Political Bias Analysis

Arkajyoti Misra
Target Corporation
Stanford University
arkajyot@stanford.edu

Sanjib Basak
Digital River Inc.
Stanford University
sbasak@stanford.edu

Abstract

The two major political parties in US are polarized between either the liberal and conservative point of view on a multitude of socio-economical and environmental issues. An algorithmic approach towards detection of such bias is both intellectually challenging and useful in areas like election prediction. There exists very few studies in the literature where modern deep learning techniques are applied to detect the personal opinion or bias of an individual. In this work, we have developed an LSTM network that achieved an F1 score of 0.718 on a data set consisting of statements made in the recent past by US election candidates.

1 Introduction

The political atmosphere in the US is deeply polarized between two predominant ideologies: liberal and conservative. The twp major parties maintain differences in opinion in different issues like global warming, gay rights, abortions, foreign policies and immigration, to name a few. While the liberals encourage active role for government in society and believe in environmental regulations, conservatives like to have a limited role for government in the society and argue against imposing environmental regulations.

In the age of big data, with the help modern technology, this information is being captured in both structured and unstructured form at an unprecedented scale. Particularly, the year of 2016 being a US presidential election year, there is no shortage of active and engaging political discussions in the media fed by the plethora of public debates, interviews and speeches. The sudden eruption of activity in the area of opinion mining and sentiment analysis, which deals with the computational treatment of opinion, sentiment, and subjectivity in text, has generated increased interest in building new applications that deal directly with opinion [1]. The agenda of the political parties are obviously biased over the different issues, but the bias in the different forms of public communication like news media, journals, news channels are at times quite difficult to comprehend from a cursory look. In many cases, it is certainly challenging, if not impossible, to manually decipher every bit of such information available in the public domain that can be quite useful in predicting the outcome of an election, for example.

Our hypothesis is that a sophisticated deep learning algorithm can be trained to detect such bias automatically. Natural Language Processing (NLP) and Information Retrieval are becoming increasingly popular in detecting private states such as opinions, sentiment, and beliefs [2], [3] from text. However, the task of detecting political bias is quite non trivial and provided a unique challenge to the NLP community over years. The goal of our project is to build an analyzer capable of detecting political bias from a given body of text.

Previous work on bias or ideology detection from a textual document is relatively rare in literature. Most prior work on the topic are based on a bag-of-words representations of the document together with a multitude of hand crafted rule based on deep understanding of the structure of the language. For example, Gerrish and Blei [4] predicted the voting patterns of Congress members based on

bag-of-words representations of bills and inferred political leanings of those members; whereas Gentzkow and Shapiro [5] derived a slant index to rate the ideological leaning of newspapers.

With the recent success of deep neural network models in the field of natural language processing, researchers have started to apply the state of the art models in the field of abstract theme or opinion detection [6] and achieved higher accuracy (an F1 score of close to 70.) Models like Recurrent neural network deploy deep architecture with multiple hidden layers and perform well for sequential prediction of tasks. Within the NLP field, sentences are viewed as constructed sequence of tokens that need to carry forward the meaning from the beginning of the sentences to the end. With this view, those models have been successfully applied to tasks such as language modeling [7], and spoken language understanding [8]

Recursive neural networks is another class of deep neural net architecture within that has been applied to parsing [9], sentence-level sentiment analysis [10], and paraphrase detection [11]. The major inspiration of the present work comes from the recent work by Iyyer et al [12] where the authors applied a Recursive Neural Network (RvNN) based on the Stanford tree parser and achieved a 69% accuracy on a binomial classification of the two prominent political ideologies in the US.

2 Data

The data set we used for the project is called the Ideological Books Corpus (IBC), which was compiled by a group of researchers from the University of Maryland [12]. The data set was based on publicly available US congressional floor debate transcripts from 2005 [13]. The final data set [14] was a compilation of sentences hand picked to be most expressive of political sentiment and manually labeled by a majority voting scheme. We are using the entire data set for this project that consists of 2025 liberal and 1701 conservative biased sentences.

A typical example of a conservative biased sentence is "Even he would have been in complete shock over Al Gore 's ability to scare billions of people into believing humans are killing themselves and suffocating all life on the planet by driving their carbon-emitting cars back and forth to work instead of riding bicycles ." The example highlights the difficulty of the task at hand. It would be easy for a real human to categorize the sentence as anti-liberal based on prior knowledge of Al Gore's political position or that carbon emitting cars are bad for the environment is a predominantly progressive point of view. However, any classifier that needs to effectively identify the political bias of an example like this must have (a) the ability to capture long distance correlation between words and (b) knowledge of proper context.

A typical liberal biased example - "Though the practice can be defended as generating revenue used to improve the college 's academic program , the effect is to favor middle-class and less needy applicants ." - also supports the fact that political bias may not be explicitly expressed through one or many words and knowledge of the general perception about the middle class of the major US political institutions is of paramount importance in classifying the sentence correctly.

We also collected data from ontheissues [15] (OTI) that collects political speeches, dialogues and debate transcripts, from candidates of both the major parties, on a multitude of issues. A majority of the content of the data set being statements from the political candidates on an election campaign, the texts consist of generally smaller sentences and are more expressive of the bias of the speaker compared to the previous data set. For each of the issues we created two documents - one with liberal views and another with conservative views. We did not assign a label to a text based on the political affiliation of the speaker; instead we carefully labeled them based on the actual bias expressed in the text. This is a departure from the IBC data set where the text was labeled based on the political affiliation of the speaker.

In the OTI data set, a typical conservative biased example on the issue of gun control is: 'Restrictive gun laws don't stop culture of violence'; while a liberal biased statement on environment is: 'Adopt the Clean Cars Act to fight global warming'.

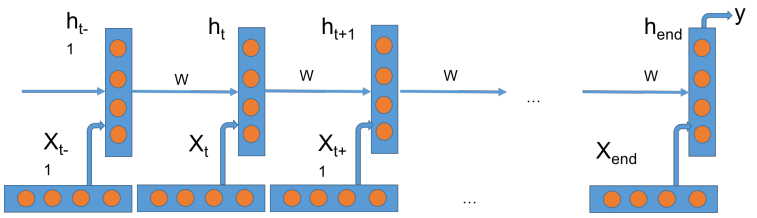

Figure 1: A rolled out RNN network used in binary classification.

3 Model

Recurrent neural networks (RNN) have completely revolutionized the world of Natural Language Processing (NLP) in the last few years. In extremely difficult NLP challenges like machine translation, question answering and text generation to name a few, RNN have recently outperformed the conventional techniques based on a deep understanding of the structure of the language. Conventional RNNs are generative model where the network is built to always predict the next word in a sequence, thereby learning the semantic structure of the language. For the task at hand, we are using the network as a classifier where the network is tasked with predicting the label at the end of a sequence, at shown in Fig. 1. The task at hand demands that the model capture long range correlation between words in the text. That is because the data loss does not get contribution from every word in the network and is calculated only at the very last time step and the gradient of the loss has to backpropagate to the beginning of the sentence. However, conventional RNNs stuggle to back propagate the gradient over many time steps because of the well known problems of vanishing or exploding gradients [16]. More recently, RNN variants like GRU and LSTM have solved the problem and therefore, an LSTM network was chosen for the task at hand.

An LSTM unit is described by the set of activations in Eqs. (1), where h_t is the hidden state at time step t; i, f and o stand for the input, forget and the output gate respectively. The network learns the various W and U matrices and c_t controls how much of the past information has to be retained in the network. The symbol $\circ$ denotes the Hadamard product.

$$\begin{aligned} i &= \sigma(x_t U^i + h_{t-1} W^i) \\ f &= \sigma(x_t U^f + h_{t-1} W^f) \\ o &= \sigma(x_t U^o + h_{t-1} W^o) \\ g &= \tanh(x_t U^g + h_{t-1} W^g) \\ o &= \sigma(x_t U^o + h_{t-1} W^o) \\ c_t &= c_{t-1} \circ f + g \circ i \\ s_t &= \tanh(c_t) \circ o \end{aligned} \tag{1}$$

The binary cross entropy loss is calculated at the last time step (t_{end}) of the network as follows:

$$\text{loss}_{CE}(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i),$$

where y is the true label and the predicted label is calculated using a sigmoid function applied on the hidden state at t_{end}:

$$\hat{y} = \sigma(h_{t_{end}} U + b) \tag{2}$$

4 Results and Discussion

In a binary classification framework, both the ROC-AUC number and the F1-score are widely accepted to be robust measures. The AUC is the Area Under the Curve of a plot of the true positive rate (TPR) vs the false positive rate (FPR). The metric varies between 0.5 for a completely random prediction and 1 for a perfect prediction. For data sets having highly imbalanced classes present in

it, the AUC is a good measure as it properly weights the correct prediction of the minority class too. The F1-score, on the other hand, is defined as the geometric mean of the precision and recall [17]

$$F1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}, \tag{3}$$

where precision is the fraction of retrieved instances that are relevant, while recall is the fraction of relevant instances that are retrieved [18]. The F1 score varies between 0 and 1 and the standard goal in a classification project is to maximize the score.

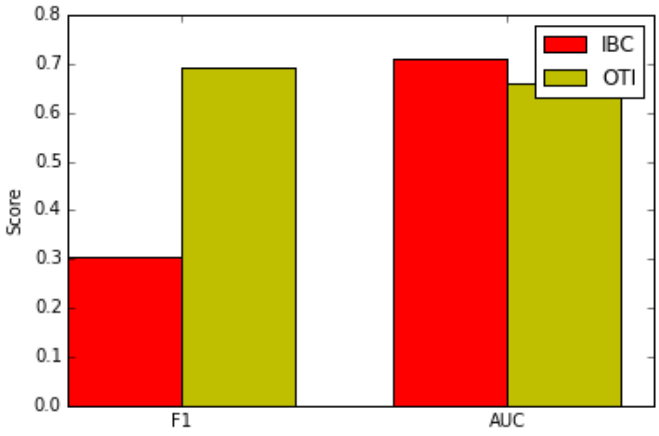

Figure 2: Performance of the Naive Bayes model measured by both F1 and AUC metrics on both the data sets used in the study.

In our experiment we shall focus on the F1 score as it is most common in the literature but will also track the AUC score as a guide against the model predicting completely random outcomes. Although the F1 score is more frequently reported in classification tasks similar to the present one, the AUC actually looks at all possible prediction thresholds instead of the best possible one. The IBC data set being so challenging, we felt it is important to track both metrics for higher confidence in the prediction.

The previous work [12] on the IBC data set consistently used accuracy as the measure of their model prediction. However, we have experimented with another data set (OTI) that tends to have a larger imbalance between the labels on some topics. So we decided to focus on the F1 and AUC metrics instead.

A baseline was established for both sets of data using a Naive Bayes bag of words model that works on the frequency of the unigram and bigrams present in the text. Fig. 2 shows the performance of both the IBC and the OTI data set in terms of F1 and AUC. The basic model performed very poorly with the F1 metric on the IBC data set. Both the metrics perform in a similar fashion for the OTI data set. The F1 score on the IBC data sets a very low baseline clearly illustrating the challenges with the content of the data set.

We used a single layer LSTM model for our classification task, where each hidden unit in fig. 1 is replaced by an LSTM unit governed by the set of activations described in eqn. 1. The binary cross entropy loss described in eqn. 2 was minimized by 'Adam' optimizer. We let the optimizer run for an unlimited number of epochs and used a 10 step early stopping scheme to terminate each individual model run. We kept our batch size fixed at 32 for all our experiments.

We stuck to a five fold cross validation scheme for all our experiments, where in each fold the model was trained on 80% of the data and a validation score was calculated on the remaining 20%. Exploratory runs with the IBC data revealed that for learning rates larger than 0.001 the model fits nicely to the training data, while the validation loss kept increasing slowly over hundreds of epochs. We believe the reason for the model's failure to decrease the validation loss lies in the very nature

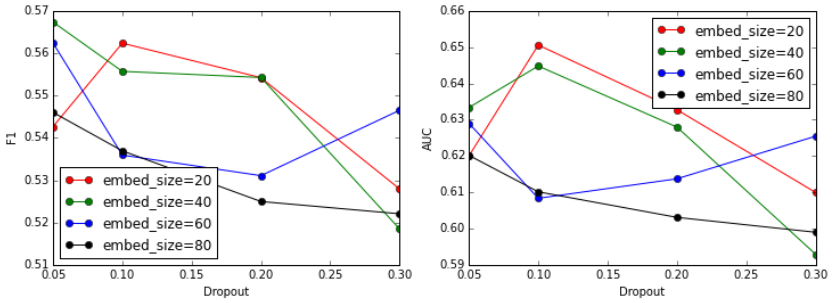

Figure 3: Model performance on the IBC Dataset: Variation of F1 and AUC measures as a function of the dropout rate for a range of hidden layer sizes.

of the data set as in every run the training data looks quite different from that of the validation data. The poor F1 score of the bag of words model shown in Fig. 2 supports this claim.

Fig. 3 shows the performance of the model on the IBC data set. We were careful not to overfit the training data and hence kept the size of the hidden layer at a maximum of 80. The dropout rate had to be pushed to quite extreme limits for the model to start fitting. Again, the extreme low values of dropout, namely 0.1, where the model performs best on both the F1 and AUC metrics suggests the extreme risk of overfitting as discussed before. However, such a large amount of regularization leads to a poor model that cannot even fit the training data well. The best F1 score of 0.568 was obtained by a model with a hidden size of 40 and a dropout rate of 0.05.

However poor the F1 scores appear for the model trained on the IBC data set, it must be noted that this is not unprecedented. Using a combination of bi-directional RNN and a bidirectional RvNN Isroy et al [19] obtained a best F1 score in the range of 0.5-0.6 on an opinion extraction task.

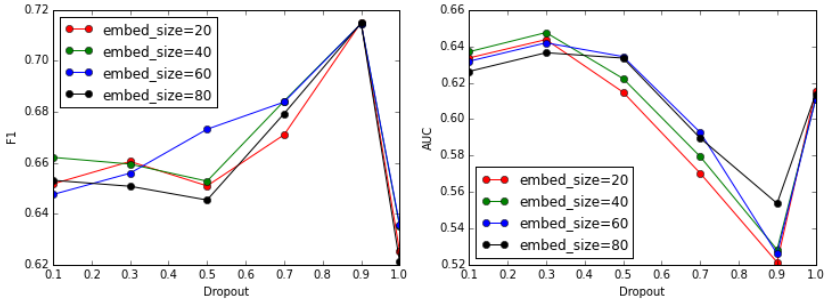

Figure 4: Model performance on the OTI Dataset: Variation of F1 and AUC measures as a function of the dropout rate for a range of hidden layer sizes.

Nonetheless, the results presented in fig. 3 does not show beyond any reasonable doubts, that a state of the art LSTM network is up to the task of capturing implicit meaning of a text. As mentioned before, we strongly believe it is not a shortcoming of the model, rather a combination of complexity and lack of volume of the data led to the results. It should be noted that Iyyer et al [12] extracted better performance from the data set because it was manually labeled to the node level of the parse tree associated with every sentence that was used in building their RvNN model. We argue that this method is not scalable to newer data sets and we wanted to test further on our hypothesis that an LSTM network should be able to extract hidden meaning of a text with just an overall label for the text.

To test out hypothesis, we collected and manually cleaned data publicly available on the web to create the OTI dataset. The advantage of the OTI data set over the IBC data set was that almost all the sentences showed a bias that could be detected by a normal human, thereby reducing the

possibility of 'confusing' the model. Moreover, the average length of a sentence was 11 in the OTI data set, compared to 37 for the IBC data set, which made the training the model a lot easier.

The first indication of learning by the model became obvious when we did not have to resort unreasonably strong regularization (very low dropout number). Fig. 4 shows a fundamental change in behavior of the influence of dropout on the F1 score. The best performing models had a dropout of 0.9 beyond which overfitting started and the score dropped rather dramatically. There was not a big influence coming from the hidden layer size of the network, another clear sign that the model was truly learning from the data without a need of becoming too complex. The AUC score, on the other hand, showed a nearly opposite trend. The model could obtain the best AUC value of 0.642 at a dropout of 0.3. It must be reiterated that the two metrics do not capture the same sensitivity from a model and we kept the AUC as an extra metric to look at.

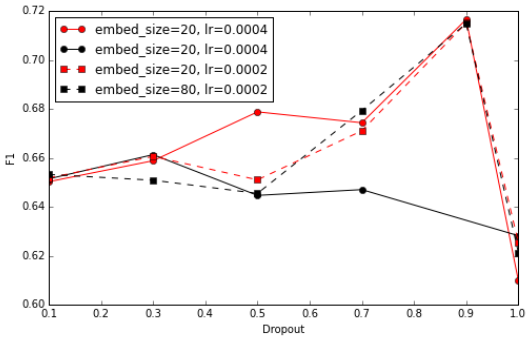

Figure 5: Effect of learning rate of the Adam optimizer as a function of dropout rate for the minimum and maximum hidden layer sizes used in the study.

We also looked at the sensitivity of our result as a function of the learning rate in the model. Fig. 5 shows that the peak performance does not suffer from doubling the learning rate from 0.0002 to 0.0004, but a smaller hidden layer size with a faster learning rate lead to poor performance.

The RNN and its variants can also be used in a bi-directional network. Meaning of a particular word in context does not always depend on the preceding words, sometimes the meaning of a word gets fully expressed from trailing words. Bi-directional networks try to capture this effect. We experimented with a bi-directional network where we concatenated the the hidden layers from the forward and backward direction. Fig. 6 shows that the network did not learn any extra information from the text. The bi-directional model performed nearly at par with its regular counterpart and the best bi-directional model also had a dropout rate of 0.9 and a hidden layer size of 80.

5 Conclusions

In this work we have showed how a state of the art model like LSTM can be used to predict the implicit political bias present in a text even if there are no specific words present in the text that obviously relates to one of the two major political ideologies. We have explored two data sets with very different content - one is extracted from speeches of US congressional floor debates while the other is a collection of statements on multiple socio-political issues by US presidential candidates in most recent history. The model performed poorly on the former data set because of two main reasons: There is very little overlap of contents in the data that led to severe training challenges but more importantly a good portion of the data did not actually show any clear political bias that can be detected by a normal human being.

We argue that the work by Iyyer et al [12] on the same data set performed better, although it is not a straight comparison because we used the F1 score instead of accuracy, with a RvNN because it was manually labeled at the node level of the parse tree, none of which we used in our model. Our goal was to build a model that is robust enough so that it can be applied to a variety of different contexts without the need of any detailed manual labeling, which is always difficult to obtain.

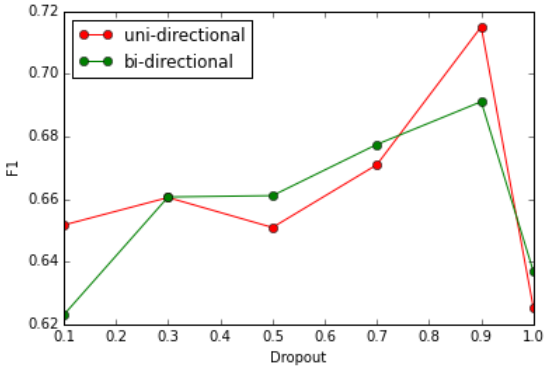

Figure 6: A bi-directional LSTM network performs very similarly as that of a uni-directional network on the OTI data set.

To test our hypothesis, we created a curated data set based on publicly available statements made by political candidates. We were able to obtain an F1 score of 0.718 with a single layer uni-directional LSTM network. This is at par with the best results on bias or opinion detection available in the literature [19]. We want to emphasize the fact that a model performs well only when there exists useful information present in the data. We would also like to point out that the method we developed is general in nature in the sense that the same model can be applied to extract bias or opinion in a different field of study without any major modification.

The LSTM model we developed works only marginally better than a bag of words based approach on the OTI data set. It is already extensively reported in the literature that deep learning based models need substantially larger amount of data to outperform the traditional methods of the domain. We, therefore, strongly believe the model will outperform the bag of words model substantially if we have more training data at our disposal.

References

[1] URL: http://www.cs.cornell.edu/home/llee/omsa/omsa.pdf.

[2] J. Wiebe T. Wilson and P. Hoffmann. "Recognizing contextual polarity in phrase-level sentiment analysis". In: (2005).

[3] B. Pang and L. Lee. "Opinion mining and sentiment analysis. Foundations and trends in information retrieval". In: (2008).

[4] Sean Gerrish and David Blei. " Predicting legislative roll calls from text". In: *28th International Conference on Machine Learning* (). URL: http://www.cs.columbia.edu/~blei/papers/GerrishBlei2011.pdf.

[5] Matthew Gentzkow and Jesse M Shapiro. "What drives media slant? evidence from us daily newspapers". In: *Econometrica* 78(1) (2010), pp. 35–71.

[6] URL: http://www.cs.cornell.edu/~oirsoy/files/emnlp14drnt.pdf.

[7] Tomas Mikolov et al. "Extensions of recurrent neural network language model. In Acoustics, Speech and Signal Processing (ICASSP)". In: *2011 IEEE International Conference* (2011), 55285531.

[8] Yoshua Bengio Deng. "Investigation of recurrent- neural-network architectures and learning methods for spoken language understanding". In: *Interspeech* (2013).

[9] Cliff C and Lin, Andrew Ng, and Chris Manning. "Parsing natural scenes and natural language with recursive neural networks". In: *Proceedings of the 28th International Conference on Machine Learning (ICML-11)* (2011), 129136.

[10] Richard Socher et al. "Semi-supervised recursive autoencoders for predicting sentiment distributions". In: *In Proceedings of the Conference on Empirical Methods in Natural Language Processing* (2011), 151161.

[11] Richard Socher et al. “Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In Advances in Neural Information Processing Systems”. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (2011), 801809.

[12] Mohit Iyyer et al. “Political Ideology Detection Using Recursive Neural Networks”. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics* (2014), pp. 1113–1122.

[13] URL: http://www.cs.cornell.edu/home/llee/data/convote.html.

[14] URL: http://cs.umd.edu/miyyer/ibc.

[15] URL: http://www.ontheissues.org/default.htm.

[16] Bengio et all. “Learning Long-Term Dependencies with Gradient Descent is difficult”. In: *IEEE Transactions on Neural Network Vol 5 No. 2* (1994).

[17] URL: https://en.wikipedia.org/wiki/F1_score.

[18] URL: https://en.wikipedia.org/wiki/Precision_and_recall.

[19] URL: https://arxiv.org/pdf/1312.0493.pdf.