Detecting Fake News with Sentiment Analysis and Network Metadata

Maniz Shrestha
Department of Computer Science
Earlham College
Richmond, IN 47374
mashres15@earlham.edu

ABSTRACT

With the proliferation of the Internet and social media, there is now a voluminous amount of news, articles, and other text available online. These resources, while making communication and information flow easier and faster, have jeopardized the veracity of the news that is being distributed. The impact of fake news has been so deep-rooted in society that it even affected the US election of 2016. This project used a combination of sentiment analysis and network metadata in a Machine Learning model to classify fake news. The study created a scraping tool to accumulate data related to the news and leveraged four different sub-pipelines for feature engineering and feature extraction. The fake news model was trained on the Random Forest Classifier. Results show that the proposed model achieved a F1-score greater than 88%. The study has also developed a web interface to take a URL of the news and display whether the news is fake.

KEYWORDS

Machine Learning, Sentiment Analysis, Fake News, Random Forest, Metadata

1 INTRODUCTION

Fake news is any form of false story or content spread on the internet to influence people's view to gain inimical benefits[24]. Detecting fake news in the digital world is a significant challenge in overcoming the widespread dissemination of rumors and biases. Although there has been significant progress in fake news detection, a concrete set of solutions is yet to be established as the standard. Companies such as Facebook, Twitter and Google are facing challenges in tackling this problem to ensure a platform where people can trust the newsfeed content.

Fake news affects human judgment and behavior. In the spring of 2018, there was an article spreading the news that "Cadbury chocolate is infected with HIV-Positive Blood" with a video of boxes of the chocolate being burnt. This post gained traction on Facebook, especially in South-Asian countries such as India. Rumors began to spread, damaging Cadbury's reputation, and even people who did check other sources to make sure that the news was false became hesitant about buying those chocolates[25]. Thus, it is imperative to detect and limit the spread of fake news.

This project built a classification model using machine learning to detect fake news and implemented the machine learning model using a web-based application. In this project, the fake news detection is a binary classification problem - news is either fake or reliable. A user-friendly web interface was built to enable users to easily query a news source using a URL and determine if the news is fake (the application does not provide a percentage probability score for the classification).

The term "fake news" can be applied to various forms of content. It can be a misleading video, an altered image, a biased news article that does not represent the reality or even posts in social media that are blatantly false. This project is focused on the text and metadata of the fake news, and pictures and videos for fake news detection were not considered. Fake news can also be found in different domains such as politics, finance, sports, entertainment, and others. Because different domains differ in the content type and the choice of words, there are challenges associated with building a classifier that works for all kinds of domains. This issue was addressed in the study by using a fake news dataset that covered different domains ranging from politics, finance, sports, entertainment, blogs, etc.

In this study, I implemented a hybrid approach, which combines the sentiment analysis with network approach by using metadata to detect fake news. While most of the research[1, 5, 9, 19, 22] has focused on either linguistic cue approaches (with machine learning) or network analysis approaches, this innovative hybrid approach can outperform previous methods by using the merits of both the methods. Specifically, the intuition of the hybrid model is that the feature sets of linguistic cues will complement the metadata to reveal a visible pattern in fake news.

The following are the major contributions of this research project:

- This study used Facebook Analytics metadata for fake news detection.
- The machine learning model implemented feature engineering for text and metadata.
- The project built a scraping tools to accumulate new source and social media data for article content, metadata, and social media popularity.
- The project developed a web-based application that takes a URL to classify the news or article as fake or reliable.
- The web-based application stores the query with fake news label in a dataset which over time can be used as an extension of the Fake News Corpus.

The rest of this paper is organized as follows. Section 2 discusses the strategies of prior researchers and highlights the merits and

Earlham College, Fall 2018, Richmond, IN

demerits of their approaches. Section 3 explains the fake news detection algorithms and its implementation details. Section 4 presents extensive evaluation and experimental results of the proposed classification model. Finally, Section 5 draws the conclusions and presents future work in the area of research.

2 RELATED WORK

Automating the process of detecting fake news requires algorithms that can look at different features exhibited by fake news. In a broad perspective, researchers primarily used two prominent approaches, namely Linguistic Approaches and Network Approaches. This section looks at some related works that have leveraged these techniques to detect fake news and explores the literature on the Linguistic Approach, the Network Approaches and the machine learning classifiers used in implementing the models.

2.1 Linguistic Approach

Linguistic approach uses linguistic cues to find patterns in fake news and uses those patterns to detect fake news. Research has shown that deceivers use language in a specific way to make fake news seem legitimate. For instance, Singh et al. found that fake news tends to be shorter in length, convey less expertise, appear more negative in tone, and denote lesser "analytical thinking with more informal, personal, here-and-now, and narrative thinking"[21]. They concluded that fake news articles also appeared to be more authentic.

Similarly, Feng and Hirst claimed that when people lie, language "leakage" occurs, which is the use of negative emotion words, similar conjunction and different patterns of the pronoun[9]. These verbal patterns, although hard to monitor, provide information that has helped people to identify fake stories or lies. The linguistic approach uses these language leakages and deceptive cues found in the content to detect fake news[6]. In Machine Learning and Natural Language Processing, words have to be represented in numerical form for machines to understand and apply algorithms. The following are representations that researchers have used to convert raw text to a sensible vector representation.

2.1.1 Bag of Words Data Representation. Bag of words is the representation in analyzing cues and frequencies of a word in a text. The bag of words is like a JSON object containing a list of words as keys and the word frequency as the value. Using single words and n-grams prevalent in fake articles, this approach can check for dominance of such words in the text to conclude whether the material is fake[1]. Further, researchers have used part of speech tagging of words[6] as well as location-based[12] tagging to reveal cues.

The simple bag of words representation succumbs to the high frequency of common words such as pronouns (I, he, she, they), articles (a, an, the), etc. The Term Frequency-Inverse Document Frequency (tf-idf) transformation overcomes this inefficiency by penalizing the frequency of the words that are used heavily in a document; it provides a better representation for the frequency of the words in the documents that have been adjusted for words occurring frequently.

The limitation of the bag of words approach is that it is solely based on using words in the document without taking into consideration any context information. Alternate word representations such as word vector representation (see Section 2.1.4) consider the word context in a document.

2.1.2 Syntax Analysis. Since analysis of words is not enough for predicting fake news, other linguistic approaches such as analysis of the syntax and grammar of the language have to be taken into consideration. Researchers have used Probability Context-Free Grammars (PCFG) to transform sentences into parse trees that describe sentence structure [6][8]. Feng et al. used PCFG to transform sentences to a recursive syntax structure (such as noun and verb phrases) parse tree for deception detection with an accuracy of 85% [8]. Syntax Analysis is widely-used for sentiment analysis. Third party tools such as the Stanford Parser[7] and AutoSlog-TS syntax analyzer[6] are often combined with other linguistic and network approaches for better performance[6][9].

2.1.3 Semantic Analysis. In most cases, the truthfulness of certain reviews or text can be predicted by examining the comments and similar articles. If most similar articles are not in line with the news, it is most likely that the news might be biased or fake. Similarly, the comments on the article can be used to evaluate whether the facts in the article are reliable.

Semantic Analysis approach is like an extension of the n-gram model plus syntax model along with profile compatibility features[6][9]. Feng and Hirst applied this approach by aligning author's profiles with veracity assessment and gained an accuracy of 91% in detecting deception[9]. The major drawbacks of this approach includes the difficulty in automatically finding similar articles, checking profile compatibility, and accounting for different words that imply the same thing.

2.1.4 Word Vector Representation. While there are traditional representations such as one hot word encoding which maps a word to a unique vector, there are many limitations of such representations. One hot representation is a sparse matrix containing N dimensions for N words which suffers from the curse of dimensionality as the vocabulary grows[28]. Moreover, the representation suffers from data sparsity as it cannot handle words that are not seen in the training set[26].

Mikolov et al. proposed alternate word to vector models, Skip-gram and Continuous bags of word (CBOW), to combat the previously-mentioned challenges with word representation[13]. They proposed a two-layered neural network model to process raw text into a word vector. The skip-gram and CBOW models group similar words embedding closer in a vector space so that similar words such as "Man and King" will appear closer than "Man and Car". In the skip-gram model, a word in the sentence is used to predict a larger context while the context of a sentence is used to predict a target word in the CBOW.

As an extension to the word vector models, Mikolov proposed a vector representation (Doc2Vec[11]) for a document rather than a sentence. This extension adds a paragraph or a document token along with the words in the documents. The paragraph token can be thought of as another word, but it acts as the topic of the document[11]. Implemented using the feedforward and recurrent neural networks, the Doc2Vec model achieves an error rate of 7.42%[11] which is 1.3% absolute improvement over the best previous result of Wang & Manning [27]. In this project, the Doc2Vec model was used to represent the text because of its superior performance.

2.2 Network Approaches

The internet contains huge amount of metadata that can be leveraged to predict the reliability of the source. Twitter, Facebook, and Google have large network dataset associated with every user that can be used in the network approaches.

2.2.1 Metadata. The network approach examines the metadata such as URL, authors, social media likes, etc. to get insight into whether the source is reliable. Chu et al. used metadata to determine the behavior of questionable sources[4].

My model takes into account the metadata of the site in conjunction with the features generated through sentiment analysis. The convergence of these sets of distinct features provides the machine learning more patterns to discover which ultimately leads to a better classification model.

2.2.2 Linked Data and Fact Checking. The veracity of the news can also be determined by checking facts mentioned in the news. Drawing a network relation for facts checking is one approach for this. Fact checking is merely a way of using the knowledge networks to check facts. Certain factual statements containing data or relationships can be verified using trusted sources. One of such methods is to use knowledge networks and publicly available structured data, such as DBpedia ontology, or the Google Relation Extraction Corpus (GREC)[6].

Although fact checking has several advantages in correctly classifying fake news, it is a separate research domain on its own and, given the time constraint, this research project does not look into fact checking.

2.3 Machine Learning Classifiers

Researchers have utilized machine learning algorithms with complex mathematical models to learn patterns from the labeled training data. For instance, the clustering algorithm can be used to find the numeric distance between texts to classify fake news. Zhang et al. built a model using Support Vector Machines (SVM) that uses different clustering methods to calculate distance functions between the text data in a linear space [29]. Support Vector Machines (SVM)[10] use the training data to find an optimal hyperplane in n-dimensional space that can be used for classification.

Oraby et al. used the Naïve Bayesian algorithm to classify text based on the correlation between a given variable such as syntax and the other variables present in the model[14].

Natali et al. developed a Capture, Score, and Integrate model (CSI) using recurrent neural networks[18]. This system is one of the few hybrid implementations that considers news article representation for linguistic analysis and scores the users based on their behaviors.

Random forest is an ensemble learning method that uses a collection of decision trees trained on a random subset of the data for prediction[2]. A decision tree is a tree with a set of rules or questions that are traversed in a particular order to output a label. The random forest trains multitudes of decision trees, all of which vote for a classification. My project implements the random forest algorithm to train the classification model (see Section 3.3 for details).

3 DESIGN AND IMPLEMENTATION

I propose an end to end system to predict whether a news article is fake. It accepts a URL and classifies the news as fake or reliable. This section dives deeper into the system architectures, the machine learning pipeline and the integration of the machine learning model with the whole system.

3.1 Overview

As shown in Figure 1, there are four major components in the system: front-end web user interface, back-end flask server, machine learning model and database layer.

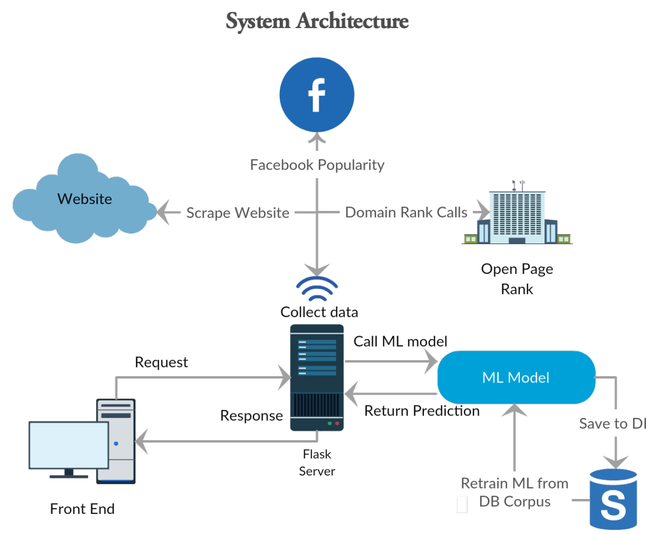

Figure 1: System Architecture Diagram.

Front-end Input: The web interface takes a URL as input and requests the Flask back-end for a prediction.

Flask Back-end: The back-end takes a URL and makes API calls to collect the text content, metadata and Facebook trend metrics (reactions, shares, and comments). Python's Newspaper library is used to scrape the website for text content and name of the author. The metadata such as domain score and domain rank are fetched through an API call to the Open PageRank server. The domain metadata are collected from the Open PageRank because it is an open source page ranker and is a substitute for the former Google PageRank (Google closed the Google PageRank API to the public in 2016 [23]). For the popularity metrics in social media, the system uses Facebook's Graphs API to collect the total counts of reaction, shares and comments of a particular business or page in Facebook. After 2015, Twitter has removed the API to track the share count of a URL and so this system does not use Twitter as a source of share metrics.

Machine Learning Model: The accumulated text and metadata of the news source are passed to the trained machine learning model to predict the reliability of the news. Section 3.2 explains details of the machine learning pipeline used in the model.

Database Entry and Result: After the prediction is made, the back-end stores the data along with the prediction in the corpus database and returns the result to the front end.

3.2 Machine Learning Pipeline

My machine learning pipeline is composed of four sub-pipelines: text pipeline, sentiment pipeline, numeric pipeline, and hashing pipeline. Each of these sub-pipelines implements data preprocessing and feature engineering to extract features used for training the classifying algorithm. Using the sub-pipelines (see Figure 2), I was able to structure the code in a way that is modular, easier to read and avoids code duplication. The inputs for my model are listed in Table 1.

Table 1: The input features for the model

Content	The text of the article/news
Title	The title of the news
Reactions count	Facebook reaction count (like, happy, sad, angry) for the particular domain
Comments count	Facebook comments count for the particular domain
Shares count	Facebook shares count for the particular domain
Domain score	The score of the domain collected from Open PageRank
Rank	The overall rank of the domain
Domain	The domain of the news source
Authors	The authors of the news

3.2.1 Text Pipeline. The text pipeline transforms the text content into a vector format that the algorithm can use. Gensim[16], a python NLP library, was used to implement the document to vector(doctoVec) model using the hierarchical softmax algorithm[13]. This specific model was chosen because it was previously shown to perform significantly better than models such as Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA) [13]. The output of doctoVec is a spare matrix that could easily be used in the classification algorithm.

3.2.2 Sentiment Pipeline. The sentiment pipeline is a feature engineering pipeline that takes the text content and implements Natural Language Processing (NLP) to calculate polarity and subjectivity of the text. The polarity of the text indicates to the classifying algorithm the degree of positive or negative emotional sentiments in the text. Fake news have a strong positive or negative sentiment of hate, anger or resentment [21], and, in my model, the polarity of the text captures these cues. Likewise, the subjectivity of the text conveys the use of personal tones or expressions in the text, and these cues can help in bias detection in my model. Python's NLP library, Textblob, is used to calculate the polarity and subjectivity of the content. The pipeline also normalizes the new features using the Standard Z-score normalization for better performance and results of the classification algorithm.

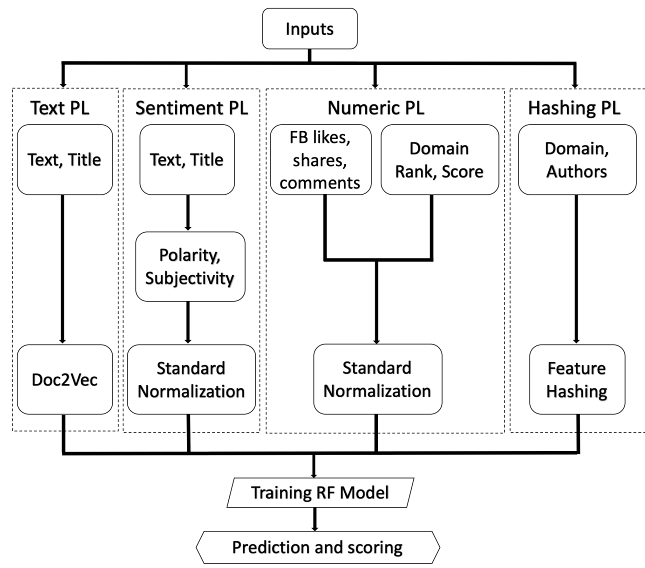

Figure 2: Machine Learning Pipeline.

3.2.3 Numeric Pipeline. The numeric pipeline takes the numeric data - namely domain rank and score, Facebook likes, shares and comments - and normalizes it using the standard Z-score normalization. I normalized numeric features because features in most machine learning algorithms perform better with normalized data.

3.2.4 Hashing Pipeline. The hashing pipeline takes as input the domain and authors of the news source and implements feature hashing to uniquely map the respective features into a unique feature hash vector. This pipeline implements feature extraction to convert domain and authors in a vector form that can be passed on to the machine learning classifier for training.

The sub-pipelines generate a feature set that trains the machine learning algorithm. The features extracted from the sub-pipeline are aggregated and passed on for training the random forest classifier.

3.3 Random Forest Implementation

Random Forest performs well on classification problems [3]. This algorithm was chosen for four main reasons. First, the intuition of traversing a set of questions using decision trees makes more sense given the numerical and categorical feature set. For example, if the domain score and Facebook popularity metrics are meager, then it is a good indicator that the news might not be as reliable. A similar comparison of the word vector will help in finding a pattern in fake news. Second, the random forest also supports different feature types including binary, categorical, numerical and especially spare matrix which is used to represent the word vector. Third, since random forest uses a collection of decision trees that are trained on a subset of the dataset, it is very rare for the model to overfit. Overfitting is a problem that is difficult to detect and fix, and any option to limit overfitting is a step toward building a better classifier.

Finally, random forest works well on a large data set, and as the corpus expands, this is a suitable algorithm for the task.

It is important to note that the random forest algorithm, like any other ensemble algorithm, takes more time to train than prominent algorithms such as Logistic Regression and Decision Trees. However, this issue can be tackled by using more workers in a parallel and distributed system environment.

4 EVALUATION AND RESULTS

The experiments were conducted on the Earlham College Cluster Node, Pollock. Pollock has AMD Opteron(tm) Processor and 132GB memory. Two datasets, the FakeNewsCorpus[24] and Getting Real about Fake News (GRFN) [17], were used for evaluation of the machine learning model. The FakeNewsCorpus dataset contains more than 9 million articles with nine labeled categories indicating the news as reliable or fake (i.e., fake, satire, reliable, unreliable, etc.). The Getting Real about Fake News dataset contains more than 13,000 fake news articles. Since the FakeNewsCorpus had a large dataset with a variety of news sources with both reliable and fake news, it was chosen as the primary training dataset and the GRFN was used as a testing set for one of the two tests.

4.1 Data Preprocessing

The Random Forest algorithm was implemented using Sci-kit Learn [15], an open-source machine learning library for Python. Since Scikit-learn has memory and performance limitations with the large dataset, PySpark was used to randomly select a subset of both reliable and fake news from the FakeNewsCorpus to create a smaller dataset (referred as the modified FakeNewsCorpus) containing 200,000 entries of labeled news sources. The articles from FakeNewsCorpus were randomly selected because the distribution of the dataset was unknown and random selection minimized the chances of having a biased dataset.

Since the features in the dataset are unique for every article, the entries with missing features or articles were removed from both the FakeNewsCorpus and GRFN dataset. The Facebook Graphs API was used to integrate the likes, comments, and share counts data to the dataset. The domain related metadata were accumulated from the Open PageRank.

After the data pre-processing stage, the modified FakeNews-Corpus had 192,926 article entries and GRFN had 12,345 article entries.

4.2 Results

Two tests were conducted for evaluating the proposed model. F1-score, accuracy, precision and recall were used as the metrics for assessing the model performance. Precision is a good measure to determine the costs of False Positive (reliable news that was tagged as false news) while recall is a better measure for the cost of False Negatives (fake news that was not tagged as fake)[20]. F1-score is a metric that tries to balance precision and recall and is better for evaluation of the model.

For the first test, the modified FakeNewsCorpus was divided as 70% training and 30% testing set. Using this dataset, the model achieved an accuracy of 99.9% and F1-score of 99.99%. The run-time for this trial was 2 hours and 32 minutes. The classification report is summarized in Table 2.

Table 2: Classification Report for FakeNewsCorpus 70-30 test-training

	precision	recall	f1-score	support
0 (reliable)	1.00	1.00	1.00	29874
1 (fake)	1.00	1.00	1.00	28004
avg/total	1.00	1.00	1.00	57878

Given that the model achieved an F1-score of 99.99%, I was concerned about over-fitting issues. Although, Random Forest usually does not over-fit, a second test was conducted using the GRFN dataset to see how the model performs on a different dataset. Since GRFN only contained fake news articles, I added 5,000 reliable news articles from the FakeNewsCorpus that were not used in training the algorithm. For this evaluation, the model was trained on the modified FakeNewsCorpus and tested with the GRFN dataset. The model achieved an accuracy of 87.9% and F1-score of 87.90% as summarized in Table 3. The run-time for this trial was 2 hours and 26 minutes.

Table 3: Classification Report - training with FakeNewsCorpus and testing on GRFN

	precision	recall	f1-score	support
0 (reliable)	0.70	1.00	0.83	5000
1 (fake)	1.00	0.83	0.91	12345
avg/total	0.91	0.88	0.88	17345

The results of the experiment with the GFRN dataset shows that the proposed model is not as accurate as shown in the first experiment. The F1-score decreased to 88.41%, indicating a overfitting problem. However, an F1-score of 88.41% is still a decent score. The precision of 91% reveals that 91% of the articles classified as fake were indeed fake. Similarly, the recall of 88% reveals that the model correctly classified 88% of fake article as fake. In the evaluation, recall is an important metric because it specifies the percentage of fake news articles detected by the model.

Overall, the results of the evaluation proves that the proposed model for fake news detection can predict fake news with reliable accuracy.

5 CONCLUSION AND FUTURE WORKS

Fake news detection is an interdisciplinary problem that is challenging to both the computer scientist and the linguists. One of the reasons for the hurdles in this problem space is the low inter-rater reliability (the general agreement among people) for the scope of fake news. Therefore, every fake news detection model is susceptible to its training dataset for the definition of fake news. The proposed hybrid model using linguistic cues and network metadata is no different. However, the dataset used in training the model contains news sources from various fields and sources to accommodate a broader definition of fake news. The proposed model reached a high accuracy rate (91% precision, 88% recall, 88% F-score) and can reliably predict the integrity of unseen news sources. The web user interface presents the applicability of the model in limiting the rapid spread of unreliable content on the Internet.

The project can be extended in the future to include fact-checking and deep syntax analysis, as well as recommending similar credible articles. Fact-checking can be implemented using knowledge networks such as DBpedia ontology. Similarly, the Google Relation Extraction Corpus can also be used to verify factual statements. In syntax analysis, Probability Context-Free Grammars can be used to create a recursive parse tree for deception detection[8]. Finally, similar credible articles recommendation can be implemented by computing the cosine distance between the document vector representations. In the future version of the project, fact-checking will be the most important addition for fake news detection as it will improve the assessment of factual content in the news.

ACKNOWLEDGMENTS

This project was supported by the Department of Computer Science at Earlham College. The author would like to extend sincere gratitude to Dr. Xunfei Jiang, Dr. Charles Peck, Dr. David Barbella and Dr. Ajit Chavan for their guidance in the design and evaluation of the project.

REFERENCES

[1] Hadeer Ahmed, Issa Traore, and Sherif Saad. 2017. Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. In *International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*. Springer, 127–138.

[2] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[3] Leo Breiman and Adele Cutler. 2018. https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm

[4] Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. 2010. Who is tweeting on Twitter: human, bot, or cyborg?. In *Proceedings of the 26th annual computer security applications conference*. ACM, 21–30.

[5] Giovanni Luca Ciampaglia, Prashant Shiralkar, Luis M Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. 2015. Computational fact checking from knowledge networks. *PloS one* 10, 6 (2015), e0128193.

[6] Niall J Conroy, Victoria L Rubin, and Yimin Chen. 2015. Automatic deception detection: Methods for finding fake news. In *Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community*. American Society for Information Science, 82.

[7] Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, Vol. 6. Genoa Italy, 449–454.

[8] Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. Association for Computational Linguistics, 171–175.

[9] Vanessa Wei Feng and Graeme Hirst. 2013. Detecting deceptive opinions with profile compatibility. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. 338–346.

[10] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their applications* 13, 4 (1998), 18–28.

[11] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*. 1188–1196.

[12] David M Markowitz and Jeffrey T Hancock. 2014. Linguistic traces of a scientific fraud: The case of Diederik Stapel. *PloS one* 9, 8 (2014), e105937.

[13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[14] Shereen Oraby, Lena Reed, Ryan Compton, Ellen Riloff, Marilyn Walker, and Steve Whittaker. 2017. And that's a fact: Distinguishing factual and emotional argumentation in online dialogue. *arXiv preprint arXiv:1709.05295* (2017).

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[16] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[17] Megan Risdal. 2018. Getting Real about Fake News. https://www.kaggle.com/mrisdal/fake-news

[18] Natali Ruchansky, Sungyong Seo, and Yan Liu. 2017. CSI: A Hybrid Deep Model for Fake News. *arXiv preprint arXiv:1703.06959* (2017).

[19] Katherine Schulten. 1443797475. Skills and Strategies | Fake News vs. Real News: Determining the Reliability of Sources. //learning.blogs.nytimes.com/2015/10/02/skills-and-strategies-fake-news-vs-real-news-determining-the-reliability-of-sources/

[20] Koo Ping Shung. 2018. Accuracy, Precision, Recall or F1? https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

[21] DSKR Vivek Singh, Rupanjal Dasgupta, and I Ghosh. 2017. Automated fake news detection using linguistic analysis and machine learning. In *International Conference on Social Computing, Behavioral-Cultural Modeling, & Prediction and Behavior Representation in Modeling and Simulation (SBP-BRiMS)*. 1–3.

[22] Sneha Singhania, Nigel Fernandez, and Shrisha Rao. 2017. 3HAN: A Deep Neural Network for Fake News Detection. In *International Conference on Neural Information Processing*. Springer, 572–581.

[23] Danny Sullivan. 2016. Introduction to Bayesian Statistics. Retrieved November 2, 2018 from https://searchengineland.com/rip-google-pagerank-retrospective-244286

[24] Maciej Szpakowski. 2018. FakeNewsCorpus: A dataset of millions of news articles scraped from a curated list of data sources. https://github.com/several27/FakeNewsCorpus

[25] TruthorFiction. 2018. Cadbury Chocolate Eggs Are Infected With HIV-Positive Blood-Fiction. https://www.truthorfiction.com/cadbury-products-infected-hiv-blood/

[26] Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 384–394.

[27] Sida Wang and Christopher D Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. Association for Computational Linguistics, 90–94.

[28] Will. 2015. Beyond One-Hot: an exploration of categorical variables. http://www.willmcginnis.com/2015/11/29/beyond-one-hot-an-exploration-of-categorical-variables/

[29] Hu Zhang, Zhuohua Fan, Jia-heng Zheng, and Quanming Liu. 2012. An Improving Deception Detection Method in Computer-Mediated Communication. *JNW* 7, 11 (2012), 1811–1816.