# Fake News Detection in Social Media

KUMARAN GUNASEKARAN, kugunase@eng.ucsd.edu,A53207045
GOWTHAM GANESAN, gganesan@eng.ucsd.edu,A53203033
SUDARSHAN SRINIVASA RAMANUJAM, sus046@eng.ucsd.edu,A53220043
BALASUBRAMANIAM SRINIVASAN, bsriniva@eng.ucsd.edu, A53210041

**ABSTRACT**

Owing to the growing popularity of social media, spammers have made these their prime target to mislead others with an intent to make financial or political gains. In this work, we have developed a fake news detection system which allows users to filter and flag potentially deceptive information. The prediction that a particular news article is deceptive is based on the analysis of previously seen news as well as information available online from reliable sources. A scarcity of deceptive news for predictive modeling, is a major stumbling block in this field. Our model using a Random Forest Classifier achieves an accuracy of 94.87% using a combination of features such as n-grams, shallow syntax, tf-idf and a novel online relevance score feature. For the same model we achieve a BER score of 0.115. These promising results encourages us to probe further into this issue to eliminate fraud through fake news.

**Keywords** Fake news detection, n-grams, tf-idf, shallow syntax, Online Relevance Score, Jaccard Similarity, Classification, SVM, Random Forests, Balanced Error Ratio

## 1 INTRODUCTION

Fake news has been a troublesome issue leading to millions of people being misinformed all over the world. Especially in this day and age of excessive reliance on social media, the spread of fake news has been much faster and wider. Recently there has been plenty of debate in the political arena as well, regarding which news are real and which ones are not. In this work, we explore detection of fake news based solely on the content of the news and not on any of it's meta characteristics(such as time of the news, the author of the news,etc). All the relevant work in this area concentrate on identifying deceptive news mainly based on the type of language used. To truly identify fake news we have to give equal importance to the information being presented, which can be hard. One simple heuristic that we have considered in this project, to assess the veracity of the news, is to consider the number of search results returned by a Bing search query and a relevance score calculated based on the similarity between the news and the results obtained. This feature will ensure that our model is not naively making predictions based on the language features and rather takes into account the veracity of the content presented. This work assumes responsibility in social media such as Facebook, Whatsapp, Twitter where a lot of information is shared/forwarded/re-tweeted without much thought. This work therefore aims to prevent fake news from trending on social media and enables us to make informed decisions especially during high impact events.

In this work apart from the online data we have also used a combination of features related to the syntactic features of the text. Studies on similar literature were conducted and are elucidated in the section 2. In section 3, we explore the intrinsics of the dataset used, perform a thorough breakdown of the data and offer a better visualization of the text data. In section 4, the features and model used for classification are discussed. In section 5, the results observed are analyzed followed by conclusions and future work in section 6 and 7 respectively.

## 2 RELATED WORK

Interpersonal Psychology and Communication studies have shown that humans are generally not very successful in distinguishing deceptive news even when they are alerted to this possibility [2]. This necessitates the requirement of filtering and flagging dubious information by intelligent systems.

Myle Ott, et al. in their work [5] have explored fictitious opinions which are written to sway consumer judgment . One possible solution to detect biased opinions which is elucidated in [5] is to use the output of the Linguistic Inquiry and Word Count software tool as a feature.

Song Feng, et al. in their work [3], apart from using shallow lexico-syntactic features and words (such as uni-grams, bi-grams, n-grams) demonstrated the gain in fake review detection performance by drawing from features based on Context Free Grammars (CFG) parse trees. Towards this approach they had experimented using four different production rules, each having their own advantages.

Niall Conroy, et al. in their work [2] have elucidated on two approaches - one using linguistic cues and the other using network-based behavioral data. Deep Syntax (using PCFG), semantic analysis, as well as rhetorical structure analysis were used as linguistic cues whereas social network behavior and inherent linked structure derived from previous corpus contributed towards the network approaches.

Mathieu Cliche in his sarcasm detection blog has described the detection of sarcasm on twitter through the use of n-grams, words learnt from tweets specifically tagged as sarcastic. His work also includes the use of sentiment analysis as well as identification of topics (words that are often grouped together in tweets) to improve prediction accuracy.

In this work we have built upon the simple existing linguistic model (we haven't used the model also based on social network behavior due to it not being completely open source) and introduced the concept of an online relevance score.

## 3 DATA EXPLORATION AND VISUALIZATION

### 3.1 Data Collection using Web scraping

We have extracted data from the following different sources to get a diverse flavor of news data.

(1) Twitter Feed (Fake and Real)
(2) BBC Database
(3) UCI News Dataset
(4) Kaggle fake news dataset

BBC's news dataset was used off the shelf. Kaggle's fake news dataset had a list of links to fake news articles in a CSV files. We used these links to scrap the fake news data using Beautiful Soup. Additionally we subscribed to Twitter REST Api and extracted the tweets from various twitter handles. For example we extracted tweets from '@TheOnion', '@fakingnews', '@ClickHole', '#rumour' etc. for fake news and from channels like '@CNBC', '@washingtonpost', '@HuffingtonPost',etc for genuine content. With all these extracted data we had close to 100,000 data points with 40% of those representative of fake news class. 30% of the collected data was saved for testing our models. This test data was solely extracted from the Twitter feeds mentioned above.

### 3.2 Data Preprocessing

The data obtained from these different sources were in different formats and organized in different ways. We preprocessed the data to bring uniformity in the data before they can be used to train any model. The data obtained from the twitter feed contained re-tweets, twitter handles and hyper-links. These had to be removed. Also tweets with supporting images were removed from the dataset since at this point of time we are not interested in identifying fake news from images or other media content. We also preprocessed the data obtained from non-twitter sources and trimmed the length,taking only an excerpt of the news. This was done so as to avoid skew in the content length, since tweets are much shorter in comparison to news articles. Also we removed tweets which where very small(less than 5 distinct words) as they do not provide much information to the prediction. Average length of data snippets used for the training was close to 120 characters.

### 3.3 Data Visualization

After the preprocessing phase, exploratory analysis of the data was done. We initially began by generating a word cloud of the data from fake and real news as shown in Figures 1 and 2. A quick glance through the images show that the fake news is mostly related to the 2016 US elections while the true news is not specific to any particular genre.

Fig. 1. Word cloud for true news

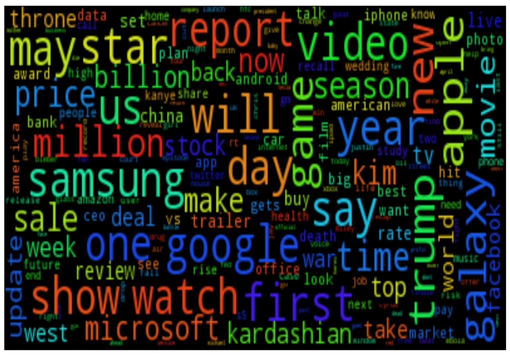
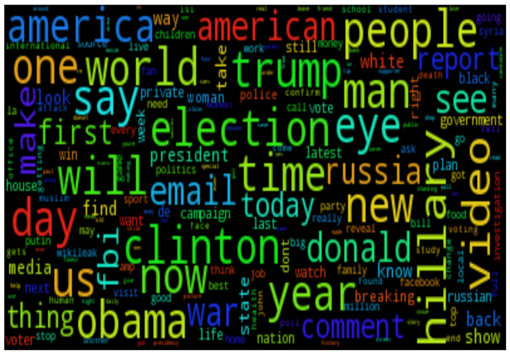


Fig. 2. Word cloud for fake news



As we stated in the previous section, most of the existing work on fake news detection concentrates on the syntactic and semantic features of the content. While in the real world we could have a fake news well disguised under the features of a real news. A trivial predictor would miss out on identifying such fake contents. So we have tried to use a new feature where we measure the current relevance of the given news snippet. We are doing this by finding the Jaccard similarity between the given news and the results from a Bing search. For this purpose, we scrapped data from Bing search query results using Beautiful Soup. We hypothesized that true news articles will have a higher similarity while the fake articles will have low similarity. As a proof of correctness we tested our hypothesis on a small set of 20 fake news snippets and 20 true news snippets and found that the results support our hypothesis as shown in Fig.3(here the blue bubbles are fake news and red bubbles are true news).

Fig. 3. Relevance scores for Fake and Real News

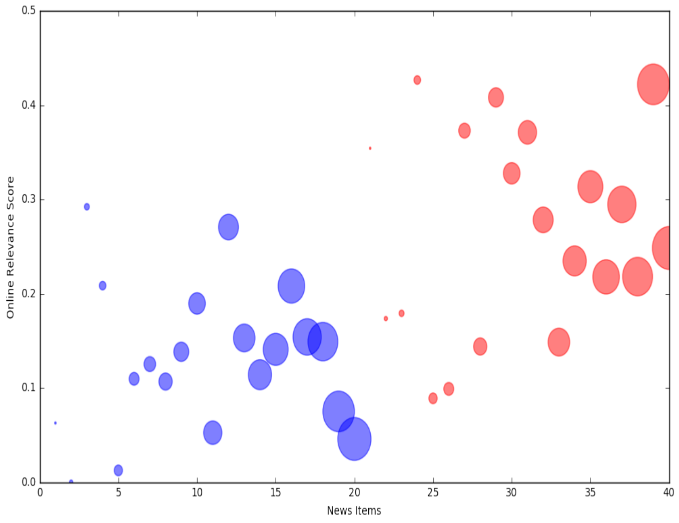


Fig. 4. Centering Resonance analysis for fake news

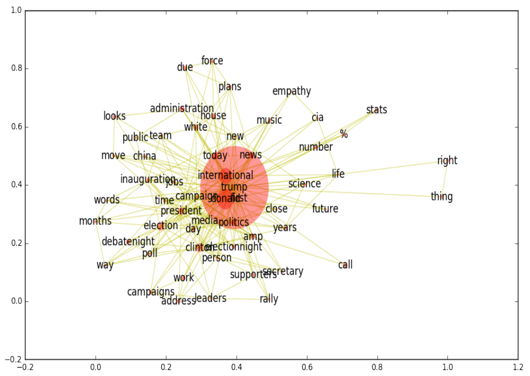
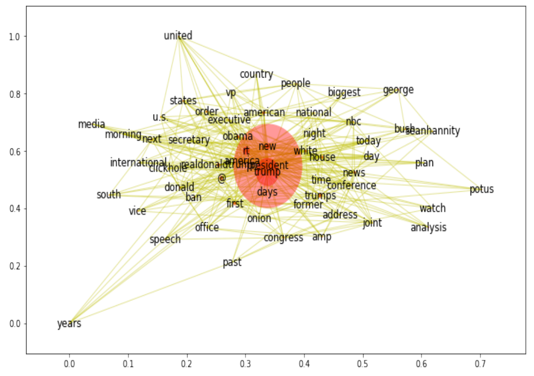


Fig. 5. Centering Resonance analysis for true news



Figures 4 and 5 show a networkX model of the fake and real news snippets. As we can see these graphs are densely connected which shows that a bi-gram model could work well in this classification task. These networks are generated using a technique called centering resonance analysis, a type of network based text analysis which represents the content of large data sets by identifying the most important words that link other words. In these two figures we have included only sentences with the word "President" to have close look at the edge distribution of the network. We also wanted to find the correlation between the sentiment of the news article and it's type. We used NLTK sentiment analyzer library to compute the sentiment scores for the news articles and got the results as shown in Table 1. As we can see, there isn't statistically significant difference between true and fake news in terms of the sentiment of the news.

| News Type | positive | negative | neutral |
|---|---|---|---|
| True News | 0.0543 | 0.0516 | 0.8792 |
| Fake News | 0.0664 | 0.0571 | 0.8710 |

Table 1. Sentiment analysis for fake and true news

## 4 CLASSIFICATION TASK

### 4.1 Task description

To predict whether a news article is fake, it is natural to consider the content of the news as an important factor. In this experiment, the data set consists of news taken from the twitter feed (140 characters maximum). For the data set scrapped from other sources, the headlines were alone taken. We aim to perform the classification based on Summarized news data in this project.

Since the training data comes from a variety of sources in the internet, the complexity of the classification problem is increased and there is a need for better generalization in classification. A portion of news data from twitter (real and fake) were set aside for testing purpose.

We explored a variety of models from the literature for building our feature set. Based on the data exploration explained in the previous section, we arrived at the following basic features:

(1) Unigrams (baseline model)
(2) Bigrams
(3) Trigrams
(4) Automated Readability Index (ARI)
(5) Syntactic information using POS
(6) TF-IDF scores
(7) Online Relevance Score(ORS) as described in eq. 8

### 4.2 Baseline

Our baseline model is to simply predict whether a news item is fake depending on the presence or absence of words, characteristic to it. For this, we use a bag of words (BoW) model with unigram probabilities computed for each word on the training set.

$$unigram\_prob_{w,e} = \begin{cases} \frac{|x_{w,e}|}{|n_w|} & \text{if word 'w' occurs in dataset} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $|x_{w,e}|$ is the count of occurrences of word $w$ in the $e_{th}$ example and $|n_w|$ is the count of all occurrence of the word $w$ in the training set.

The top highly probable unigrams in the training set for the real and fake news is listed in Fig.1 and Fig.2 respectively.

## 5 MODEL DESCRIPTION

In this section, we illustrate the proposed model for fake news classification task. For the task, we consider two important classes of features: textual features and credibility of the news article using Online Relevance Score (ORS).

### MODEL

### 5.1 SVM classifier

A SVM classifier with RBF kernel was applied to this problem for the classification task. The general formula for a soft SVM is listed in the following 3 equations.

$$PRIMALmin_{w \epsilon R^p, b \epsilon R, \xi \epsilon R^n} \frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \xi y^{(i)} \left( w.x^{(i)} + b \right) \geq 1 - \xi_i // x \tag{2}$$

$$Constraint1 - y^{(i)}\left(w.x^{(i)} + b\right) \geq 1 - \xi_i \quad (3)$$

$$Constraint2 - \xi > 0 \quad (4)$$

The slack variable for soft SVM (C) was set at 100 in order to improve generalization of the model. The gamma parameter of the RBF kernel was set to a relatively low value (0.001) as our data set is one with high variance. Intuitively we expect the SVM to have a high spread influence.

The kernel trick (dot product in the dual of the SVM) enables us to exploit the RBF kernel which is capable of building the classifier in an infinite dimensional space.

## 5.2 Random Forest Classifier

Random forest was tried as a classifier for the same problem. It yielded a higher accuracy value value when all features were used. The number of classifiers to combine was set to 1000 and the rest of the parameters were set to SK learn default.

## FEATURES

## 5.3 N-gram probabilities

The baseline model described in section 4.3 only takes into consideration, the presence/absence of individual words. However, the complete context of the words are ignored. To improve this, we used bigram model determining a set of unique pairs of words and their probabilities. Just like in our baseline model, the text was processed in lowercase, punctuations were ignored, English stopwords were removed while building the bigram model to keep a check on the final dimensionality. From Ref. [1], it is clear that using bigram produces stark distinctions in the data compared to unigrams. On experimentation, we observe that using a combination of bigram and unigram on SVM classifier leads to better performance (in terms of accuracy) as compared to using only bigrams.

To gather more context, we experiment by extending our model to include trigrams with the combination of unigrams and bigrams.

## 5.4 Using TF-IDF

To distinguish the features of fake and real news we use TF-IDF weighting model, typically used to describe documents in Vector Space Model (VSM). Based on the Ref. [7], we use TF-IDF to identify the words that discriminate between the two classes of news articles. TF-IDF distinguishes the fake and real news by distinguishing word frequency across various news articles. For instance, common words across news articles are filtered out and the words which help to discriminate between classes are given more weight.

$$weight_{w,e} = log(tfidf_{w,e} + 1) * log(\frac{n}{x_w}) \quad (5)$$

where $tfidf_{w,e}$ is the tf-idf value of the word $w$ in the $e$ example in training set, $n$ is the number of examples, $x_w$ is the number of examples where the word $w$ occurs.

## 5.5 Automated Readability Index(ARI)

From our observations on the training set, the fake news items, especially satirical articles were hard to understand because of the language complexity. This motivates us to use Automated Readability Index(ARI) as one of the features in building the model. ARI is defined as,

$$ARI = 4.71 * (\frac{characters_e}{words_e}) + 0.5 * (\frac{words_e}{sentences_e}) - 21.43 \quad (6)$$

where $characters_e$ is the number of characters, $words_e$ is the number of words, $sentences_e$ is the number of sentences in the $e^{th}$ example.

## 5.6 Syntactic information using POS tag

To capture the syntactic information of the news content, we used Parts of Speech (POS) lexical features of the text. PyStat Parser (Ref. [4]) was used to parse the news content and generate corresponding POS tags. The probabilies of POS tags under each category was then used as a feature vector for the classification task.

$$F < POS_{e,c} >= \frac{|POS_{e,c}|}{|POS_c|} \quad (7)$$

where $F$ is the feature vector, $POS_{e,c}$ is the POS tag of $e_{th}$ example and $c_{th}$ POS category, $|POS_c|$ is the count of the POS category $c$ in the training set.

On experimentation with bigram POS tagging, we observe that the performance in terms of accuracy on the validation set drops down and results in a sparse matrix.

## 5.7 Online Relevance Score (ORS)

The description of search results of Bing search query for a specific news headline was scraped from the internet and stored. The Jaccard similarity measure between the query and the description of search results are taken for every news headline.

We define a term "ORS (Online Relevance Score)", which is equal to the average of the Jaccard Similarities between the input text and the news description of the top K links returned.

$$ORS\,(Q,k) = \frac{1}{k}\sum_{i=1}^{k} Jaccard(Q, D_i) \quad (8)$$

In the above equation $Q$ denotes the query (input) and $D_i$ denotes the description text from the top k links returned by the search engine API.

On close inspection, a distinct difference was observed between the similarity measures seen for fake news data and true news data. For a start the first 10 results which appeared in Bing search were taken as the measure to be compared against.

The first few results which appeared did not contribute significantly to classification (some relevant search results always appeared). Nevertheless, as we took text from higher values of k, we started seeing a significant difference between the scores for real and fake news.

## 6 RESULTS AND DISCUSSION

### 6.1 Test set

Different models were trained using SVM and Random Forest classifiers on a training set of close to 45k real news and 25k fake news items aggregated from different sources mentioned in section 3.1. To evaluate the performance of our classifier models, we used a test set of 30k news articles fetched from Twitter and the performance metrics (Balanced Error rate and Accuracy) were compared against the baseline model (uni-gram). The following sections compares in detail the results of our classifier against the baseline model.

### 6.2 Evaluation of classifier

To evaluate the classifiers, we use classification accuracy and Balanced Error Rate(BER) measure to quantify the proportion of misclassifications.

$$BER = 1 - \frac{1}{2}(TPR + TNR) \tag{9}$$

where TPR is the True Positive Rate and TNR is the True Negative Rate

### 6.3 Results on SVM classifier

The components discussed in section 5.3 were used as features for our SVM models in an incremental fashion. On varying the parameters of the SVM classifier with the RBF kernel, we observed that using low values of C resulted in huge number of mis-classifications on the validation set. We identified that setting the parameters as C=100 and $\gamma = 0.001$ resulted in the best performance.

When only the bi-gram or tri-gram probabilities were used as features, there was a loss in accuracy, in comparison with the baseline model. However, their unions with the uni-gram probabilities helped the model outperform the baseline model.

Using TF-IDF along with the union of an uni-gram and bi-gram features, resulted in an increase in accuracy. TFIDF helps reduce the impact of common words across news headlines and increases weight on the words which helps to distinguish between news headlines'.

We added the relevance of news articles as a feature based on their ORS score discussed in section 5.5. Fake articles tend to provide less ORS score because of their irrelevance in the search results and this was evident on our results as the combination of ORS score with the other features discussed above produced 7% improvement in performance from the baseline model.

| Results | Accuracies | BER |
|---|---|---|
| Unigram | 85.73 | 0.169 |
| Bigram | 81.62 | 0.197 |
| Trigram | 71.79 | 0.224 |
| Unigram+Bigram | 86.27 | 0.151 |
| Unigram+Bigram+Trigram | 86.84 | 0.149 |
| Words+pos tags+TFIDF | 89.23 | 0.141 |
| Words+pos tags+TFIDF+ORS | 93.36 | 0.134 |

Table 2. SVM Results

### 6.4 Results on Random Forest classifier

A Random Forest Classifier was built using uni-gram, bi-gram, TF-IDF and ORS features in an incremental fashion and we observed an increase in accuracies, in a pattern similar to the SVM Classifier. We recorded the lowest balanced error rate when the number of estimators was set to 1000. However, building a model with an increased number of estimators made the training process computationally hard.

| Results | Accuracies | BER |
|---|---|---|
| Unigram | 86.12 | 0.162 |
| Bigram | 81.39 | 0.196 |
| Trigram | 70.06 | 0.213 |
| Unigram+Bigram | 87.94 | 0.149 |
| Unigram+Bigram+Trigram | 88.65 | 0.142 |
| Words+pos tags+TFIDF | 90.35 | 0.137 |
| Words+pos tags+TFIDF+ORS | 94.87 | 0.115 |

Table 3. Random Forest Results

## 7 CONCLUSION AND FUTURE WORK

The fake news detection system achieved the highest accuracy for a Random Forests Classifier with the number of estimators as 1000. This model was built with a number of features ranging from n-grams to ORS, with the ORS feature significantly improving the performance.

As future work, the idea presented in this paper can be extended to classify fake news into multiple categories such as serious fabrications, large-scale hoaxes and humorous fakes as done in [6]. Another problem that we encountered was the inability to handle very small/very large news snippets by ORS. This was because a very small snippet got a nice ORS score regardless of whether the data is true or fake and a very large snippet could not fit in the query string. A few heuristic measures can be tried to overcome these difficulties. To counter the effect of small data we could give more weights to the important parts of the data such as the subject and the object. To handle large snippets of data we could consider summarizing the data into smaller chunks and take an average of the ORS scores of each chunk. Also one could try to implement more sophisticated models such as Probabilistic Context Free Grammar (PCFG).

## REFERENCES

[1] Maria Fernanda Caropreso, Stan Matwin, and Fabrizio Sebastiani. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. *Text databases and document management: Theory and practice*, 5478:78–102, 2001.

[2] Niall J Conroy, Victoria L Rubin, and Yimin Chen. Automatic deception detection: methods for finding fake news. *Proceedings of the Association for Information Science and Technology*, 52(1):1–4, 2015.

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

[4] Emilio Monti. Pystat parser, 2014.

[5] Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 309–319. Association for Computational Linguistics, 2011.

[6] Victoria L Rubin, Yimin Chen, and Niall J Conroy. Deception detection for news: three types of fakes. *Proceedings of the Association for Information Science and Technology*, 52(1):1–4, 2015.

[7] Pascal Soucy and Guy W Mineau. Beyond tfidf weighting for text categorization in the vector space model. In *IJCAI*, volume 5, pages 1130–1135, 2005.