FakeNewsCNN: A Convolutional Neural Network for Fake News Detection

Muhammad Haseeb Ahmad, Carlos Mattoso, Surya Tamraparni
{A53241491, A53236176, A53254448}
{m8ahmad,climamat,stamrapa}@eng.ucsd.edu

***Abstract*—Infiltration of social networks into every corner of the society has resulted in a facility that allows rapid spread of information. Consequently, a serious need for detecting and containing spread of false and potentially harmful information has been created. In this paper, we seek to build a classification model capable of detecting if a given news article is fake or real.The challenges associated with building a fake news detection system are exacerbated due to the unstructured nature of textual data. We take two approaches 1) Metadata driven, where we train a logistic regression model using only meta-features of news articles, and 2) Content driven, where we use the content of the text to classify articles. Both approaches involve extensive text processing and feature engineering. To bypass these labor-intensive requirements, we proceed to train a model that uses raw text to detect fake news. It is observed that by using a simple Convolutional Neural Network, it is possible to attain the accuracy levels obtained by using a Text Frequency-Inverse Document Frequency model without having to perform any complex pre-processing of the text. It was also observed that using deeper neural nets further enhances the performance.**

Keywords: Fake news detection, Natural Language Processing, Logistic Regression, TF-IDF, Convolutional Neural Networks.

I. Introduction

Over the past decade, social networking services have made such deep inroads into the societal structure that the World Wide Web has effectively been transformed from a web of hypermedia articles to a web of real people. This has directly resulted in an arrangement where information spreads so fast that it quickly becomes near impossible, yet equally important to contain the spread of false information. Individuals with malicious intent have recently leveraged to great success that friction-less nature of the web to spread fake news, for both personal and political gains. This was particularly evident during the 2016 U.S. Presidential election, the prevalence of fake news during which has led to increasing public pressure on technology giants [1]. Such pressure has even led to a commitment by Facebook to increase its content filtering team by 50% [2].

Some of the key challenges of this class of problems are caused by the unstructured nature of text. The unconstrained nature of the domain set – news articles generated by any source – adds to the difficulty. If the domain of the text was known, subject knowledge could be used to make reasonable assumptions. Save for human annotators, which are costly, we need to rely on Machine Learning models to do this job for us. For example, if the data was insurance specific, numbers following the words "claim amount of..." can be used to deduce the claim.

We begin with an analysis of the news articles in the dataset, and simultaneously extract meta features. They will subsequently be used to train a logistic regression model for one of our approaches, i.e. the one without considering the content of the text. The results, specifically the weights obtained by this process are used to develop an insight about the nature of fake news sources. We then proceed to train a Term Frequency-Inverse Document Frequency (TF-IDF) based logistic regressor model. This forms the second approach adopted in which only the content of the text is considered.

After examination of the two approaches, we apply CNNs to combine the benefits of the two. This hybrid approach takes into account both, the structural aspects of the data and the actual content. Description of all approaches is followed by an evaluation of their results and inference thereof.

The authors are graduate students at the Department of Computer Science and Engineering, University of California – San Diego, La Jolla, CA, 92093 USA

December 3rd, 2017

II. Related work

Our work was motivated by a Medium article [4] written by data scientist Yunus Gene. He employed multiple traditional methodologies to discriminate fake news from real news, ultimately settling on a TF-IDF logistic regressor that achieved 95% accuracy. We extend his work by doing further pre-processing of the news article text before training our baseline logistic regressor. Furthermore, the text processing that results in higher accuracy is observed to be labor-intensive and requires knowledge of the nature of fake versus real news. This motivated us to explore Neural Networks (NNs), given their recent successes in diverse applications, from image captioning [5] to text translation [6].

We also rely on work on the nature of fake news [7]. In that paper the authors contribute a set of criteria that should hold in predictive tasks on fake news. We adhered to such criteria in constructing our dataset, in particular with regards to: homogeneity in length, topic and language.

A paper from Stanford's graduate level Natural Language Processing course [10] has also been referred to. The author used the same fake news dataset, but a different dataset for real news. He experimented with multiple deep learning methods, ultimately achieving an F1 score of 84% with his best model with respect to that metric. We note that his real news articles are from a different timeframe than fake news, so that is likely to have impacted the results.

III. DATASET AND ANALYSIS

For fake news, we used a dataset available on Kaggle that contains 12,999 news articles and associated metadata [8], scraped from 244 websites from October to November 2016. Real news datasets were scraped from the websites of The Guardian and The New York Times. These articles range from March 2016 to December 2016. We restricted the scraped articles to sections related to United States news about politics, the economy, and business, in order to achieve a thematically homogeneous set of articles whose main topic was the ongoing Presidential Election. As a consequence, the model is limited in knowledge to such themes and textual constructs, and would likely fail to capture topics and expressions not belonging to these categories and timeframe. In all, the dataset consists of 44,666 news articles, with 12,999 labelled fake and 31,667 labelled real.

We operate under the assumption that the articles contained in The Guardian and the New York Times are real. We also assume that the only available parts of articles across websites are their headline and body text. Hence, we choose to utilize only those features, and discard several other fields such as date, writer, category etc.

For the purposes of this paper we refer to a definition of fake news from a recent study by Stanford [3]: “We define ‘fake news’ to be news articles that are intentionally and verifiably false, and could mislead readers.” This definition applies to the dataset we used and is important in that it accounts for the intent of the news writer to present false information.

A. Data processing

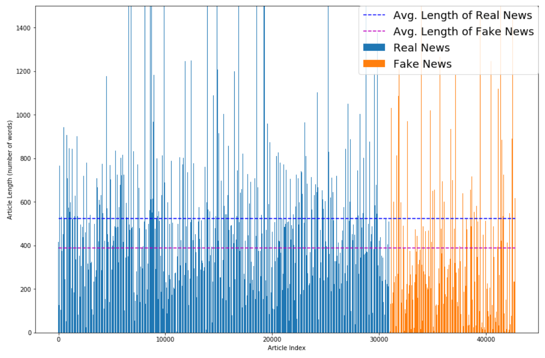
To improve the generalization and avoid overfitting in the logistic regression models, the lemma of each word in the text is extracted. Each article is filtered for stopwords to increase content density and each word in the resulting text is stored as a separate token. Furthermore, tokens containing punctuation only are removed and their counts recorded.

B. Data analysis

The meta-feature extraction process uses general guesses about the nature of the sources of news articles. These guesses implicitly undergo validation in logistic regression. The rationale behind these guesses and consequent features are presented below:

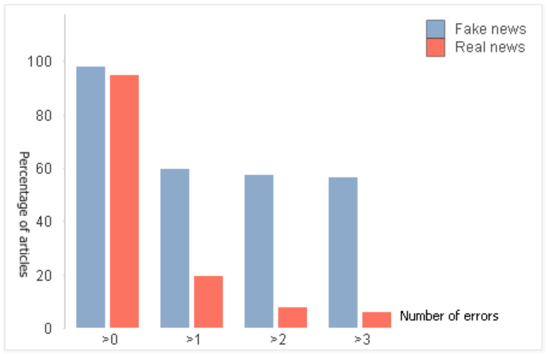
1) Article length: Real news articles are likely to contain observations or reasons leading to inferences while fake news would lack such descriptions. This would cause fake news to be smaller in length.

Fig. 1: Article length per class.



2) Spelling errors: Real news is likely to undergo multiple rounds of verification and checks before they are published, in contrast to fake news. That is likely to incur a greater number of spelling errors in fake news.

Fig. 2: Spelling errors per class.



3) Average word length: Fake news authors tend to resort to excessive use of thesaurus to make the content seem sophisticated, resulting in a greater average word length than that observed in real news.

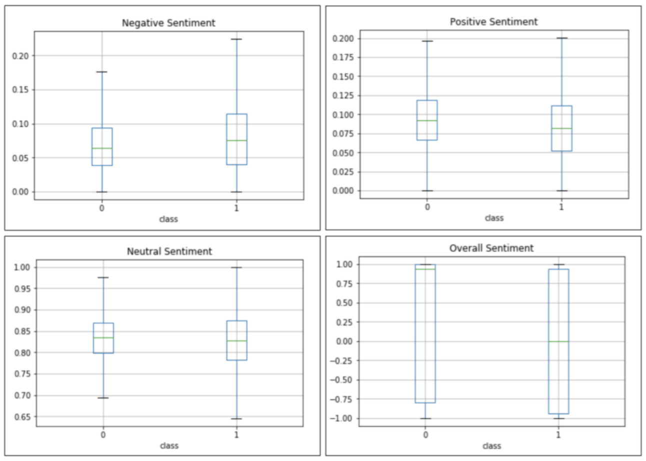
TABLE I: Summary statistics of word length per class

	Fake News	Real News
Average	4.805	4.603
Std. Deviation	1.270	0.440

4) Count of exclamation mark: As we browsed through the samples, we found that several fake news articles made grammatically inappropriate use of the exclamation symbol (!). This can be explained by the absence of a requirement of formality in fake news, as opposed to real news. It was found that 3% of real news articles contained exclamation marks, while roughly 10% of the fake articles contained at least 3 exclamation marks.

5) Sentiment analysis: To extract sentiment from the texts we used Python’s Natural Language Processing Toolkit (nltk) implementation of VADER - *Valence Aware Dictionary and sEntiment Reasoner* [11]. This is a rule based system for sentiment analysis. It yields four metrics: positiveness, negativeness, neutrality and overall sentiment (a measure from -1 to 1.) A visualization of the distributions of each sentiment is presented. Note how fake news are more negative on average, whereas real news are more positive.

Fig. 3: Nature of sentiment and its intensity per class.



6) Other features: We also extracted:

- Number of adjectives used in the article.
- Number of digits used in the article.
- Number of superlatives used in the article.
- Number of words with all capital letters.
- Average length of sentences.

IV. Methodology

A majority class classifier was used as the baseline model. The majority class for our dataset used was real news; all articles were therefore classified as real. We will now describe the various approaches taken for classifying articles:

A. Approach I - Meta-feature Logistic Regression

Using the features extracted during data analysis (as mentioned in the previous section), a balanced logistic regression model with L2-regularization and inverse regularization parameter of 100 was trained on one half of the dataset and validated on the remainder.

We will briefly describe how logistic regression works. Logistic regression is employed in supervised classification tasks. It transforms linear outcomes to the range $[0, 1]$ using the Sigmoid function:

$$\sigma(x) = \frac{1}{1 + e^{\theta \cdot x}}$$

Where θ denotes the weights for the model and x denotes the feature vector for each data point. The log loss function to be minimized, is given by:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} y^i \log(h_\theta(x^i)) + (1-y^i)\log(1-h_\theta(x^i)) + \lambda\theta^\top\theta$$

Where m is the number of data points, y is the actual label and λ is the regularization constant. The corresponding update rule for θ is defined as:

$$\theta_{j,t+1} = \theta_{j,t} - \alpha * \frac{\partial}{\partial\theta_j} J(\theta)$$

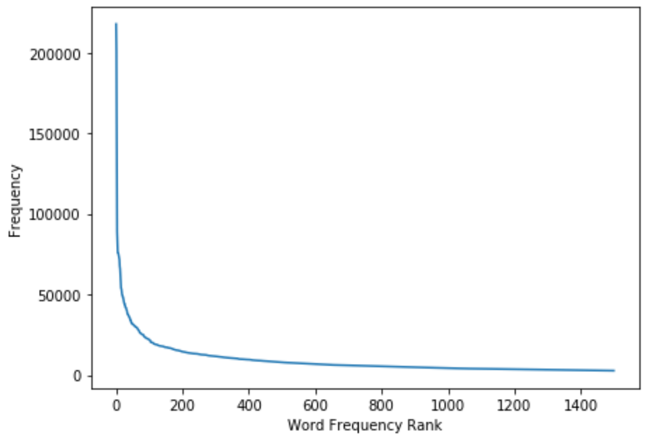
Where α denotes the learning rate, which may be arbitrarily set or determined sarcastically. It controls the convergence rate for logistic regression.

For classifying the articles, a Sigmoid activation of greater than or equal to 0.5 was labelled as fake news, and less than 0.5 as real.

The above procedure was replicated for other logistic regression variations in the next two approaches.

B. Approach II a: Logistic regression using Word Counts

Fig. 4: Word frequencies.



An examination of frequency of the words in the corpus indicated that words which appear less frequently than the 1000 most common words have insignificant frequencies. It is postulated that such words contain information unique to the articles containing them and hence would not capture fakeness or realness of the article. With this, each article is represented as a vector of counts for the 1000 most common words, i.e. the feature vector used in logistic regression.

C. Approach II b: Logistic regression using TF-IDF

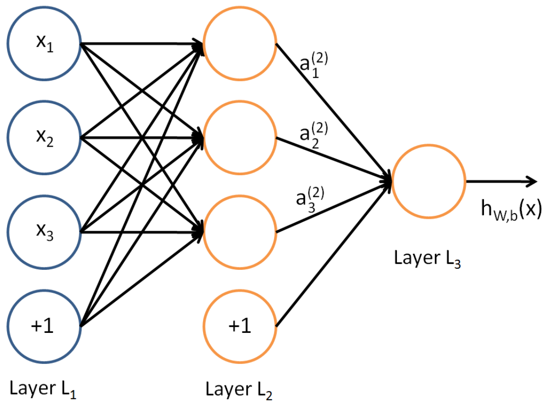
The TF-IDF model captures information in terms of frequency of word occurrences weighted by their contribution to the context of the article or its "influence", which is measured as Inverse Document Frequency (IDF):

$$idf(t, D) = log\frac{N}{|\{d \epsilon D : t \epsilon d\}|}$$

Where t is the word, d is the current document, D denotes all documents in the corpus. To get the TF-IDF of a given word in a specific article, multiply the frequency of that word in that article by its corresponding IDF value. Effectively, each text is represented as a vector of products of counts in that article of the top-1000 words and their IDF weights. A logistic regressor is trained using the obtained vectors on the text of the training set.

D. Approach III: Convolutional Neural Network

Fig. 5: A neural network with one hidden layer.



Taken from [13].

1) Priors: A neural network consists of a stack of layers of neurons, which are small computational units that carry some value (like variables in programming languages.) Each neuron is connected to every other neuron in the following layer, as depicted in figure 5. These connections are associated to a set of weights, which enable a weighted sum to be carried over the previous layer to set the neurons of the subsequent layer. These connections establish what is called fully connected layers in a neural network. They can be formally defined as:

$$z = (xW_l + b)$$

$$h_{l+1} = f(z)$$

Where x is the matrix of inputs, W_l if the matrix of weights for the transition from layer l to layer $l+1$. The neurons in layer $l+1$ are then set to the resulting activation of z.

The f function is called an activation function. It controls whether or not the neuron is triggered by the incoming signals. Moreover, in the case of convolutional neural networks, convolutions are performed over the input. For the case of 1-D convolution the formula is the same as above, i.e. an inner product of the input by a matrix of weights, typically called a kernel.

In our solution we employed two activation functions, Rectified Linear Units (ReLU) and softmax, defined as follows:

$$f_{ReLU}(z) = max(0, z)$$

Where z is the input to a neuron.

$$f_{softmax}(z)_j = \frac{e^z_j}{\sum_k = 1^K e^z_k}$$

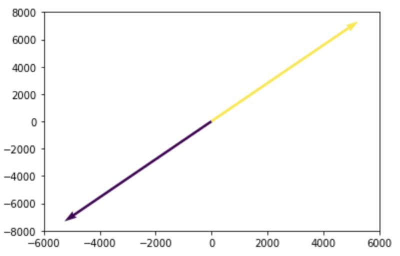
Where z is the vector of outputs, one per class, for a total of K classes. It yields a probability and is therefore applied on the last layer to get the probabilities of the classes. In our problem, $K = 2$.

An interested reader can study these concepts in greater detail at [13].

For the task of text classification at hand, we must first perform a simple pre-processing step of the text. The text must be tokenized and converted to a numerical representation, based on an index of the most common 25,000 words. For consistency, we truncate all texts to their first 500 words, and should they be shorter than that, the resulting vector representation is left-padded with zeroes. Aside of tokenizing the text, a requirement of the framework we employed, no further processing of the text is required of us.

Another feature of Neural Networks employed is that of embeddings. These are latent space representations trained jointly with the neural network over the words that make up our vocabulary. Each text is then represented as a 500x256 matrix of column-vector embeddings of all words which becomes the input to the network. This enables the model to discriminate data samples across classes. See in figure 6 a 2D t-SNE visualization of the embeddings of a positive (yellow) and negative (purple) examples.

Fig. 6: Embeddings of fake (yellow) and real (purple) news.



Finally, to avoid overfitting, we employ a strategy called dropout. For each step in training a set percentage of neurons is eliminated at random from the layer to which it is applied. In doing that, we force the neurons to actually learn meaningful patterns in the data distribution, instead of just memorizing the training data.

2) Approach III A: Convolutional Neural Network over Embeddings: The model we propose is a simple CNN with the following set of properties:

- An embedding layer of size 256; one embedding per word, with 500 words total [500x256]
- A stack of 1D-convolutional layers:
  - First with output of size 128.
  - Second with output of size 64.
  - Third with output of size 32.
  - Kernel of size 3 for all (i.e. we are using trigrams)
  - Activation function for all: ReLU

- A Fully Connected (FC) layer of size 256.
  - Activation function: ReLU
- A FC layer of size 2, one neuron per class.
  - Activation function: Softmax
- Further hyper-parameters:
  - Dropout: 0.2
  - Batch size: 32
  - Class weight: 1.3 to fake news, 1.0 to real news.
  - Optimizer: Adam
  - Default parameters otherwise.

The above hyperparameters were determined by tuning through a range of values to obtain those that gave the best results on the validation set.

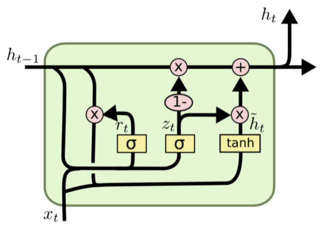
We used the Keras framework, with a TensorFlow backend to solve this task. These models were trained on a *p2.xlarge* instance on Amazon Web Services Elastic Computer 2. It is highly encouraged that the reader uses a GPU instance to replicate this work. We recommend training for just 2 epochs, to avoid overfitting to the training data, and making sure to use batches of size 32. Finally, use ReLU for the hidden layers, softmax for the output layer, and dropout of 0.2 on the hidden layers (to avoid overfitting.)

3) Approach III B: CNN with Gated Recursive Unit over Embeddings: We propose the following extension to the FakeNewsCNN model. It is a much more complex and deeper model than the one described in Approach III A, but achieves better results overall. It integrates a CNN with a Recurrent Neural Network using a Gated Recursive Unit (GRU). The latter is a type of time-sensitive Recurrent Neural Network (RNN). The formulation for RNNs is as follows:

$$h_t = (x_t W_x + h_{t-1} W_h + b)$$

The term t has been introduced to create a dependence between the network's output at time t and the input at time $t-1$. A GRU is an extension of this type:

Fig. 7: Schematic of a Gated Recurrent Unit.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure taken from [12].

The four fundamental equations for a GRU are presented above. Note how the output from time $t-1$, h_{t-1}, feeds into the computation over the input at time t, x_t. Moreover, the neuron now has gates that are activated based on the similarity between the current input and previous output. This enables the neuron to keep track of long-term relationships in the text, propagating previous knowledge or restarting the engine [12].

This extension has a more complex structure:

- An embedding layer of size 256; one embedding per word, with 500 words total [500x256]
- A stack of 1D-convolutional layers over the input:
  - Three concatenated layers, each of output 128.
    - Independent convolutions of kernel size 5 on each (i.e. 5-grams)
  - A convolution of output 256, kernel of size 5.
  - Final convoluton of output 128, kernel of size 5.
- Activation function: Rectified Linear Unit (ReLU)
- A parallel GRU over the input with default parameters.
- A stack of FC layers:
  - Input: Concatenation of the GRU output with the output of final convolutional layer, i.e. the one with size 128.
  - First FC with output of size 128.
  - Second FC with output of size 64.
  - Third FC with output of size 32.
  - Activation: ReLU (in each one.)
- An output FC layer of size 2:
  - Activation: Softmax
- Other hyper-parameters:
  - Dropout: 0.2
  - Batch size: 32
  - Class weight: 1.3 to fake news, 1.0 to real news.
  - Optimizer: Adam
  - Default parameters otherwise.

V. Evaluation

A. Metrics

The performance of each model is reported in terms of Accuracy, F1-score, False Positive Rate (FPR) and False Negative Rate (FNR).

$$F1 = \frac{2.0 * precision * recall}{precision + recall}$$

In fake news detection, both precision and recall are equally important. F1-score was accordingly chosen, as it combines the measures of both precision and recall.

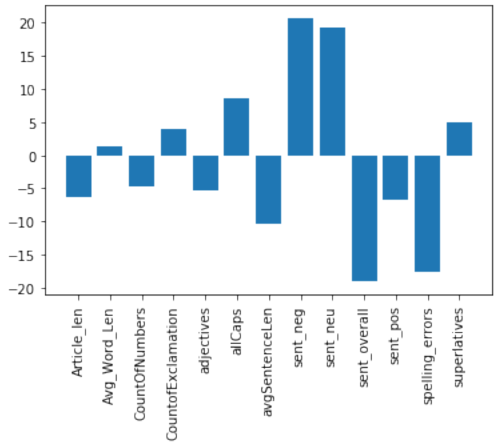
TABLE II: Model Performance

Model Name	**Accuracy**	**F1-Score**	**FPR**	**FNR**
Majority Class(Baseline)	0.723	NA	0.000	1.000
Logistic Regression (Metadata)	0.950	0.918	**0.005**	0.072
Logistic Regression (Word Counts)	**0.969**	0.945	0.028	0.041
Logistic Regression (TF-IDF)	**0.969**	**0.946**	0.021	0.055
FakeNewsCNN	0.954	0.925	0.053	**0.027**
FakeNewsGRUCNN	0.966	0.942	0.024	0.054

As can be seen, the more complex CNN was able to achieve results on par with the logistic regressor based on TF-IDF, falling short of it by just 0.3%! The simple CNN was actually superior in detecting fake news, yielding the smallest FNR among all models. These results highlight the predictive power of CNNs, which were able to achieve high accuracy and F1-score when being given just embeddings of the raw, tokenized text – requiring no feature engineering on our part.

B. Inferences from Logistic regression of Metadata

Fig. 8: Weights learned over metadata by logistic regressor.



From the weights obtained by logistic regression, the following inferences can be made about the general nature of real and fake news.

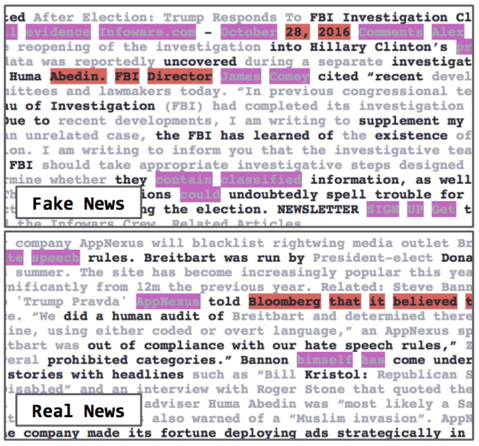
- Fake news articles are highly likely to contain a high negative sentiment and unlikely to contain positive sentiment.
- Fake news articles are likely to be shorter in length relative to real news, the possible explanation for which could be absence of research that usually builds content.
- Fake news is likely to contain longer words which could result from excessive use of thesaurus.
- Real news is likely to contain numbers as statements are usually justified by using numbers.
- Fake news is likely to contain greater number of (!) and all capital words, as real news are often required to use formal style.
- Counter to our assumption, spell errors feature has been assigned a negative weight, which can be interpreted as meaning real news are more likely to contain spell errors. This may be attributed to greater use of nouns and uncommon words in real news, than in fake news, that were wrongly labelled as errors in the feature construction process.

C. Inferences from FakeNewsCNN

Finally, we present a visualization of the models attention to specific locus of the text with respect to the output class i.e. the words it focused on to make its call. Words highlighted in red have the highest attention, those in magenta the second highest, those in bold the third highest, and finally those in gray the lowest. This visualization is applied to two instances of stories that mention Hillary Clinton and Huma Abedin.

D. Inferences from Logistic regression of Metadata

Fig. 9: CNN's attention when deciding on fake and real news.


```
ted After Election: Trump Responds To FBI Investigation Cl
l evidence Infowars.com - October 28, 2016 Comments Alex
 reopening of the investigation into Hillary Clinton's pr
lata was reportedly uncovered during a separate investigat
 Huma Abedin. FBI Director James Comey cited "recent devel
nittees and lawmakers today. "In previous congressional te
au of Investigation (FBI) had completed its investigation
Due to recent developments, I am writing to supplement my
an unrelated case, the FBI has learned of the existence of
on. I am writing to inform you that the investigative tea
 FBI should take appropriate investigative steps designed
rmine whether they contain classified information, as well
Th                    ions could undoubtedly spell trouble for
t                     ng the election. NEWSLETTER SIGN UP Get t
l the Infowars Crew. Related Articles
```

```
 company AppNexus will blacklist rightwing media outlet Br
te speech rules. Breitbart was run by President-elect Dona
 summer. The site has become increasingly popular this yea
nificantly from 12m the previous year. Related: Steve Bann
 'Trump Pravda' AppNexus told Bloomberg that it believed t
e. "We did a human audit of Breitbart and determined there
ine, using either coded or overt language," an AppNexus sp
tbart was out of compliance with our hate speech rules," Z
eral prohibited categories." Bannon himself has come under
stories with headlines such as "Bill Kristol: Republican S
isabled" and an interview with Roger Stone that quoted the
                      adviser Huma Abedin was "most likely a Sa
t                     also warned of a "Muslim invasion". AppN
e company made its fortune deploying ads strategically in
```

Fake news (top) vs Real news (bottom)

The model is observed to be focusing on patterns that substantiate claims being made. Additionally, it seems to look for an erudite style of writingnotice how its triggered by well-constructed sentences like "[...] out of compliance with out hate speech rules." The mention of trending topics (e.g. Hillary's email server), personalities or institutions also activates the model in either direction: Bloomberg, a reputed source not often mentioned in fake news, leans the model towards classifying the article as real news. On the other hand, a mention of Alex Jones or Huma Abedin in conjunction with the FBI sways the model strongly to a prediction of fake news. This goes to show that the model not only picked up stylistic patterns: it encoded some understanding of themes generally discussed in either class of articles. In line with this, observe that the highlighted portions could make up a succinct summary of the text. In essence, the model seems to have learned to skim over the text, and to identify the key pieces of information necessary to understand and classify the text.

VI. Conclusion

In our travails through the muddy waters of fake news we experimented with multiple models. Were we able to answer our motivating research question? Yes. We successfully trained Convolutional Neural Networks that achieve high accuracy and F1-score, finely balancing false negatives and false positives. The most important highlight is that these results were obtained without requiring any deep processing of the text or feature engineering dependent on expert knowledge of the material. Moreover, by visualizing the CNNs' activations, we were able to glean some insights on the structural and semantics patterns it learned to discriminate articles.

We identify the following limitations with our proposed CNN models:

- **Duped by style**: by visualizing our models its clear the network is paying attention particular structures in text that are typical in real news articles. In the examples presented the models attention is focused on claims that substantiate the statements made, as well as genuinely erudite writing style. A fake news article that adheres to these patterns will easily deceive our model.
- **Lack of knowledge base**: our model does not consult an up-to-date knowledge base to confirm the authenticity of the stories. It is limited to the to the themes and conversations revolving around the 2016 election. Fake news articles that dupe the model on style, but made factually incorrect claims wouldnt be caught by the model. NNs attached to a (differentiable) knowledge base is an active area of research which should be explored in the context of this problem [14].

Further work should explore extensions of our models that tackle these constraints in the context of fake news detection.

Acknowledgments

The authors would like to thank Prof. Julian McAuley for approachable but in-depth overview of Machine Learning throughout the quarter, in particular their applications to recommender systems. We'd also like to thank Yunus Gene for documenting the scraping process of the New York Times and the Guardian, which helped us in getting a sizeable dataset of real news. That enabled us to quickly get onto the fun part: munging data and training models! Finally, we thank the Teaching Assistants for their patience and support they've given us on Piazza throughout the quarter.

References

[1] Ewing, Philip. (November 2, 2017). Tough Questions, Hours Of Hearings But No Silver Bullet On Russian Tech Interference. *National Public Radio*. Retrieved from: https://www.npr.org

[2] Balakrishnan, Anita. (October 31, 2017). Facebook pledges to double its 10,000-person safety and security staff by end of 2018. *CNBC*. Retrieved from: https://www.cnbc.com

[3] Allcott, Hunt & Gentzkow, Matthew. (Spring 2017). Social Media and Fake News in the 2016 Election. *Journal of Economic Perspectives, Vol. 31, Number 2, pages 211236*.

[4] Gene, Yunus (May 23, 2017). Detecting Fake News With NLP. Retrivied from: https://medium.com/@Genyunus

[5] Karpathy, Andrej & Fei-Fei, Li. (2017). Deep Visual-Semantic Alignments for Generating Image Descriptions. *IEEE Trans. Pattern Anal. Mach. Intell., 39, 664-676*.

[6] Wu, Yonghui et al.(2016). Google's Neural Machine Translation System: Bridging the Gap between. *CoRR, Vol. abs/1609.08144*.

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

[8] Risdal, Megan. (2016). Getting Real about Fake News. Retrieved from: https://www.kaggle.com/mrisdal/fake-news

[9] Fan, Rong-En et al. (2008). LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research 9 1871-1874*. Updated in September 2017.

[10] Bajaj, Samir. (2017). The Pope Has a New Baby! Fake News Detection Using Deep Learning. Project for CS224n at Stanford University.

[11] Hutto, C.J. & Gilbert, Eric. (2015). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. *Proceedings of the 8th International Conference on Weblogs and Social Media*, ICWSM 2014.

[12] Olah, Christopher. (August 27, 2015). Understanding LSTM Networks. Retrieved from: http://colah.github.io/

[13] Ng, Andrew et al. Multi-Layer Neural Network. Retrieved on December 03, 2017 from: http://ufldl.stanford.edu/tutorial/supervised/MultiLayerNeuralNetworks/

[14] Kumar, Ankit et. al. (2015). Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. *CoRR, abs/1506.07285*.