RAMANAND ARYA D.A.V COLLEGE

(Affiliated to University of Mumbai)

MUMBAI-MAHARASHTRA-400042

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled, "**Machine Learning for Detection of Fake News**" is bonafied work of **RAJ UDAY BAING** bearing Seat. No: **(208)** submitted in partial fulfilment of the requirements for the award of **M.sc** in **INFORMATION TECHNOLOGY** from University of Mumbai.

Internal Guide **Coordinator**

External Examiner

Date: **College Seal**

Raj Uday Baing, Roll No: 208

Machine Learning for Detection of Fake News

I would like to express my special thanks of gratitude to my teacher (Name of the teacher) as well as our principal (Name of the principal) who gave me the golden opportunity to do this wonderful project on the topic (Write the topic name), which also helped me in doing a lot of Research and i came to know about so many new things I am really thankful to them. Secondly I would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

Abstract

Recent political events have lead to an increase in the popularity and spread of fake news. As demonstrated by the widespread effects of the large onset of fake news, humans are inconsistent if not outright poor detectors of fake news. With this, efforts have been made to automate the process of fake news detection. The most popular of such attempts include “blacklists” of sources and authors that are unreliable. While these tools are useful, in order to create a more complete end to end solution, we need to account for more difficult cases where reliable sources and authors release fake news. As such, the goal of this project was to create a tool for detecting the language patterns that characterize fake and real news through the use of machine learning and natural language processing techniques. The results of this project demonstrate the ability for machine learning to be useful in this task. We have built a model that catches many intuitive indications of real and fake news as well as an application that aids in the visualization of the classification decision.

Contents

List of Figures

Chapter 1

Introduction

The rise of fake news during the 2016 U.S. Presidential Election highlighted not only the dangers of the effects of fake news but also the challenges presented when attempting to separate fake news from real news. Fake news may be a relatively new term, but it is not necessarily a new phenomenon. Fake news has technically been around at least since the appearance and popularity of one-sided, partisan newspapers in the 19th century. However, advances in technology and the spread of news through different types of media have increased the spread of fake news today. As such, the effects of fake news have increased exponentially in the recent past and something must be done to prevent this from continuing in the future.

I have identified the three most prevalent motivations for writing fake news and chosen only one as the target for this project as a means to narrow the search in a meaningful way. The first motivation for writing fake news, which dates back to the 19th century one-sided party newspapers, is to influence public opinion. The second, which requires more recent advances in technology, is the use of fake head- lines as clickbait to raise money. The third motivation for writing fake news, which is equally prominent yet arguably less dangerous, is satirical writing. While all three subsets of fake news, namely, clickbait, influential, and satire, share the common thread of being fictitious, their widespread effects are vastly different. As such, this paper will focus primarily on fake news as defined by poli-tifact.com, “fabricated content that intentionally masquerades as news coverage of actual events.” This definition excludes satire, which is intended to be humorous

and not deceptive to readers. Most satirical articles come from sources like “The Onion “, which specifically distinguish themselves as satire. Satire can already be classified, by machine learning techniques according to . Therefore, our goal is to move beyond these achievements and use machine learning to classify, at least as well as humans, more difficult discrepancies between real and fake news.

The dangerous effects of fake news, as previously defined, are made clear by events such as in which a man attacked a pizzeria due to a widespread fake news article. This story along with analysis from provide evidence that humans are not very good at detecting fake news, possibly not better than chance. As such, the question remains whether or not machines can do a better job.

There are two methods by which machines could attempt to solve the fake news problem better than humans. The first is that machines are better at detecting and keeping track of statistics than humans, for example it is easier for a machine to detect that the majority of verbs used are “suggests” and “implies” versus, “states” and “proves.” Additionally, machines may be more efficient in surveying a knowledge base to find all relevant articles and answering based on those many different sources. Either of these methods could prove useful in detecting fake news, but we decided to focus on how a machine can solve the fake news problem using supervised learning that extracts feature of the language and content only within the source in question, without utilizing any fact checker or knowledge base. For many fake news detection techniques, a “fake” article published by a trustworthy author through a trustworthy source would not be caught. This approach would combat those “false negative” classifications of fake news. In essence, the task would be equivalent to what a human face when reading a hard copy of a newspaper article, without internet access or outside knowledge of the subject (versus reading something online where he can simply look up relevant sources). The machine, like the human in the coffee shop, will have only access to the words in the article and must use strategies that do not rely on blacklists of authors and sources.

The current project involves utilizing machine learning and natural language processing techniques to create a model that can expose documents that are, with

high probability, fake news articles. Many of the current automated approaches to this problem are centered around a “blacklist” of authors and sources that are known producers of fake news. But, what about when the author is unknown or when fake news is published through a generally reliable source? In these cases it is necessary to rely simply on the content of the news article to make a decision on whether or not it is fake. By collecting examples of both real and fake news and training a model, it should be possible to classify fake news articles with a certain degree of accuracy. The goal of this project is to find the effectiveness and limitations of language-based techniques for detection of fake news through the use of machine learning algorithms including but not limited to convolutional neural networks and recurrent neural networks. The outcome of this project should determine how much can be achieved in this task by analyzing patterns contained in the text and blind to outside information about the world.

This type of solution is not intended to be an end-to end solution for fake news classification. Like the “blacklist” approaches mentioned, there are cases in which it fails and some for which it succeeds. Instead of being an end-to-end solution, this project is intended to be one tool that could be used to aid humans who are trying to classify fake news. Alternatively, it could be one tool used in future applications that intelligently combine multiple tools to create an end-to-end solution to automating the process of fake news classification.

Chapter 2

Related Work

2.1 Spam Detection

The problem of detecting not-genuine sources of information through content-based analysis is considered solvable at least in the domain of spam detection [7], spam detection utilizes statistical machine learning techniques to classify text (i.e. tweets [8] or emails) as spam or legitimate. These techniques involve pre-processing of the text, feature extraction (i.e. bag of words), and feature selection based on which features lead to the best performance on a test dataset. Once these features are obtained, they can be classified using Nave Bayes, Support Vector Machines, TF-IDF, or K-nearest neighbors classifiers. All of these classifiers are characteristic of supervised machine learning, meaning that they require some labeled data in order to learn the function (as seen in [9])

$$f(message, \theta) = \begin{cases} C_{spam} & \text{if classified as spam} \\ C_{leg} & \text{otherwise} \end{cases}$$

where, m is the message to be classified and is a vector of parameters and Cspam and Cleg are respectively spam and legitimate messages. The task of detecting fake news is similar and almost analogous to the task of spam detection in that both aim to separate examples of legitimate text from examples of illegitimate, ill-intended texts. The question, then, is how can we apply similar techniques to fake news detection. Instead of filtering like we do with spam, it would be beneficial to be able

to flag fake news articles so that readers can be warned that what they are reading is likely to be fake news. The purpose of this project is not to decide for the reader whether or not the document is fake, but rather to alert them that they need to use extra scrutiny for some documents. Fake news detection, unlike spam detection, has many nuances that arent as easily detected by text analysis. For example, a human actually needs to apply their knowledge of a particular subject in order to decide whether or not the news is true. The “fakeness” of an article could be switched on or off simply by replacing one person’s name with another person’s name. Therefore, the best we can do from a content-based standpoint is to decide if it is something that requires scrutiny. The idea would be for a reader to do leg work of researching other articles on the topic to decide whether or not the article is actually fake, but a “flagging” would alert them to do so in appropriate circumstances.

2.2 Stance Detection

In December of 2016, a group of volunteers from industry and academia started a contest called the Fake News Challenge [10]. The goal of this contest was to encour- age the development of tools that may help human fact checkers identify deliberate misinformation in news stories through the use of machine learning, natural language processing and artificial intelligence. The organizers decided that the first step in this overarching goal was understanding what other news organizations are saying about the topic in question. As such, they decided that stage one of their contest would be a stance detection competition. More specifically, the organizers built a dataset of headlines and bodies of text and challenged competitors to build classi- fiers that could correctly label the stance of a body text, relative to a given headline, into one of four categories: “agree”, “disagree”, “discusses” or “unrelated.” The top three teams all reached over 80% accuracy on the test set for this task. The top team’s model was based on a weighted average between gradient-boosted decision trees and a deep convolutional neural network.

2.3 Benchmark Dataset

[11] demonstrates previous work on fake news detection that is more directly related to our goal of using a text-only approach to make a classification. The authors not only create a new benchmark dataset of statements (see Section 3.1), but also show that significant improvements can be made in fine-grained fake news detection by using meta-data (i.e., speaker, party, etc.) to augment the information provided by the text.

Chapter 3

Datasets

The lack of manually labeled fake news datasets is certainly a bottleneck for advancing computationally intensive, text-based models that cover a wide array of topics. The dataset for the fake news challenge does not suit our purpose due to the fact that it contains the ground truth regarding the relationships between texts but not whether or not those texts are actually true or false statements. For our purpose, we need a set of news articles that is directly classified into categories of news types (i.e. real vs. fake or real vs parody vs. clickbait vs. propaganda). For more simple and common NLP classification tasks, such as sentiment analysis, there is an abundance of labeled data from a variety of sources including Twitter, Amazon Reviews, and IMDb Reviews. Unfortunately, the same is not true for finding labeled articles of fake and real news. This presents a challenge to researchers and data sci- dentists who want to explore the topic by implementing supervised machine learning techniques. I have researched the available datasets for sentence-level classification and ways to combine datasets to create full sets with positive and negative examples for document-level classification.

3.1 Sentence Level

[11] produced a new benchmark dataset for fake news detection that includes 12,800 manually labeled short statements on a variety of topics. These statements come from politifact.com, which provides heavy analysis of and links to the source

documents for each of the statements. The labels for this data are not true and false but rather reflect the “sliding scale” of false news and have 6 intervals of labels. These labels, in order of ascending truthfulness, include ’pants-fire’, ’false’, barely true, ’half-true’, ’mostly-true’, and true. The creators of this database ran baselines such as Logistic Regression, Support Vector Machines, LSTM, CNN and an augmented CNN that used metadata. They reached 27% accuracy on this multiclass classification task with the CNN that involved metadata such as speaker and party related to the text.

3.2 Document Level

There exists no dataset of similar quality to the Liar Dataset for document- level classification of fake news. As such, I had the option of using the headlines of documents as statements or creating a hybrid dataset of labeled fake and legiti- mate news articles. shows an informal and exploratory analysis carried out by combining two datasets that individually contain positive and negative fake news examples. Genes trains a model on a specific subset of both the Kaggle dataset and the data from NYT and the Guardian. In his experiment, the topics involved in training and testing are restricted to U.S News, Politics, Business and World news. However, he does not account for the difference in date range between the two datasets, which likely adds an additional layer of topic bias based on topics that are more or less popular during specific periods of time.

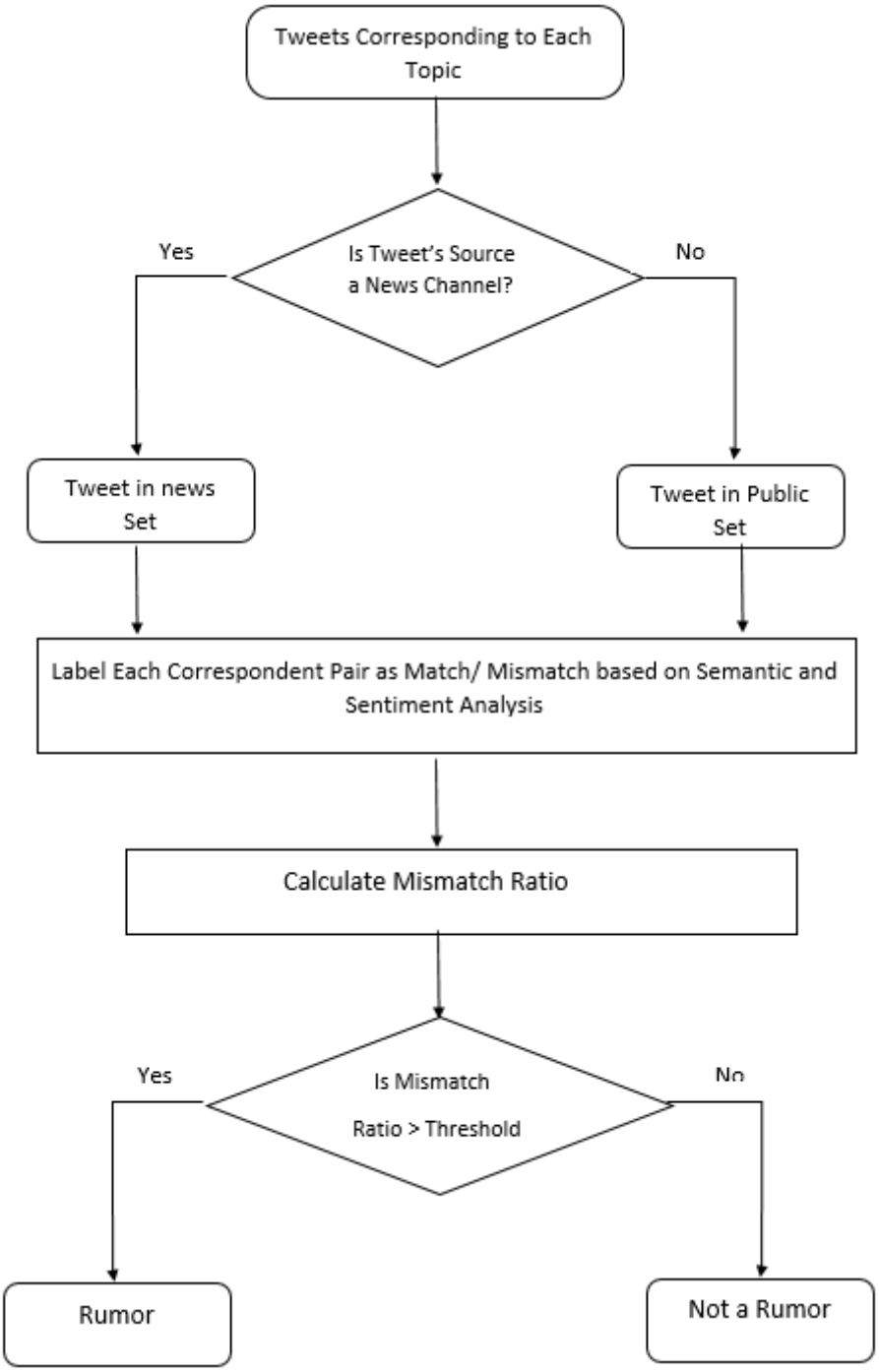
We have collected data in a manner similar to that of Genes , but more cautious in that we control for more bias in the sources and topics. Because the goal of our project was to find patterns in the language that are indicative of real or fake news, having source bias would be detrimental to our purpose. Including any source bias in our dataset, i.e. patterns that are specific to NYT, The Guardian, or any of the fake news websites, would allow the model to learn to associate sources with real/fake news labels. Learning to classify sources as fake or real news is an easy problem but learning to classify specific types of language and language patterns as fake or real news is not. As such, we were very careful to remove as much of

the source-specific patterns as possible to force our model to learn something more meaningful and generalizable.

We admit that there are certainly instances of fake news in the New York Times and probably instances of real news in the Kaggle dataset because it is based on a list of unreliable websites. However, because these instances are the exception and not the rule, we expect that the model will learn from the majority of articles that are consistent with the label of the source. Additionally, we are not trying to train a model to learn facts but rather learn deliveries. To be clearer, the deliveries and reporting mechanisms found in fake news articles within New York Times should still possess characteristics more commonly found in real news, although they will contain fictitious factual information.

3.2.1 Fake news samples

contains a dataset of fake news articles that was gathered by using a tool called the BS detector (which essentially has a blacklist of websites that are sources of fake news. The articles were all published in the 30 days between October 26 2016 to November 25, 2016. While any span of dates would be characterized by the current events of that time, this range of dates is particularly interesting because it spans the time directly before, during, and directly after the 2016 election. The dataset has articles and metadata from 244 different websites, which is helpful in the sense that the variety of sources will help the model to not learn a source bias. However, at a first glance of the dataset, you can easily tell that there are still certain obvious reasons that a model could learn specifics of what is included in the “body” text in this dataset. For example, there are instances of the author and source in the body text, as seen in Section 3.1. Also, there are some patterns like including the date that, if not also repeated in the real news dataset, could be learned by the model.


Tweets Corresponding to Each Topic
Is Tweet's Source a News Channel?
Yes
No
Tweet in news Set
Tweet in Public Set
Label Each Correspondent Pair as Match/ Mismatch based on Semantic and Sentiment Analysis
Calculate Mismatch Ratio
Is Mismatch Ratio > Threshold
Yes
No
Rumor
Not a Rumor

All of these sources and authors are repeated in the dataset. Additionally, the presence of the date/title could be an easy cue that a text came from this dataset if the real news dataset did not contain this metadata. As such, the model could easily learn the particulars of this dataset and not learn anything about real/fake news itself in order to best classify the data. To avoid this, we removed the author, source, date, title, and anything that appeared before these segments. The dataset also contained a decent amount of repetitive data and incomplete data, we removed any non-unique samples and also simples that appeared incomplete (i.e., lacked a source). This left us with approximately 12,000 samples of fake news. Since the Kaggle dataset does not contain positive examples, i.e. examples of real news, it is necessary to augment the dataset with such in order to either compare or perform supervised learning.

3.2.2 Real news samples

As suggested by , an acceptable approach would be to use the APIs from reliable sources like New York Times and The Guardian. The NYT API provides similar information to that of the kaggle dataset, including both text and images that are found in the document. The Kaggle Dataset also provides the source of each article, which is trivial for the APIs of specific newspaper sources.

pulled articles from both of these sources in the same range of dates that the fake news was restricted to (October 26, 2016 to November 25, 2016). This is important because of the specificity of the current events at that time - information that would not likely be present in news outside of this timeframe. There were just over 9,000 Guardian articles and just over 2,000 New York Times articles. Unlike the Kaggle dataset, which had 244 different websites as sources, our real news dataset only has two different sources: The New York Times and The Guardian. Due to this difference, we found that extra effort was required to ensure that we removed any source-specific patterns so that the model would not simply learn to identify how an article from the New York Times is written or how an article from The Guardian is written. Instead, we wanted our model to learn more meaningful language patterns that are similar to real news reporting, regardless of the source.

Chapter 4

Methods

4.1 Sentence-Level Baselines

I have run the baselines described namely multi-class classification done via logistic regression and support vector machines. The features used were n-grams and TF-IDF. N-grams are consecutive groups of words, up to size “n”. For example, bigrams are pairs of words seen next to each other. Features for a sentence or phrase are created from n-grams by having a vector that is the length of the new “vocabulary set,” i.e., it has a spot for each unique n-gram that receives a 0 or 1 based on whether or not that n-gram is present in the sentence or phrase in question. TF-IDF stands for term frequency inverse document frequency. It is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. As a feature, TF-IDF can be used for stop-word filtering, i.e., discounting the value of words like “and,”, “the”, etc. whose counts likely have no effect on the classification of the text. An alternative approach is removing stop- words (as defined in various packages, such as Pythons NLTK). The results for this preliminary evaluation are found in Table 4.1

Table 4.1: Preliminary Baseline Results

Model	Vectorizer	N-gram Range	Penalty, C	Dev Score
Logistic Regression	Bag of Words	1-4	0.01	0.2586
Logistic Regression	TF-IDF	1-4	10	0.2516
SVM w. Linear Kernel	Bag of Words	1	10	0.2508
SVM w. RBF kernel	Bag of Words	1	1000	0.2492

Additionally, we explored some of the characteristic n-grams that may influence Logistic Regression and other classifiers. In calculating the most frequent n-grams for “pants-fire” phrases and those of “true” phrases, we found that the word “wants” more frequently appears in “pants-fire” (i.e., fake news) phrases and the phrase “states” more frequently appears in “true” (i.e., real news) phrases. Intuitively, this makes sense because it is easier to lie about what a politician wants than to lie about what he or she has stated since the former is more difficult to confirm. This observation motivates the experiments in Section 4.2, which aim to find a fuller set of similarly intuitive patterns in the body texts of fake news and real news articles.

4.2 Document-Level

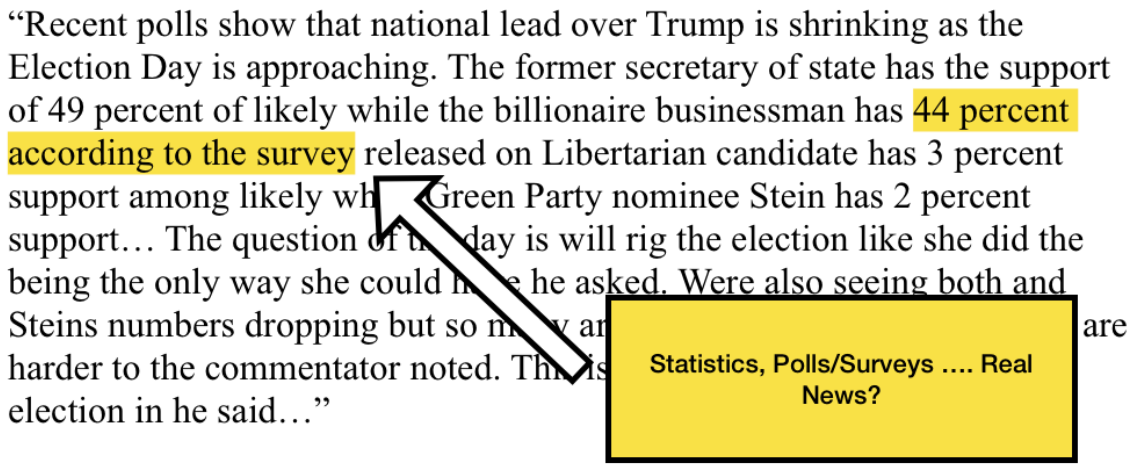
Deep neural networks have shown promising results in NLP for other classi- fication tas CNNs are well suited for picking up multiple patterns, and sentences do not provide enough data for this to be useful. However, a CNN baseline modeled off of the one described for NLP did not show a large im- provement in accuracy on this task using the Liar Dataset. This is due to the lack of context provided in sentences. Not surprisingly, the same CNN performance on the full body text datasets we created was much higher.

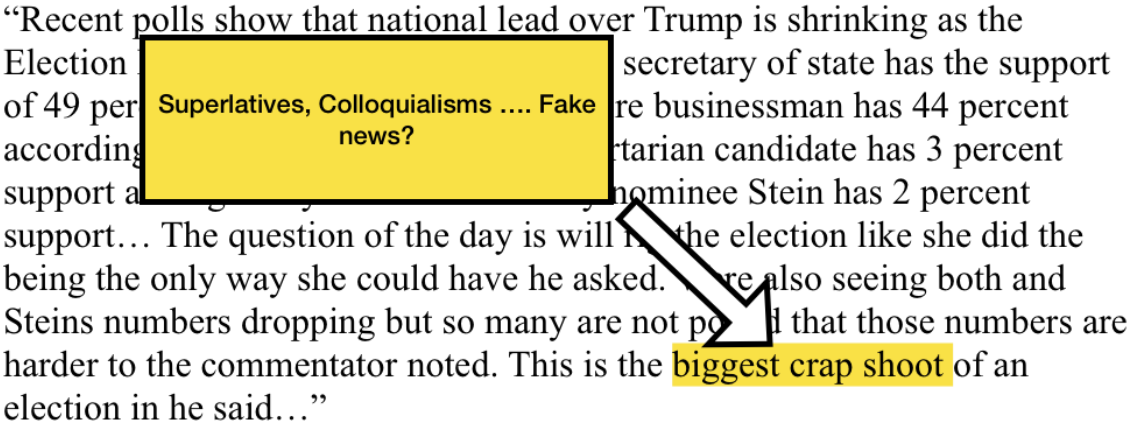
4.2.1 Tracking Important Trigrams

The nature of this project was to decide if and how machine learning could be useful in detecting patterns characteristic of real and fake news articles. In accordance with this purpose, we did not attempt to build deeper and better neural nets in order to improve performance, which was already much higher than expected. Instead, we took steps to analyze the most basic neural net. We wanted to learn what patterns it was learning that resulted in such a high accuracy of being able to classify fake and real news.

If a human were to take on the task of picking out phrases that indicate fake or real news, they may follow guidelines This and similar

guidelines often encourage readers to look for evidence supporting claims because fake news claims are often unbacked by evidence. Likewise, these guidelines encourage people to read the full story, looking for details that seem "far-fetched." Figures 4.1 and 4.2 show examples of the phrases a human might pick up on to decide if an article is fake or real news. We were curious to see if a neural net might pick up on similar patterns.





this intricacy, the model was able to learn overlapping segments. For example, the 4-gram “Donald Trump’s presidential election” could be learned in addition to the trigrams “Donald Trump’s presidential” and “Trumps presidential election”. To avoid this overlapping, we simplified the network to only look at filter size 3, i.e., trigrams. We found that this did not cause a significant drop in accuracies; there was less than one half percent decrease in accuracy from the model with filter sizes
= [3,4,5] to the model with filter sizes. We limited the data to 1000 words because less than ten percent of the data was over this limit and found most of the time the article was longer than 1000 words it contained excess information at the end that was not relevant to the article itself. For example, lengthy ads were sometimes found at the end of articles, causing them to go over 1000 words. There were no noticeable drops in accuracy across trials when we restricted the document length to 1000 words.

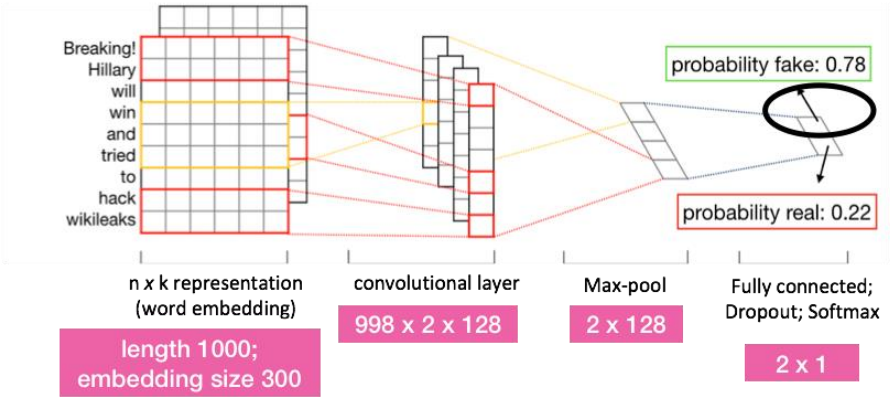
In order to obtain the trigrams that were most important in the classification decision, we essentially had to back-propagate from the output layer to the raw data (i.e. actual body text being classified)
We did this in a manner similar to . For any body text being evaluated by the CNN, we can find the trigrams that were “most fake” and “most real” by looking at the $weight_i \times activation_i$ for each of the individual neuron, i, when that text was evaluated. I will explain the process for finding the most real trigrams, and the same process can be used to find the most fake trigrams. The only difference is which column of the 2-columns in each layer you choose to look at.

The first step in this process is looking at the max pool layer where you will find a down sampled version of the convolutional layer (See Figure 4.4. Each of the 128 values are selected as the max of 998 values in the previous layer. Due to the dropout probability, we expect that a different pattern will cause the highest activation for each of these neurons. As such, the max-pool layer represents the value of the trigram that was closest to this pattern and made the neurons activation the highest.

Each value in the max-pool layer is representative of the neuron, i, $weight_i \times$

activation for that text. Therefore, we can select the neurons with the highest (most positive) $weight_{i]} \times activation$ to ultimately find the "most real" trigrams or we can select the neurons with the lowest (most negative) $weight + i \times activation_i$ to ultimately find the "least real" trigrams.

Depending on which we were looking at ("most real" or "least real"), we would pick a select number of neurons to trace backwards. For a selected neuron, say neuron number 120, we can find the 119th index out of the 128 dimensions in the output of the convolutional layer with ReLU function applied. Now, we have 998 values to look at. One of these values was chosen to be the max-pooled value, so we must look at all of them and find the match. Once we find the matching number, we have its index. Its index is representative of the trigram index in the original text. So if the index is 0, we look at the first trigram (words at indices 0,1, and 2) and if the index is 1, we look at t



appear very frequently in the real news dataset, but these are topics that you can imagine would not be written about, or rarely be written about, in fake news. The “hot topics” of fake news present another issue in this task. We do not want a model that simply chooses a classification based on the probability that a fake or real news article would be written on that topic just like we would never tell a person that every article written about Hillary is fake news or every article written about love is real news.

The way we accounted for these differences in the dataset was by separating our training set and tests sets on the presence/absence of certain words. We tried this for a number of topics that were present in both fake news and real news but had different proportions in the two categories. The words we chose were “Trump”, “election”, “war”, and “email.”

To create a model that was not biased about the presence of one of these words, we extracted all body texts which did not contain that word. We used this set as the training set. Then, we used the remaining body texts that did contain the target word as the test set. The accuracy of the model on the test set represents transfer learning in the sense that the model was trained on a number of articles about topics other than the target word and had to use what it learned to classify texts about the target word. The accuracies were still quite high, as demonstrated in section 5. This shows that the model was learning patterns of language other than those specific words. This could mean that it learned similar words because of the word embeddings, or it could mean that it learned completely different words to “pay attention” to, or both.

Figure 4.7: Words exclusively common to one category (Fake/Real)

(a) Fake News Frequent Words

(b) Real News Frequent Words

4.2.2 Cleaning

Pre-processing data is a normal first step before training and evaluating the data using a neural network. Machine learning algorithms are only as good as the data you are feeding them. It is crucial that data is formatted properly and meaningful features are included in order to have sufficient consistency that will result in the best possible results. for computer vision machine learning algorithms, pre-processing the data involves many steps including normalizing im- age inputs and dimensionality reduction. The goal of these is to take away some of the unimportant distinguishing features between different images. Features like the darkness or brightness are not beneficial in the task of labeling the image. Similarly, there are portions of text that are not beneficial in the task of labeling the text as real or fake.

The task of pre-processing data is often an iterative task rather than a linear one. This was the case in this project where we used a new and not yet standardized dataset. As we found certain unmeaningful features that the neural net was learning, we learned what more we needed to pre-process from the data.

Non-English Word Removal

Two observations that lead us to more pre-processing were the presence of run-on words and proper nouns in the most important trigrams for classification. An example of a run-on word that we saw frequently was in the "most fake" trigram category was "NotMyPresident" that came from a trending "hashtag" on twitter. There were also decisive trigrams that were simply pronouns like "Donald J Trump." Proper nouns could not possibly be helpful in a meaningful way to a machine learning algorithm trying to detect language patterns indicative of real or fake news. We want our algorithm to be agnostic to the subject material and make a decision based on the types of words used to describe whatever the subject is. Another algorithm may aim to fact check statements in news articles. In this situation, it would be important to maintain the proper nouns/subjects because changing the proper noun in the sentence "Donald J. Trump is our current president" to "Hillary Clinton is our current president" changes the classification of true fact to false fact. However, our purpose is not fact checking but rather language pattern checking, so removal of proper nouns should aid in pointing the machine learning algorithms in the right direction as far as finding meaningful features.

We removed "non-English" words by using PyEnchants version of the English dictionary. This also accounted for removal of digits, which should not be useful in this classification task, and websites. While links to websites may be useful in classifying the page rank of an article, it is not useful for the specific tool we were trying to create.

Source Pattern Removal

Another observation was that the two real news sources had some specific patterns that were easily learnable by the machine learning algorithms. This was more of an issue with the real news sources than the fake news sources because there were many more fake news sources than real news sources. More specifically, there were 244 fake news sources and only 128 neurons so the algorithm couldnt simply attune one neuron to each of the fake news sources patterns. There were only two

real news sources, however. Therefore, the algorithm was able to pick up easily on the presence or absence of these patterns and use that, without much help from other words or phrases, to classify the data.

There were a few separate steps in removing patterns from the real news sources. The New York Times articles of a particularly common section often started off with “Good morning. (or evening) Heres what you need to know:” This, along with other repeated sentences were always in italics. To account for the lack of consistency in the exact sentences that were repeated, we had to scrape the data again from the URLs and remove anything that was originally in italics. Another repeated pattern in the New York Times articles was parenthetical questions with links to sign up for emails, for example “Want to get California Today by email? Sign up.)“. Another pattern was in The Guardian, articles almost always ended with “Share on FacebookShare on TwitterShare via EmailShare on LinkedInShare on PinterestShare on Google+Share on WhatsAppShare on MessengerReuse this content” which is the result of links/buttons on the bottom of the webpage to share the article. When removing the non-English words, we were left with “on on on on on this content” which was enough of a pattern to force the model to learn classifica- tion almost solely based on its presence or absence. Note that this was a particularly strong pattern because it was consistent throughout the Guardian articles from all sections of the Guardian. Also, the majority of articles in our real news set are from the Guardian.

4.2.3 Describing Neurons

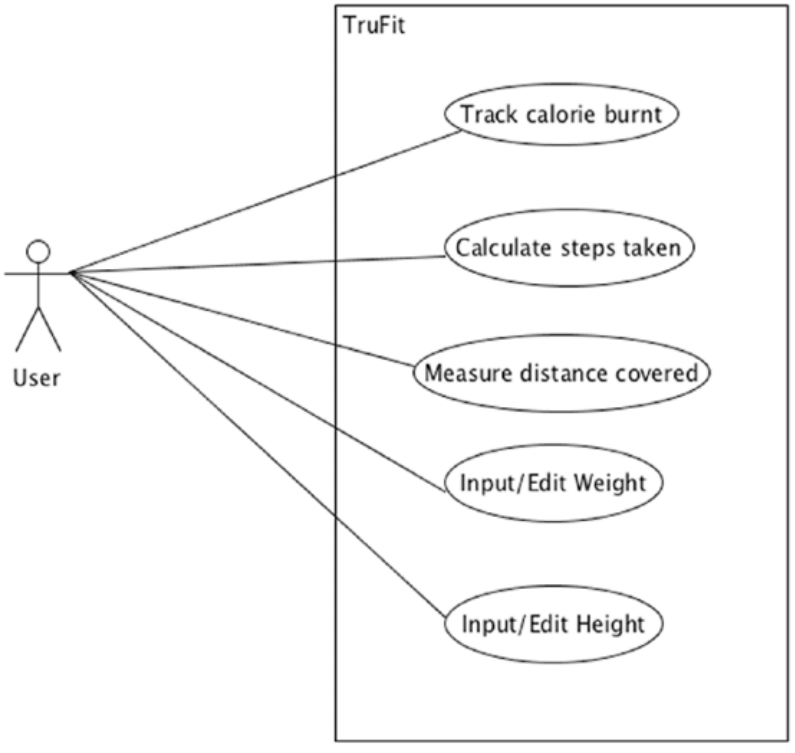
Although the accuracy was high in the classification task even after extensive pre-processing of the data, we wanted a way to more qualitatively evaluate how and what the neural net was learning the classification. Understanding and visualizing the way a CNN encodes information is an ongoing question. It is an infinitely more challenging pattern when there are more than one convolutional layer, which is why we kept our neural net shallow. For CNNs with one convolutional layer, shows a way to visualize any CNN single neuron as a filter in the first layer, in terms of the

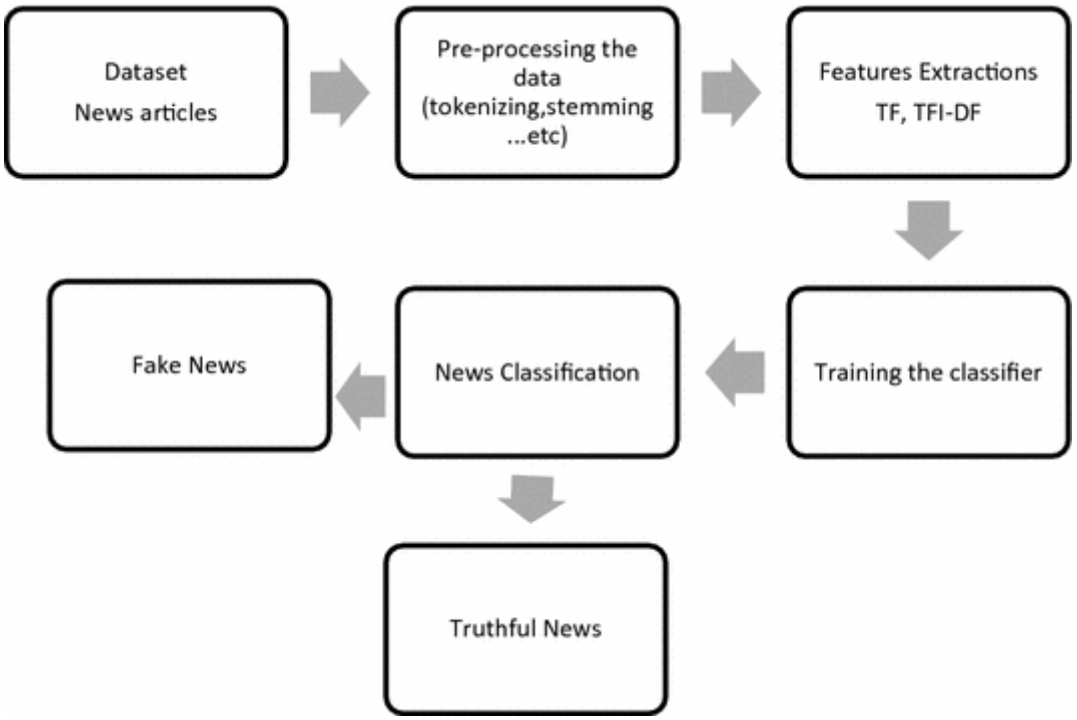
image space. We were able to use a similar method to “visualize” the CNN neurons as filters in the first (and only) layer in terms of text space.

Instead of finding the location in each image of the window that caused each neuron to fire the most, we find the location in the pre-processed text of the trigram (or length 3 sequence of words) that caused each neuron to fire the most. As the authors of [19] were able to identify patterns of colors/lines in images that caused firing, we were able to identify textual patterns that caused firing. Textual patterns are more difficult to visualize than image space patterns. While similar but non- identical RGB pixel values look similar, two words that are mathematically “similar” in their embedding but non-identical do not look similar. They do, however, have similar meanings.

In order to get a general grasp of the meaning of words/trigrams that each neuron was firing most highly for, we followed similar steps to those described in the section of 4.2.1. However, instead of finding those neurons that had the high-est/lowest *weight × activation*, we looked at each neuron, and which trigram in each body text resulted in the pooled value for that neuron. Then, we accumulated all of the trigrams for each neuron and summarized them by counting the instances of each word in the trigram. Our algorithm reported the words with the highest counts, excluding stopwords as described by NLTK (i.e. words like “the”, “a”, “by”, “it”, which are not meaningful in this circumstance). We were able to observe some clear patter

Use case Diagram





Class diagram

Chapter 5

Experimental Results

The accuracy of the model we believe is the most representative of how machine learning can handle fake news / real news classification task based simply on language patterns is 95.8 %. This model was trained and tested on a sample of the entire dataset, without any topic exclusion as described in section 4.2.2. This accu- racy can be represented by the following confusion matrix that shows the counts of each category of predictions. The rest of the accuracies and confusion matricescan be found in Table 5.1 in the Appendix.

Table 5.1: Confusion matrix from our "best" model

Actual Fake	2965	98
Actual Real	134	2307

To better understand which types of Fake news were being properlyclassified and which more were difficult to classify, we used [20] to gather different "types" of Fake

News. According to [20], fake news is separate from other categories such as clickbait, junkscience, rumor, hate, satire, etc. However, our dataset included sources that are listed as types other than straightforward “fake news.” The ma- jority of the 244 sources were listed in /citeopensources mapping of sources to their corresponding categories. Figure 5.1 shows the different categories that were in- cluded in our fake news dataset and their corresponding rate of misclassification. We excluded one category from this chart that was not misclassified. Table 9.1 expands on this data.

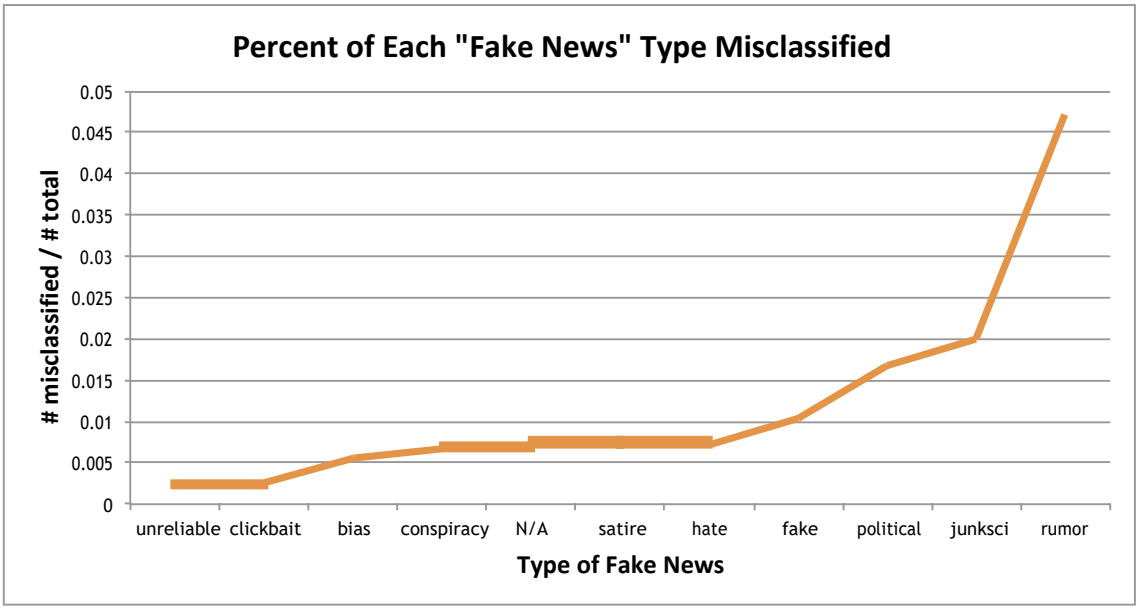
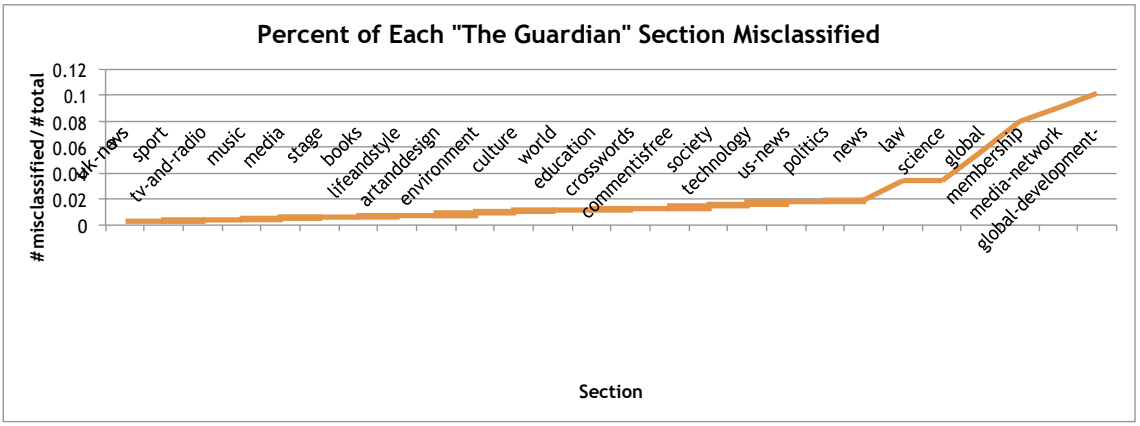
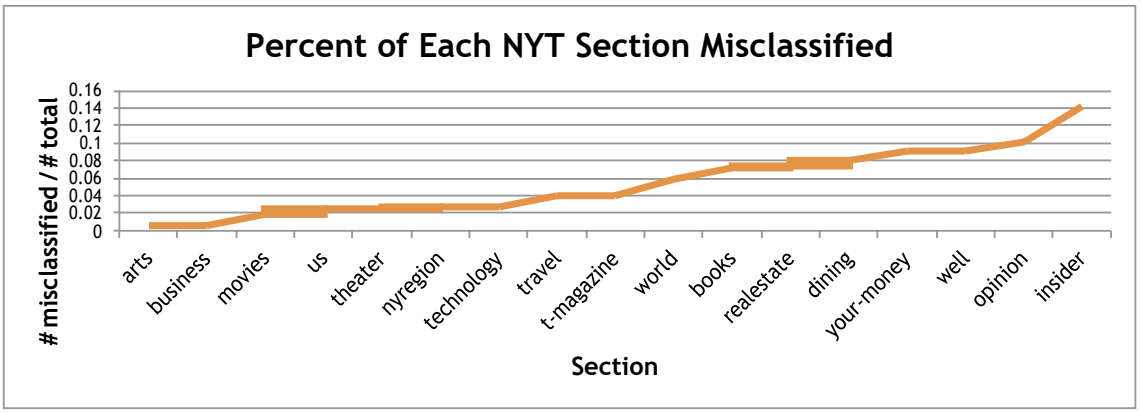


Figure 5.1: Fake News Types, and their misclassification rates.

We followed a similar procedure to identify which real news sections were most commonly misclassified as fake news. We obtained the section of news by taking it out of the URL. The sections are diverse, and some may be overlapping as a result of this. Additionally, the section names for the New York Times and The Guardian are distinct, so we have created two different plots to show the rate of misclassification for each. We have excluded from these charts any sections that made up <1 % of the full set from that news source or had a <1 % rate of misclassification. See below Figures 5.2 and 5.3 as well as Tables 9.2 and 9.3.





The New York Times sections, and their misclassification rates.

5.1 Tracking Important Trigrams

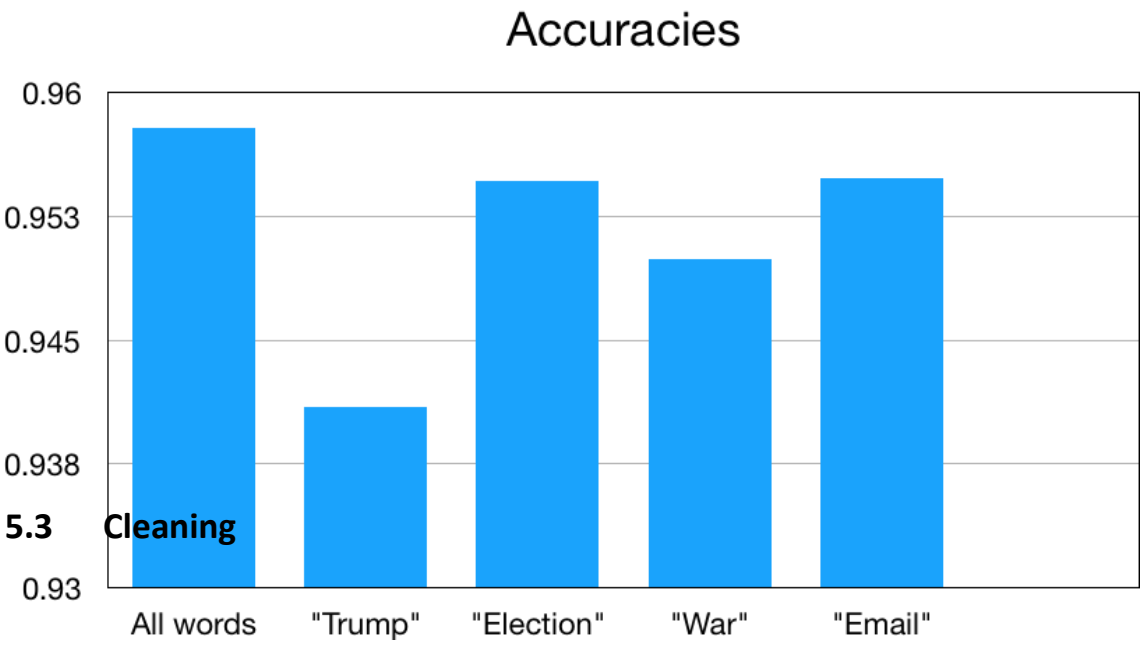
Throughout all of the different body texts, we captured the 10 trigrams whose weight * activation for each category was the most positive and most negative. For real news, the most positive weight * activation, we called “most real” and the most negative weight * activation, we called “least real”. We used the same terminology for fake news (i.e., “most fake” and “least fake”). To summarize our findings, we combined the “most real” with the “least fake” trigrams and combined the “most fake” with the “least real” trigrams. Within these two groups, we collected the 1000 most common words from the trigrams captured by the model. Then we took out the words that were common to both categories, to get those that were uniquely found as “fake” or “real” indicators. In Table 9.4, we have separated these words by part of speech to more easily compare the types of words chosen as indicative of fake and real.

5.2 Topic Dependency

We took some words that were more common in real news, some that were more common in fake news, and some that were similarly common in both real and fake news. Table 5.2 shows the distribution of each word in the fake and real news datasets. Also, note that other forms of the word were included such as plural

	Real Dataset Count	Fake Dataset Count
“Trump”	1926	3664
“election”	5658	5120
“war”	2143	3211
“email”	777	2408

The accuracies in show how well a model performed on the test set including only articles that contained the given word, after being trained on a dataset that only included articles that did not contain the given word.



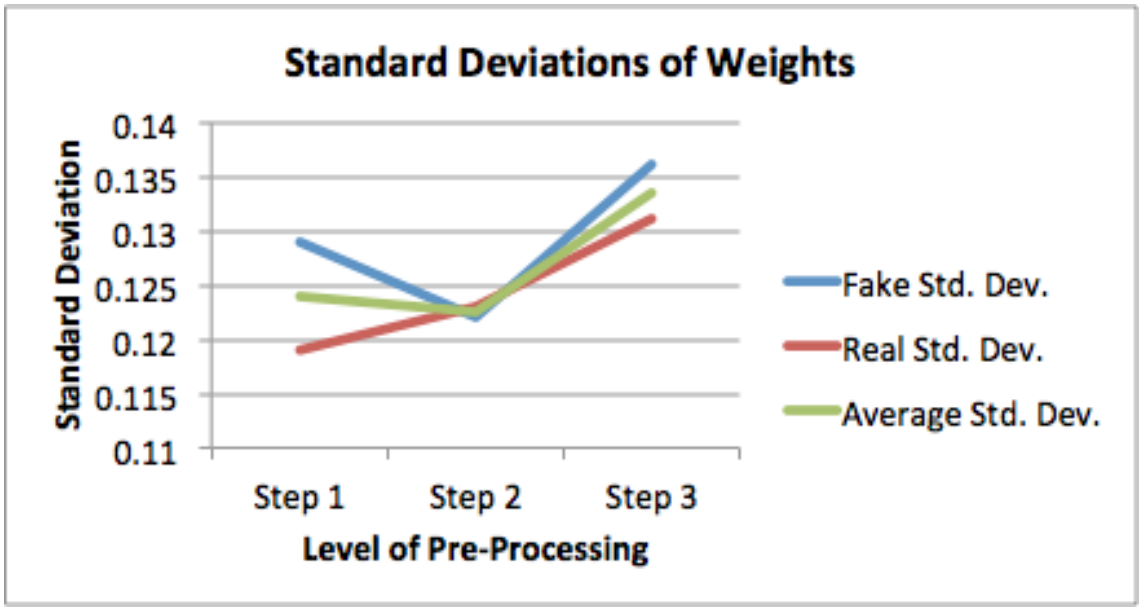
5.3 Cleaning

Although pre-processing our data to rid it of any distracting features was an iterative process, we have split it up into three major steps. These incremental steps each have corresponding models that were trained and tested on the data that was pre-processed at the level represented by the step name. All of the steps build on each other, such that the second step includes the first steps pre-processing and the third step includes the first two pre-processing methods. The first step is simple pre-processing (i.e. tokenization cleaning of data from citeyoonkim with the addition of our removal of source, author, title, and date from our own cleaning). The second

step is removing any non-English words, as described in the final step was removing the end of guardian articles which all said the same “Share on x, y, z.”

Figure 5.5 shows how the distribution of weights changed as the text was cleaned more. We anticipated that as we removed the easy words which were like cheat codes for classifying the text, there would be more neurons that contributed to the decision of classification and this was confirmed by the standard deviations. The final output of a fully connected layer is computed by summing $w_i * a_i$ for each neuron over all neurons, i. Therefore, the higher the absolute value of$w_i * a_i$ of a particular neuron, the more importance it had in the final classification decision.

Figure 5.7 shows how the accuracies of the model changed with more cleaning. We describe how this relates to the standard deviations and vocab size, as seen in Figure 5.6, in Section 6.3



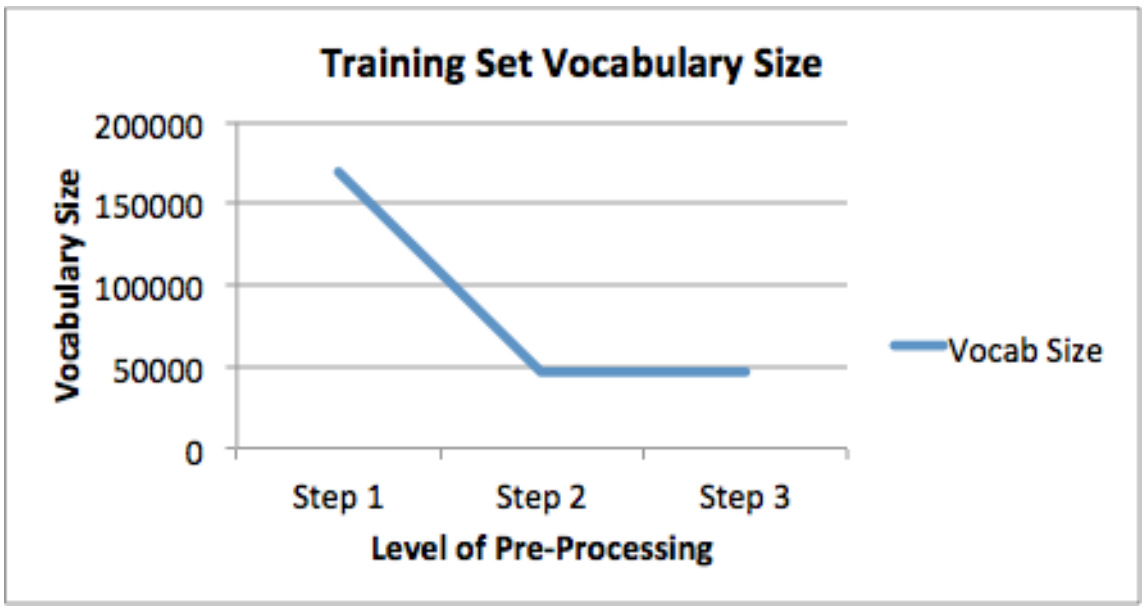
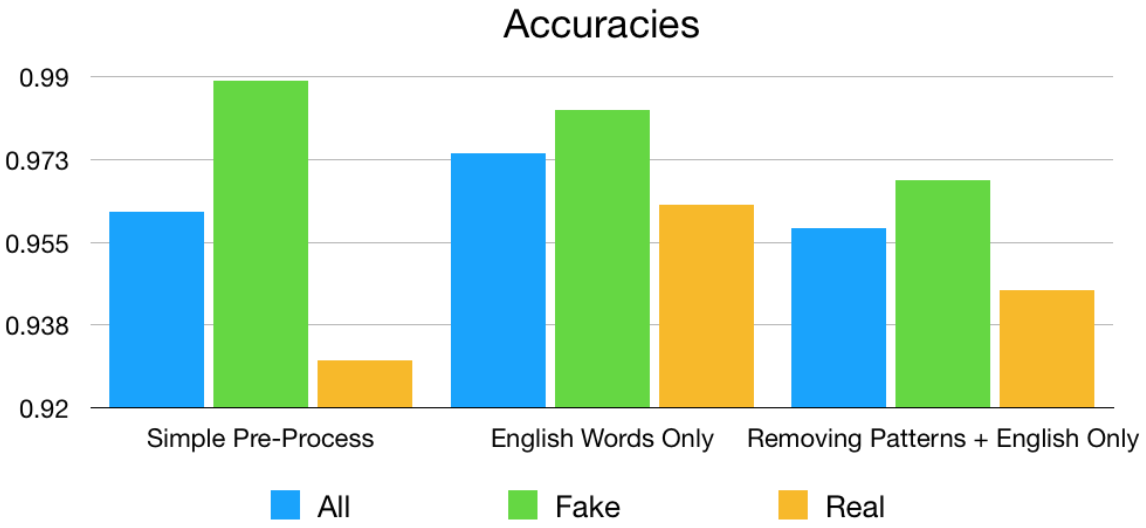


Figure 5.6: Vocab Size with Cleaning.



We accumulated all of the trigrams that resulted in the pooled value for each of the different body texts. Then, we found the most frequent words in the trigram set for each neuron, subtracting nltks stopwords from our set to remove articles like “the”, “a”, and other similarly common words. We claim that this set of words summarizes the pattern that a given neuron was detecting. Below are some examples of the most common words for a neuron with a “descriptor” word that indicates how

we think the words are related. We show this for the “all words” case (see Table 5.3 and also for the “election” case (see Table 5.4. The other words cases show similarly cohesive results.

“seasonal”	New, short, home, live, thanksgiving, autumn, posted, ms, sharp, us
“sports”	New, biggest, coach, live, coal, says, league, home, v, posted
“transformation”	Feel, read, change, affected, climate, new, like, shape, said, scene
“Directions; england”	England, elections, ms, north, v, wales, read, east, mp, oxford
“Political - media”	Nations, presidential, video, democratic nominee, image, trumps, via, post, propaganda
“Entertainment; negotiations”	J, trump, deal, games, drama, league, trade, theme, tackle, premier
“corruption”	Peoples, corrupt, media, theres, evil, mainstream, thats, source, terrorists, cant
“References; citations”	Article, posted, twitter, related, translated, articles, originally, source, change, loading
“spokespeople”	Read, said, twitter, live, election, spokesman, ms, phone, mp, spokeswoman
“evolving”	Team, played, read, games, last, growth, said, live, year, transition,

“References”	moved, referring, readers, referred, convicted, “may, understanding, flag, reference, author
“Democrat/politics	democratic, democrats, nominee, presidential, party, campaign, peoples, candidate, democrat, media
“numbers”	four, five, prospect, turned, announcement, next, drawn, running, three, demonstrate
“Impeded”	twitter, made, four, denied, decisions, declined, way, years, struggled, past,
“Political issues”	gave, cost, risk, edge, new, says, climate, live, jobs, questions
“Measurement”	greater, wider, freedom, spokeswoman, genuine, range, start, first, autumn, new,
“challenge”	difficult, challenge, court, performance, sales, guardian, autumn, high, challenging, ban
“Taking over”	reported, “emails, “soviet, “ten, “august, “observed, “hacked, “cant, seized
“Timespan”	year, last, twitter, next, friend, miles, friends, week, early, three
“Possibilities”	likely, willing, unlikely, less, keen, could, would, optimistic, try, war