CS579 - Online Social Media Analysis

Fake News Classification Report

Tinh Cao Co
A20476426

Kajol Tanesh Shah
A20496724

Department of Computer Science
Illinois Institute of Technology
Chicago, IL - 60608

Introduction

Considering the explosive growth of fake news and its negative impact on authenticity, journalism, and trust, many researches have been conducted to study the nature of fake news and to design a system that can intervene and prevent the spread of it in time. Our target scope is limited to online social media as the main news ingestion platform since its structure allows fake news to disseminate quickly at a very low cost. Many detection algorithms have been put in place to detect fake news using text-vectorization and other metrics from information retrieval fields to quantify the importance and relevance of string representations in documents. In this report, we also explore many of the natural language processing (NLP) techniques and popular machine learning algorithms to create models and determine which algorithms would give the best result in terms of accuracy.

Problem Statement

A general fake new classification task can be broken down into making a decision whether an incoming news is real or not. Here we are given two news articles A and B, where A is a known fake news article and B is a coming news that we need to classify on. A and B will be compared based on this, we will classify B into the below mentioned categories.

Three labels are used for comparing the two articles A and B: 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

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/neural networks.

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

1. Data preprocessing using NLP techniques
2. Bag of words and TF-IDF transformer
3. Data augmentation
4. Model training using machine learning algorithms
5. Model testing
6. Model evaluation using different evaluation metrics

1] Data preprocessing using NLP techniques

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 using NLTK library functions:-

- Converting strings to lowercase

  As lowercase and uppercase words are treated differently by machine, changing the cases of the words will allow the machine to interpret the text better

- Removal of stopwords

  Stopwords are words which are not important for your text and occur frequently like the, an, they, there, etc.

- Removal of punctuation marks

  Punctuation marks like commas, exclamation marks, question marks, ect are removed as they are not going to add any value to the text

- Lemmatization

In this process, the words are brought to their root form

2] 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 tex/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

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

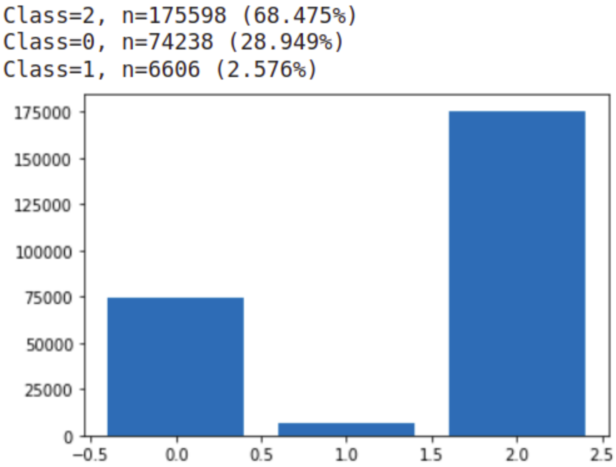
$$IDF(t) = log\frac{N}{1+df}$$

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

Figure 1: Tf-idf formula

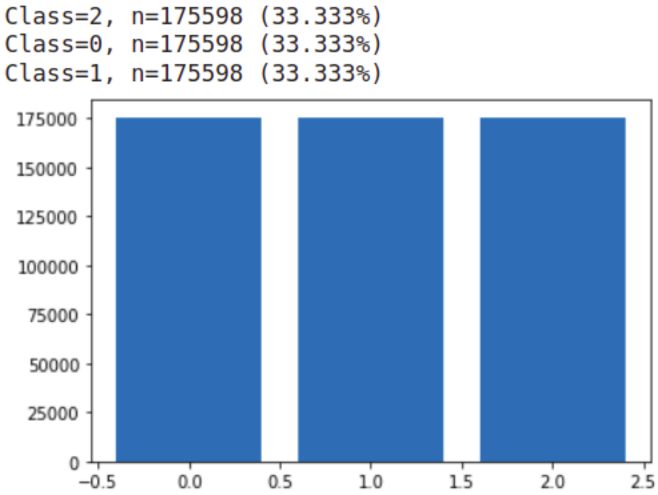
3] Data Augmentation

As in our case the classes are imbalanced, we have used data augmentation techniques to balance all the three classes. Our dataset contains approximately 70% of data which has been labeled as ‘disagreed’ whereas the remaining 30% of the data has ‘agreed’ and ‘disagreed’ labels. So, using the Imblearn library, we have oversampled the original training dataset and have balanced all the three classes so that our model does not get trained only on a specific class.



```
Class=2, n=175598 (68.475%)
Class=0, n=74238 (28.949%)
Class=1, n=6606 (2.576%)
```

Figure 2: Before data augmentation, we can see that the classes are heavily imbalance



```
Class=2, n=175598 (33.333%)
Class=0, n=175598 (33.333%)
Class=1, n=175598 (33.333%)
```

Figure 3: After data augmentation, all classes are balanced

4] Model training using machine learning algorithms

Here, we used two approaches to train the model:-

- Training the model using original training dataset
- Training the model using oversampled data

For both the approaches, we have used the following machine learning algorithms:-

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

We also used the Long-Short Term Memory **(**LSTM) model for our classification problem.

Long-Short Term Memory for Multi-Class Text Classification

Long-Short Term Memory (LSTM) is an improvement from the original Recurrent Neural Network (RNN) models by allowing the information to persist if it's determined as important. The model avoids the vanishing gradient problem faced by traditional RNN, where information learned diminished proportional to the number of network layers it passed through as determined by the loss function and its shortcoming in learning from previous information when it reappears in a later time state.

One of the main reasons for considering LSTM from the RNN family for our task is its ability to store information for an arbitrary duration, and be resistant to noise coming from the randomness of the inputs that are irrelevant to output prediction. In addition, word dependency is maintained and available for future retrieval when incoming news is similar to learned information. This will help with predicting class labels.

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.

5] Model testing

We tested our trained models on validation data to check the performance of the models and use the best model to predict labels for our test dataset. We also used k-fold cross validation.

6] Model evaluation using different evaluation metrics

As we saw earlier, there is class imbalance in our dataset. The metrics which we used to evaluate our model are:-

- Accuracy
- Precision
- Recall
- F-1 score

Results

Results of 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

Results of 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

LSTM results

Using LSTM after 5 epochs and a batch size of 64 units, the loss and accuracy of LSTM over time is captured in the figure below.

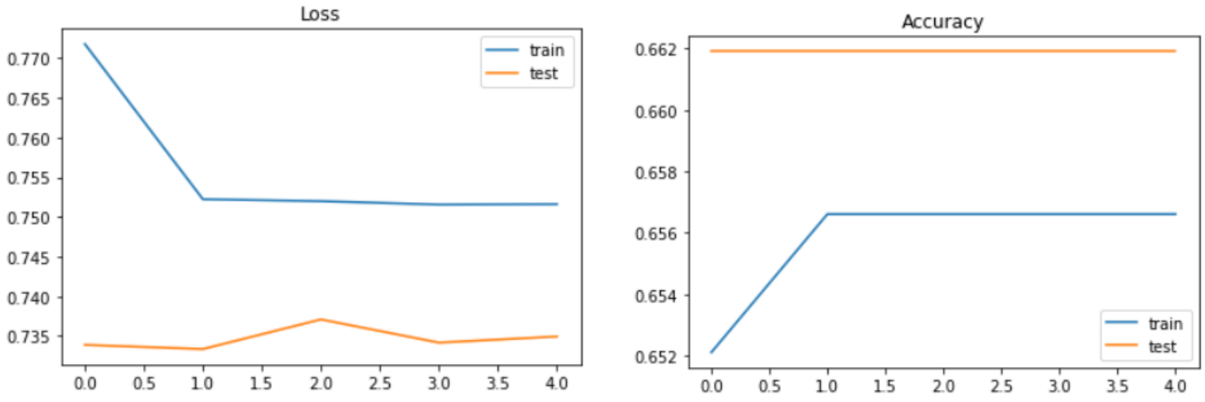
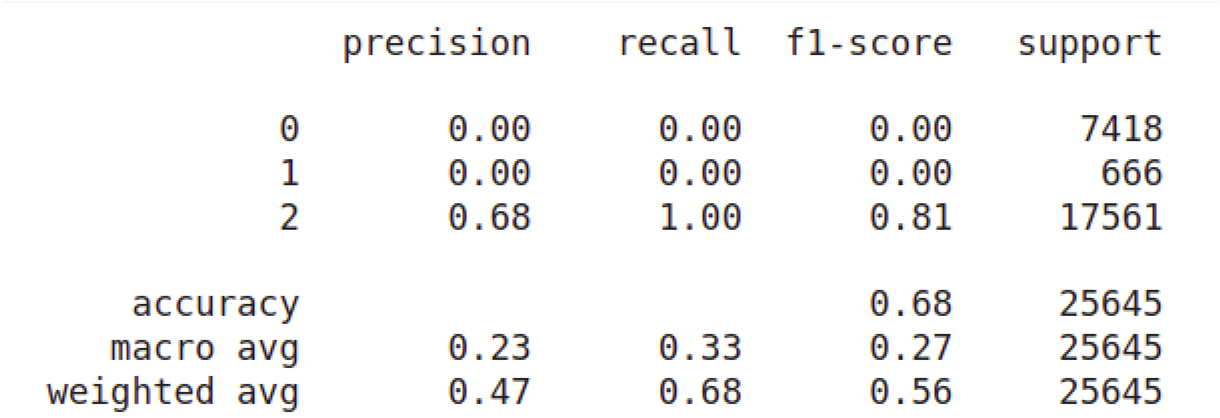


Figure 4: As we can see,there was no significant improvement in our loss and accuracy after the training, which may be because of class imbalance.

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


```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      7418
           1       0.00      0.00      0.00       666
           2       0.68      1.00      0.81     17561

    accuracy                           0.68     25645
   macro avg       0.23      0.33      0.27     25645
weighted avg       0.47      0.68      0.56     25645
```

Figure 5: Classification report for LSTM

Below is the link for our trained LSTM model:-
https://drive.google.com/drive/folders/18CPAzcaAy9fooanqFGkfht9sS_4KP2u-?usp=sharing

Conclusion

Using different machine learning algorithms and the 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

Noticing the significant issue introduced by class imbalance, we want to focus on data-level methods for handling such issues in future experiments. Pouyanfar et al. [8] introduced dynamic data sampling in his paper, where the sampling rates are adjusted according to the class-wise performance. The majority class will be undersampled and the minority will be oversampled to increase the weights that associate with unlearn events just like how humans learn.

One great advantage of dynamic sampling is its ability to self-adjust sampling rates to adapt to different complexity and class imbalance with little hyperparameter tuning. By removing samples that have already been captured by the network parameters, gradient updates will be driven by the more difficult positive class samples, hence making the Deep Neural Network better in catching infrequent events.

One of the drawbacks of this method lies in the validation set to calculate the class-wise performance metrics that control the sampling rate. Setting aside a portion of the minority class in the validation set may reduce the important information in our training set, defeating the purpose of using such a method. One may need to solve the problem of maximizing the available training data for relaxing this constraint.

Broadly speaking, we can see the future of fake news classifications being used in industry settings, 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 decisions and intervene in a timely manner can prevent those ill-practices from harming the community.

Responsibilities/Contributions

Tasks	Name
Data Preprocessing and Data Augmentation	Kajol
Bag of words and Tf-idf	Kajol
Training and Testing of Model using different Machine Learning Algorithms	Kajol
Text Preprocessing for LSTM	Tinh
Training and Testing of LSTM model	Tinh
Presentation	Tinh, Kajol
Report	Tinh, Kajol

References

[1] https://iamtrask.github.io/2015/11/15/anyone-can-code-lstm/

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

[3] Hassim Sak, et al., Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling, 2014

[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.