Fake News Classification

Team 17
Anubhav Sharma - 2018114007
Arushi Agarwal - 2019201015
Shreya Vanga - 2019201087
Yash Bhansali - 2018101068

Abstract

Fake news, deliberate misinformation, hoaxes, satires etc are various ways to mislead people in order to damage an agency, entity or person, and/or gain financially or politically. Fake news has been in the spotlight as its prevalence has increased with the rise of social media. In this project, we have experimented with various neural network and machine learning methods to classify news as fake or real.

1 Introduction

Nowadays, internet has become an integral part of our lifestyle.The role of traditional information channels such as newspapers and television has become less prominent on how we collect and consume information. Certainly the growth of social media platforms have played a major role here. A lot of people use social media platforms not only to keep in touch with their friends and family, but also to gather information and news from around the world. Indeed, social networks have proved to be extremely useful especially during crisis situations, because of their inherent ability to spread breaking news much faster than traditional media.

However, this positive impact of the social media comes at a cost: the absence of control and fact-checking over posts makes social media a fertile ground for the spread of unverified and/or false information. People often publish posts or share other people's posts verifying neither the source nor the information validity and reliability. Often times, an attractive headline is sufficient for an article to be shared thousands of times, despite a possibly unsubstantiated or false content.

Probably, one of the most striking examples of how fake news can influence opinions has been the U.S. presidential campaign in 2016. Nonetheless, we can argue that fake news and, more broadly, disinformation are becoming a huge problem on the web, and might have an important social cost in the future. Hence, we provide a comprehensive analysis of our approach to fake news detection using various detection techniques involving machine learning models and neural networks.

1.1 Research Problem

The project is concerned with identifying a solution that could be used to detect out articles containing fake news for purposes of helping users to avoid sharing the news. It is imperative that such solutions are identified as they will prove to be useful to readers.

1.2 Demonstration of the proposed solution

The proposed solution to the issue concerned with fake news includes the use of a web app framework Streamlit, which presents the user with an interactive dashboard where they can see the news classification in action. The user can enter the news title / content, which gets redirected to our best performing ML model in the back end and classifies the said news title/content into fake or real.

To run or deploy the Streamlit app from Google's Colaboratory, first install it using

```
!pip install streamlit
```

Then we'll have to use Ngrok for providing secure tunnels from the local system to the public. Follow these commands:

```
!wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip
!unzip ngrok-stable-linux-amd64.zip
get_ipython().system_raw('./ngrok http 8501 &')
!curl -s http://localhost:4040/api/tunnels | python3 -c \
'import sys, json; print(”Access the app from the following URL: ” +json.load(sy
```

Then simply execute the respective python file to deploy on the Streamlit app

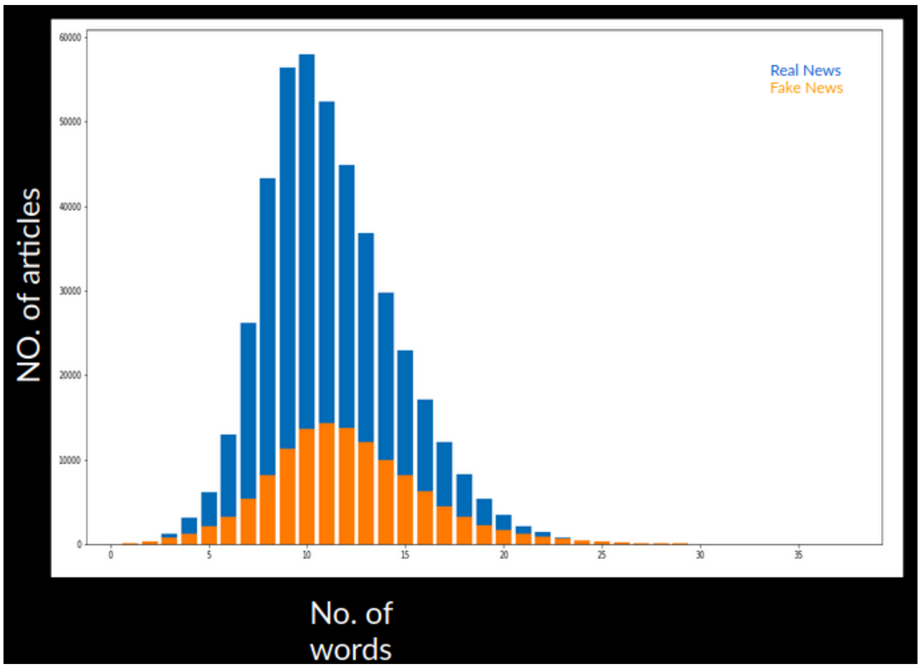
```
!streamlit run /content/file.py
```

2 Dataset Analysis

We have used NELA-GT-2019, a large multi-labelled news dataset, for the study of misinformation in news articles. This dataset contains 1.12M news articles from 260 sources collected between January 1st 2019 and December 31st 2019. Also, the sources have a wide range of mainstream news sources and alternative news sources.

In the dataset, the sources are classified into 3 classes, namely Reliable(0), Mixed(1) and Unreliable(2). For our research purpose, we have considered only the reliable and unreliable sources. We categorised all the articles of the class 0 (reliable) as ”real” and all the articles of the class 2 (unreliable) as ”fake”. Articles of the class 1 (mixed) were dropped. For the rest of the paper, fake news will be represented by class 1 and real news by class 0.

The dataset consists of 1,26,009 Fake articles and 4,46,251 Real articles in toto. The average length of title is 12 and the average length of body is 511. Majority of the titles will fall under the length of 25, as is evident from below graph.



Pre-processing was done on the data before we began with our experimentation. The pre-processing step included Tokenization, Dropping Null values, Stemming and Removal of Stop Words

3 Proposed Solution

For a given sample to be classified as fake or not, the model needs to learn the features for classification. Hence, we experimented with the traditional classifiers and some state-of-the-art classifiers like BERT and Roberta for Sequence classification using different hyperparameters. We tried different approaches such as running different classifiers for different features. These approaches were experimented with all the different dataset distributions that are mentioned in Section 2.
The initial baseline work was conducted on the Kaggle dataset (url) that consists of 21K samples. Preprocessing such as removing of null entries, stop-words removal, tokenizing and stemming was performed on the data. This preprocessed data was further used for training the model for classification. To process natural language text and to extract useful information from it, a sentence using machine learning and deep learning techniques requires the text to be converted into a set of real numbers and this process of representing words or phrases from vocabulary to a corresponding vector of real numbers is called vectorization. The preprocesed data is vectorized using TF-IDF (term frequency- inverse document frequency). The vectorzed data is fed into standard classifiers standard classifiers like SVM, Multinomial Naive Bayes Classifier for training the model. The accuracy and f1-scores of the baseline is as follows:

Model	F1-Score	Accuracy
Multinomial NB Classifier	0.900	0.891
Passive Aggressive Classifier	0.949	0.942
SVM	0.945	0.939

To improve the scores, we used different approaches based on the different representation of features. The different representations are as follows: a)Tf-idf b) Pretrained Glove-Embeddings c) Retrained Word2Vec Embeddings d) Transformers based Approach e) Nela Features f) Ensembled Approaches. Approaches used in each feature representation are mentioned the below subsection:

3.1 TF-IDF

TF-IDF, which stands for term frequency - inverse document frequency, is a scoring measure widely used in information retrieval (IR) or summarization. TF-IDF is intended to reflect how relevant a term is in a given document. We now experimented on the NELA-GT dataset that is briefly explained in Section 2. The preprocessing was done similar to the baseline work. Here we considered only the 'body' feature of the dataset for training. We then vectorized this feature using TF-IDF and this vectorized dataset was given as an input to classifiers like SVM (Support Vector Machines), Logistic Regression and Random Forest. The best accuracy and F1-scores of the classifiers used were obtained for these data distributions:

Random Forest Classifier

Dataset-Used	F1 score(weighted)	Accuracy
50k Samples (Equal Distribution)	0.816	0.833
180k Samples (Equal Distribution)	0.852	0.852

Support Vector Machines(SVM)

Dataset-Used	F1 score(weighted)	Accuracy
50K	0.782	0.767

Logistic regression

Dataset-Used	F1 score(weighted)	Accuracy
50K	0.766	0.761
100K	0.798	0.788

3.2 Pretrained Glove Embeddings

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. We used pretrained glove embeddings for vectorizing the dataset as glove is trained on a huge corpus of 840B tokens. Experimented with classifiers such as RNN (using Bi-LSTM), Logistic Regression and SVM. This model performed better than the TF-IDF model (Section 3.1) by achieving a better accuracy and F1-score. The results are as follows:

Logistic regression

Classifier-Data Distribution	F1 score(weighted)	Accuracy
NN 100k(normal)	0.800	0.801
SVM 50K(normal)	0.824	0.805
Logistic regression 50k(normal)	0.779	0.781

3.3 Transformers based Approach

As we know the drawbacks of classifiers like RNN which are very slow in computations, it becomes very difficult to process sequences that are very long. Hence transformers are used. It is a novel architecture that aims to solve sequence-to-sequence tasks while handling long-range dependencies with ease. The idea behind Transformer is to handle the dependencies between input and output with attention and recurrence completely. We use Google's BERT - (Bidirectional Encoder Representations from Transformers)

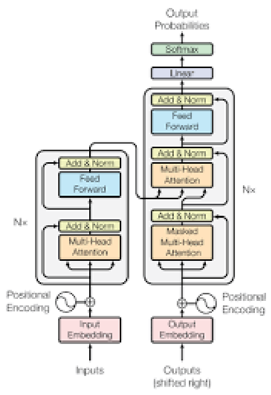


Figure 1: Transformer - Model Architecture

framework, a new language representation model from Google AI, uses pre-training and fine-tuning to create state-of-the-art models for a wide range of tasks. We also experimented with RoBERTa (Robustly optimized BERT approach) - , introduced by FaceBook, is a retraining of BERT with improved training methodology. The table below shows a detailed list of the variations of BERT and RoBERTa that we experimented with. The optimal paramater values of epochs are 2 to 4 (after which a spike in validation loss was observed), learning rate , optimizer - adam and batch size of 32 was kept constant.The following variations were used :

- Bert-base-uncased (12-layer, 768-hidden, 12-heads, 110M parameters)
- Bert-large-uncased(24-layer, 1024-hidden, 16-heads, 336M parameters)
- Roberta-base (12-layer, 768-hidden, 12-heads, 125M parameters)
- Roberta-large(24-layer, 1024-hidden, 16-heads, 355M parameters)

The results of the variations on various data distributions are as follows :

- Bert-base-uncased

Dataset	F1 score(weighted)	Accuracy
100k samples (Normal Distribution) (2 epochs,24)(title)	0.852	0.86
50k samples (Equal Distribution) (4 epochs)(title)	0.8691	0.8783
20k samples (Equal Distribution) (2 epochs)(head+title+tail)	0.8645	0.8647
20k samples (Equal Distribution) (2 epochs)(body)	0.8721	0.8777

- RoBerta-base

Dataset	F1 score(weighted)	Accuracy
60k samples (Equal Distribution) (4 epochs)(title)	0.8701	0.8718
100k samples (Normal Distribution) (2 epochs)(title)	0.8843	0.8872

- RoBerta-Large

Dataset	F1 score(weighted)	Accuracy
180k samples (Equal Distribution) (2 epochs)(title)	0.89205	0.89207
100k samples (Normal Distribution) (2 epochs)(title)	0.89116	0.89279
100k samples (Equal Distribution) (4 epochs)(title)	0.88218	0.88217
100k samples (Normal Distribution) (2 epochs)(title)(lr was halved)	0.89097	0.89279
180k samples (Equal Distribution) (2 epochs)(title)(padding till 48)	0.89490	0.89411

3.4 NELA Features

Feature representation was a difficult task as embeddings in general would not result in a dense representation of the entire article . So we decided to experiment with NELA Features, It has one such paramater that given a text it returns hand-crafted, text-based features for news veracity detection.The News Landscape (NELA) Features is an open source toolkit for the systematic exploration of the news landscape. The goal of NELA is to both speed up human fact-checking efforts and increase the understanding of online news as a whole.The features can be broken down into 6 groups: a. Style, b. Complexity, c. Bias, d. Affect, e. Moral, f. Event. The results obtained by using NELA Features on the entire dataset with various classifiers was as follows:

Dataset	Algorithm	Accuracy	F1-score
Entire Title	SVM	0.644	0.641
Entire Body	Logistic Regression	0.779	0.779
Entire Title	Logistic Regression	0.704	0.700
Entire Body	SVM	0.777	0.773

This is how we experimented with different data representations, different classifiers and their hyperparameters and different data distributions. We obtained the best model using 24 layer roberta 180k equal sample one with the extended padding length . The final results of each approach is mentioned in the Results Section.

4 Results

With the NELA-GT 2019 dataset, we got the best result with the 24 layer Roberta 180k equal sample (with the extended padding length) for the title, and with the TF-IDF Approach along with Random Forest Classifier 180k equal samples for the body.

All the outcomes from our multiple experiments are compiled and displayed in the below table. Please note that the F1 scores in the table are in this order: macro, weighted and micro.

Table 1: Results

Model	**Classifier**	**F1 score**	**Accuracy**	**Approach**
roberta-large	RobertaForSequenceClassification	0.89206 0.89205 0.89207	0.89207705	90-90 dataset
bert-base-uncased	BertForSequenceClassification	0.826564 0.857648 0.859462	0.85946275	70-30 dataset
bert-base-uncased	BertTokenizer	0.86431 0.86430 0.86465	0.86465	Body + title 10-10 (title + head+ tail)
bert-base-uncased	BertTokenizer	0.87765 0.87768 0.87765	0.87765	Body 10-10
Functional		0.75877 0.75883	0.75883	Neural network 30-30 Glove embedding on title
roberta-base	RobertaForSequenceClassification	0.87178 0.87179 0.87179	0.87179	Roberta 30-30
roberta-base	RobertaForSequenceClassification	0.86086 0.88565 0.88717	0.88717	Roberta 70-30
roberta-large	RobertaForSequenceClassification	0.86816 0.89116 0.89279	0.89279	Roberta 70-30
roberta-large	RobertaForSequenceClassification	0.88215 0.88218 0.88217	0.88217	Roberta 50-50
	RandomForestClassifier	0.76767 0.81612 0.83306	0.83306	TF-IDF 30-30
	RandomForestClassifier	0.8538 0.8537 0.8538	0.8538	TF-IDF 90-90
roberta-large	RobertaForSequenceClassification	0.89409 0.89410 0.89411	0.89411	90-90 dataset With max$_{l}en = 48$
roberta-large	SVM Logistic Regression	0.644 0.704		Nela features On title
roberta-large	SVM Logistic Regression	0.777 0.779		Nela features On body

After running on 2 epochs, we observed that the validation loss was increasing. So, we've ran mostly on 2 epochs. Adam optimizer has been used everywhere.

We have kept the batch size for the transformer as 32. When running transformer on title, max_len is kept 24(average is 12 , maximum is 89). When running transformer on body, max_len is kept 256.

While applying logistic regression, we experimented with many parameters and have displayed the best results in the table.

5 Conclusion

Our best model was the 24 layer roberta 180k equal sample one with the extended padding length . The data , despite simple looking has more concepts to learn, thus deeper the model , better does it perform.

We saw multiple concepts on which our particular best performing model (Roberta-large) was learning to classify on . We found out that when it comes to the Declarative sentences with facts rather than opinions stated in them(Declarative sentences simply states a statement or expresses an opinion. In other words, it makes a declaration. This kind of sentence ends with a period), the news articles were correctly classified.This means that the underlying concepts are learnt better with less noisy words.For example : `Gregory Cochran On Gay Genes` was correctly classified as fake news, whereas `Trump's Other Base` was correctly classified as Real News.

While observing more, it was found that there were multiple sentences with multiple parts , i.e in the sentence ***Akala tells Owen Jones: 'The black-on-black violence narrative is rooted in empire' – video*** has two parts in which the part before the semicolon was acting as a reference for the further text . In these types of sentences , the concept learned is that since there is adequate reference given for the article , hence mostly are real.