Fake News Finder

Matt Robinson [*1], **Adeela Huma** [†1], **and Sergio Sainz-Palacios** [‡1]

[1]**7054 Haycock Road in Falls Church, VA**

ABSTRACT

With the rise in social media platforms such as twitter, facebook and online availability of news information spreads very fast. It's hard to tell that the information shared is legitimate or not and fake news stories can have real life consequences. In 2016's presidential election fake news of presidential endorsement got a lot of attention and people initially believed it. Fake news is becoming a major problem. In this paper, we explore fake news detection using document topic models. We use document vectorization techniques: TFIDF, Word2vec, Doc2vec and subsequently classification using support vector machines, multinomial baye's and random forest. The best results are shown with TF-IDF and random forests.

Keywords: Machine Learning, Fake News, Word2Vec, Doc2Vec, TFIDF, SVM, Random Forest, Naive Baye's Classifier

1 INTRODUCTION

The proliferation of fake news has recently emerged as a major problem for social media platforms such as Facebook. While the motivation for producing fake news varies, the goal is normally to generate advertising revenue by luring users into clicking on an article. Broadly speaking, there are two strategies that fake news sites use to attract clicks. First, they try to masquerade as a legitimate news source by mimicking the layout and language of a particular news site as closely as possible. Second, they use sensational article titles to grab the attention of users. Identifying fake news is an important problem, because their presence on social media platforms leads to poor user experience and the spread of disinformation. In order to address this problem, we plan to assess the feasibility of using machine learning models to identify fake news.

Our methodology for assessing this problem involves building a series of fake news classifiers, using three different vector representations of the news articles and three different classification techniques. The three vector representations that we evaluate are term frequency inverse document frequency (TFIDF), Word2Vec based dense vector representation and Doc2Vec. For each vector representation technique, we will build a classifier using Multinomial Naive Bayes, Random Forests and Support Vector Machines. We will evaluate the models according to their F1-score. In order to test the robustness of the results, we will evaluate the performance of the model on 100 random partitions of the data set. These approaches will be explained in more detail below.

The remainder of this section will include an overview of past work on document classification in general, and fake news classifiers in particular. We will also explain how we obtained our data set. In the next section, we perform exploratory analysis of the data set and compare the fake news corpus to the real news corpus. Next, we provide a detailed explanation of our methodology, including the assumptions that underly each of the models. Included in this section is a discussion of the strengths and weaknesses of each approach. After discussing our methodology, we present our model evaluation results. We conclude by discussing the meaning and relevance of our results, and suggesting several avenues for future research.

1.1 Related Work

In [15] Martin et al. describe the three different paradigms for fake news detection: fake news detection based on knowledge, on context, and on style. Fake news detection based on knowledge, also known as fact checking is done, by utilizing techniques from informational retrieval, semantic web and linked

[*] robinmw@vt.edu

[†] ahuma@vt.edu

[‡] ssainz@vt.edu

open data. Context based fake news detection uses social network analysis where the spread of false information is analyzed. This techniques is also known as rumor spreading. Style based fake news detection uses computational linguistics and natural language processing to detect false information at sentence level. In [15] Martin et al do fake news detection by style based text categorization i.e classifying news text as fake or real. Knowledge-based Fake News Detection: Etzioni et al. [8] proposed to use Text Runner tool to extract and index factual knowledge from the web and match it against the indexed facts to identify inconsistencies. Magdy et al. [13] proposed a statistical model to check factual statements from a given document and then checked how frequently they are supported by documents retrieved from web. Ginsca et al. [9] addressed the issue with [[8], [13]] of a website's reputation and reliability. Assuming factual knowledge is available for a domain, Wu et.al [19] try to estimate the truthfulness of a fact by perturbing it. Ciampaglia et al. [7] propose fact finding as a problem of finding shortest path between two concepts. Shi and Weninger [18] solve fake news detection as a link prediction task. Context-based Fake News Detection: Acemoglu et al. [1] models spread of misinformation in social networks. In [[5], [14]] authors purpose models to limit the spread of misinformation. Kwon et al. [11] studied the spread of misinformation on Facebook during presidential election. Style-based Fake News Detection: In this approach there are two ways to detect fake news a) deception detection in text and b) style based text categorization. Deception detection originates from forensic linguistics and builds on the Undeutsch hypothesis— a result from forensic psychology asserting that memories of real-life, self-experienced events differ in content and quality from imagined events. The hypothesis leads to the development of forensic tools to assess testimony at the statement level, such as Criteria-based Content Analysis (CBCA) and Scientific Content Analysis (SCAN). Authors in [[11],[6], [16]] propose techniques to detect tweets that convey uncertain information, crap detector for news and fake news detection using rhetorical structure theory respectively. Style based text categorization was proposed by Argamon-Engelson et al. [3] as an alternative to topic based text categorization. Afroz et al. [2] attempt to detect texts whose authors tried to confuse their writing style to avert author identification. Badaskar et al. [4] using language models identified real and fake news. Rubin et al. [17] utilized variants of tf-idf weighted lexical vector space model to identify fake news. Martin el al. [15] employs bag of words model for topic based classification model.

1.2 Data

The data for this project was collected through a web scraper developed by the research team. Articles were scraped over a two week period in March 2017. In our data set, articles are labeled as real or fake based on the source of the article. Given this labeling strategy, one assumption that underlies our research is that trustworthy news is produced by trusted sources. Of course, this assumption may not always hold in reality. Generally untrustworthy sources may occasionally produce high quality articles. Likewise, since their revenue model is largely driven by advertisement, mainstream news outlets have a financial incentive to sensationalize some news stories. Nevertheless, we believe that this is a reasonable labeling strategy. The alternative would be to fact check and hand label each of the articles, which is prohibitively time consuming.

Table 1. Data Sources

Fake News	Real News
abcnews.com.co	washingtonpost.com
usatoday.com.co	nbcnews.com
infowars.com	cnn.com
prntly.com	foxnews.com
naturalnews.com	cnbc.com
nationalreport.net	msnbc.com
	bloomberg.com
	cbs.com
	nytimes.com
	wsj.com
	bbc.com

A list of the news sources for this project appears in Table 1, and is partitioned by type. Here, it is worth noting how we selected news sources for the real and fake news corpora. In general, types of sources appear in the fake news category. The first are sites that are obviously designed to mimic the layout of a real news website, such as abcnews.com.co. Since these sources are trying to masquerade as another news site, they are obviously illegitimate. The second grouping of fake news articles consists of highly biased sources, such as infowars.com. These sites have a track record of producing dubious content and peddling conspiracy theories. The real news corpus consists entirely of main stream news sources. Content from these sources undergoes fact checking in accordance with news industry standards. While some of these sources have a reputation for producing partisan editorial material, it would be unusual for any of these sources to intentionally publish outright falsehoods.

Over the course of two weeks, our scraper collected about 35,000 real news articles and 1,620 fake news articles. In order to provide a balanced data set for the classifier, we sampled 1,620 as negative examples. This data is the basis of all of the analysis that follows.

2 EXPLORATORY DATA ANALYSIS

For Exploratory Data Analysis of Fake news detector two types of plots are drawn:

- **Word-frequency plot** - displays the frequency of words in documents.
- **Word cloud** - Word-cloud or tag cloud is a visual representation of text data. Tags are usually single words and the importance of each tag is shown with font size or color. This format is useful for quickly perceiving the most prominent terms.

The word frequency plots for fake and real news are shown below. The word frequency is almost same in both types of news article.

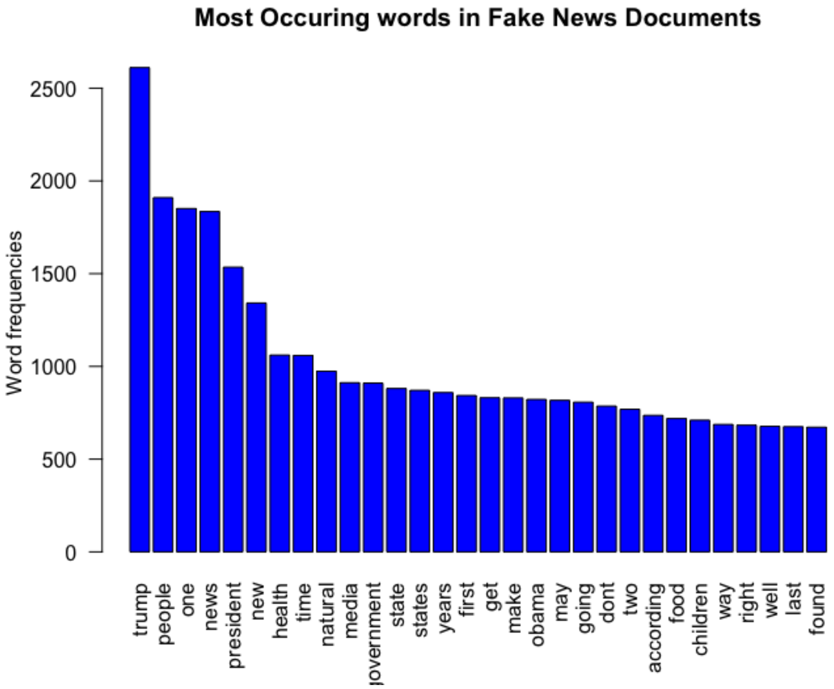

Figure 1. Fake news word frequency

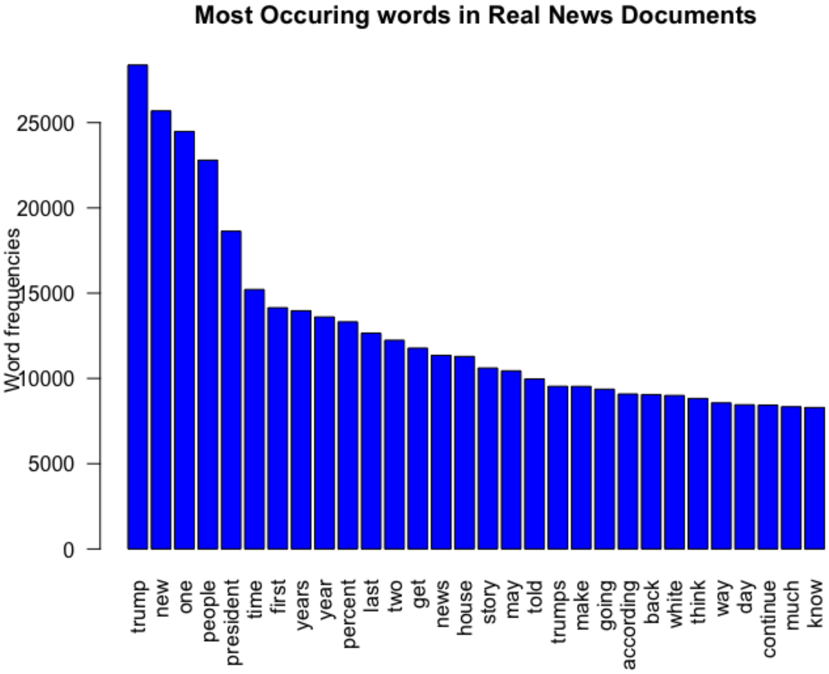

Figure 2. Real news word frequency

In addition to word frequency plots, we produced a word cloud for fake and real news articles. The word trump is most used word in fake and real news articles and some of the other frequent words are also same. That means usage of words is similar in both classes. This may be intentional, since it makes fake news seems real.

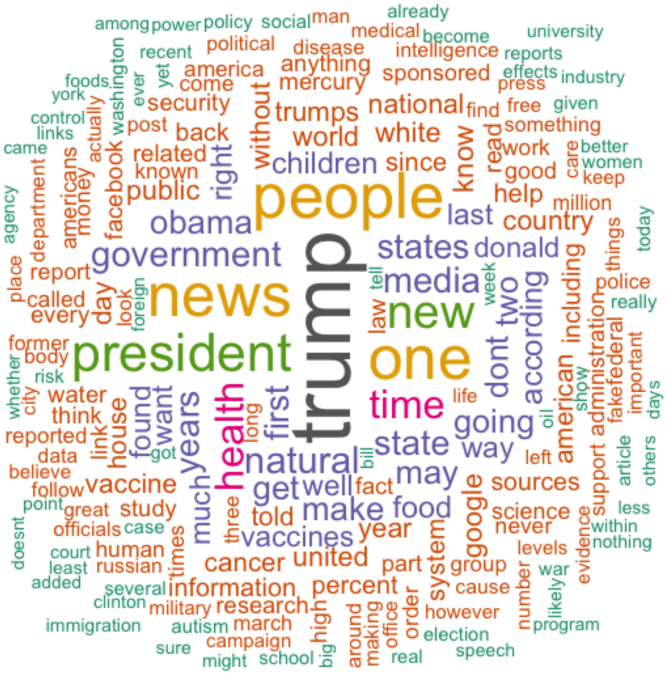

Figure 3. Fake news word cloud

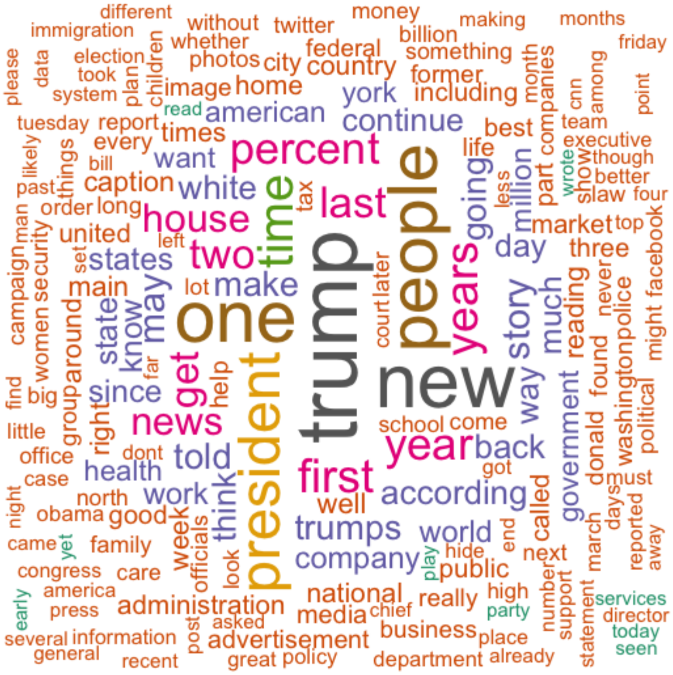

Figure 4. Real news word cloud

3 METHODS

3.1 Vector Representations

In order to build features for documentation, we will use three techniques for mapping news articles to vectors. The first technique is Term Frequency-Inverse Document Frequency (TF-IDF). Intuitively, TF-IDF provides a count of how often a word appears in a document, weighted by how important that word is in the corpus. Specifically, the term frequency of a word is how often that word appears in a document. The inverse document frequency, which measures how important the word is in the corpus, is computed by dividing the total number of documents by the total number of documents containing the term, and then taking the logarithm of that value. Combining these two components, the measure becomes $w_{i,j} = tf_{i,j}\log(\frac{N}{df_i})$. The advantage of representing the articles using TF-IDF is that it helps the classifiers identify the difference in the specific wording of fake news articles. For example, if the word 'conspiracy' is used heavily in fake news but not in real news, it could be an important feature for the classifier. The disadvantage is that it has no understanding of the semantic content of an article. Since it treats the content of the article as a bag of words, similar words are not necessarily close to one another in the TF-IDF vector. As a result, the classifier may not focus on distinct words that fake news articles use in the same context.

Our second technique uses Word2Vec to generate feature vectors for the news article. Word2Vec vector representations for a word are the result of a neural network that maps words to a dense vector space. Since the technique is designed to recognize when words appear in the same context, similar words are close to one another in the Word2Vec vector space. In order to generate vector representation for a news article from Word2Vec, we take the maximum value for each position in the vector, as suggested by De Boom, Van Canneyt, Demester and Dhoedt. The advantage of Word2Vec is that it is less sensitive to differences in wording, because articles with similar words will produce similar vectors for the news article. As a result, the Word2Vec may perform well if fake news consistently addresses different topics than real news. A disadvantage is that if real news and fake news use similar words in similar contexts, the classifier may have difficulty distinguishing them. It could, for instance, be the case that very specific wording differences are the best way to separate fake news from real news. If that is the case, TF-IDF could turn out to be more effective.

The third technique uses Doc2vec. Doc2Vec is described by Le and Mokolov [12]. Doc2Vec is similar to Word2Vec in that it also uses neural networks to generate feature vectors. Doc2Vec first executes Word2Vec internally to generate word vectors. Next, it runs a second phase of training while also adding one feature vector that represents the document. The second phase has as input the document vector concatenated with the context related words vectors and its output is to predict the next word within the context (this happens iteratively on all words of the context). The advantage of Doc2Vec is that it preserves additional information about the news article. An example of this information is the ordering of the words within context. Meanwhile, in the Word2Vec technique we just take the maximum word vector and it could be repeated with other articles. The Doc2Vec vectorization still depends on the kinds of words that appear on the articles, and therefore if both fake news and real news discuss similar topics, classifiers might not distinguish between the two.

3.2 Classification Methods

In addition to the three techniques for converting documents to vectors, we will evaluate three different classification techniques: Multinomial Naive Bayes, Support Vector Machines and Random Forests. The Multinomial Naive Bayes model maintains strong assumptions about the independence of words in the documents. In particular, it assumes that the words that appear in a document are independent of one another, given the class of the article. Of course, this assumption never holds in reality. However, if the dependence between words is weak, it could be a reasonable approximation. In addition, Multinomial Naive Bayes could be advantageous in terms of training time, depending on the size of the corpus. Another advantage of Multinomial Naive Bayes is that it does not require as much tuning as other methods. If the performance of a Naive Bayes classifier is similar to that of a more expensive classifier, then may be preferable for both of these reasons.

Support Vector Machines classify documents by finding a separating hyperplane that maximizes the margin between the two classes. Unlike Multinomial Naive Bayes, Support Vector Machines allow for dependence between features, given the document class. This flexibility, however, comes at a cost in terms of model tuning. In particular, the effectiveness of the SVM may depend on choosing a suitable kernel function. Selecting the best kernel function is non-trivial, and may involve significant experimentation. As a result, it may be more cost effective in terms of modeling time to choose a method with fewer hyperparameters to adjust.

Finally, Random Forests are an ensemble method that averages over a number of decision trees. Each decision tree is based on a subset of the feature space, and usually considers $\sqrt{n}$ features, where n is the dimension of the feature space. Unlike Multinomial Naive Bayes, Random Forests do not make strong assumptions about the dependence of the features and can fit highly non-linear classifiers. While there are some hyperparameters to tune, they are easier to calibrate than kernel functions. As a result, Random Forests provide a good middle ground between the advantages and disadvantages of Naive Bayes and SVMs.

3.3 Model Evaluation

During our experiments, we will evaluate all three vectorization techniques in combinations with all three classification algorithms, for a total of 9 possible modeling choices. Each combination will be evaluated by computing the F1-score for the model over 100 random partitions of the data set. For each random partition, the model will be trained on 80% of the data set and tested on 20% of the data set. A random seed will be set so the experiments are repeatable. The F1-score is computed as $F = 2\frac{pr}{p+r}$ where p is the precision for the model and r is the recall. Once an F-score is computed for each partition, we perform a two-sample t-test between each pair of models. This will help determined if the difference in performance between two models is statistically significant. We will recommend a classification techniques based on the results of these experiments.

3.4 Vectorization parametrization

For Word2vec and Doc2vec, we have several parameters to tune: context size, feature size and minimum word count. The context size (also known as window size) is the number of words that comprise the context. And context is the number of words that will be analyzed and trained together. During training the context is shifted one position at the time until all words in the document are covered. The feature size is the number of features the final feature vectors will have. Minimum word count is the minimal number of times a word appear for it to be considered for training.

We tested the performance of Word2Vec and Doc2Vec against the dataset with 326 positive and 326 negative samples. We vectorized the dataset with the chosen parameters, then apply random forest classifier and calculate f-score as a measure of performance. We tried first a combination of window size and feature size, and then we pick the the best window size parameter and tried another combination of feature size and minimum word count.

The values chosen for subsequent experiments are: Word2Vec: window size: 4, feature size: 500, minimum word count: 10. For Doc2Vec: window size: 10, feature size: 3000, minimum word count: 50.

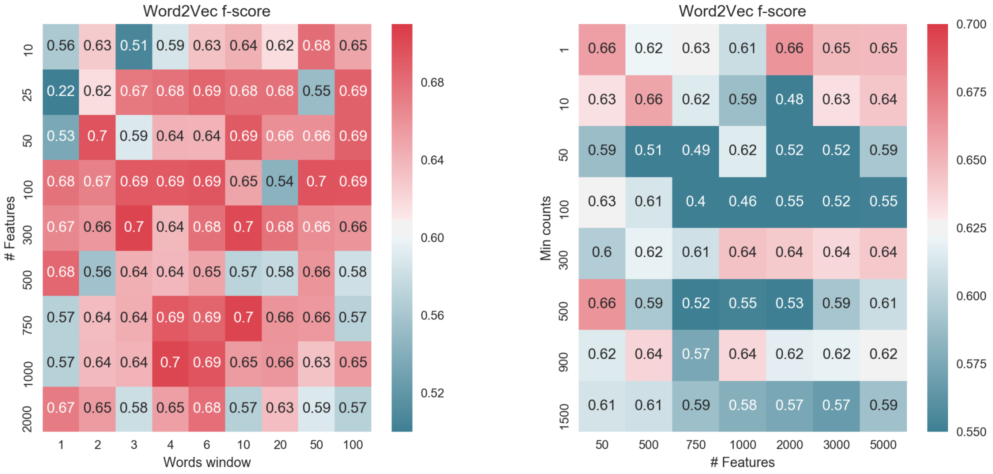

Figure 5. Word2Vec parametrization. Best parameters: window size: 4, feature size: 500, minimum word count: 10

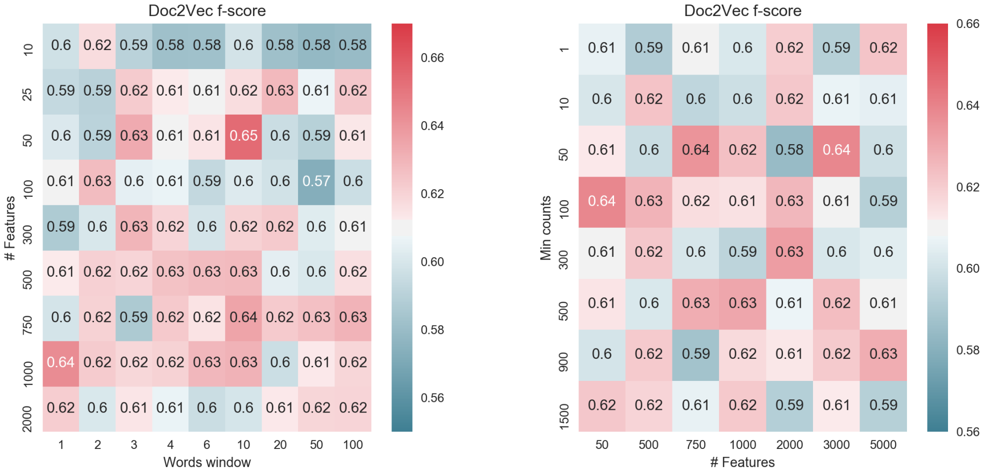

Figure 6. Doc2Vec parametrization. Best parameters: window size: 10, feature size: 3000, minimum word count: 50

4 RESULTS

Two datasets were classified, one balanced dataset comprised of 326 positive and 326 negative entries. A second unbalanced dataset comprised of 326 positive and 4,713 negative samples. For the first dataset, we find F1-scores of around 0.60 for Word2Vec vectorization, 0.61 for doc2vec and approximately 0.84 for

TFIDF vectorization. Meanwhile, second unbalanced dataset vectorized with Word2Vec and Dov2Vec render low performance for all classifiers, and for SVM it classifies all values as real news (this causes the F1-score to become undefined). The second dataset vectorized with TFIDF renders F1-score values of around 0.06:

Fake news classification (326 fake news, 326 real news)			
F1-Score	Vectorizers		
Classifiers	TF-IDF	Word2Vec	Doc2Vec
Random forests	0.841	0.60	0.616
SVM	0.840	0.60	0.623
Multinomial Bayes	0.808	0.524	0.573

After computing these scores, we recomputed them on 100 random partitions of the dataset in order to test the statistical significance of the results. While the Random Forest with TF-IDF had a higher mean F1-score than SVM, the differences in the scores were not statistically significant over the 100 runs. The advantage of both of these methods over Naive Bayes, however, was significant. For Word2Vec and Doc2Vec, no classifier had a statistically significant advantage over any other. Doc2Vec, however, did systematically outperform Word2Vec.

5 CONCLUSION

We found it is possible to identify fake news using document vectorization and different classifiers with F1-score of about 0.81. This is good performance when compared with random guess. Also, when the training set was unbalanced the performance drastically dropped.

Our results show that the fake news classifier is very sensitive to how the feature space is represented, but not very sensitive to what classification technique is used. Based on our exploratory data analysis, we know that real and fake news cover roughly similar topics. This leads us to believe that dense vector representations of news articles perform poorly because many fake and real news articles are conceptually similar. Likewise, the strong performance of TF-IDF suggests that certain specific words are more common in real news than in fake news. Taken as whole, this leads to the conclusion that real and fakes news articles speak differently about the same events.

5.1 Applications

Media companies could use this methodology to classify fake news. Nevertheless, we should consider the disadvantages: it requires plenty of computing and storage resources to produce the document vectors. It requires N documents times M features and W words times M features of storage. Another consideration is that as time goes by, the topics and words that fake news and real news articles write about will change. Therefore there is no guarantee that the classifiers will perform in the same way as present. For example, if tomorrow fake news share larger number of words and word frequency as real news, the classifier would not work as well as today.

5.2 Future Research

Further research is needed in two directions: (1) how can we better classify the fake news when the dataset is unbalanced? In our experiments we saw that when there are many more real news compared with fake news, the classifier will classify most samples as real news. (2) Generate text style features to characterize the document style, similar to Horne et. al. did [10], for example how many verbs, nouns, stop words, how deep is the syntax tree, average length of sentence, of word. These document style features will not be as variable in time as the word topic models (Word2vec, Doc2vec). Additionally the results should be validated against larger datasets.

Another interesting question, more along the lines of the research presented in this paper, is whether or not machine learning methods can differentiate between the real and fake version of the same site, for instance abcnews.com and abcnews.com.co. The difficulty in building a classifier for this task is that the fake news version of a site often produces a low volume of news relative to the real version, so it may be difficult to build a training set with a sufficient number of positive examples to be useful. Nevertheless, this more limited version of the classifier we developed in this paper could be quite useful for blocking unwanted content.

6 CONTRIBUTION

- Adeela Huma explored related work on fake news detection, exploratory data analysis of dataset
- Matt Robinson wrote the web scraper, the TFIDF model and a class that handled testing the statistical significance of the differences in model performance
- Sergio Sainz-Palacios helped with Word2vec and Doc2vec parametrization as well as perform experiments.

REFERENCES

[1] Acemoglu, D., Ozdaglar, A., and ParandehGheibi, A. (2010). Spread of (mis) information in social networks. *Games and Economic Behavior*, 70(2):194–227.

[2] Afroz, S., Brennan, M., and Greenstadt, R. (2012). Detecting hoaxes, frauds, and deception in writing style online. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 461–475. IEEE.

[3] Argamon-Engelson, S., Koppel, M., and Avneri, G. (1998). Style-based text categorization: What newspaper am i reading. In *Proc. of the AAAI Workshop on Text Categorization*, pages 1–4.

[4] Badaskar, S., Agarwal, S., and Arora, S. (2008). Identifying real or fake articles: Towards better language modeling. In *IJCNLP*, pages 817–822.

[5] Budak, C., Agrawal, D., and El Abbadi, A. (2011). Limiting the spread of misinformation in social networks. In *Proceedings of the 20th international conference on World wide web*, pages 665–674. ACM.

[6] Chen, Y., Conroy, N. J., and Rubin, V. L. (2015). News in an online world: The need for an "automatic crap detector". *Proceedings of the Association for Information Science and Technology*, 52(1):1–4.

[7] Ciampaglia, G. L., Shiralkar, P., Rocha, L. M., Bollen, J., Menczer, F., and Flammini, A. (2015). Computational fact checking from knowledge networks. *PloS one*, 10(6):e0128193.

[8] Etzioni, O., Banko, M., Soderland, S., and Weld, D. S. (2008). Open information extraction from the web. *Communications of the ACM*, 51(12):68–74.

[9] Ginsca, A. L., Popescu, A., Lupu, M., et al. (2015). Credibility in information retrieval. *Foundations and Trends® in Information Retrieval*, 9(5):355–475.

[10] Horne, B. D. and Adali, S. (2017). This just in: Fake news packs a lot in title, uses simpler, repetitive content in text body, more similar to satire than real news. The 2nd International Workshop on News and Public Opinion at ICWSM.

[11] Kwon, S., Cha, M., Jung, K., Chen, W., and Wang, Y. (2013). Prominent features of rumor propagation in online social media. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 1103–1108. IEEE.

[12] Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In Jebara, T. and Xing, E. P., editors, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196. JMLR Workshop and Conference Proceedings.

[13] Magdy, A. and Wanas, N. (2010). Web-based statistical fact checking of textual documents. In *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, pages 103–110. ACM.

[14] Nguyen, N. P., Yan, G., Thai, M. T., and Eidenbenz, S. (2012). Containment of misinformation spread in online social networks. In *Proceedings of the 4th Annual ACM Web Science Conference*, pages 213–222. ACM.

[15] Potthast, M., Kiesel, J., Reinartz, K., Bevendorff, J., and Stein, B. (2017). A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*.

[16] Rubin, V. L., Conroy, N. J., and Chen, Y. (2015). Towards news verification: Deception detection methods for news discourse.

[17] Rubin, V. L., Conroy, N. J., Chen, Y., and Cornwell, S. (2016). Fake news or truth? using satirical cues to detect potentially misleading news. In *Proceedings of NAACL-HLT*, pages 7–17.

[18] Shi, B. and Weninger, T. (2016). Fact checking in heterogeneous information networks. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 101–102. International World Wide Web Conferences Steering Committee.

[19] Wu, Y., Agarwal, P. K., Li, C., Yang, J., and Yu, C. (2014). Toward computational fact-checking. *Proceedings of the VLDB Endowment*, 7(7):589–600.