst results with an accuracy rating of 0.82 and an F1 score of 0.81 and 0.83 for fake and real news respectively.

BERT achieved an accuracy rating of 0.91 and an F1 score of 0.90 and 0.91 for fake and real news respectively.

RoBERTa was the best of the three models. It achieved an accuracy rating of 0.97 and an F1 score of 0.96 and 0.97 for fake and real news respectively.

Table 1: LSTM: Classification Report

| Class | Precision | Recall | F-score | Support |
|---|---|---|---|---|
| Fake | 0.80 | 0.83 | 0.81 | 2143 |
| Real | 0.84 | 0.82 | 0.83 | 2437 |
| Accuracy | | | 0.82 | 4580 |

Table 2: BERT: Classification Report

| Class | Precision | Recall | F-score | Support |
|---|---|---|---|---|
| Fake | 0.89 | 0.91 | 0.90 | 2143 |
| Real | 0.92 | 0.90 | 0.91 | 2437 |
| Accuracy | | | 0.91 | 4580 |

Table 3: RoBERTa: Classification Report

| Class | Precision | Recall | F-score | Support |
|---|---|---|---|---|
| Fake | 0.96 | 0.97 | 0.96 | 2143 |
| Real | 0.97 | 0.96 | 0.97 | 2437 |
| Accuracy | | | 0.97 | 4580 |

## 6 Conclusion

### 6.1 Analysis of Results

The LSTM model performed poorly as compared to the BERT based models as these models had the benefit of being trained on a large corpus, prior to training on the data. Also, the tokenization of the sentences were more sophisticated in BERT based models, with self-attention being a core factor that optimized the results.

In our case, more than two LSTM layers resulted in higher parameters and also over-fitting, which led to poor validation accuracy. Dropout values larger than 0.2, resulted in lower training accuracy as the models tooks longer to learn.

From the experimental results, we observed that BERT model was prone to over-fitting. However, this was corrected in RoBERTa. This might be because BERT used static masking, where the same words in a sentence were masked in all the epochs. However, RoBERTa made use of dynamic masking, which allowed it to switch up the words being masked in every epoch.

Another factor for better accuracy in RoBERTa was due to it being pre-trained on 160 GB of data, while base BERT was just trained on 16 GB.

Table 4: RoBERTa: Classification Report - Real World News

| Class | Precision | Recall | F-score | Support |
|---|---|---|---|---|
| Fake | 0.50 | 0.53 | 0.51 | 2143 |
| Real | 0.56 | 0.53 | 0.55 | 2437 |
| Accuracy | | | 0.53 | 4580 |

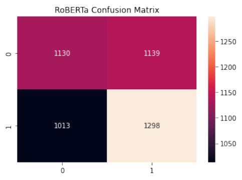

Fig. 10: Roberta: Confusion Matrix - Real World News

## 6.2 Analysis on Current Data (Real-World News)

While the models performed exceedingly well with what they were given, they were abysmal when it came to predicting anything that was a little out of the way from what they had come across.

On current news or unseen data that was not part of the dataset, none of the models performed well. A total of nine models were tested, having variations in two categories. The first variation was on the text passed to the model for training, being it just title, text and title together and just text. The second variation was the architecture of the model: LSTM, BERT or RoBERTa.

The RoBERTa model which was trained on both title and text performed best though it had an accuracy rating of of only 54%. Statistically speaking, this was akin to a coin-toss, as the models just didn't have enough information flow or context to determine what was true and what is not.

This leads us directly to the belief that this problem cannot have a definitive solution at this stage with the approaches we currently have. One cannot say whether a piece of text is "fake or not", a program can only suggest how much of the article might be true based on the limited knowledge it has.

However, it will be possible when we achieve artificial general intelligence which is the singularity we might achieve someday.

## References

1. Stahl, K.: Fake news detection in social media. (2018)
2. Shirwandkar, N.S., Kulkarni, S.: Extractive text summarization using deep learning. In: 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA). (2018) 1–5
3. Liu, Z., Lin, Y., Sun, M. In: Representation Learning and NLP. Springer Singapore, Singapore (2020) 1–11
4. Maulud, D., Zeebaree, S., Jacksi, K., M.Sadeeq, M., Hussein, K.: A state of art for semantic analysis of natural language processing. Qubahan Academic Journal **1** (03 2021)
5. M, I.P., Srikanth, G.U.: Survey of sentiment analysis using deep learning techniques. 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT) (2019) 1–9
6. Wang, J.H., Liu, T.W., Luo, X., Wang, L.: An LSTM approach to short text sentiment classification with word embeddings. In: Proceedings of the 30th Conference on Computational Linguistics and Speech Processing (ROCLING 2018), Hsinchu, Taiwan, The Association for Computational Linguistics and Chinese Language Processing (ACLCLP) (October 2018) 214–223
7. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need (2017)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2018)
9. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach (2019)