CS230

Hate in Politics: Can Transfer Learning on Political Tweets Help Detect Hate Speech?

Sergio Charles
Department of Computer Science
Stanford University
sergioc1@stanford.edu

William Ellsworth
Department of Computer Science
Stanford University
willells@stanford.edu

Matthew Kolodner
Department of Computer Science
Stanford University
mkolod@stanford.edu

Bharath Raj Namboothiry
Department of Computer Science
Stanford University
brn@stanford.edu

1 Introduction

The Internet has revolutionized the way Americans process information. Social media platforms such as Twitter and Facebook have enabled discourse among citizens in a way never before possible. Often, these discussions are largely driven by partisan divide, where individuals sway towards predetermined schools of thought. Normally, it is healthy to discuss current events and analyze perspectives from all sides. However, social media has also allowed people to engage negatively with others and has provided a platform for hate speech. As the country has become increasingly polarized over certain issues, hate speech as become all the more common. Given the relationship between politically-charged rhetoric and hate speech, we seek to determine whether this relationship can be observed through transfer learning. In this paper, we study the effect of pre-training on partisan/non-partisan political tweets for a hate speech classification model. We hypothesize that pre-trained parameters from a partisan classification problem will perform better in hate speech classification than randomly initialized parameters.

2 Related work

1. Media Bias Detection using Deep Learning Libraries in Python: This blog post had a related, but distinct, goal: to extract political bias from news articles. One of the clever elements of this approach was the number of the optimizations the author used in order to produce the most accurate data results. Oversampling was used and hyperparameters (e.g. learning rate) were tuned.
2. Political Bias Analysis: This paper also had the related goal of identifying political bias in a given text. The authors primarily focused on using a LSTM network as a means of predicting implicit political bias. This group used US Congressional floor debates and statements on certain issues from presidential candidates in order to train their data. However, one of the weaknesses of this approach can be seen in their results, as the performance was only slightly better than a bag of words approach. With more training data, however, they believe they would have been able to produce better results. This use of LSTM is state-of-the-art in the sense that it is a new approach implemented by this group.
3. Quantifying News Media Bias through Crowdsourcing and Machine Learning Dataset: This group gathered data by both using crowdsourcing and by using a large-scale logistic

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

regression model in order to gather information on which articles were political news articles before being classified by issue and hand-assigned a certain bias value by distributing the results to a group of individuals.

4. Politic News Bias Detection Using Machine Learning: This paper discussed a different method for analyzing political bias. More specifically, they implemented a classifier using a Multilayer Perceptron model. Matrix representations were taken of sentences and classified using a classifier module, where articles were broken down into individual sentences, formatted, and sent to be classified. Although there was success in the training data, there was also a Google Chrome extension that struggled to be applicable due to most of the websites being classified as neutral and being unable to capture negative phrases and metaphors.
5. Political Ideology Detection Using Recursive Neural Networks: In this paper, Iyyer et al. created a recursive neural network (RvNN) in order to identify a given news article as politically biased. Supervised learning was then implemented using a regression and parameters were then optimized in order to minimize cross-entropy loss. One great benefit of this experiment is that it surpassed the bag-of-words method of bias detection.
6. DeepNews.AI: Detecting Political Bias: In this paper, Misra et al. used a recurrent neural network (RNN) and concurrent neural network (CNN) in order to identify political bias. In their results, they found that four-layer CNNs were more effective in detecting political bias than RNNs, although they performed on a similar level.
7. Measuring Ideological Proportions in Political Speeches: In this paper, Sim et al. used a hidden markov model in order to classify speeches as biased or not. In the model, Sim focused on using cue-lag ideological proportions, where cues represented ideology terms and lags correspond to lengths of non-cue words. Although the model had some success, it had difficulty due to always associating a given candidate's name with a political position.
8. Hate speech detection: Challenges and solutions: This paper focused on the hate-speech aspect of our model that we are building up. The paper discusses in depth the difficulty in identifying hate speech, but also proposes several sources that contain labeled data of hate speech. The paper also explores several options for building the model, such as Naive Bayes, BERT, and C-GRU models before selecting a Multi-view SVM(Support Machine Vector) model. The selected model had difficulty identifying hate in examples where there was implicit hate as well as examples where more context was needed. A F1 score of 71% was obtained for this model.

3 Dataset

This project required the inclusion of two datasets - social media posts labeled as partisan or non-partisan, and social media posts labeled as hate speech or neutral. Initially, we attempted to scrape our own data (for milestone 1 & 2). However, it often proved challenging to extract content from articles without including specific features of a given data source. As a solution, we have adapted the following online datasets for this final report:

1. Social Media Political Bias: We leveraged the social media political bias Kaggle dataset from Crowdflower's Data For Everyone Library, consisting of 5000 Twitter and Facebook posts from US Senators and American politicians, to classify speech as either partisan or neutral. Each message in the dataset was labeled by contributors according to audience (national or the politician's constituency), bias (neutral or partisan), and the purpose (informational, announcement, attack). The dataframe used a 26/74 split between partisan and neutral messages, respectively [9].
2. Twitter Hate Speech: We adapted data from Davidson, Warmsley, Macy, and Ingmar's 2017 AAAI paper on detecting hate speech tweets. This dataset also leveraged Crowdflower's workforce to label tweets as hate speech, mere offensive language, or neutral text. For the purposes of this project, we were concerned with only hate speech and neutral text, so we seleceted those portion of the dataset. At the end of sculpting, our working dataframe had 5593 tweets, with a 26/74 split between hate speech and neutral tweets, respectively [10].

Both datasets use a 90-10 split between training/test data.

4 Naive Bayes

For the baseline method, we used a Naive Bayes Bag of Words model with add-one Laplace smoothing as described in [9]. We represent the text as a bag of words: this method essentially extracts a vocabulary from a document without regard for the sequential relationships of words. We then train a Naive Bayes classifier on our bag of words, and use a Laplace smoothing. This model is an effective baseline that has been effectively used for sentiment analysis, which is a similar to to political bias analysis.
The Naive Bayes method of classification computes the probability of a given document D belonging to a given class C as follows:

$$P(C|D) = \frac{P(D|C) \times P(C)}{P(D)}$$

. However, since $P(D)$ is the same for each class, we can drop the $P(D)$ term. We perform operations in log space in order to avoid underflow and increase speed. We can determine $P(C)$ by computing $\frac{N_c}{N_{docs}}$. Similarly, we can compute $P(D|C)$ by measuring the frequencies of each word for each class. We also use add-one Laplace smoothing since likelihoods are multiplied together and a probability of 0 would cause problems.

5 Recurrent Neural Networks

We used recurrent neural networks with an architecture adapted from Yenter and Verma [11]. Our architecture consisted of 1) an embedding layer using pre-trained GloVe vectors, which fed into 2) four branches, each consisting of Conv-ReLu-MaxPool-Dropout-BatchNorm-RNN, and 3) a concatenation of the four branches, which finally fed into 4) a fully-connected layer with a sigmoid activation. The dropout rate was 0.5. Each conv layer used 128 filters and L2 regularization with a regularization factor of 0.01. The only difference between branches was the kernel size, which varied between 3, 5, 7, and 9 among the four branches. Two separate architectures were tested: one used a GRU for the "RNN" unit, while the other used an LSTM for the "RNN" unit. Both used 128 units. Note that the only significant differences between our architecture and that described in [10] are 1) the use of pre-trained 100-dimensional GloVe vectors, as opposed to the 32-dimensional embedding layer in [10] and 2) the use of GRU rather than LSTM in one architecture. This is intentional: our primary focus was not experimenting with architectures, but rather, studying the effects of transfer learning on the Twitter Hate Speech dataset. We implemented these architecture using the tf.keras API on TensorFlow 2. We trained our architectures for 15 epochs with early stopping on validation loss (patience of 2 epochs). We used a batch size of 32 and the Adam optimizer.

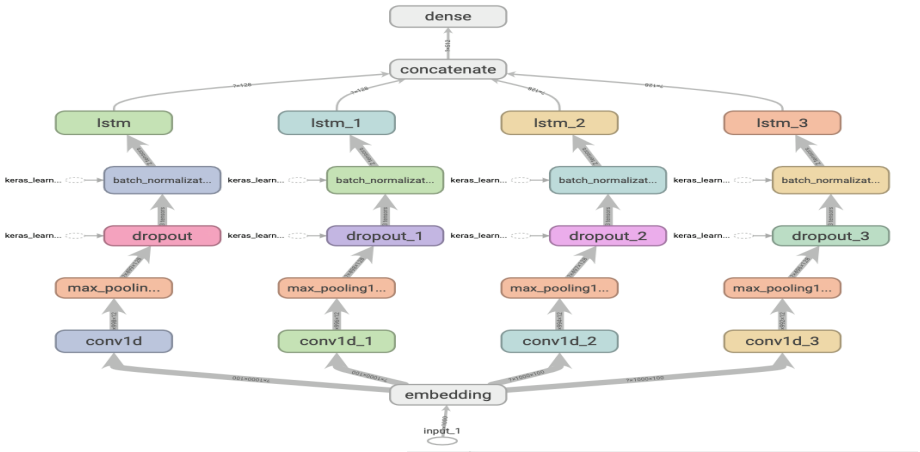

Figure 1: LSTM Model Architecture

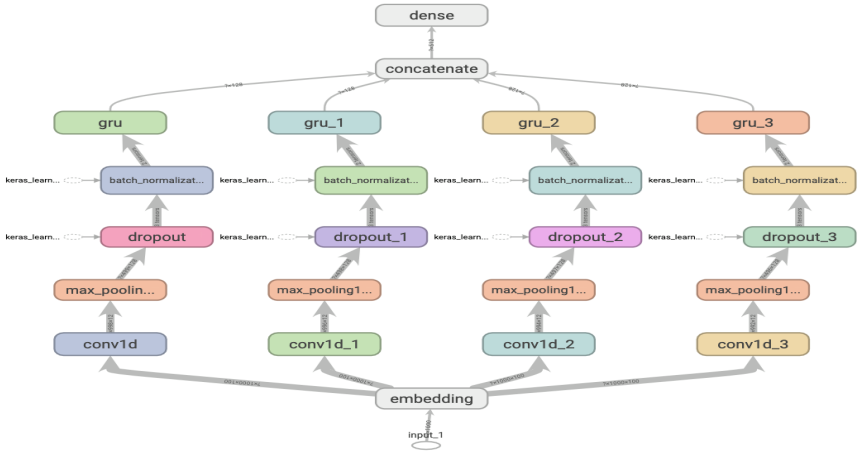

Figure 2: GRU Model Architecture

6 Results

After training and evaluating both bias detection methods, we arrived at the following train and validation accuracy metrics:

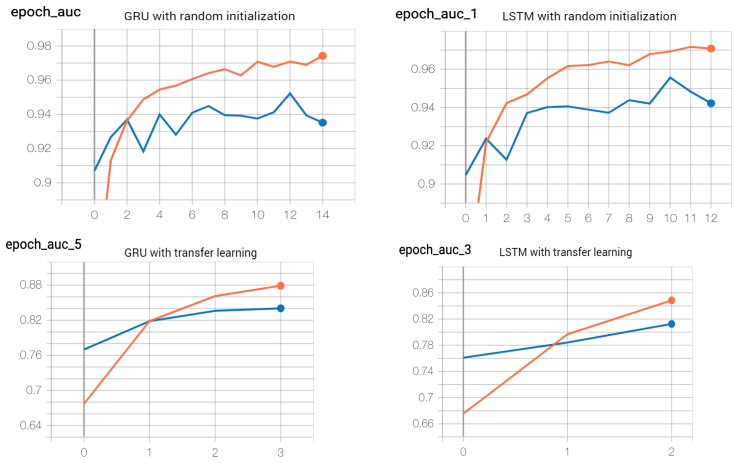

Figure 3: Plots of Epoch vs AUROC, where orange is train set and blue is validation set

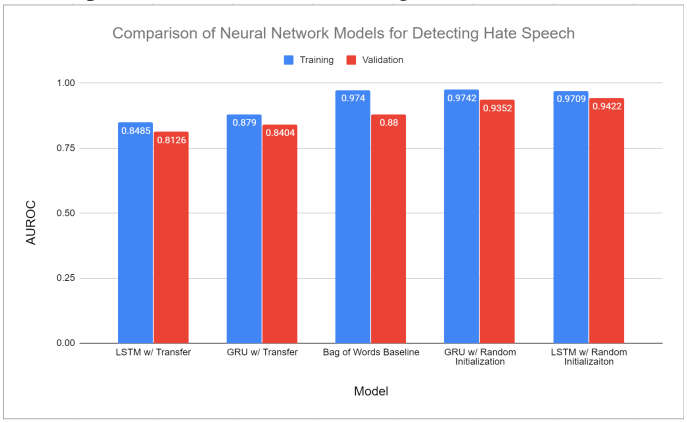

Figure 4: Comparison of AUROC based on model

We find that the best performing models are the recurrent neural nets using random initialization. Transfer learning from a political dataset seems to harm performance, with these models even underperforming the baseline Naive Bayes models.

7 Analysis and Conclusion

In this project, we sought to explore the relationship between partisan speech and hate speech on social media platforms. Our idea was to use train parameters on a partisan classification problem and use them as transfer learning parameters in a hate speech classification problem. Our hypothesis was that these transferred parameters would perform better than randomly initialized parameters when classifying hate speech, as partisan topics and ideologies can commonly manifest as hate speech. To test this, we ran experiments on GRU and LSTM networks, with and without transferred parameters. After running these networks on our hate speech data, we recorded AUROC to evaluate the relative efficacy of these networks. Our results show that, independent of network choice, transfer learning under-performed random initialization and our Bag-of-Words control. Moreover, results seemed largely independent of network choice (LSTM and GRU dev set performance differ by 3.4% with transfer learning and 0.7% without). From these results, we conclude that our hypothesis was false, and in fact there is not as strong of a transferable relationship between hate speech and partisan speech on social media.

We would acknowledge the following potential causes for error in our conclusions

1. As hate speech identification is a largely subjective, human based task, its hard to know where the Bayes optimal error lies. From our randomized parameter tests and our control model, we are led to believe that it lies somewhere above 97%. However, it is hard to reconcile this with the reality that classifying text as hate speech is not so simple. Our data was labeled using Crowdflower's aggregate opinions, and our AUROC is based on those labels. However, the transfer-trained networks that had a lower AUROC may not necessarily be performing "worse", but rather employing a different decision boundary that is less quick to identify a given tweet as hate speech.

2. All of our networks, transfer-trained and random, had a lower AUROC than the Bag-of-Words control. This seems to indicate that we have some sort of bottleneck on our networks, which we believe to be lack of data. Our hate speech corpus only had 5593 tweets, of which only 5034 were used in training. Neural networks' performance generally scales with the amount of available data. Had we used more, perhaps our results would have differed.

8 Future Work

When further investigating this problem space, we plan on doing the following:

1. Gathering/accessing more data to better train and evaluate our neural networks

2. Conducting a survey or some other human experiments to get a better sense of where Bayes error lies within a particular cultural communities (i.e. American college students).

3. Gathering/accessing social media posts labeled as conservative/liberal/neutral rather than simply partisan/non-partisan, and explore the transfer relationship between the political spectrum and hate speech.

9 Contributions

- William Ellsworth: Model architecture and Implementation, Web Scraping (Milestone 1), GloVe Encodings, Data Cleaning
- Sergio Charles: Data Set Prep, Web Scraping (Milestone 1), Problem and Model Architecture Brainstorming, Data Cleaning
- Bharath Raj Namboothiry: GloVe Encodings, Data Prep, Web Scraping (Milestone 1), Model Brainstorming, Result Analysis, Video Presentation
- Matthew Kolodner: Baseline Model, Web Scraping (Milestone 1), Data Cleaning, Visualization of Results

References

[1] Yao, Hao-Ren, et al. 2019. Hate Speech Detection: Challenges and Solutions. Plos One, Public Library of Science.
[2] Saenger, V. 2019. Media Bias Detection using Deep Learning Libraries in Python.
[3] Misra, A., Basak, S. (n.d.). Political Bias Analysis (pp. 1–8).
[4] Budak, C., Goel, S., Rao, J. M. (n.d.). Quantifying News Media Bias through Crowdsourcing and Machine Learning Dataset.
[5] Vu, M. 2018. Political News Bias Detection using Machine Learning.
[6] Mohit Iyyer, Peter Enns, Jordan Boyd-Graber, and Philip Resnik. 2014. Political ideology detection using recursive neural networks. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vol. 1. 1113–1122
[7] Yanchuan Sim, Brice DL Acree, Justin H Gross, and Noah A Smith. 2013. Measuring ideological proportions in political speeches. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. 91–101.
[8] Jurafsky, D., Martin, J. H. 2014. Speech and language processing. Upper Saddle River, NJ: Prentice Hall, Pearson Education International.
[9] Figure Eight. 2016. Political Social Media Posts (v1). Crowdflower Data for Everyone. Retrieved from https://www.kaggle.com/crowdflower/political-social-media-posts.
[10] Davidson, Thomas, et al. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In Proceedings of the International AAAI Conference on Web and Social Media, North America.
[11] Yenter, Verma. 2017. Deep CNN-LSTM with combined kernels from multiple branches for IMDb review sentiment analysis. In Proceedings of the 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON).