Detection of Fake News Using Supervised Learning Techniques

Karnati Sai Abhishek, Wira Azmoon Ahmad, Ivan Andika Lie, Priyan Rajamohan, Sim Yu Jie, Jerry Zhang Zhuoran

School of Computing
National University of Singapore
Singapore
{saiabhishek.karnati, wira.azmoon, ivanandika, priyan_rajamohan, sim.yu.jie, jerry.zhang}@u.nus.edu

Abstract

Fake news is a prevalent issue worldwide and most certainly present in Singapore, with the elderly in Singapore being more susceptible. This paper aims to show the development of a possible intelligent application that makes use of common machine learning models and techniques to identify fake news. These models include Support Vector Machines, Naive Bayes, with techniques such as Term Frequency-Inverse Document Frequency (TF-IDF) that have been shown to produce promising results in the field of fake news detection. Comparison of experimental results between numerous models and variations revealed that the usage of Passive Aggressive Classifier with TF-IDF feature selection technique outperformed other classification models in detecting fake news. Models were tested against a combined dataset composed of FakeNewsCorpus, LIAR dataset and publicly available datasets from Kaggle. Despite high classification accuracy, results may be improved in several ways that will be discussed later in this report.

I. Introduction[1]

With the development of the Internet and social media, information and news have been made increasingly available and accessible for the common masses. Coupled with the widespread use of mobile technology, people can gain access to online news with much greater ease.

Given the huge influence that social media and news outlets have on society, it is only natural to see fake news becoming more prevalent as malicious people seek to influence the public's opinion on certain matters or simply for monetary reasons. (BBC Trending, *The rise and rise of fake news* 2016) Fake news is defined as content that has had the credibility of the information jeopardised, due to falsification of said information (Aphiwongsophon and Chonstitvatana 2018). The spread of fake news can contribute negatively to people's safety and security, especially when the news can touch on sensitive topics such as religion and politics. In a multi-racial and multi-harmonious society like Singapore, it is undeniably important for insensitive information, especially in the form of fake news, to be detected early to prevent it from spreading discord (Zhuo 2019).

However, one of the greatest challenges when tackling the issue of fake news detection is the speed at which fake news propagates. It has been found that false news travels faster than real news (Vosoughi, Roy, and Aral 2018) with the help of social media, making fake news detection a complicated task. Moreover, it can be difficult for humans to detect fake news (Ahmed, Traore, and Saad 2017) as a vast knowledge of the covered topic is required. In Singapore, it has been reported that many people are unable to discern the difference between fake and real news (Huiwen 2018).

Given the difficulties related to manual fact-checking, machine learning models may be the solution to address the complex problem of fake news detection. Machine learning models are capable of learning from large volumes of data to differentiate between reliable and unreliable news sources (Horowitz 2021). This is reflected in how research interest in utilizing machine learning techniques for fake news detection has increased in recent years. In (Ahmed, Traore, and Saad 2017), the authors proposed multiple models and techniques to detect fake news. In (Khan et. al. 2019), the authors analysed several models for online fake news detection.

The goal of this paper is to examine and compare the performance of various machine learning models while also supporting the notion that fake news detection can be resolved with machine learning. This paper also differs from others as the models are tested on numerous datasets while also exploring how application services can be integrated with Machine Learning models to defend users

against fake news. In our project, we have integrated our model into a Telegram Bot which can help users verify if a particular news source is potentially unreliable, as well as provide alternative news sources.

II. Proposed Application

Our application is a simple Telegram Bot named FakeNews! SmartBot[2]. The source code is hosted on Github[3]. Telegram was our platform of choice as it is one of the popular instant messaging applications used in Singapore. While there are resources to teach users how to identify fake news, there are no easy applications that can help users detect them.

Our project requirements include the need for extensive data in the backend to train a reliable and consistent model for prediction. Our project also needs to be online with a fast prediction speed to reply to the users promptly. As such, we have developed two core features that have been implemented on the SmartBot.

The main feature is a "Verify News" command which prompts the user to send a URL. The Bot uses our implemented Machine Learning model to predict if the news is fake. If the news is reliable, a message will be sent to the user indicating to assure the user that the news is reliable. However, if the news is unreliable, a message will be sent to the user to indicate that the news might not be reliable and the Bot will search the Internet for reliable news articles that are most similar to the contents of the news article before sending them as a suggestion to the user.

The second feature is a "Suggest News" command which prompts the user to suggest some keywords which will be used by the Bot to search the Internet for reliable news articles. The Bot will then find the closest results before sending them to the user.

III. Studied Machine Learning Techniques

In related literature (Ahmed, Traore, and Saad 2017), various models and techniques were investigated, and they had found that the best performance came from having the feature extraction technique being TF-IDF, and the classifier being the Linear Support Vector Machine. A separate study found that instead of Linear Support Vector Machine, Naive Bayes with TF-IDF had performed better (Khan et. al. 2019), out of the traditional machine learning approaches, so there is room to study multiple models and techniques for our application. Since we would like the application to be based on a machine learning model that is relatively simple to implement, using an easily available and highly maintained library such as scikit-learn was a priority. In this project, we used Naive Bayes as the benchmark which we use to compare other implemented Machine Learning models. Subsequently, we can use Ensemble Learning to further improve the accuracy of the model. Passive-Aggressive Learning can also be used to deal with the constant stream of news on the internet.

A. Support Vector Machine (SVM)

To understand SVM as a classifier, there are only four basic concepts that need to be covered (Noble 2006) - (i) the separating hyperplane, (ii) the maximum-margin hyperplane, (iii) the soft margin and (iv) the kernel function. The separating hyperplane can be imagined as a line that separates positive and negative training data. The data can be of any number of dimensions, so the hyperplane acts as a barrier between data in its appropriate dimension. The maximum-margin hyperplane then does as the name suggests, aiming to select the separating hyperplane which is 'in the middle' of all other hyperplanes to be able to classify new unseen examples better. The soft margin then allows a few anomalous or noisy data to fall on the 'wrong side' of the separating hyperplane. This assists in the final result not being affected, since it has been so far assumed that the data can be separated with a straight line, which may not be the case. Lastly, we have the kernel function, which is a mathematical trick that allows data from a lower dimension to have a higher dimensional classification. A good choice of such a function can result in the data being separable in the higher dimension, even if they were not in the lower dimension.

Advantages of SVM include being effective in high-dimensional spaces and memory-efficient (Khan et. al. 2019). In terms of detecting fake news, there are many similarities between the classification of fake news and the SVM. The maximum-margin separating hyperplane could be used to split the data given, the soft margin can account for anomalous data, and the kernel function could be the simplest linear kernel, which is the kernel that we chose.

B. Naive Bayes (NB)

Naive Bayes classifier is a subset of Bayesian classifiers, which assign the most likely class to a given example using its features (Rish 2001). The Naive Bayes classifier greatly simplifies learning by assuming that features are independent given any class. That is, we have the formula $P(X|h) = \prod_{i=1}^{n} P(X|C)$, where $X = (X_1, ..., X_n)$ and h is a class. While this assumption is unrealistic and generally does not hold, its final classification can end up being correct even if the probabilities calculated are inaccurate. Rish proceeds to explain that the success of Naive Bayes

[2] https://telegram.me/fakenews_smartbot

[3] https://github.com/therealcs1010/FakeNews-SmartBot

could be attributed to the optimal classification not needing to be related to the fit of the probability distribution, but instead that the actual and estimated distributions just have to agree and the most probable class. Naive Bayes was then found to work best in 2 cases: when the features are completely independent, and when the features are functionally dependent.

In terms of classifying fake news, the features present using TF-IDF could naturally use the Naive Bayes classifier, where the words used in every news article are generally dependent, and it can end up with an accurate classifier.

C. Passive-Aggressive (PA) Classifier

Online Learning is a classification that extends the use of SVM (Ezukwoke and Zaerian 2019). Instead of taking data in batches, it can update weights using one example at a time. The classification is done in rounds, whereby an instance is observed and a binary output predicted. This is then compared with the true label, with an instantaneous loss to the algorithm suffered, and subsequently, it improves the prediction rule for future rounds. This algorithm is fascinating because it can handle new data streaming and updates itself to make informed predictions. This is in line with our final product being a Telegram Bot.

D. Ensemble Learning

Ensemble Learning has been shown to reduce both bias and variance of learning algorithms and classifiers (Dietterich 2002, Polikar 2012). This is because assuming the different classifiers make different errors, but agree on their correct classifications, averaging the outputs reduces the error by averaging the error components out. We found it particularly interesting that the accuracy of our previously studied models could be even more accurate with the simple congregation of the results using Ensemble Learning.

There are various ways of combining results of various classifiers, after training, which can also depend on discrete or continuous output. Majority voting can be used to combine the result of classifiers with discrete outputs, while Continuous output classifiers could be combined using the Mean rule, averaging out all classifiers' outputs, and where the final prediction is that of the class that has the highest probability. In scikit-learn, VotingClassifier can be used to combine classifiers with hard voting, using majority voting, or soft voting, using the averaging of probabilities.

IV. Proposed Experiment Methodology

A. Our Proposed Approach

In our experiment, there were several key phases which include the preprocessing of the data and text, training of the models, and interpretation of the classifications (Lorent 2019). Afterwards, the best performing model was chosen and saved so it can be deployed to our proposed application. The design choices for each phase will be discussed in their respective subsections.

In the first phase, data were obtained from several different publicly available online sources and preprocessed by removing rows with empty fields or unused features. Python libraries such as Natural Language Toolkit (NLTK) were also utilized to remove stop words, punctuation and to break down the text in the dataset.

In the second phase, feature extraction utilizing TF-IDF as a metric was conducted before the data was separated into 70% for training and 30% for testing. A variety of different supervised classification models such as Naive Bayes and SVM were chosen and their performance was measured to choose the best performing model.

In the final phase, we evaluated the chosen models using several techniques and attempted to further optimize them through various methods. The best performing model was deployed for use in a Telegram Bot for manual testing.

B. Datasets Used

For our experiment, we combined two datasets: “FakeNewsCorpus” dataset and Kaggle’s fake news dataset. “FakeNewsCorpus” dataset[4] comprises over six million records of articles and their content are categorized into 11 categories: fake, satire, bias, conspiracy, state, junk science, hate, clickbait, unreliable, political and reliable. This dataset was used in other literature such as (Kurasinski and Mihailescu 2020). For simplification, we took only records labelled as fake, reliable, political, bias and unreliable and relabelled each record as either fake or reliable.

Our second dataset is a fake news dataset available on Kaggle[5]. There are a total of 4009 records, with 2137 labelled as fake and 1872 labelled as real. This dataset was chosen over other publicly available datasets on Kaggle as it contained features that were similar to the features available in the “FakeNewsCorpus” dataset.

As both datasets have different structures, they were preprocessed to ensure similar formatting, for ease of training. Although the “FakeNewsCorpus” dataset is much larger than the Kaggle dataset, we were only able to use a subset of the collected “FakeNewsCorpus” dataset due to memory constraints. As such, all experiments were done on a total of 879,747 records from either of the two datasets.

[4] “FakeNewsCorpus Dataset,” GitHub. [Online]. Available: https://github.com/several27/FakeNewsCorpus

[5] “Fake News detection dataset” Kaggle, 07-Dec-2017. [Online]. Available: https://www.kaggle.com/jruvika/fake-news-detection

C. Data Preprocessing Methods

Before utilizing the compiled data for training, we need to utilize certain processing techniques to further refine it. This is required as it aids us in reducing the size of the actual data and removing any possible irrelevant data that exists within the dataset.

Removing Stop Words

Stop words are a set of commonly used words in a language. Examples of stop words in English are "is", "are", "a", and "the". By removing the stopwords, the remaining text would contain more important words. It also reduces the number of features to consider which helps to keep the model relatively smaller.

Stemming

Stemming is the process of reducing inflection in words (e.g. troubled, troubles) to their root form (e.g. trouble). It is useful for dealing with sparsity issues as well as helping to standardize the vocabularies within the dataset.

D. Feature Extraction Using TF-IDF

Multiple studies have already compared the Term Frequency-Inverse Document Frequency feature extraction technique with other common techniques in Natural Language Processing (NLP), such as TF (Ahmed, Traore, and Saad 2017), Bag of Words (BOW) (Pimpalkar and Raj 2020), and Probabilistic Context-Free Grammar (PCFG) (Dyson and Golab 2017), and there is a consensus that TF-IDF performs the best out of the various techniques. Thus, for this project, all preprocessing of text makes use of TF-IDF.

The technique is a weighting metric that measures the importance of a term in a document within a dataset, by counting the frequency that it appears within the document, and countering this with the frequency of the term in the corpus (Ahmed, Traore, and Saad 2017). TF-IDF is also readily available within Python's scikit-learn library by making use of the TfidfVectorizer class.

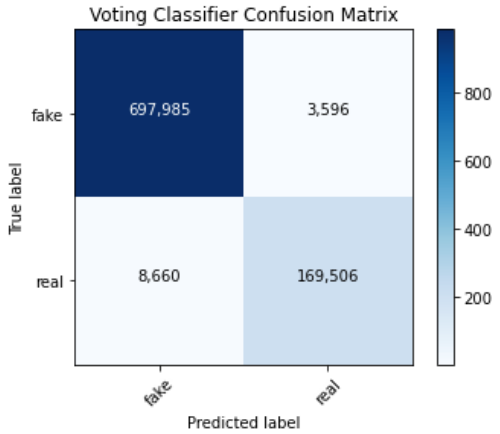

Fig 1. Confusion Matrix from predictions.

E. Performance

Table 1 below showcases the various scores and accuracy of the models and techniques chosen. Performance metrics such as precision and recall, together with the F1-score are used to evaluate the models (Lorent 2019). These can be combined in a confusion matrix, displayed in Fig 1.

Fig 1 shows a confusion matrix used to summarise the prediction results of the voting classifier. True negatives and positives represent the number of times, the model predicts correctly. Similarly, false negatives and positives represent the number of times the model predicts incorrectly. By computing the number of correct and incorrect predictions in such a matrix, we can observe how classification models are confused when making predictions. The goal of creating robust models is to keep the number of false positives and negatives low.

Using such a confusion matrix, we can compute other performance metrics like precision, recall and f1 score.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Precision describes the number of positive predictions that belong to the positive class while recall gives us an idea of how high the cost associated with false negatives is. F1 score depicts the balance between precision and recall giving us an idea of how even or uneven the class distribution is.

V. Experimental Results and Analysis

Overall, we find that the best classifier proved to be the Voting Classifier. However, when testing the model on new, real-life data, there were a few false negatives. This could be attributed to overfitting or the fact that fake news are more advanced and closely imitate real news.

From the results of our experiment, PA Classifier emerged as the best performing model compared to Multinomial Naive Bayes and Linear SVM with Stochastic Gradient Descent. In addition, we ran multiple tests utilizing PA Classifier and different parameters for TF-IDF to determine that the best results were achieved by using bi-gram analysis and unlimited features. Lastly, we attempted to optimize the PA Classifier by changing the loss function between 'hinge' and 'squared_hinge' and training epochs but there were little differences.

Classifier	Feature Selection & Parameters	Accuracy	Precision	Recall	F1-Score
Multinomial Naive Bayes	TF-IDF ngram_range=(1, 2) max_features=None	0.7703	0.9018	0.7703	0.8007
Linear SVM using Stochastic Gradient Descent		0.7661	0.9087	0.7661	0.7994
PA Classifier **('hinge', max_iter=1000)**		0.9449	0.9476	0.9449	0.9453
PA Classifier **('hinge', max_iter=1000)**	TF-IDF ngram_range=(1, 2) max_features=5000	0.8520	0.8551	0.8520	0.8530
PA Classifier **('squared_hinge', max_iter=1000)**	TF-IDF ngram_range=(1, 2) max_features=None	0.9448	0.9475	0.9448	0.9453
PA Classifier **('hinge', max_iter=5000)**		0.9451	0.9473	0.9451	0.9455
Voting Classifier with LSVM, NB, PA		**0.9860**	**0.9863**	**0.9860**	**0.9861**

Table 1. Comparison of Performance between different Classification Models.

Lastly, we paired TF-IDF using the best performing parameters with a Voting Classifier combined with the three aforementioned models to balance out individual weaknesses and improve generalizability. This resulted in higher precision and recall scores compared to the previous 3 models.

In our SmartBot, the consequences of false positives are less severe than false negatives. This is due to the remedies that our Telegram Bot takes. In the event of a false negative, the bot would send a message indicating that the fake news is a reliable source of information. This could result in the propagation of fake news. However, in the event of a false positive, our bot would source for more reliable news sources which best fits the current news article.

VI. Challenges Faced and Improvements

A. Limitation of Hardware Resources

While increasing the upper limit of *n_gram* range helps to include more context to the features generated, this also puts a strain on the hardware resources at our disposal. As such, we could only explore till bigrams and not beyond.

B. Time Required for Data Preprocessing

To make our models as robust as possible, we collected a variety of datasets. However, each dataset had a different format and combining them required much time and effort. Furthermore, it was not feasible to train the models on raw data due to the sheer size of the datasets. To deal with this, we used NLP libraries like NLTK to extract compact meaningful information from the data. This helped to reduce the size of the training datasets allowing us to considerably decrease the amount of time required for training the models.

C. Data Overfitting for Certain Models

Overfitting happens when the model fits too closely with the training data, including all the erroneous/noisy training data (Hawkins 2004). We incorporated early stopping to ensure the model training is stopped when it starts to overfit the data and lost the generalization power. The use of ensemble methods greatly reduces overfitting by training multiple models and combining their predictions. Combining the predictions from multiple models adds a bias that in turn counters the variance of a single model. Therefore, the predictions are less sensitive to the specifics of the training data and the choice of the training scheme.

D. Lack of Native Dataset

The global datasets with which our models were trained were not based in Singapore and as such, it is rather difficult to detect fake news in Singapore, where the application for this project is based. Hence, a large portion of training could be improved by obtaining more Singaporean sources.

E. Kernel Tricks

The kernel trick is commonly used in SVMs to help map the data from a lower dimensional space to a higher dimensional space (Cristianini and Shawe-Taylor 2000). The rationale for this is to allow a more efficient and inexpensive way of transforming data into higher dimensions, which may allow us to uncover a decision surface that can separate different classes during classification. Common functions to be used for kernels are polynomial kernel and radial basis function kernel.

VII. Conclusion

This project managed to showcase the common ML models, algorithms, and techniques used in determining fake news. The results seem to agree with previous literature, in terms of LSVM (Ahmed, Traore, and Saad 2017), but we found that the use of an alternative classifier, PA classifier, gave improved results. Combining this with Ensemble Learning, this provided impressive results. We have learnt that the ideation, creation, and implementation of an AI model to solve a pertinent problem can be a non-trivial task, and have learnt various ML models, techniques, and algorithms not known before. These include the inner workings of SVM and Naive Bayes models, the usage of TF-IDF as a feature extractor, and the advantages of Ensemble Learning to further improve performance output by combining various models.

VIII. Acknowledgements

This project would not have been possible without the combined effort of every member of the team. Weekly meetings were held to take note of each member's progress, and we are satisfied that all members contributed their best in their assigned tasks.

All the team members were involved in carrying out research and writing the report. Abhishek was responsible for training the voting classifier and data collection. Wira explored different models and literature reviews. Ivan experimented with CNN models and preprocessed the datasets. Priyan helped with data preprocessing, TF-IDF implementation, gradient boosting and PA classifier. Yu Jie constructed the NB and Linear SVM classifiers as well as the Telegram Bot. Jerry worked on experimenting with stacked classifier, data preprocessing and parameter tuning.

IX. References

Ahmed, H.; Traore, I.; and Saad, S. 2017. Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques, ECE Department, University of Victoria, Victoria, BC, Canada.

Aphiwongsophon, S., and Chongstitvatana P. 2018. Detecting Fake News with Machine Learning Method. In *15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, 528-531. Chiang Rai, Thailand: IEEE.

BBC News, 2016. The rise and rise of fake news. [Online]. Available: https://www.bbc.com/news/blogs-trending-37846860.

Cristianini, N., and Shawe-Taylor, J. 2000. Kernel-Induced Feature Spaces. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Dietterich, T. G. 2002. Ensemble Learning. Arbib, M. A. (ed.),. *The Handbook of Brain Theory and Neural Networks, Second edition*. Cambridge, MA: The MIT Press.

Dyson, L., and Golab, A. 2017. Fake News Detection Exploring the Application of NLP Methods to Machine Identification of Misleading News Sources, Final Project for CAPP 30255 Advanced Machine Learning for Public Policy, University of Chicago.

Ezukwoke, K. I., and Zareian S. J. 2019. Online Learning and Active Learning: A comparative study of Passive-Aggressive Algorithm with Support Vector Machine (SVM), Department of Computer Science, University Jean Monnet, Saint-Etienne, France.

Hawkins, D. M. 2004. The Problem of Overfitting. *Journal of Chemical Information and Computer Scientists* 44(1): 1-12.

Huiwen, N. 2018. 4 in 5 Singaporeans confident in spotting fake news but 90 per cent wrong when put to the test: Survey. [online] The Straits Times. Available at: <https://www.straitstimes.com/singapore/4-in-5-singaporeans-confident-in-spotting-fake-news-but-90-per-cent-wrong-when-put-to-the>

Horowitz B. T. 2021. Can AI Stop People From Believing Fake News? *IEEE Spectrum: Technology, Engineering, and Science News*. [Online]. Available: https://spectrum.ieee.org/tech-talk/robotics/artificial-intelligence/ai-misinformation-fake-news.

Jakkula V. 2006. Tutorial on Support Vector Machine (SVM), School of EECS, Washington State University, USA.

Khan, J. Y.; Khondaker, M. T. I.; Iqbal, A; and Afroz, S. 2019. A Benchmark Study on Machine Learning Methods for Fake News Detection, Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Bangladesh.

Kurasinski, L., and Mihailescu, R. C. 2020. Towards Machine Learning Explainability in Text Classification for Fake News Detection. In *19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 775-781, Miami, FL, USA.

Lorent, S. 2019. Fake News Detection Using Machine Learning, Thesis, Faculty of Applied Science, University of Liege, Belgium.

Noble, W. S. 2006. What is a support vector machine? *Nature Biotechnology* 24(12): 1565-1567.

Pimpalkar, A. P., and Raj, R. J. R. 2020. Influence of Pre-processing Strategies on the Performance of ML Classifiers Exploiting TF-IDF and BOW Features. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal Regular Issue* 9(2): 49-68.

Polikar, R. 2012. Ensemble Learning. Zhang C., and Ma, Y. (eds.), *Ensemble Machine Learning: Methods and Applications*. Springer.

Rish, I. 2001. An empirical study of the naive Bayes classifier. In *Proceedings of ICJAI-01 Workshop on Empirical Methods in AI* 3: 41-46.

TODAYonline 2020. Study finds seniors, public flat dwellers more vulnerable to fake news, with two-thirds of respondents failing experiment. [online] Available at: <https://www.todayonline.com/singapore/study-finds-seniors-public-flat-dwellers-more-vulnerable-fake-news-two-thirds-respondents-fail>

Vousoughi, S.; Roy, D.; and Aral, S. 2018. The spread of true and false news online. *Science* 359:1146-1151.

Zhuo, T. 2019. NUSSU spoof page misquotes Shanmugam on religion and politics; bent on sowing discord and hatred, says minister's press secretary. [online] The Straits Times. Available at: <https://www.straitstimes.com/singapore/nussu-spoof-page-misquotes-shanmugam-on-religion-and-politics-bent-on-sowing-discord-and>