FAKE NEWS IDENTIFICATION
CS 229: MACHINE LEARNING : GROUP 621

Sohan Mone
Department of Civil Engineering
Stanford University
sohanm@stanford.edu

Devyani Choudhary
Department of Civil Engineering
Stanford University
devyani@stanford.edu

Ayush Singhania
Department of Civil Engineering
Stanford University
ayushs@stanford.edu

ABSTRACT

Due to recent events in American Politics, fake news, or maliciously-fabricated media has taken a central role in American political discourse. A litany of verticals, spanning national security, education and social media are currently scrambling to find better ways to tag and identify fake news with the goal of protecting the public from misinformation. Our goal is to develop a reliable model that classifies a given news article as either *fake* or *true*. Our model is designed to emulate the functionality of the *BS Detector*, a popular extension for Chrome that automatically flags articles, websites and content as *BS*. Our simple models give us accuracy of up to 82%.

1 INTRODUCTION

With the advent of technology, information is free for everyone. This is an advancement in human history, but at the same time it blurs the line between true media and maliciously fabricated generated media. A freely available tool to verify the trustworthiness of a news is needed to filter the information we receive everyday. With this motivation, through building our knowledge of python and machine learning, we worked on our final project. Given the text of a news article and it's headline as input, we have developed an algorithm that can litigate the difference between the fake and true news with an 83 percent accuracy. This project can consists of numerous binary classification (true/fake) algorithm. We use a combination of python and MATLAB both to create our model. Taking 8071 true news samples (New York Times, CNN, BBC), and 4094 fake news samples, we convert our words into numbers using NLP in python. Having this big dataset, we pre-processed and cleaned the data and tested it using our Naive Bayes, SVM, logistic regression. Based on the identification of indicative tokens through Naive Bayes, we used a 1 layer deep neural network, and 2 layer deep neural network for fake news identification.

2 RELATED WORK

2.1 TEXT PROCESSING

Text processing has evolved from simple tokenization to contextual clustering with time. The most common approach, simply tokenizing the data and and using classification algorithms is computationally cheap in preprocessing, but increases the feature space significantly (Mihalcea & Strapparava, 2009). Centering resonance analysis (CRA), is one such method that identifies the key nodes (dominant words), and identifies the most important words correlated with the node (Papacharissi & Oliveira, 2012). For our project we use simple tokenization with removal of stop words, and also take into account lemmatization. This served a middle ground of the two approaches described above. The context of the words, however, is not taken into account. There are third party tools like Stanford GloVe (Jeffrey Pennington, Richard Socher, & Christopher D. Manning, 2014), that help with creating contextual word tokens.

2.2 MACHINE LEARNING TOOLS

The simplest and most popular classification algorithms are Naive Bayes, and Support Vector Machines (SVM). Naive Bayes classifier used with Laplace smoothing helps understand the basic nature of given data (Oraby et al., 2015), and we use this method to identify the causal features in our project. With SVM, experimenting with different kernels such as polynomial kernel, RBF kernel function, sigmoid kernel, and gaussian kernel is an option (Zhang et al., 2012). The above mentioned methods are both helpful for classification, and so we used these methods to analyze which features are most causal. Further up, we play with deep learning activation functions to obtain our classification model.

3 Dataset and Features

We gathered our fake news data of 4094 samples (headlines+body) from $Kaggle$[7]. This fake data was obtained from a chrome extension called *BS Detector* that allows chrome users to tag a news article as *BS* as they are reading the news. We obtained our true news data of 8071 samples from $Kaggle$[8] as well. These are news articles from trusted news sources. The data obtained was in excel form, and needed a lot of manual, as well as syntactic preprocessing. We have a total of 12,165 samples which we distributed to train: validation: test set in ratio 70:25:5. Each sample corresponds to a news article headline and body. We used NLTK in python to tokenize the body and title. Removing the stop-words (referencing the nltk stop-word list), we lemmatized the rest of the data. To obtain the labeled sentence list for a particular course, we apply the following processing steps:

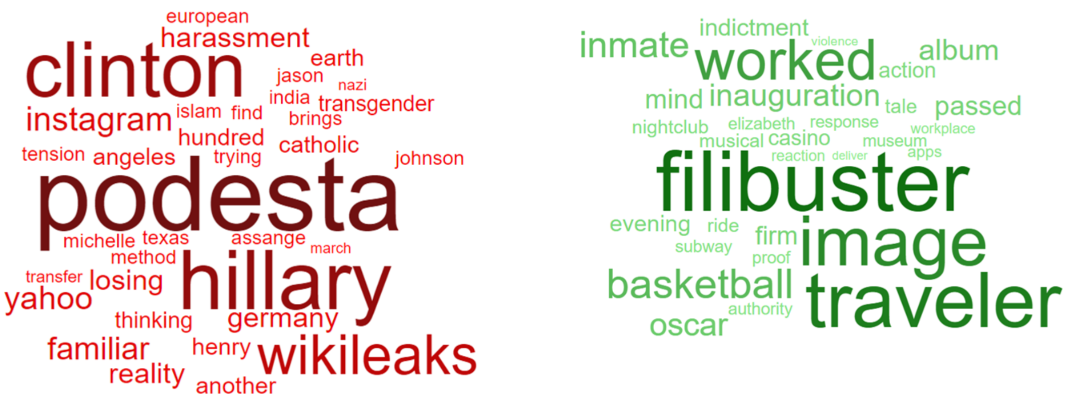
1. Tokenize the body and headline with the Punkt statement tokenizer from the NLTK NLP library. This tokenizer runs an unsupervised machine learning algorithm pre-trained on a general English corpus, and can distinguish between scentence punctuation marks, and position of words in a statement.

2. Tokenize words with our algorithm, and take care of lemmatization.

3. Tag each sample with the tokens obtained from entire headline set, and body set.

For titles, we kept only the tokens that had a frequency more than 10 over the entire title dataset, and for body, we kept only the tokens that had a frequency of more than 200 over the entire dataset. This leaves us with a total of 5261 tokens. In order to remove the symbols that came along with data (for example, xd,*,dxl), we only kept tokens with string size greater than 3.

We used Naive Bayes to obtain the tokens with high posterior probability, which we then used for deep learning and logistic regression.



(a) Fake news tokens

(b) True news tokens

Figure 1: Causal tokens based on posterior probability

4 Methods

Two models were developed independently in order to achieve our desired objective: an Average-Hypothesis model, and a Neural Network. In this section, the methodology and theory behind each model will be discussed in detail.

4.1 AVERAGE-HYPOTHESIS MODEL

Our average hypothesis model combines the hypotheses obtained from Nave Bayes, Logistic Regression and SVM by averaging the output probabilities obtained from each model. The aim of averaging is to obtain a model that is less susceptible to over-fitting compared to a model that only uses one of the constituent methods. Given our large feature set consisting of 5,078 features, certain judgment calls were used and validated to integrate this models. Within the Average-Hypothesis model, the Nave Bayes algorithm (which includes Laplace smoothing) and SVM algorithm was run using all 5,078 tokens, while Logistic Regression was performed using only the 20 tokens that were determined to be most indicative to a sample's classification. The following sections delineates the theory used in our implementations of these three learning algorithms.

4.1.1 NAIVE BAYES WITH LAPLACE SMOOTHING

Given the size of our feature space, we determined that Naive Bayes was an appropriate method to begin our analysis. Drawing from the lecture notes, the maximum-likelihood estimates for the model parameters are:

$$\phi_{j|y=1} = \frac{\sum_{i=1}^{m} 1\{x_j^{(i)} = 1 \wedge y_j^{(i)} = 1\} + 1}{\sum_{i=1}^{m} 1\{y_j^{(i)} = 1\} + 2}; \quad \phi_{j|y=0} = \frac{\sum_{i=1}^{m} 1\{x_j^{(i)} = 1 \wedge y_j^{(i)} = 0\} + 1}{\sum_{i=1}^{m} 1\{y_j^{(i)} = 0\} + 2}$$

Using our Naive Bayes algorithm, we identified the top-k tokens that were found to be the most indicative on the classification of the example. This was computed by finding the k/2 tokens which have the highest posterior probability of being in fake news, and the k/2 tokens with the lowest posterior probability of being in fake news. The following expression was used to rank the tokens by their indication of fake news:

$$Token\ \ Rank = \frac{exp(\phi_{j|y=1})}{exp(\phi_{j|y=0})}$$

The k/2 most indicative tokens for each class was used to form a new feature space for our Logistic Regression model. These tokens were also examined heuristically to ensure they pass the eye-test given our team's knowledge of contemporaneous fake news.

4.1.2 SVM

Due to it's robustness, a support vector machine (SVM) was used as the second algorithm in our Average-Hypothesis model. The SVM algorithm used uses a hinge loss that seeks to maximize the margin between the two classes of data. The SVM algorithm uses a second-order Gauss kernel that operates on the full 5078 token feature space. The expression for this kernel is given by the following expression:

$$G(x; \sigma) = \frac{1}{\sqrt[2]{2\pi\sigma}} exp(\frac{x^2}{2\sigma^2})$$

Note that this expression is provided for the 1-D case. In retrospect, the selection of this high-order kernel seems rather naive, since it may have caused the SVM model to over fit the training set.

4.1.3 LOGISTIC REGRESSION

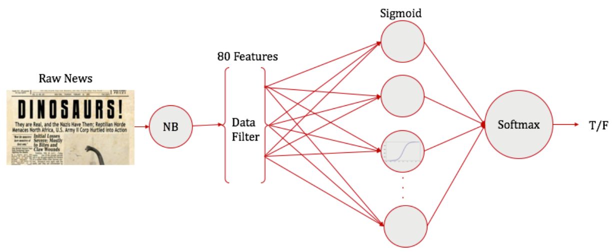
Due to its simplicity and elegance, Logistic Regression (LR) was used as the third algorithm within the Average-Hypothesis model. The LR model uses gradient descent to converge onto the optimal set of weights (θ) for the training set. Where J is the loss function and alpha is the learning rate. For our model, the hypothes used is the sigmoid function:

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + exp(-\theta^T x)}$$

4.2 Neural Network

A one-layered neural network model was used on the 80 tokens identified to be most causal to a sources classification. The hidden layer neurons uses sigmoid activation functon and, the output layer uses the softmax activation.

Also, ReLU and tanh function were tested for the activation function of the hidden layer. Although the results from sigmoid are not good enough to be used as compared to other models discussed above, it was better than ReLU and tanh activation function.



5 Discussion and Results

5.1 Final Results

Using the average hypothesis model, we observed an accuracy of 83 percent on our training set. As mentioned in the methods section, the average hypothesis model is constituted by three separate learning algorithms: Naive Bayes, SVM and Logistic Regression. Due to the poor performance of Logistic Regression on the dev. set, it's hypothesis was omitted from the average accuracy model.

Feature Set	Naïve Bayes Accuracy	SVM Accuracy	Logistic Regression Accuracy	Averaged Accuracy
Body + Title	0.8316	0.8165	0.6588	0.8302
Body	0.8253	0.8165	0.6588	0.8294
Title	0.6805	0.6624	0.6657	0.6805

5.2 Discussion

5.2.1 Average-Hypothesis Method

The first algorithm used for classification was Naive Bayes (with Laplace smoothing), where no hyperparameter was required. This helped to set a reference point for further analysis. As indicated before, the top k indicative tokens were recovered by using Naive Bayes. These tokens were then used in the Logistic Regression algorithm and the neural network. Naive Bayes was followed by SVM model where we selected the normalizing parameter (τ) as 12. The model was trained starting from a smaller value of $\tau = 4$, because the larger the τ the larger number of features influencing the output. However, the model did not converge for any τ smaller than 12. Another hyper parameter used in SVM was Lagrange multiplier (λ). A λ value of 1/64 was used which gave the best result. Any value smaller than this was not converging.

Third model was Logistic Regression, where the only parameter used was learning rate (α). The learning rate between 5 to 12 was giving same convergence point, hence value of 10 was used. However, this model resulted in exceptionally low accuracy so it was weighted zero in our average hypothesis model. This also prompted us to try neural network on the 40 most causal words.

Initially, in the average hypothesis model, all the three models discussed above - Naive Bayes, SVM, and Logistic Regression - were equally weighted. After discovering the poor performance of our Logistic Regression algorithm, this algorithm was not considered as part of the average hypothesis.

To determine the sensitivity of our model to sample size, we generated a loss curve for the model.

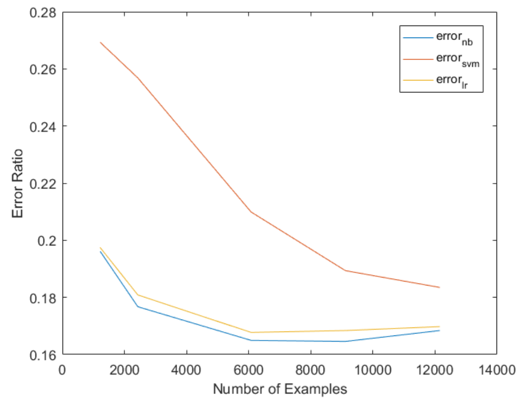
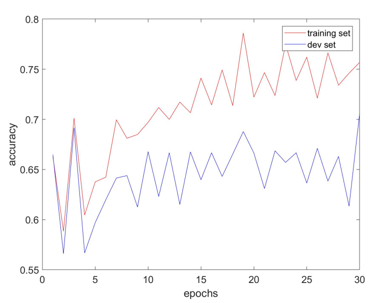


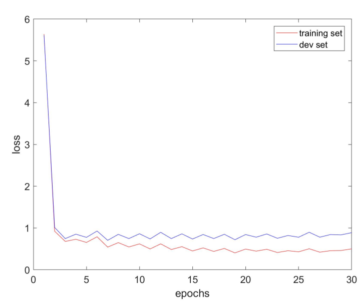
Figure 2: Naive Bayes, SVM, Average-Hypothesis Learning Curve

5.2.2 Neural Network

The final model considered was Neural Network, where number of hidden layers, depth of hidden layer (neurons), learning rate, regularization parameter (λ), batch size, number of features, and number of epoch were selected based on different analysis performed. Initially a 2 hidden layers neural network was used with a 1000 neurons in first hidden layer and 50 neurons in the second hidden layer was used. This resulted in exponentially high number of parameters which ended up being higher than the training set. To encounter this problem the model was reduced to single layer and also the depth of the hidden layer (neuron) was reduced to around 300. In this case it was found that the training accuracy reached around 75 % however the dev accuracy was 66 % which is the base line accuracy of the model. Therefore, the number of parameters were reduced by reducing the neurons to 150 and also the value of regularization parameter was increased which resulted in less variance. However, the accuracy was as low as 70%. Hence it was concluded that the training set was only about 8500 samples and it was found that the neural network model was not the best method for training the model.



(a) $\lambda = 0.0005, \alpha = 5$



(b) $\lambda = 0.0005, \alpha = 5$

Table 1: Variation in loss and accuracy with hyperparameters for NN

	True	Fake
True	57%	8.6%
Fake	8.2%	26.1%

	True	Fake
True	59.8%	12.8%
Fake	5.4%	21.8%

	True	Fake
True	1.4%	65%
Fake	32%	1.3%

Table 2: (a) Naive Bayes (b) SVM (c)NN

As we see, the false positives are higher in SVM, and are highest in NN. We need to try to reduce them further by adding constraints in future.

6 FUTURE WORK

A lot of our results circle back to the need for acquiring more data. Generally speaking, simple algorithms perform better on less (less variant) data. Since we had less data, SVM and Naive Bayes outperformed Neural Networks, and Logistic Regression did not perform well. Given enough time to acquire more fake news data, and gain experience in python, we will try to better process the data using n-grams, and revisit our deep-learning algorithm. We tried using our own codes for the project, and the algorithms were relatively slow. To tweak all knobs of various algorithms, we shall use available robust packages in the future.

7 CONTRIBUTIONS

Overall, work was well-distributed between the team members throughout the project. Since he had experience with Python, Ayush lead the effort to process of the data using NLTK. Devyani modified these algorithms to work for a larger set of examples. Sohan helped debug the data processing algorithms. For modeling, Sohan and Ayush worked on the Average-Hypothesis model which Devayani debugged. The neural Network was developed by Ayush and Devyani and the poster was formatted by Sohan. All team members were involved in the preparation of this report.

REFERENCES

[1] Papacharissi, Z. & Oliveira, M. (2012). The Rhythms of News Storytelling on Egypt. Journal of Communication. 62. pp. 266–282.

[2] Mihalcea, R. & Strapparava, C. (2009). The Lie Detector: Explorations in the Automatic Recognition of Deceptive Language. Proceedings of the ACL–IJCNLP Conference Short Papers, pp. 309–312.

[3] Chen, Y., Conroy, N. J., & Rubin, V. L. (2015). News in an Online World: The Need for an Automatic Crap Detector. In The Proceedings of the Association for Information Science and Technology Annual Meeting (ASIST2015), Nov. 6–0, St. Louis.

[4] Jeffrey Pennington, Richard Socher, & Christopher D. Manning. (2014). GloVe: Global Vectors for Word Representation.

[5]Zhang, H., Fan, Z., Zeng, J. & Liu, Q. (2012). An Improving Deception Detection Method in Computer-Mediated Communication. Journal of Networks, 7 (11)

[6]Conroy, N. J., Rubin, V. L., Chen, Y. (2015). Automatic deception detection: Methods for finding fake news. Proceedings of the Association for Information Science and Technology, 52(1), 1–4.

[7] kaggle Fake News NLP Stuff. https://www.kaggle.com/rksriram312/fake-news-nlp-stuff/notebook.

[8] kaggle All the news .https://www.kaggle.com/snapcrack/all-the-news.