Fake News Detection Using Trained Datasets

Srinivasachar G[1], Shreeram G Hegde[2], Ajit Narayana Naik[3], Santosh R[4]
[1]Assistance Professor
[1, 2, 3, 4] Atria Institute of Technology, Visvesvaraya Technological, University, Bangalore

Abstract- *This paper explores the application of natural language processing techniques for the detection of 'fake news', that is, misleading news stories that come from non-reputable sources. Using a dataset obtained from Signal Media and a list of sources from OpenSources.co, we apply term frequency-inverse document frequency (TF-IDF) of bi-grams and probabilistic con- text free grammar (PCFG) detection to a corpus of about 11,000 articles. We test our dataset on multiple classification algorithms - Support Vector Machines, Stochastic Gradient Descent, Gradient Boosting, Bounded Decision Trees, and Random Forests. We find that TF-IDF of bi-grams fed into a Stochastic Gradient Descent model identifies non-credible sources with an accuracy of 77.2%, with PCFGs having slight effects on recall.*

Keywords- Natural language processing, Machine learning, Classification algorithms, Fake-news detection.

I. INTRODUCTION

In 2016, the prominence of disinformation within American political discourse was the subject of substantial attention, particularly following the election of President Trump [1]. The term 'fake news' became common parlance for the issue, particularly to describe factually incorrect and misleading articles published mostly for the purpose of making money through page views. In this paper, we seek to produce a model that can accurately predict the likelihood that a given article is fake news.

Facebook has been at the epicenter of much critique following media attention. They have already implemented a feature for users to flag fake news on the site [2]; however, it is clear from their public announcements that they are actively researching their ability to distinguish these articles in an automated way. Indeed, it is not an easy task. A given algorithm must be politically unbiased – since fake news exists on both ends of the spectrum – and also give equal balance to legitimate news sources on either end of the spectrum. In addition, the question of legitimacy is a difficult one. We need to determine what makes a new site 'legitimate' and a method to determine this in an objective manner.

In this paper, we compare the performance of models using three distinct feature sets to understand what factors are most predictive of fake news: TF-IDF using bi-gram frequency, syn- tactical structure frequency (probabilistic context free gram- mars, or PCFGs), and a combined feature union. In doing so, we follow the existing literature on deception detection through natural language processing (NLP), particularly the work of Feng, Banerjee, and Choi [3] with deceptive social media reviews. We find that while bi-gram TF-IDF yields predictive models that are highly effective at classifying articles from unreliable sources, the PCFG features do little to add to the models' efficacy. Instead, our findings suggest that, contrary to the work done in [3], PCFGs do not provide meaningful variation for this particular classification task. This suggests important differences between deceptive reviews and so-called 'fake news'. We then suggest additional routes for work and analysis moving forward.

Section II briefly describes the past work done in the field of text classification and fake news detection. Section III describes the dataset used for training the classifier. Section IV illustrates the feature generation methodology and pre- processing steps. Section V delineates the actual modelling procedure and compares the outputs from the different al- gorithms. Finally, Section VI presents the conclusions and briefly illustrates the potential for further improvements in the proposed methodology.

II. RELATED WORKS

There exists a sizeable body of research on the topic of machine learning methods for deception detection, most of which have been focused on classifying online reviews and publicly available social media posts. Particularly since late 2016 during the American Presidential election, the question of determining 'fake news' has also been the subject of particular attention within the literature.

Conroy, Rubin, and Chen [4] outline several approaches that seem promising toward the aim of correctly classifying misleading articles. They note that simple content-related n- grams and shallow part-of-speech (POS) tagging have proven insufficient for the classification task, often failing to account for important context information. Rather, these methods have been shown useful only in tandem with more complex methods of analysis. Deep Syntax analysis using Probabilistic Context Free Grammars (PCFG) have been shown to be particularly valuable in combination with n-gram methods. Feng, Banerjee, and Choi [3] are able to achieve 85%-91% accuracy in decep- tion related classification tasks using online review corpora.

Feng and Hirst [5] implement a semantic analysis looking at 'object: descriptor' pairs for contradictions with the text on top of Feng's initial deep syntax model for additional improve- ment. Rubin, Lukoianova and Tatiana [6] analyze rhetorical structure using a vector space model with similar success. Ciampaglia et al. [7] employ language pattern similarity net- works requiring a pre-existing knowledge base.

TABLE I: Comparison of top unreliable and reliable sources by article frequency.

Top Five Unreliable News Sources		Top Five Reliable News Sources	
Before It's News	2066	Reuters	3898
Zero Hedge	149	BBC	830
Raw Story	90	USA Today	824
Washington Examiner	79	Washington Post	820
Infowars	67	CNN	595

III. DATA PREPARATION

A. Dataset Description

Conroy, Rubin and Chen [4] outline several requirements for a helpful corpus for use in these contexts (shortened for relevance):

1. Availability of both truthful and deceptive instances.
2. Verifiability of 'ground truth'.
3. Homogeneity in lengths.
4. Homogeneity in writing matter.
5. Predefined timeframe.
6. The manner of delivery (e.g. sensational, newsworthy).

To deal with some of these challenges, we outsource some of corpus definitions to the website OpenSources.co [8] which compiles an ongoing list of fake and trusted news sources. It is by no means perfect and has some detractors, as any list like this might. Ultimately, our modeling approach should be data source independent and capable of using a better corpus or corpora when they are available.

Obtaining a corpus of news articles is notoriously difficult due to copyright issues. We found a dataset published by Signal Media in conjunction with the Recent Trends in News Information Retrieval 2016 conference to facilitate conducting research on news articles [9]. The dataset contains about 1 million articles from a variety of news sources from September 2015. Sources include major news outlets like Reuters [10] as well as local news sources and blogs. From this dataset, we filter to include articles from verified reliable sources (labeled as 0) and verified unreliable sources (labeled as 1).

Our cleaned dataset contains 11051 articles. 3217 (29%) are labeled as fake. The reliable articles come from 14 unique sources. The unreliable articles come from 61 unique sources. In particular, for fake news our examples are heavily drawn from one source: Before It's News.

B. Resampling to Account for Skewed Distributions

In order to limit the extent to which our models will primarily learn the difference between 'Before It's News' and Reuters [10], we force the distribution to cover a more limited range by randomly re-sampling the largest source contributors for a smaller n.

We choose *n-max* of 500 articles for our implementation as it seemed prudent, though it is non-empirically based. We also do not drop low frequency sources in the interest of maintaining some heterogeneity of sources. The correct *n-max* (or a potential n-min) is an interesting research question in its own right. Additional avenues of research may consider varying this number to achieve an optimal result if facing similar corpus difficulties. We notice that before and after

TABLE II: Naive and random model performance across metrics.

Model	Accuracy	Precision	Recall
Naive	67.89%	54.22%	54.22%
Random	56.42%	32.18%	32.18%

our re-sampling of the distributions we see a slight drop in precision, indicating that we did fit to particular sources rather than the classes themselves with a more skewed distribution of sources.

IV. FEATURE GENERATION

Our approach evaluates the performance of models trained on three feature sets:

1. Bigram Term Frequency-Inverse Document Frequency.
2. Normalized frequency of parsed syntactical dependencies.
3. A union of (1) and (2).

For feature generation, we rely on the Spacy Python package [11] to conduct tokenization, part-of-speech tagging, syntactical parsing, and named entity recognition. Spacy [11] is implemented in Cython (a superset of the Python language that allows C code for be generated from Python using the Python/C API) [12], allowing for very fast performance compared to other NLP packages such as NLTK [13].

Several evaluations from peer-reviewed journals find that Spacy [11] achieves performance on parsing and entity recog- nition tasks that is comparable to other widely-used tools, while having a significant advantage with respect to speed [14]. This is why we chose to use Spacy [11] over more established options such as the Java implementation of Stanford's Proba- bilistic Context Free Grammar. [15]

From the raw article text, we use Spacy [11] and SciKit Learn [16] [17] to generate the relevant features. We utilize Spacy's [11] support for multi-threading to parallelize the feature generation process and SciKit Learn's Pipeline feature[17]to create fit-transform and transform methods that can be used on the training data and then applied to the test set.

A. Preprocessing

We scrub the articles of any mention of the name of the source. Because the reliable/unreliable classification is determined at the source level, this step is necessary to ensure the model does not just learn the mappings from known sources to labels. We also strip Twitter handles and email addresses (which often show up in journalist biographies) for the same reason.

B. Term Frequency-Inverse Document Frequency

The first feature set is vectorized bigram Term Frequency- Inverse Document Frequency. This is a weighted measure of how often a particular bigram phrase occurs in a document relative to how often the bigram phrase occurs across all documents in a corpus.

Because of the political nature of our corpus, we want to limit the model's knowledge of the people and institutions

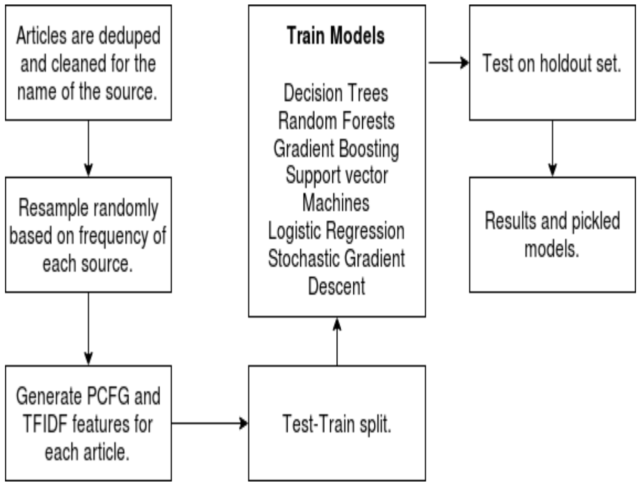
Articles are deduped and cleaned for the name of the source.
Resample randomly based on frequency of each source.
Generate PCFG and TFIDF features for each article.
Test-Train split.
Train Models
Decision Trees
Random Forests
Gradient Boosting
Support vector Machines
Logistic Regression
Stochastic Gradient Descent
Test on holdout set.
Results and pickled models.

Fig. 1: Pipeline representation.

TABLE III: Average model performance with both PCFG and TF-IDF bi-gram features at 0.7 score threshold for categorization.

Model	Area Under Curve	Precision	Recall	Accuracy
Bounded Decision Trees	65.9%	66.9%	37.9%	67.6%
Gradient Boosting	75.6%	40.2%	16.1%	65.7%
Random Forests	80.0%	84.2%	18.4%	64.8%
Stochastic Gradient Descent	87.5%	74.1%	71.7%	65.7%
Support Vector Machine	84.3%	80.9%	44.5%	73.6%
Baseline	-	32.18%	32.18%	67.89%

TABLE IV: Average model performance with only TF-IDF bi-gram features at 0.7 score threshold for categorization.

Model	Area Under Curve	Precision	Recall	Accuracy
Bounded Decision Trees	60.7%	58.5%	23.3%	66.1%
Gradient Boosting	79.4%	41.0%	22.3%	68.7%
Random Forests	78.8%	82.9%	25.3%	67.6%
Stochastic Gradient Descent	88.3%	88.8%	45.3%	77.2%
Support Vector Machine	85.6%	81.3%	48.1%	76.2%
Baseline	-	32.18%	32.18%	67.89%

TABLE V: Average model performance with only PCFG features, classifying the top 5% of scores as positive (k = 0.05).

Model	Area Under Curve	Precision	Recall	Accuracy
Bounded Decision Trees	50.0%	40.9%	10.8%	60.1%
Gradient Boosting	50.0%	40.9%	10.8%	60.1%
Random Forests	50.0%	40.9%	10.8%	60.1%
Stochastic Gradient Descent	50.0%	40.9%	10.8%	60.1%
Support Vector Machine	50.0%	40.9%	10.8%	60.1%
Baseline	-	32.18%	32.18%	67.89%

mentioned in the article text. Otherwise, we risk the model sim- ply learning patterns such as 'Clinton corrupt' which describe the topic and viewpoint of the text, rather than the outcome of interest (is this source reliable or not). Additionally, these patterns will be highly sensitive to the particular news cycle. To address this concern, we introduce a step during tokenization to use Spacy's [11] named entity recognition to replace all mentions of named entities with a placeholder, e.g. <-NAME-> or <-ORG->.

We use SKLearn [16] to calculate the TF-IDF for each bigram within each document and build a sparse matrix of the resulting features. To keep the dimensionality of our data to a manageable size, we limit the vocabulary to only consider the top 3000 terms ordered by term frequency across the entire corpus. We did not experiment with different methods or thresholds for selecting the terms included in the vocabulary, or with different lengths of n-grams, but this may be an area to explore in future work.

C. Normalized Syntactical Dependency Frequency

We use Spacy [11][18] to tokenize and parse syntactical de- pendencies of each document. Spacy's algorithm is a transition- based, greedy, dynamic oracle using Brown clusters [19] [20] that is comparable in accuracy to Stanford's PCFG [15] but dramatically faster and more lightweight [14].

Each token is tagged with one of 46 possible syntactic dependency relations, such as 'noun subject' or 'preposition'. We count the frequency of occurrences of each dependency tag and normalize by the total number of dependencies in the document. Again, we use SKLearn [16] to convert these frequencies into sparse matrices suitable for training models.

In total, we obtain 3000 features in TF-IDF family and 46 in the grammar family. Meaningful feature names are not currently available, limiting our ability to evaluate what specific characteristics of a document appear to be predictive of its legitimacy. This would allow us to better determine if the classifier is learning topical patterns or the outcome of interest and allow for a more thorough assessment of the generalization potential of our results.

V. MODELING AND EVALUATION

A. Our Pipeline

After cleaning the data and generating features, we execute a 90/10 random test-train split on the dataset and feed it into a modeling pipeline. This pipeline iteratively fits models varying the tuning parameters with which they are executed up to 50 times, depending on the number of possible permutations for that model. These models are then tested on the 10% holdout data to understand their performance.

B. Baseline Models for Comparison: Naive and Random

As a baseline comparison for understanding the perfor- mance of our models, we look at two methods. First, a Naive Bayes model that predicts all majority class; in this case, all articles are from reliable news sources. Second, a model that randomly selects a classification for each article as either reliable or unreliable based on the posterior probability of that class in the training set. These are the Naive and Random models, respectively. We detail their performance in Table II.

C. Combining PCFG and TF-IDF bi-gram features

Combining both feature sets, our models perform well above our baseline as seen in Table III.

We note that our best models tended to be Stochastic Gradient Descent (SGD) models, which, given that they tend to perform well with sparse and highly dimensional data, is not surprising. In particular, SGDs far outperform on precision *while retaining a high recall*, meaning that these models would work well both as identification of high priority articles in addition as 'fake news' filters.

D. TF-IDF Bigram Only Model Performance

Removing the PCFG features allows us to understand in more depth the value of those features in achieving these combined feature results. The results from this more limited feature run are displayed in Table IV.

The removal of PCFGs *improves* most of the metrics across our models. This is surprising, indicating that the PCFG features add little predictive value to the models. Indeed, the only noticeable decrease in performance is in our recall figures for Decision Trees and SGDs.

E. PCFG Only Model Performance

The removal of TF-IDF bi-gram features allows us to isolate the predictive value of PCFGs for our application. The results are displayed in Table V.

Surprisingly, all of our models give the same result. Diving into the individual predictions, we find that all models produce the same rank order of scores. We've switched from a 0.70 threshold for classification to a top-k of 0.05 because the distribution of scores for these models have a particularly low mean with a tight range, such that determining an appropriate threshold for categorization was tedious and the 0.70 results weren't illuminating.

All this goes to indicate that in the case of this classification task, PCFGs do not add a strong source of information for classification on their own.

VI. CONCLUSION AND FUTURE SCOPE

The results obtained above are very promising. This method demonstrates that term frequency is potentially predictive of fake news - an important first step toward using machine classification for identification. The best performing models by overall ROC AUC are Stochastic Gradient Descent models trained on the TF-IDF feature set only. We observe that PCFGs do not add much predictive value, but balance the Recall for our top performing model. This indicates that PCFGs are good for a Fake-News Filter type implementation versus, say, targeting fake news sites for review. TF-IDF shows promising potential predictive power, even when ignoring named entities, but we remain skeptical that this approach would be robust to changing news cycles. However, this would require a more complete corpus.

Despite the high performance of our classifier, there is definitely scope for improvement. We evaluated our models using absolute probability thresholds, which may not be the most reliable for models where probability scoring is not well-calibrated. While TF-IDF performs better, we are possibly overfitting to topics/terms important in the ongoing news cycle. Also, a vectorized approach like ours makes it technically hard to see which individual features are most important, thus hampering our analysis. These issues limit our analysis and thus prevent broader generalizability. We plan to address these issues in a future work.

REFERENCES

[1] S. Maheshwari, *How fake news goes viral: A case study*, Nov. 2016. [Online]. Available: https://www.nytimes. com / 2016 / 11 / 20 / business / media / how - fake - news - spreads.html (visited on 11/08/2017).

[2] Mosseri, *News feed fyi: Addressing hoaxes and fake news*, Dec. 2016. [Online]. Available: https://newsroom. fb . com / news / 2016 / 12 / news - feed - fyi - addressing - hoaxes-and-fake-news/ (visited on 11/08/2017).

[3] S. Feng, R. Banerjee, and Y. Choi, "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, 2012, pp. 171–175.

[4] N. J. Conroy, V. L. Rubin, and Y. Chen, "Automatic deception detection: Methods for finding fake news," *Proceedings of the Association for Information Science and Technology*, vol. 52, no. 1, pp. 1–4, 2015.

[5] V. W. Feng and G. Hirst, "Detecting deceptive opinions with profile compatibility.," in *IJCNLP*, 2013, pp. 338–346.

[6] V. L. Rubin and T. Lukoianova, "Truth and deception at the rhetorical structure level," *Journal of the Association for Information Science and Technology*, vol. 66, no. 5, pp. 905–917, 2015.

[7] L. Ciampaglia, P. Shiralkar, L. M. Rocha, J. Bollen,F. Menczer, and A. Flammini, "Computational fact checking from knowledge networks," *PloS one*, vol. 10, no. 6, e0128193, 2015.

[8] *Opensources*. [Online]. Available: http : / / www . opensources.co/ (visited on 11/08/2017).

[9] Corney, D. Albakour, M. Martinez, and S. Moussa, "What do a million news articles look like?" In *Proceedings of the First International Workshop on Recent Trends in News Information Retrieval co-located with 38th European Conference on Information Retrieval (ECIR 2016), Padua, Italy, March 20, 2016.*, 2016, pp. 42–47. [Online]. Available: http://ceur-ws.org/Vol-1568/paper8.pdf.

[10] *Business financial news, u.s international breaking news*. [Online]. Available: http : / / www. reuters . com/ (visited on 11/08/2017).

[11] Explosion, *Spacy*, Sep. 2017. [Online]. Available: https: //github.com/explosion/spaCy (visited on 11/08/2017).

[12] S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D. Seljebotn, and K. Smith, "Cython: The best of both worlds," *Computing in Science Engineering*, vol. 13, no. 2, pp. 31–39, 2011, ISSN: 1521-9615. DOI: 10 . 1109 / MCSE.2010.118.

[13] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[14] J. D. Choi, J. R. Tetreault, and A. Stent, "It depends: Dependency parser comparison using a web-based evaluation tool.," in *ACL (1)*, 2015, pp. 387–396.

[15] Chen and C. Manning, "A fast and accurate depen- dency parser using neural networks," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 740–750.

[16] 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, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[17] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa,A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly,B. Holt, and G. Varoquaux, "API design for machine learning software: Experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108– 122.

[18] M. Honnibal and M. Johnson, “An improved non-monotonic transition system for dependency parsing,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 1373–1378. [Online]. Available: https://aclweb.org/anthology/D/D15/D15-1162.

[19] M. Collins, “Discriminative training methods for hid- den markov models: Theory and experiments with perceptron algorithms,” in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, Association for Computational Linguistics, 2002, pp. 1–8.

[20] T. Koo, X. Carreras Pérez, and M. Collins, “Simple semi-supervised dependency parsing,” in *46th Annual Meeting of the Association for Computational Linguis- tics*, 2008, pp. 595–603.