Fake News Classification

By: Kajol Tanesh Shah (A20496724)
Tinh Cao Co (A20476426)

Introduction

Fake news/misinformation is nothing new, but it has made its prominent entrance through the 2016 US Presidential election, where it became an important tool to deliver false or even misleading political stories on social media. Even recently, the COVID pandemic has brought an upheaval of disinformation targeting public fear of the virus, its transmission rate, vaccinations, and post-infection recovery.

Despite existing in many forms, we will put our focus on the online social media platform as it is one of the most powerful tools that we use to communicate right now.

Problem Statement

We define our problem as the ability to classify incoming news B as fake news or not by comparing it with a known fake news A on the same topic. Three labels are used for comparing the two articles: agreed, disagreed, and unrelated.

- Agreed: B talks about the same fake news as A
- Disagreed: B refutes the fake news in A
- Unrelated: B is unrelated to A

With the ability to determine whether an incoming news article are fake new or not, we hope to minimize the spread of fake news and rebuild the trust for the online community.

Proposed Solution

To classify the news article B into one of the three categories which are agreed, disagreed and unrelated, we created a machine learning model and trained it using NLP techniques and machine learning algorithms.

Below are the steps that were used to create the model:-

- Data preprocessing using NLP techniques
- Bag of words and TF-IDF transformer
- Data augmentation
- Model training using machine learning algorithms
- Model testing
- Model evaluation using different evaluation metrics

Data Preprocessing

In this step, we applied some NLP preprocessing techniques to both the train and test datasets in order to make our datasets light. Using such techniques, we only kept the information which will be useful and has some meaning and removed the data which will not be important for prediction. Below are the techniques which were applied:-

- Converting strings to lowercase
- Removal of stopwords
- Removal of punctuation marks
- Lemmatization

Bag of words and TF-IDF Transformer

- Bag of words creates a set of vectors which contains the count of the word occurrences in the text/document. For this, we have used CountVectorizer from scikit-learn
- TF-IDF means term frequency - inverse document frequency and is used to find how important a word is to a document/text in a corpus. We implemented this by using TfidfTransformer from scikit-learn

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$

$$IDF(t) = log\frac{N}{1+df}$$

$$TF-IDF(t,d) = TF(t,d) * IDF(t)$$

Approaches

We used two different approaches for training our model. As our training data was highly imbalanced, we were not sure whether our model will be able to make correct predictions as it will be trained on a majority of data which has label as ‘unrelated’. So, we used two approaches to solve this problem:-

- Using the original dataset to train our model
- Using data augmentation to balance all the three classes i.e. the % of each class label in the training dataset will be 33.33%

Approach 1

In this approach, we used the original dataset to train our model using different machine learning algorithms and neural networks. For this, we made use of scikit-learn library in Python.

Below are the machine learning algorithms which we used:-

- Naive Bayes
- Stochastic Gradient Descent Classifier
- Logistic Regression
- Random Forest
- Linear Support Vector Machine Classifier

Approach 1 - LSTM

We also used LSTM model which is a Recurrent Neural Network architecture. LSTM has feedback connection which is not present in other standard neural networks and so, LSTM can process entire sequences of data. We used Keras for creating LSTM model in Python.

LSTM unit consists of :-

- A cell
- Input gate
- Output gate
- Forget gate

The gates regulate the flow of information and the cell remembers the data for some time intervals.

LSTM is used in many applications like time series, speech recognition, handwriting recognition, etc.

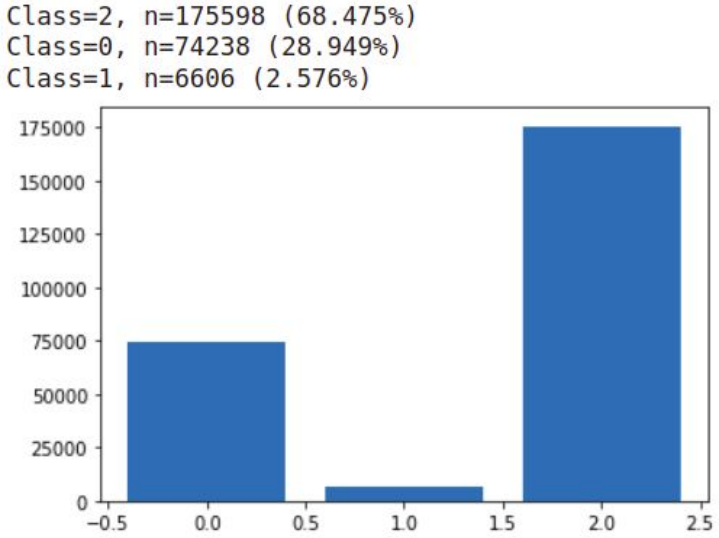
Approach 1- LSTM

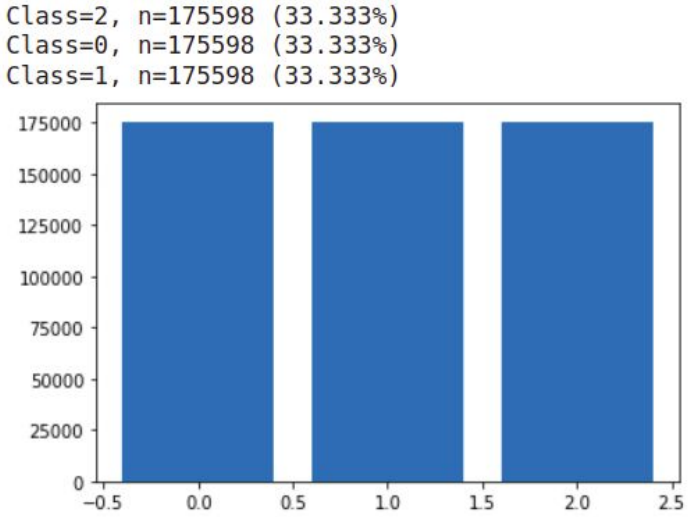
Text-preprocessing for LSTM: We use Keras' Tokenizer to convert the title string into a sequence of integers representing each word. The corpus is built upon our combined titles' vocabulary.

We choose to represent words as a sequence in an attempt to capture the word similarity that exists between title A and title B as well as the word dependency for its semantic meaning. In addition, the ability to process news that make reference to a long-time ago information also makes LSTM a prominent candidate. From this we hope to catch certain patterns that fake news follows to better classify them.

Approach 2

In this approach, we used data augmentation to solve the class imbalance problem as approximately 70% of our training data had class label as ‘unrelated’ and the remaining data 30% included of both ‘agreed’ and ‘disagreed’ label. We used the Imblearn library for this.





Approach 2

After doing data augmentation, we made use of the same machine learning algorithms to train our model which were used in the first approach. They are as follows:-

- Naive Bayes
- Stochastic Gradient Descent Classifier
- Logistic Regression
- Random Forest
- Linear Support Vector Machine Classifier

Evaluation

We tested our model on the validation data in order to find the best model to predict the class labels of our test data. We also used k-fold cross validation.

The metrics which we used to evaluate our model are:-

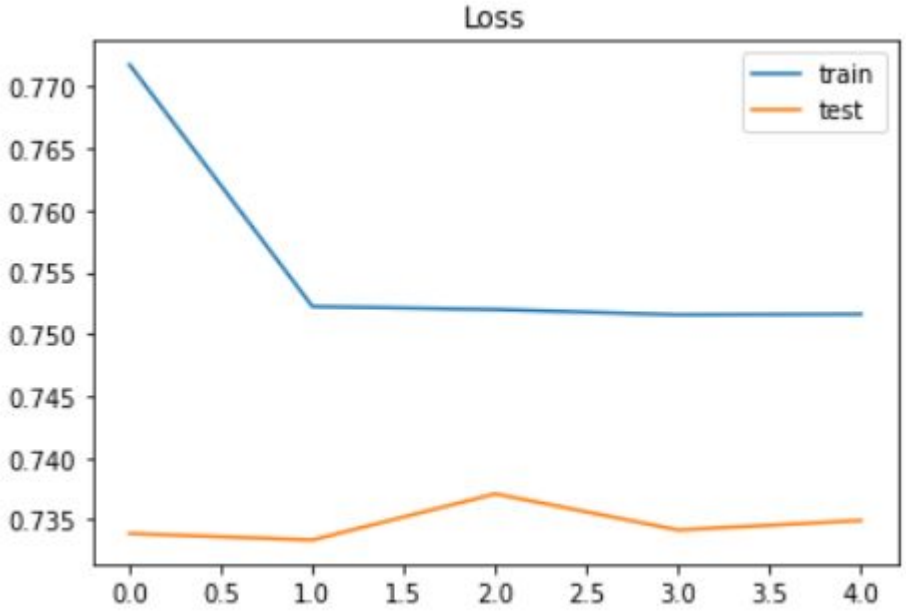
- Accuracy
- Precision
- Recall
- F-1 score

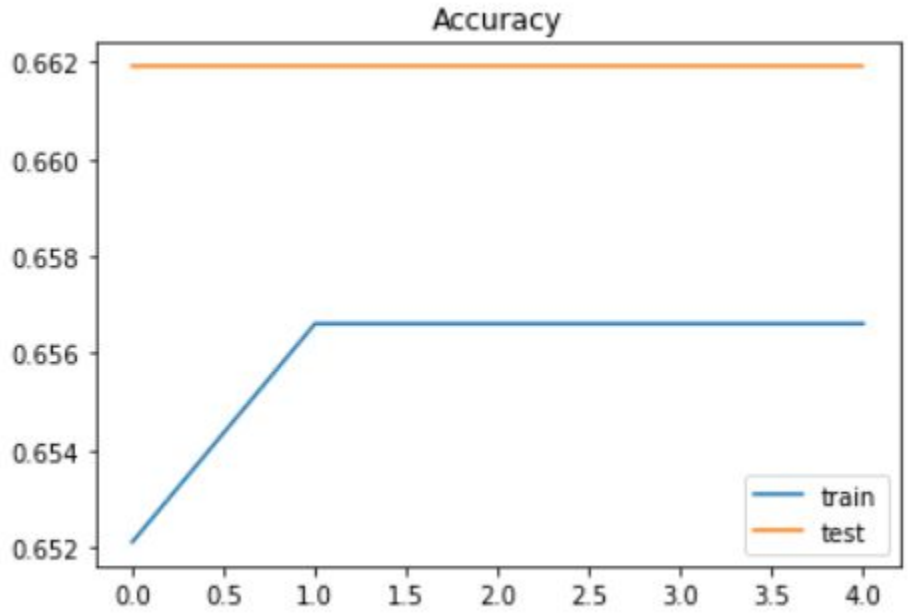
Results - Approach 1

Method	Accuracy	Precision	Recall	F-1 score
Naive Bayes	71.58	0.73 0.00 0.72	0.11 0.00 0.98	0.19 0.00 0.83
SGD Classifier	69.74	0.00 0.00 0.70	0.00 0.00 1.00	0.00 0.00 0.82
Logistic Regression	73.02	0.57 0.73 0.77	0.44 0.10 0.87	0.49 0.18 0.82
Random Forest	69.74	0.00 0.00 0.70	0.00 0.00 1.00	0.00 0.00 0.82
Linear SVC	73.02	0.57 0.64 0.78	0.45 0.12 0.87	0.50 0.20 0.82

LSTM - Result

Using 5 epochs and a batch size of 64, below are the results after the training.





LSTM result

As we saw in the previous slide, our LSTM model is not giving better accuracy than the other machine learning models. Also, as there is class imbalance, our LSTM model is only predicting the class label as ‘unrelated’ and does not score well on Precision, Recall and F-1 score.

Results - Approach 2

Method	Accuracy	Precision	Recall	F-1 score
Naive Bayes	75.91	0.71 0.89 0.69	0.81 0.81 0.65	0.76 0.85 0.67
SGD Classifier	63.84	0.62 0.65 0.70	0.78 0.94 0.20	0.69 0.76 0.31
Logistic Regression	86.00	0.81 0.91 0.86	0.84 0.97 0.77	0.82 0.94 0.81
Random Forest	53.77	0.42 0.78 0.54	0.61 0.54 0.46	0.50 0.64 0.50
Linear SVC	86.64	0.81 0.91 0.87	0.85 0.98 0.77	0.83 0.94 0.82

Conclusion

- Using different machine learning algorithms and LSTM model, we were able to create a model which can classify whether a given news B talks about the same fake news as A or not
- As we saw earlier in the results section, we achieved the best accuracy with Logistic Regression and Linear Support Vector Classifier

Future Scope

On the technical side, we believe that our models can better classifies fake news by employing dynamic sampling techniques introduce by Pouyanfar [8]. This put heavier weights on rare but significant events where fake news share the same sentiments, and less on unrelated fake news.

Broadly speaking, we can see the future of fake news classifications being used in industry setting, such as detecting fraudulent activities on Instagram, to scammers on Facebook Groups/ Posts that leverage the loopholes of online platforms for personal interest. The ability to make decision and intervention in timely manner can prevent those ill-practices from harming the community.

References

[1] H. Sak, A. Senior, F. Beaufays, " Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling" in Interspeech 2014

[2] Andrej, Karpathy: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[3] https://vitalflux.com/text-classification-bag-of-words-model-python-sklearn

[4] NLTK. Bird, Steven, Edward Loper and Ewan Klein (2009), *Natural Language Processing with Python*. O'Reilly Media Inc

[5] Tensor Flow. LSTM. https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM

[6] Keras. Project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System). https://keras.io/

[7] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

[8] Pouyanfar S, et al.. Dynamic sampling in convolutional neural networks for imbalanced data classification. In: 2018 IEEE conference on multimedia information processing and retrieval (MIPR). 2018. p. 112–7. https://doi.org/10.1109/MIPR.2018.00027.

[9] Jeff Reback, et al. (2020). pandas-dev/pandas: Pandas 1.0.3 (v1.0.3). Zenodo. https://doi.org/10.5281/zenodo.3715232

[10] https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer