Fake News Classifier

Joshua Andrew Mack
Department of Electrical and Computer Engineering
University of Arizona
jmack2545@email.arizona.edu
Srishti Sinha
Department of Electrical and Computer Engineering
University of Arizona
ssinha@email.arizona.edu

Abstract

In recent times, there has been a large amount of discussion about the topic of "fake news". From the context of machine learning, there are a large number of subtopics within the scope of "fake news" that could be studied almost exclusively, ranging from adversarial training and dataset integrity to language modeling and knowledge representations. In our work, we take for granted the integrity of two datasets from Kaggle [4][5] and seek to simply explore how feasible it is to classify news articles as real or fake based on simple metrics that can easily be scraped by an automated web crawler. To this end, after performing initial dataset manipulation, we utilized Labeled Latent Dirichlet Allocation [3] with a few different hyperparameter setups to train and test on a combined dataset of close to 430,000 articles from the years 2014 and 2016. In the end, we achieved a maximum testing accuracy of 84.72% and a Matthews Correlation Coefficient of 0.355 and identified areas of future work that may improve these results further.

1 Introduction and Related Work

In this paper we consider the problem of Fake news, which has become a hot and potentially profitable area with many companies working on this. Different companies have different motives behind classification of fake news but the real stakeholders in this battle are all motivated by money. Companies like Facebook and Google are suffering from advertisers pulling their ads that were programmatically placed on sites that proved to provide fake news or other offensive content. Beyond that they have general reputational risk with their users who may visit less often if they are exposed to blatantly false or offensive material. Furthermore, there are many news agencies that take raw information feeds and convert them into news stories. They would suffer greatly if their material were to be compromised by falsehoods.[11]

There have been many different approaches for fake news classification including linguistic cue approaches (with machine learning), and network analysis approaches. We see promise in an innovative hybrid approach that combines linguistic cue and machine learning, with network-based behavioral data[6].Linguistic Approaches in which the content of deceptive messages is extracted and analyzed to associate language patterns with deception and Network Approaches in which network information, such as message metadata or structured knowledge network queries can be harnessed to provide aggregate deception measures. Both forms typically incorporate machine learning techniques for training classifiers to suit the analysis. In this approach, sets of word and category frequencies are used for subsequent automated numerical analysis. One common use is for the training of classifiers like Support Vector Machines (SVM) and Naïve Bayesian models. The use of different clustering methods and distance functions between data points shape the accuracy of SVM which invites new experimentation on the net effect of these variables. Naïve Bayes algorithms make classifications based on accumulated evidence of the correlation between a given variable and the other variables present in the model.[6]

We took a similar approach for Fake news classification, but in terms of the classifier used, we utilized a supervised version of a well known topic modeling technique, Latent Drichlet Allocation (LDA).

2 Technical Approach

2.1 Data Preparation

We began by exploring the data contained within our two datasets and deciding how to best combine the two. The first dataset provided a total of roughly 18,000 samples of fake news from the year of 2016 while the second provided roughly 422,000 samples of real news articles from the year of 2014.

The first dataset provided a more detailed, if messier, set of features to work with including author, date published, title, the actual text of the article, the language of the article, the date the website was crawled, the hostname, and the "type" of fake news. As our goal was simply to perform a simple binary classification of "Fake" or "Real", we mapped all articles in this dataset to the single label of "Fake". The second dataset provided much more data, but this was at the cost of less features about that data, including features like article title, article URL, hostname, and publisher.

In combining the two, our goal was to introduce as little bias towards one dataset or the other as possible in the eventual model by minimizing the amount of information that could be used to identify which dataset the data came from. As the two datasets were from different years, there were actually no articles in common between them. As a result, we had to use the rough guideline that if a hostname from the fake dataset was contained in the UCI dataset, we labeled all articles from that hostname as "Fake". To protect from the potential bias towards hostname that may not actually exist in the real data, we began by stripping the hostnames from the data and classifying solely based on article title.

2.2 LLDA

In terms of the classifier itself, we utilized a supervised variant of a well known topic modeling technique, Latent Dirichlet Allocation (LDA). LDA itself is an unsupervised, generative model that allows for the automatic discovery of topics within a corpus of text. It is covered in more detail in [1], but the basic ideas are presented here.

LDA is based around the generation of and distributions associated with "words", "topics", "documents", and "corpora". In making the discussion rigorous, the following definitions are helpful:

- A *word* is the basic unit of data that the model considers. Each word is represented by a single entry in a vocabulary vector V, and thus each individual word is a one-hot basis vector where the component that corresponds with a given word is 1 and the rest are 0.
- A *document* is a sequence of words $(\mathbf{w}_i) = (w_1, w_2, ..., w_N)$.
- A *corpus* is a collection of documents $D = \{(\mathbf{w}_1), (\mathbf{w}_2), ..., (\mathbf{w}_M)\}$
- A *topic* is a probability distribution over the vocabulary V where each word $v \in V$ is assigned a probability of occurring within a given topic.

With these definitions in hand, the main idea of LDA is that it assumes that documents are generated by random mixtures of topic distributions where the topic distributions themselves are latent — or non-observed — model variables. Specifically, given a corpus D, LDA assumes the following generative process for each document $(\mathbf{w}) \in D$:

1. Choose a number of words $N \sim Poisson(\xi)$
2. Choose $\theta \sim Dirichlet(\alpha)$
3. For each word $w_i, i \in \{1, 2, ..., N\}$

(a) Choose a topic $z_n \sim Multinomial(\theta)$

(b) Choose the word w_i by sampling from $p(w_i|z_n, \beta)$, a multinomial probability conditioned on the topic z_n

The core work of LDA, then, is to find the parameters for each of those topics that yield models which best explain the documents that have been observed (i.e. the training set). It achieves this through an iterative Expectation Maximization (EM) procedure, where the E step is to find the optimizing values of a given set of *variational parameters*, and the M step is to maximize the resulting lower bound (provided by Jensen's inequality) on the log likelihood function. LDA has model parameters of α and β which allow the user to control the similarity between topics and similarity between words in each topic.

In moving to LLDA[3], Ramage et al. turn the unsupervised LDA into the supervised LLDA by restricting the topic model for a given document to only choose topics that correspond to a document's given label set. This approach allows for the flexibility of each document pertaining to multiple topics as well as the freedom to, with modifications, implement a version of LLDA that doesn't assume all the topics are known a priori and thus allows for the discovery of new topics other than those listed in the labeled data.

To make predictions on with this generated model, we took an incoming sample to be classified — in our case, just the bag of words contained in the article headline — and converted it to a vocabulary vector v. Each entry in v corresponds to the same location in the overall model vocabulary V, and the entry $v_i = 1$ if the i^{th} model word was contained in the headline while it is 0 otherwise. After this, we used the Euclidean dot-product as a measure of similarity between this word vector v and each of the topic distributions. For example, if one topic has the word distribution $T1 = \{``dog" : 0.5, ``cat" : 0.25, ``bird" : 0.25\}$ and our title consists of the words "Dog Cat", the corresponding word vector is $[1, 1, 0]$, and the corresponding dot-product similarity is $0.5 * 1 + 0.25 * 1 + 0.25 * 0 = 0.75$. We do this for all topics in the model (in our case, "Fake" and "Real"), and we choose as our classification the topic that has the largest dot-product correlation with our observed word-vector.

3 Results

Moving to the experimental results, we built set of Python methods on top of the implementation of LLDA given by Shuyo[7] to perform all of the dataset manipulation and data wrangling, train an LLDA model, save it to disk, and collect various metrics about its testing generalization. We utilized default model parameters of $\alpha = \beta = 0.0001$.

In initial testing, the data was preprocessed as described in the previous section, with the two datasets merged such that only the title and a binary "fake" or "real" indicator remained. Common words such as "and", "or", "the", "a", etc. were filtered out from each article title as they provide no meaningful information for the model about the topics it needs to learn. The data was split randomly into 80% testing and 20% training using sklearn's train_test_split function[8]. We trained for 5 iterations of LLDA inference as we found that the model's training metric, perplexity, converged to a stable value quite quickly. At this point, we classified our testing data using the dot-product method described above, and the results are presented in the first column of Table 1.

Table 1: Results Achieved with Described Test Setups

	No Hostname, 5 iter	Hostname, 5 iter	Hostname, 35 iter
Testing Accuracy (%)	81.19796	84.7196	84.26416
Matthews Correlation Coefficient	0.261456	0.355474	0.343544
Recall (%)	80.0224	94.82952	92.12828
Precision (%)	11.93473	16.14261	15.80593
Model Perplexity	13982.772	15985.733	15753.101

While the testing accuracy in the first configuration isn't necessarily terrible, we wanted to see if improvements could be made. At this point, we noted that the instances "Fake" titles in the dataset occupied only 3% of the overall dataset. With this in mind, we also report metrics of Matthews

Correlation Coefficient, Recall, and Precision. If TP is the number of True Positives (fake articles that are labeled as fake), FP is the number of False Positives (real articles that are labeled as fake), TN is the number of True Negatives, (real articles labeled as real), and FN is the number of Fake Negatives (fake articles labeled as real), then these three metrics are defined as follows:

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

In other words, recall is the percentage of "Fake" article titles that the model correctly labels as fake while precision is the percentage of actual "Fake" articles labeled as fake out of all the articles that were labeled as fake, and the Matthews Correlation Coefficient is a metric that varies between -1 and 1 that attempts to summarize the entire confusion matrix of the model in one number[10].

As can be seen from these metrics, the first attempt actually does a decent job at identifying fake articles as fake, but it has a very high FP rate, meaning that a large proportion of real articles are being misclassified as fake in relation to the total number of actual fake samples. To combat this, we made a small change to our data preparation methodology and tested two additional configurations. We reasoned that a large amount of information about the validity of an article may be contained in the hostname/website that the article came from. Despite intentionally removing hostnames earlier in the process, we added them back in at this point so that we could examine what kind of changes occur in the resulting model. We did this by simply prepending the hostname of each website as the first word in each article title and performing the same analysis.

The second column in Table 1 demonstrates this first attempt with the added hostnames. This change did fairly dramatically drive the recall up towards 100%, but it did little to change the Precision and corresponding high FP rate. To test if we potentially weren't using enough iterations, we tested with 35 training iterations and recorded those results in the third column of Table 1. Compared to the second model, the third almost universally overfitted to the training data and thus yielded worse generalization performance. In summary, though, precision seems to be the main metric holding the model back at this point.

We hypothesize that this is simply due to the difference in popular news topics between the years 2014 and 2016. The LLDA vocabulary distribution for the "real" topic is heavily weighted towards words that were popular in the field of news media in 2014 but not 2016 like "Kanye", "Kardashian", and "Ebola" (as it turns out, Kanye West married Kim Kardashian in 2014) while the vocabulary distribution for the "fake" topic is heavily weighted towards words that were popular in the news in 2016 (specifically the election season) but not 2014 like "Trump", "Clinton", and "Russia". Coupled with the fact that the fake news dataset doesn't appear to cover the same distribution of topics as the UCI news one does (i.e. the fake news one is heavily Election 2016 focused while UCI covers a wide range of topics across 2014). If a larger dataset could be synthesized that (i) provides a better balance of real and fake news to overcome the current class imbalance and (ii) is from the same time period as to remove bias from coverage of specific topics over others, then we believe the results may improve — or the datasets may simply be too similar to distinguish and this method may turn out to be infeasible.

4 Conclusion and Future Work

In conclusion, we have verified that it is possible, if given high integrity data, to provide better-than-guessing results in the topic of fake news classification. We have demonstrated the viability of LLDA in this subject matter, and we have presented an approach that, while not perfect, is certainly progress towards being able to perform automated integrity classification of news articles.

Potential future work includes applying the methods described here to datasets from the same time period as to minimize topic difference based on news cycle, investigating how to crowd source high integrity information to reduce the cost of building such datasets, feature selection or feature extraction techniques to improve separability of the data, and alternate methods of approaching the problem entirely (perhaps through the application of techniques such as Recurrent Neural Networks or String Kernels along with kernel-based classification methods).

References

[1] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. *Latent dirichlet allocation*. Journal of machine Learning research 3.Jan (2003): 993-1022.

[2] McAuliffe, Jon D. and Blei, David M. *Supervised topic models*. Advances in neural information processing systems. 2008.

[3] Ramage, Daniel and Hall, David and Nallapati, Ramesh and Manning, Christopher D.*Labeled LDA: A Supervised Topic Model for Credit Attribution in Multi-labeled Corpora*. 2009. http://dl.acm.org/citation.cfm?id=1699510.1699543

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

[5] UCI Machine Learning, *News Aggregator Dataset*, https://www.kaggle.com/uciml/news-aggregator-dataset

[6] Conroy, Niall J., 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 (2015): 1-4.

[7] Shuyo, Nakatani. *Python implementation of Labeled LDA*. https://shuyo.wordpress.com/2013/07/24/python-implementation-of-labeled-lda-ramage-emnlp2009/

[8] Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. et al, *Scikit-learn: Machine Learning in Python*, http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html, 2011

[9] Victoria L. Rubin, Niall J. Conroy and Yimin Chen. *Towards News Verification: Deception Detection Methods for News Discourse* HICSS2015 (2015), http://works.bepress.com/victoriarubin/6/

[10] Lettier, David. *Matthews Correlation Coefficient*. https://lettier.github.io/posts/2016-08-05-matthews-correlation-coefficient.html

[11] Vorhies, William.*Using Algorithms to Detect Fake News The State of the Art* http://www.datasciencecentral.com/profiles/blogs/using-algorithms-to-detect-fake-news-the-state-of-the-art