MGTA 415 Final Project - Fake News Prediction

Zezhi Lan
zlan@ucsd.edu

ABSTRACT

In the era of information explosion, fake news detection has become a hot topic in the field of Natural Language Processing. In this article, we conduct experiments on leveraging and features to identify if a piece of given news is fake or real with various NLP method. We find that all the detectors we build achieve impressive accuracy with high F1-score. Furthermore, to our surprise, simple Bag-of-words model with binary vectorizer achieves the best performance in this task.

1. ABOUT DATASET

The dataset we choose is the Fake News Data from Kaggle. It includes 6335 rows and 3 columns. The attributes are:

- **title**: It is the title of every piece of news in the dataset.
- **text**: This is the text content of each piece of news.
- **label**: It is the label to show whether such piece of news is fake or real.

First, we have a rough look of the anime file by showing its first 5 rows as shown in figure 1.

	title	text	label
0	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE
2	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
4	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL

Figure 1: First 5 Rows of Data

Second, we check the type of data in each column and whether there is a null value. We find that the data is complete and there are no null values. And all the data types are objects, we will perform data processing in the subsequent model training.

Then, we take a look at the label distribution in figure 2.

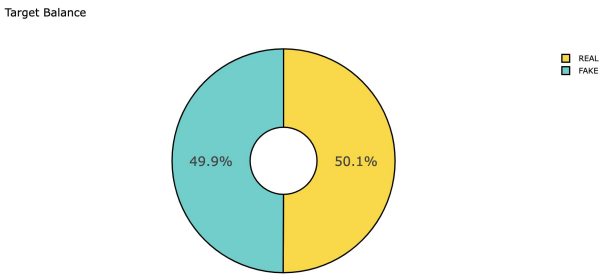

Figure 2: Label Balance

The label is really balanced. That means, in later data processing, we do not need to tackle imbalance problems.

After that, some visualizations are created to show the characteristics of our variables.

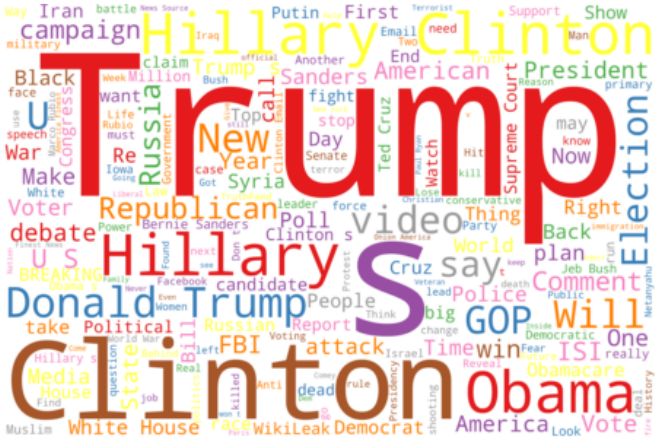

Figure 3: Title WordCloud

From WordCloud of title and text in Figure 3 and Figure 4, we can infer that the topic of news is mainly politics. With high frequency of words including 'Hillary', 'Trump', 'election' and other partisan words, the data collection time of this news database is likely to be during the 2016 presidential election.

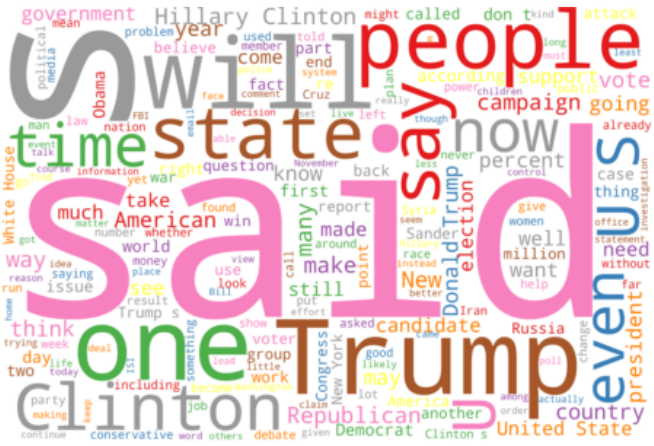

Figure 4: Text WordCloud

As the title and text data are not numerical data, we can not directly check their distributions. However, we can check the length of the title and text. It may give us some hints for the classification task. They are shown in figure 5 and figure 6.

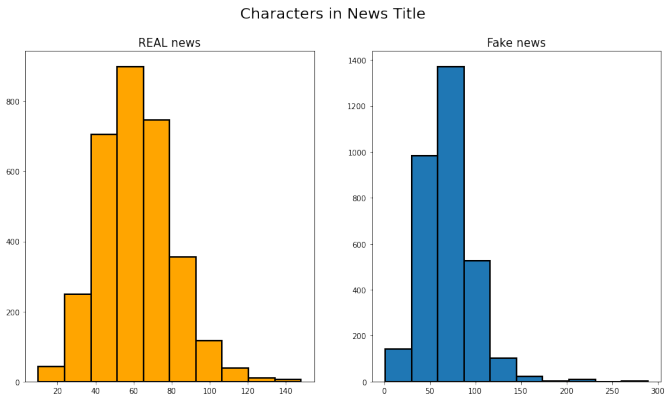

Figure 5: Title Characters

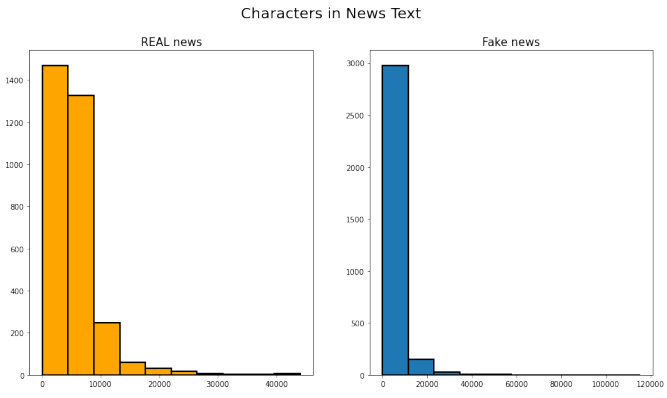

Figure 6: Text Characters

From these two figures, we can get the fact that no matter real news or fake news, most lengths of news text are below 2000 characters, and most lengths of news title are below 150 characters. The most obvious feature is that fake news tends to have more outliers in numbers than real news.

2. PREDICTIVE TASK

Fake news is nothing new, but the internet and social media have transformed how it's created and spread. We consume news through several mediums throughout the day in our daily routine, but sometimes it becomes difficult to decide which one is fake and which one is authentic. Given the massive amount of news streams, relying on human's effort to identify fake news is almost impossible and a waste of effort. Thus, automatic fake news detection is a practical Natural Language Processing (NLP) problem useful to all online content providers. To help address this issue, we are using this dataset for news classification using NLP techniques. To achieve our goal, we attempt three different approaches. They are:

- **Bag-of Words model:** In this method, we try to build vectorizer with binary, frequency and TF-IDF;
- **Word Embedding model:** We build Word2vec model with CBOW and Skip-gram, while constructing vectorizer with GloVe;
- **Language model:** We experiment N-grams model with TF-IDF.

Because of hijacked thinking or laziness, it is pretty common that people fall into fake news. Thus, to mimic the real world situation without fake news detector, the baseline we use for comparison is positive prediction, which always returns the positive label 'REAL' for all classification prediction.

Since we want to give a better measure of the incorrectly classified cases than the Accuracy Metric, the loss function and evaluation metric we use for model evaluation is micro F1-score. The model must beat our baseline to be considered as an appropriate approach for the task, and relatively higher micro F1-score indicates better performance. Micro F1-score is calculated by:

$$Micro\ F_1 = 2 \times \frac{\text{Micro-precision} \times \text{Micro-recall}}{\text{Micro-precision} + \text{Micro-recall}}$$

3. MODEL

3.1 Baseline

In real life, people tend to hold different views on the same piece of news. Some misleading messages from unverified sources would also make it more difficult to tell real news from fake ones. In order to better compare our model performance, we set a baseline model. We assume that people would take all the news as REAL without any critical thinking. Therefore, we label all the news in our dataset REAL as the baseline model predictive output. We get a score of 0.5006 since our dataset is equally labeled as 'FAKE' or 'REAL'. In the next few sections, we will introduce various models and natural language processing approaches, and then evaluate their performance based on our baseline model.

3.2 Bag-of-words Model

In this case, we train a text classifier to distinguish between real and fake news with bag-of-words models. The bag-of-words model ignores word ordering and is only concerned with whether known words occur in the document. We first pre-process the text data using tokenization and stemming. We also remove stopwords and lowercase all words. Then we split the data into training and test splits with an 80/20% ratio and build a vocabulary based on the training dataset. We consider tokens with frequency of less than 30 as rare words and replace them by a special token <UNK>. We use three document representation techniques to train the text classifier (logistic regression) as follows:

- **Binary:** each document is represented as a binary-valued vector of dimension equal to the size of the vocabulary. The value at an index is 1 if the word corresponding to that index is present in the document, else 0. This method provides us with a F1-score of 0.9479 on the test dataset.
- **Term Frequency:** a document is represented by a vector of dimension equal to the size of the vocabulary where the value corresponding to each word is its frequency in the document. This method provides us with a F1-score of 0.9392 on the test dataset.
- **TF-IDF:** each document is represented by a vector of dimension equal to the size of the vocabulary where the value corresponding to each word is its TF-IDF value.

$$TFIDF\left(t,d\right)=\log\left(f\left(t,d\right)+1\right)\times\left(1+\log\left(\frac{|D|}{df_{D}\left(t\right)}\right)\right)$$

where $f\left(t,d\right)$ is the raw frequency of the term t in the document d, and $df_{D}\left(t\right)$ represents the number of documents that contains term t in the corpus D. This method provides us with a F1-score of 0.9439 on the test dataset.

3.3 GloVe Model

GloVe stands for global vectors for word representation, which is a pre-trained model for distributed word representation. The model is an unsupervised learning algorithm developed by Stanford for obtaining vector representations for words. This is achieved by mapping words into a meaningful space where the distance between words is related to semantic similarity. It aggregates global word-word co-occurrence matrix from a corpus, and the resulting embeddings shows linear substructures of the word in vector space.

In this task, we conducted experiments on using GloVe to build word embeddings. To build the vectorizer, first, we preprocess text by removing puctuations and stopwords from nltk library, and then build 100-dimensional representation in a word2vec format with genism package. To deal with the missing value, we first transform all the matrix vectorizers to pandas series, and fill the missing value with a series of 0 with length 100. For classifier, we experiment Logistic Regression Classifier and SVC (Support Vector Classifier). This model provides us with a highest possible micro F1-score of 0.9392 using SVC on the test dataset.

3.4 Word2Vec Model

Word2vec is a collection of related models for creating word vectors. These are two-layer shallow neural networks that have been trained to reassemble linguistic word texts. Words are used to represent the network, and it must estimate the input words in neighboring spots. The bag-of-words model in word2vec assumes that the order of words is unimportant. The word2vec model can be used to map each word to a vector, which can then be used to represent the relationship between words and words in the neural network's hidden layers.

In this task, we train the word2vec model by ourselves based on the Continuous Bag of Words (CBOW) and Skip-Gram methods in order to get 100-dimension word-embeddings. These two appraoches are described as follows:

- **Continuous Bag of Words (CBOW):** In the CBOW model, it would predict words based on its surrounding word window. Therefore, it includes a context of word of continuous word window in order to predict the word in the middle.
- **Skip-Gram:** Skip-gram model uses weights of words based on its distance from the given word needed to be predicted. Words near the target words would weight more than distant words. It's reverse of CBOW algorithm in that its target word is input while context words are output.

In order to get better word-embedding representations, we do similar text preprocessing works like skipping stop words, translating punctuation to space and making all the words lowercase. The training dataset for our word2vec model is all the titles and texts in Fake New Dataset.

Comparing to CBOW method performance, Skip-Gram method performance is slightly better on Support Vector Machine model, as the micro F1-score is 0.9108 while the micro F1-score is 0.9084 in Skip-Gram method.

3.5 N-grams

N-grams model is an improvement of the typical Bag-of-Words Model that takes co-occur words into consideration, such as 'canadian government', 'high level protection'. Unigrams mean single words. Bigrams refer to pairs of words that appear next to each other. Trigrams mean a group of three words that appears in a sequence in a sentence. By making these words together, they may convey different meanings from those simply as single words.

In this task, we first process news titles and texts by converting them into lowercase and getting rid of punctuation or stop words. We extract Unigrams, Bigrams & Trigrams from the texts as our Bag-of-Ngrams Model, and build our vocabulary dictionary based on that. Looking at the words, most popular N-grams include single words such as 'people', 'president', 'republican', and phrases such as 'hilary clinton', 'obama administration', 'new york times'. We notice that the Bigram 'york times' and Trigram 'new york times' both appear in the top 1000 most popular n-grams but they almost have the same meaning, which indicates that there

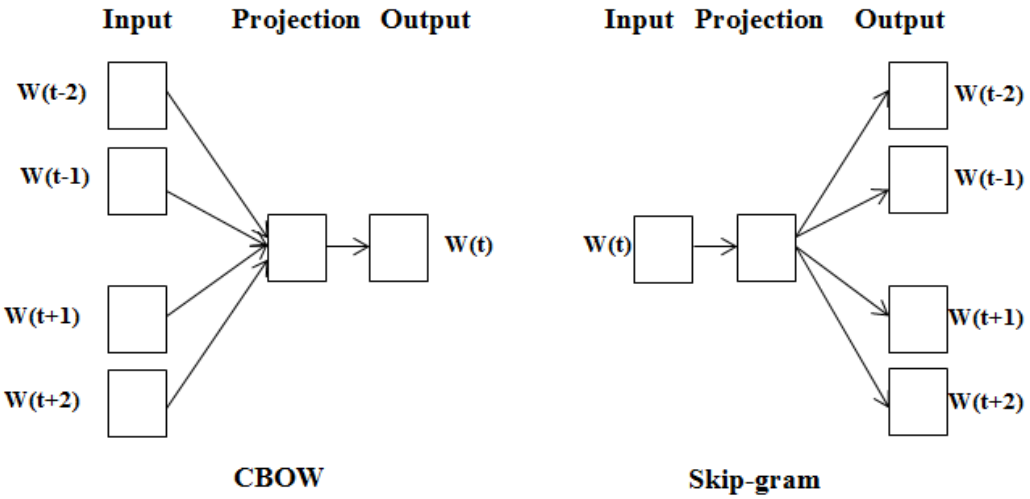

Figure 7: CBOW and Skip-Gram [1]

would be an issue of redundancy in current n-grams model. This issue could also be addressed by increasing regularization coefficient.

For each entry of review in the train set, we construct feature vector using Term-Frequency-Inverted Document Frequency (TF-IDF) and use a logistic regression model in sklearn package with a regularization coefficient of λ=100. Then we report its performance on test set. This model provides us with a micro F1-score of 0.9471 on the test dataset.

4. RELATED WORKS

Fake news have become of the biggest issues in this age of information. Several common machine learning methods have been applied to solve the problem of identifying fake news, combing with ways to processing text data. In this project, we are trying to study various approaches to extracting text features and build classifiers to identify fake news based on the features. This could potentially help to provide more insights on developing multiply combinations of word processing and classification models that could label fake news.

As described above, we are using Fake News Data from Kaggle. The data includes news title, news text and label. It is a balanced data with 50% labeled as Fake news. There are other similar datasets such as LIAR dataset which contains various statements from PolitiFact, collected by [2] for their research. Unlike their dataset which mainly focuses on political news and lebels news into six different ratings: pants-fire, false, barelytrue, half-true, mostly-true, and true, we are trying to identify Fake or Real news from multiple topics such as politics, science and sociology.

For evaluating Fake news using machine learning models, the research from [3] utilized TF-IDF of bi-grams and probabilistic context free grammar (PCFG) and tried on multiple classifiers such as Support Vector Machine (SVM) and Gradient Boosting, with Stochastic Gradient Descent model predicted from TF-IDF of bi-grams features having the highest accuracy. [4] focused on ensemble machine learning and compared Precision, Recall and F1-Score among Decision Tree, Random Forest and Extra Tree classifiers. Their ensemble approach consisting of these three tree models could reach an 100% training and testing accuracy on ISOT dataset. As for ways of processing features, [5] used google Word2Vec word embeddings to generate text features. It significantly improved model accuracy. They also proposed that for larger dataset, deep learning models such as Neural Network could also provide better performance comparing to traditional machine learning models. Recently, [6] evaluated the performance of three Deep Learning models: Long Short-Term Memory (LSTM), Neural Network with Keras (NN-Keras), and Neural Network with Tensor-Flow (NN-TF) on identifying fake news. Deep learning models outperformed all the other traditional ML models examined, with LSTM model achieved the highest average accuracy of 94.21%.

Through experiments, using word embedding such as Word2Vec or using TF-IDF to build feature vectors from texts could achieve better performance compared to other methods. While deep learning would outperformed traditional Machine Learning models, Stochastic Gradient Descent and ensemble tree models could also achieve a good performance. However, there's limited work showing comparison between different approaches to extracting text features such as GloVe, Bag-Of-Words binary feature or Bag-Of-Words terms frequency. We combined these approaches with other classification models to predict fake news. We will include detailed findings and comparisons in the next section.

5. RESULTS AND CONCLUSIONS

From our experiments and research, we conclude several findings regarding model performance, feature representations, and model parameters:

1. Micro F1-score is a reasonable evaluation metric to be chosen in our experiments. To evaluate model performance comprehensively, we should examine both precision and recall. The F1 score serves as a helpful metric that considers both of them. As we have a balanced dataset and want an easily understandable metric for overall performance, it is

Model	Micro F-1 Score
Always Predict REAL (Baseline)	0.5006
Bag-of-words (Binary)	0.9479
Bag-of-words (Frequency)	0.9392
Bag-of-words (TF-IDF)	0.9439
GloVe	0.8808
Word2vec (CBOW)	0.9084
Word2vec(Skipgram)	0.9108
N-grams(TF-IDF)	0.9471

Table 1: Comparison of Models' Performance on Test Data

reasonable to go with micro F1-score.

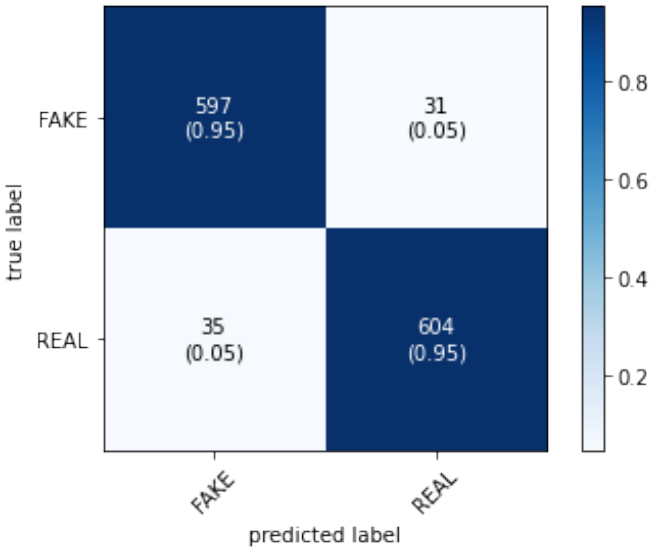

Figure 8: Confusion Matrix of Bag-of-words Model with Binary Representation Technique

2. Bag-of-words model with binary representation technique achieves the best performance among all the models. As shown in Figure 7, the prediction accuracy of this model for true and false news is approximately 95%. It is interesting for us to find that binary wins over TF-IDF, since binary seems much simpler. TF-IDF considers both term frequency and the importance of a term, while binary only focuses on the presence of a term. For a relatively small and balanced dataset with only binary classification, binary representation technique might have outstanding performance, which might not be the case for larger and more complex datasets. 1000 documents is a bit less to draw any conclusion about the vectorization technique. As the size of the vocabulary increases, TfidfVectorizer would be better able to differentiate rare words and commonly occurring words while Binaryvectorizer or Countvectorizer would still give equal weight to all words which is undesirable. So, TfidfVectorizer probably will perform better as the size of the vocabulary increases.

For TF-IDF representation technique, N-grams model performs better than bag-of-words model. As mentioned above, by taking co-occur words into consideration, N-grams is an improvement of the typical bag-of-words model.

Moreover, it is surprising for us to find that bag-of-words models outperform word embedding models. The reason might be that it is difficult for us to train accurate features in word embedding models with relatively small dataset focusing on several specific news topics.

3. For word embedding models, word2vec performs better than GloVe. GloVe is a pre-trained model, while we use the training dataset to build our own vocabulary when training the word2vec model. Since our dataset is relatively small and specific, it might happen that we cannot find corresponding vector from the pre-trained word embedding model, which leads to better performance of self-trained word2vec model.

References

[1] T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting Similarities among Languages for Machine Translation," *arXiv:1309.4168 [cs]*, Sep. 2013, arXiv: 1309.4168. [Online]. Available: `http://arxiv.org/abs/1309.4168` (visited on 03/13/2022).

[2] W. Y. Wang, ""Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection," en, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 422–426. DOI: `10.18653/v1/P17-2067`. [Online]. Available: `http://aclweb.org/anthology/P17-2067` (visited on 03/12/2022).

[3] I. S. C. on Research and i. b. Development, "Evaluating machine learning algorithms for fake news detection," in *Proceedings /*, [Piscataway, NJ] : IEEE, 2017, pp. 110–115.

[4] S. Hakak, M. Alazab, S. Khan, T. R. Gadekallu, P. K. R. Maddikunta, and W. Z. Khan, "An ensemble machine learning approach through effective feature extraction to classify fake news," *Future Generation Computer Systems*, vol. 117, pp. 47–58, 2021, ISSN: 0167-739X. DOI: `https://doi.org/10.1016/j.future.2020.11.022`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0167739X20330466`.

[5] D. Rathod, "Comparison of Techniques to Detect Fake News," *International Journal for Research in Applied Science and Engineering Technology*, vol. 8, no. 4, pp. 274–276, Apr. 2020, ISSN: 23219653. DOI: `10.22214/ijraset.2020.4043`. [Online]. Available: `http://ijraset.com/fileserve.php?FID=27456` (visited on 03/12/2022).

[6] S. A. Alameri and M. Mohd, "Comparison of Fake News Detection using Machine Learning and Deep Learning Techniques," in *2021 3rd International Cyber Resilience Conference (CRC)*, Langkawi Island, Malaysia: IEEE, Jan. 2021, pp. 1–6, ISBN: 9781665418447. DOI: 10.1109/CRC50527.2021.9392458. [Online]. Available: https://ieeexplore.ieee.org/document/9392458/ (visited on 03/12/2022).