CIS4930 Natural Language Processing with Python Project Report

Abinai Pasunuri, Ammar Syed

Abstract

The issue of fake news articles and sources in our society is a growing one, especially in a digital age. Nowadays it is becoming increasingly difficult to discern between what is legitimate and what is fake. In order to tackle this issue, this project uses Natural Language Processing and Machine Learning techniques to classify whether news articles are reliable or unreliable. Natural Language Processing techniques, such as Lemmatization, removing of stop words, and text vectorization are used to preprocess a corpus of labeled news articles. These articles are then trained with Support Vector Machine and Convolutional-Recurrent Neural Network models to construct a classifier to identify whether a news article is either reliable or unreliable. Overall, the models perform very strongly and achieve high metrics, with the Support Vector Machine model performing slightly better than the Neural Network.

Introduction

In recent years, the growth and increased availability of a wide range of news sources and social media has led to an increased amount of unreliable and "fake" news articles and sources. It has become increasingly difficult to discern between real and "fake" news articles and sometimes the information can be very convincing to many people. This can lead to dangerous consequences as fake and misleading information can spread very rapidly and lead to many people not having the proper context and full amount of knowledge and facts about certain important issues. Due to this, the problem of detecting these unreliable news sources becomes an important problem to solve. This project aims to use the techniques of Natural Language Processing to preprocess and extract useful features from news article corpora from both unreliable and reliable sources and then use various Machine Learning techniques to train classifiers on this data to accurately identify if a news article is unreliable.

Methodology

The process taken for this classification problem followed the generic strategy for classifying a text document as outlined in [1]. This process has five primary steps, including data collection, data preprocessing, feature extraction, training the Machine Learning models on the extracted features, and evaluating the performance of the trained models.

For data collection, a corpus of news articles from a wide range of sources was collected. The first set of data was obtained from a Kaggle dataset with labels that contained around 21,000 real news sources and around 23,500 fake news sources [2]. These news articles included topics of general news, US politics, and World News from 2016 and onwards. In order to diversify the dataset a little bit, recent data from this year was scraped from the Internet. The front page headlines from the beginning of April 2021 were scraped from three reliable and unreliable sources. The reliable sources included the Washington Post, the New York Times, and NBC News. The unreliable sources included InfoWars, USSA News, and Breitbart. From scraping, an additional 387 reliable articles and 442 unreliable articles are added to the overall corpus. In the dataset, the label 0 represents a reliable article and 1 represents an unreliable article.

For the preprocessing of data, this stage involved several steps. First, missing values were dropped. The title and the text of all the articles were concatenated together to form a single string and all of the article text was made to be lowercase. Following this, all of the article text was tokenized using the default NLTK tokenizer and all the stop words were removed using the NLTK English Stopwords corpus. Then all of the tokens that were a punctuation or numeric value were removed. Then, all the tokens were then lemmatized using the NLTK WordNet Lemmatizer. All of the data was then saved into a single CSV file.

Following this, the data went through feature extraction to input into the models. Each model went through different methods. The data went through the Term frequency - inverse document frequency (TF-IDF) method for the Support Vector Machine Model. TF-IDF is a statistical method to estimate a word's importance via its frequency of occurrence in the document as well as in the corpus [3]. A higher TF-IDF weight correlates with the word being less frequent. The term frequency is found by dividing the count of a particular term (word) in a particular document by the total number of terms in that particular document. This is the frequency of the word in a particular document. The inverse document frequency is calculated by finding the log of the total number of documents divided by the number of documents with the number of documents that contain that particular term. The idf represents how important the term is to the corpus. The term frequency is then multiplied by the inverse document frequency to get the TF-IDF weight. It is a common and useful feature extracted from text and it will be used as the primary feature for the classical Machine Learning algorithms. Additionally, the data went through simple one-hot encoding for the Neural Network. Each article text was converted to a list of encoded integers, where each number in the list represents a token in the article text. A vocab size of 5000 was set to make each numeric value range from 1 to 5000. Lastly, since each of the article's text were of different sizes each list was trimmed to a max size of 3000 and zero padded to make it consistent for the model.

Following these steps, the different Machine Learning models were initialized and trained. Supervised Machine Learning techniques were used because the data was labeled. The two main Machine Learning models used for classification were a Support Vector Machine and a Convolutional-Recurrent Neural Network.

One of the models used was the Support Vector Machine model. It is a widely used and very powerful supervised classical Machine Learning technique. It essentially tries to solve an optimization problem to find a maximal margin [4]. This margin is the distance between the training data points and the hyperplane, also known as a decision boundary, that separates the training data into two groups. This is why they are commonly used for binary classification, which is what this project aims to achieve. Furthermore, it is commonly used for text classification, particularly the case where there are two groups, such as "spam" and not "spam" or “happy” and “sad.” One side of the boundary would represent “reliable” while the other side would represent “unreliable” for this scenario case. Given a testing data point, the algorithm will see which side of the decision boundary it lies on and will assign it a predicted label corresponding to what that side represents. By convention, one side is represented by either -1 or 0 while the other side is represented by +1. The main hyperparameter for an SVM is the type of kernel, such as linear, polynomial, sigmoid, or radial based function. The linear kernel SVM is less likely to overfit training data compared to the other kernels and scales better with a large training data sample size. This is why the linear kernel SVM was chosen for the model, since there were well over 40,000 training data samples and it would minimize the likelihood of overfitting.

The other model used for the classification was a Convolutional-Recurrent Neural Network. Neural networks are a popular supervised Machine Learning technique. There are several types of Neural Networks used mainly Feed Forward, Convolutional, and Recurrent. For this problem, a combination of Convolutional and Recurrent layers were used for the Neural Network architecture. A Convolutional layer simply involves the multiplication of two matrices, a weight matrix with the input one-hot encoded text vector, in order to extract important features [5]. Additionally following the Convolutional layers, a Recurrent layer involves the processing data in a sequential manner, in order to retain information of what came before the current sequence. In this model, the specific Recurrent layer of Long Short-Term Memory (LSTM) is used which contains an additional carry to ensure no information loss [5]. The combination of these layers results in a model that is good at both extracting features and learning long term dependencies leading to potentially strong results. For the architecture used in this project, it first contains an Embedding layer, which is a layer that learns the relation between words and how they are used and groups words that are used similarly to each other [5]. In this instance, it produces an output vector of size 40. Following this the output vector is inputted into a 1D Convolutional Layer of 32 filters and a size of 5 and this is followed by a 1D Max Pooling of a pooling size of 2, which further reduces the vector. This process is done once more with the same operations and then the output is fed into the LSTM layer with a 100 units. The data is then renormalized with batch normalization and then fed into a standard Dense layer of 32 units with all layers having ReLU activation. Lastly, the output is passed into another Dense layer of a single unit and sigmoid activation function to get the output valued between 0 and 1. This Neural Network was trained with an Adam optimizer with a learning rate of 0.001 and the Binary Cross Entropy loss function, both of which are popular training hyperparameters. The model was trained for 2 iterations and in batches of size 32.

Following the training of the models, they were then evaluated on the 25% of data not used for training to get a sense of how well they were performing on unseen data. Since this was a binary classification problem, the primary metrics used to evaluate the models were accuracy, precision, recall, and F1 score. The accuracy metric gives a general understanding of the model performed and how many samples it correctly classifies. The precision, recall, and F1 score give more specific insight on well the models confuse reliable articles with fake ones and vice versa. Additionally, a confusion matrix and an ROC curve for each model was generated to get an overall indication of how each type of data is being classified and an overall indication of the models' overall performance through the area under the curve of the ROC.

Results

Following the steps of preprocessing the entire news article corpus, the class distribution of the articles came out to be pretty fairly balanced as seen in Figure 1. Overall there was a slightly greater number of unreliable articles, at around 52.33% and the reliable articles stands at a little lower value at around 47.67%. Overall, the dataset class distribution is pretty balanced making it ideal for Machine Learning training.

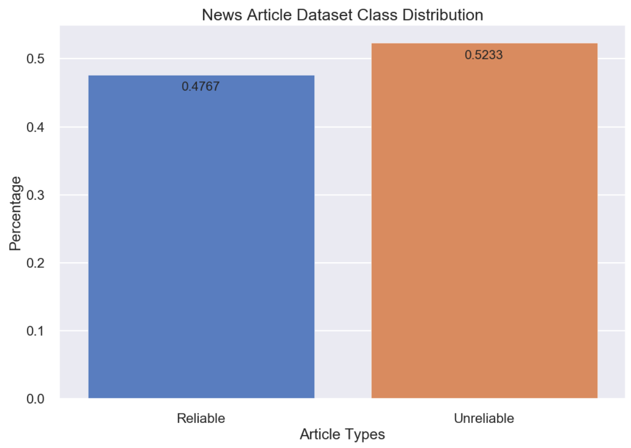

Figure 1: Bar chart showing the distribution of the class labels in the dataset.

Further analyzing the preprocessed data, taking the 2000 most frequently occurring words in the corpus it can be seen from the word cloud visualizations in Figure 2 that there is quite a bit of variation in the word choice of reliable and unreliable articles. With reliable articles, there are words, such as new, continue, allow, remain, include, which can be attributed to a pretty neutral tone. With the unreliable articles, there are words, such as become, seem, appear, which can be attributed to more speculation type language.

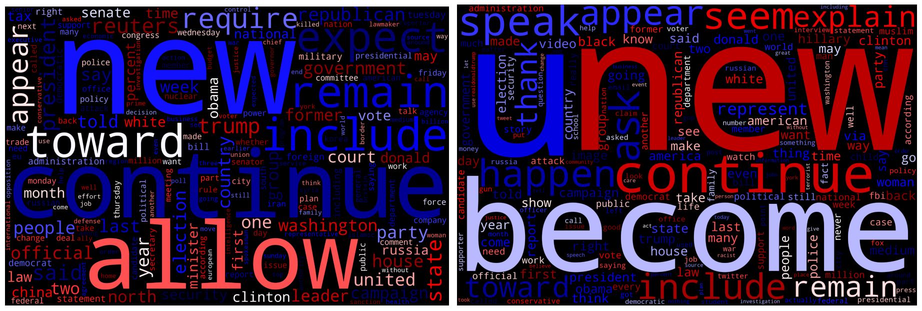

Figure 2: Reliable Word Cloud (Left) and Unreliable Word Cloud (Right)

Model Type	Accuracy	Precision	Recall	F1-Score	Area under the Curve (AUC)
Support Vector Machine	0.9885	0.9885	0.9885	0.9885	0.9885
Neural Network	0.9236	0.9308	0.9269	0.9235	0.9269

Table 1: A summary of the results produced by the two models on the testing dataset.

As seen in Table 1, the accuracy of the SVM on determining the correct label for the testing dataset was quite high at 0.9885. This high accuracy score on the testing data shows that the model is not overfitted and has low bias and low variance. The precision was 0.9885, indicating that there were very few false positives while the recall was also 0.9885, showing that there were few false negatives. Since both precision and recall were high, this means that most of the data points were correctly labeled by the classifier. The F1-score, a weighted average between precision and recall, is 0.9885 which is also very high, further supports this. The counts of true negatives, true positives, false negatives, and false positives can be seen in the confusion matrix diagram in Figure 3. It's clear that the number of false negatives and false positives are extremely small compared to the true positives and true negatives. The SVM ROC curve in Figure 3 below has an AUC that is 0.9885 and the model ROC curve is far higher than the baseline ROC curve further reinforcing that the trained SVM model was highly accurate.

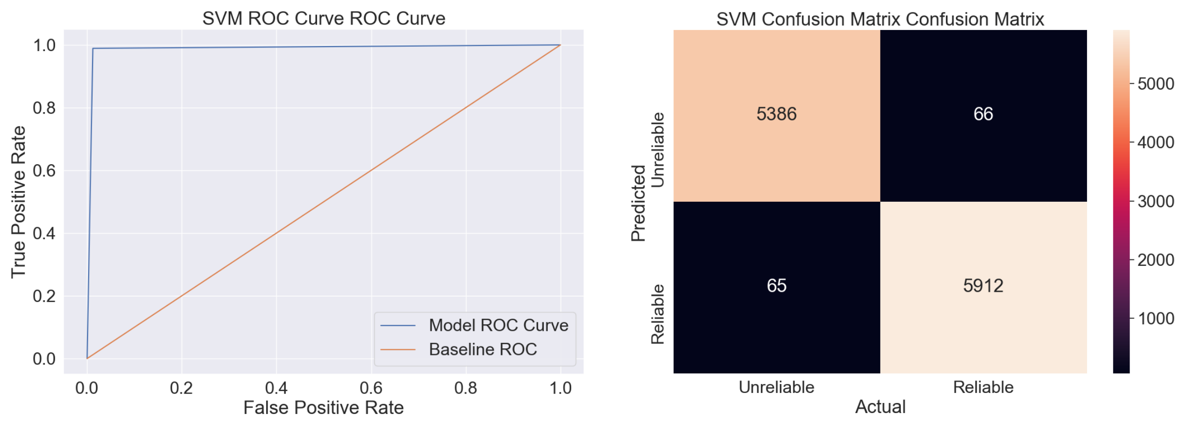

Figure 3: The ROC curve (Left) and the confusion matrix (Right) of the Support Vector Machine model.

As seen in Table 1, the Convolutional-Recurrent Neural Network model performed quite well and obtained high values on the testing data for all types of metrics. It achieved an accuracy of about 0.9236 indicating that it produces the correct result overall on the whole dataset regardless of the class label and indicates that the model has not overfitted. The precision value was 0.9308, indicating that the model is properly recognizing how to correctly identify fake news articles and not messing up with them overall. The recall value was 0.9269, which shows that there were few instances in which the model does not miss out on correctly identifying fake news articles often. Naturally the F-1 Score of the model is also high at 0.9235. In Figure 4, the confusion matrix shows that the model for the most part predicts the right label, but there are several instances of misclassification. The model most often predicts an unreliable article to be reliable, but very rarely predicts a reliable article to be unreliable. Further, the ROC curve's shape in Figure 4 shows that the model has near shape, but has a bit low True Positive rate for different thresholds. The overall AUC value is 0.9269, showing the model's strong performance. Despite the strong results obtained, the model has the potential to improve even further through more training iterations as this model was simply trained on only 2 iterations.

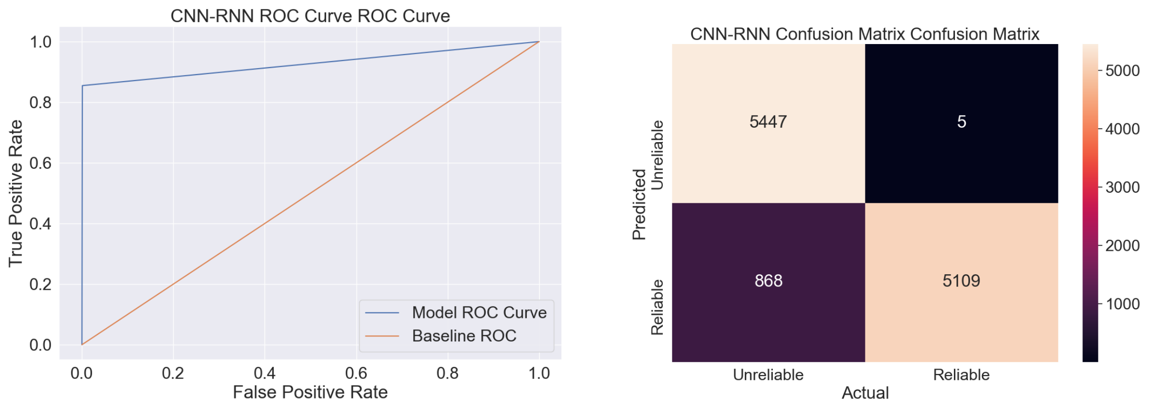

Figure 4: The ROC curve (Left) and the confusion matrix (Right) of the Convolutional-Recurrent Neural Network model.

Conclusion

Ultimately both of the proposed Machine Learning models performed very well and were able to capture the complexities of the training data without overfitting and accurately predict whether an article from the training data was reliable or unreliable. The SVM model performed better than the Neural Network model by about 7% in terms of accuracy. However, by manipulating the architecture of the Neural Network, further hyper parameter tuning, as well as simply increasing the number of epochs it is possible that the Neural Network's performance can increase. Out of curiosity, the SVM was tested against other classical ML models to simply see how it compares and it outperformed a Naive Bayes and a Logistic Regression model and had essentially the same accuracy as a Decision Tree model. Considering the fact that decision trees tend to overfit, especially as the training data gets larger or more complex, the Support Vector Machine model, with a linear kernel is a viable and effective way to predict whether an article is reliable or unreliable. If a non-classical Machine Learning algorithm is preferred, the Convolutional-Recurrent Neural Network developed would also be just as viable and effective.

References

[1] M. K. Dalal and M. A. Zaveri, “Automatic Text Classification: A Technical Review,” *International Journal of Computer Applications*, vol. 28, no. 2, pp. 37–40, 2011. https://www.researchgate.net/profile/Mukesh-Zaveri/publication/266296879_Automatic_Text_Classification_A_Technical_Review/links/54e74a0a0cf2b199060ae1c5/Automatic-Text-Classification-A-Technical-Review.pdf

[2] C. Bisaillon, “Fake and real news dataset,” *Kaggle*, 26-Mar-2020. [Online]. Available: https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset. [Accessed: 28-Apr-2021].

[3] “idf :: A Single-Page Tutorial - Information Retrieval and Text Mining,” *Tf*. [Online]. Available: http://www.tfidf.com/. [Accessed: 28-Apr-2021].

[4] S. Tong and D. Koller, “Support Vector Machine Active Learning with Applications to Text Classification,” *Journal of Machine Learning Research*, Nov. 2001. https://www.jmlr.org/papers/volume2/tong01a/tong01a.pdf

[5] J. A. Nasir, O. S. Khan, and I. Varlamis, “Fake news detection: A hybrid CNN-RNN based deep learning approach,” *International Journal of Information Management Data Insights*, 05-Jan-2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667096820300070. [Accessed: 28-Apr-2021].