CS230

Guess the Stance: Predicting Political Orientation from Congressional Statements

Saurabh Khanna
Department of Education
Stanford University
khanna90@stanford.edu

Miroslav Suzara
Department of Education
Stanford University
msuzara@stanford.edu

Abstract

We provide a comparison of models to predict USA political party affiliation, defined through a labeled text dataset containing congressional house speech data from 2005. The three models that we explored were – 1) a multi-layer perceptron (MLP), 2) a convolutional neural network, and 3) an LSTM-RNN. Performance was improved through hyperparameter tuning by applying regularization methods (L1/L2 regularization and dropout), as well as altering number and size of neural net layers and the mini batch size. Our study shows that a convolutional neural network outperforms both the simple neural network and the LSTM-RNN. This opens up opportunities for future research applying these trained models toward other related political phenomena classification such as political ideology.

1 Introduction

Our project explores a comparison of NLP models applied to USA congressional house speech data from 2005 for prediction of political party affiliation. Thus, the input to our algorithm is text data, on a per speaker basis, with political affiliation labels. Given a speech, consisting of one or more sentences, we explore the accuracy of models at predicting the output of speaker political party affiliation. Thus, we address the research question of: how might we more accurately classify political party affiliation from speech text? To achieve this, we apply three primary methods – 1) multi-layer perceptron (MLP), 2) convolutional neural network (CNN), and 3) long short-term memory recurrent neural network (LSTM-RNN). For each method, we determine accuracy of the model and tune our hyperparameters to achieve a high accuracy on our test set.

We situate this work in broader attempts for how classification algorithms might encourage more critical awareness of current information consumption patterns/sources and diverse perspective-taking. We see this work as important and fundamental to fostering digitally literate, civically engaged and informed citizenry in a healthy democracy.

2 Related work

This project builds off prior bodies of work classifying text through recursive neural networks (RNN) [3], semi-supervised recursive autoencoders, logistic regression, and multi-view Latent Dirichlet Allocation [8, 1, 6] for political classification and sentiment [2]. Within this domain, we have identified some of the ways in which prior attempts have identified and sought to address challenges related to classification and sentiment at and across local, sentence, and document-level views.

CS230: Deep Learning, Autumn 2020, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

Yan et al. (2017) [8] specifically highlight how most sentiment classifiers on political corpora perform abysmally on data that comes from a different distribution. In other words, there is a need for caution in how such methodologies are applied across domains. The fact that the test set in such contexts is often going to fall outside the purview of the training and development sets is a clear challenge, and that is something we explore through our approach.

3 Dataset and Features

We are using the Convote dataset of Congressional speech data from Thomas et al. (2006) [7]. This dataset consists of the total record of 2005 U.S. House speech data on bills from govtrack.us labeled with political party affiliation (i.e., Democrat, Republican, or independent) of contributing speech authors (see Figure 1). The data set was originally utilized to investigate whether speeches can be classified as in support or opposition to a given bill. In total, the dataset includes 53 debates comprising 3,829 speech segments. Before feeding into a deep learning model, we pre-processed our data by removing punctuation characters, special characters, numerals, single characters, and multiple spaces (see Figure 2). We also standardized multiple variants of same stemmed words to a common stem.

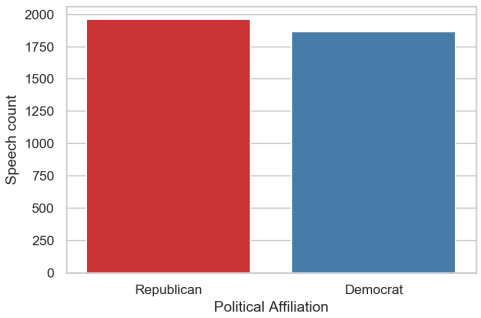


Figure 1: Political affiliation in data

	political_affiliation	speech_text	speech_text_clean	republican
0	R	mr. chairman , i am supportive , but my unders...	mr chairman am supportive but my understanding...	1
1	D	mr. chairman , i yield 5 minutes to the gentle...	mr chairman yield minutes to the gentlewoman f...	0
2	R	mr. speaker , i yield myself 2 minutes . mr. ...	mr speaker yield myself minutes mr speaker jus...	1
3	D	mr. speaker , the work of this subcommittee ha...	mr speaker the work of this subcommittee has a...	0
4	D	mr. speaker , reclaiming my time , i thank aga...	mr speaker reclaiming my time thank again the ...	0
...	...	...	...	...
3842	D	mr. chairman , i demand a recorded vote .	mr chairman demand recorded vote	0
3843	R	mr. speaker , i rise in support of s. 5 , the ...	mr speaker rise in support of the class action...	1
3844	R	i appreciate the gentleman yielding to me , mr...	i appreciate the gentleman yielding to me mr s...	1
3845	R	mr. chairman , i move to strike the last word ...	mr chairman move to strike the last word do so...	1
3846	D	mr. speaker , i rise in opposition to h. j. re...	mr speaker rise in opposition to j res which w...	0

Figure 2: Dataframe with preprocessed text

Figure 1 shows the proportions of Republican and Democrat politicians in our speech dataset. It is visible that our dataset maintains a decent balance of speeches from either category - 1964 Republican speeches and 1865 Democrat speeches. Our dataset included 9 speeches from Independent candidates - which was was not considered as a classification category due to the extremely small sample size. The fact that our dataset maintains a fair balance among both classes made us consider algorithmic accuracy as our key metric (as opposed to optimizing an F-1 Score, which is more relevant for datasets with rather skewed proportions of each category).

4 Methods

In order to kick-start our understanding of how sentiment or ideology classifiers work on political data, we began by exploring a pretrained sentiment analysis/political ideology classification module. More specifically, we explored and implemented usage of the Pattern library, which uses a sentiment analysis model based on a k-nearest neighbor approach to assign two metrics to any text corpora - polarity (ranging between -1 and 1) and subjectivity (ranging between 0 and 1). We also explored

Textblob, another Python library which directly imports algorithms from Pattern, but provides an easier integration while working with Pandas dataframes. Further, in line with our aims of testing on diverse sets, we also used the Contextual Web Search API to dynamically query data from the internet and assign it a polarity and subjectivity based on replicating the Pattern implementation. These methods helped us gain a foundational understanding of how existing libraries function to classify sentiment and ideology embedded in text with political intonations.

Building on these initial results, we intended to use our learning from the each course in our sequence. More specifically, we developed 3 models - 1) a multi-layer perceptron (MLP), 2) a convolutional neural network (CNN), and 3) a long short-term memory recurrent neural network (LSTM-RNN) model - using a corpora of political congressional speeches and compared its performance with existing models. In addition to these three models, we also implemented a simple logistic regression model that we used as our baseline model.

Our process involved training our model on a corpora of 2872 congressional speeches, and then validating each model (hyperparameter tuning, reducing overfitting) using a test set of 957 congressional speeches. Our train-test split of $75\% - 25\%$, as well as lack of a designated development set, was in lieu of the fact that our text corpora was not very extensive in terms of sample size. These would have been accommodated further if we had a larger dataset.

We used the `Tokenizer` class in `Keras` to generate a word-to-index dictionary. Each word in the corpus is used as a key in the word-to-index dictionary, and a corresponding unique index is used as the value for the key. Since the median speech length from our dataset was 931, we padded each sequence by using a maximum list size of 1000. We then used `GloVe` embeddings [5] to create a dictionary that will contain words as keys and their corresponding embedding list as values, culminating in the creation of our our feature matrix.

All four models were implemented using `Keras` – an open-source library built on top of `TensorFlow` that provides a Python interface for artificial neural networks:

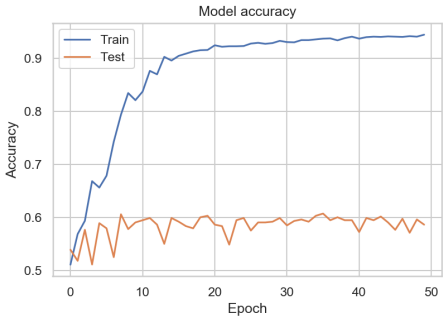
- The baseline logistic regression model uses stochastic gradient descent as an optimizer, binary cross entropy loss, and a sigmoid activation function.
- The multi-layer perceptron included 4 fully connected hidden layers, each having 24 nodes, and using the RELU activation function.
- For the convolutional neural network, we employ a one-dimensional convolutional layer with 128 kernels, with each kernel of size 5 and sigmoid being used as an activation. Then, we insert a global max pooling layer to reduce feature size. Finally, we include 4 fully connected layers for classification using RELU - similar to what we had in the multi-layer perceptron.
- For the LSTM-RNN, we include an LSTM layer with 128 neurons, with the remainder of the model similar to what we had for the multi-layer perceptron.

The output layer in all models uses a sigmoid activation to account for binary classification along probabilities on a [0, 1] scale. All models (apart from the logistic regression) use the Adam optimizer and a binary cross entropy loss.

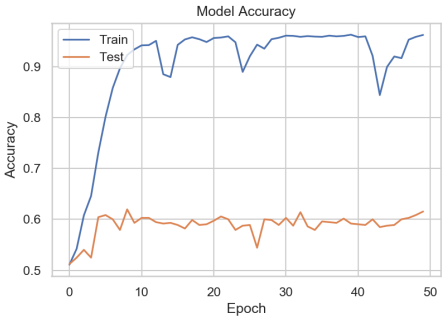
5 Results

The performance metric (i.e. accuracy) for our four models can be seen in Figure 3. Several key findings emerged from this analysis of the data. First, the training set accuracy was high (> 90%) for all models except the LSTM-RNN (where it lingered around 82%). This is not surprising since an LSTM can perform better with a much larger text corpora than we have here.

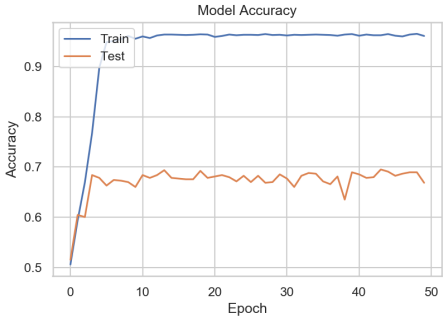
Second, we found that the Convolutional Neural Network (CNN) model performed best in terms of accuracy on the test set - with test set accuracy of around 70%. This is in comparison to the test set accuracy for other models, which lingered around 60%. This meant that our models suffered from overfitting and required effective regularization to raise test accuracy. In an attempt to address this, we tried two regularization methods (in that order) - L2 regularization using a λ value in the range of [0.001, 0.1] and dropout using probability values in the range of [0.1, 0.5]. We found that regularization was not able to fix the gap between the training and test sets, and often resulted in



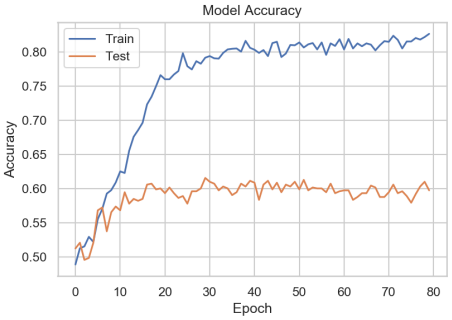
(a) Logistic Regression



(b) Multilayer Perceptron



(c) Convolutional Neural Network



(d) LSTM-RNN

Figure 3: Train and Test set accuracy plotted across epochs for each model

degenerate models. This highlights the fact that our models need to be provided with, and trained on, more data than the congressional speech dataset allowed us to utilize.

6 Conclusion

We found that a Convolutional Neural Network (CNN) approach to political affiliation text data outperformed a multi-layer perceptron and an LSTM-RNN. This result is in line with prior work on using CNNs for sentence classification [4]. That said, we urge readers to interpret our results with caution as even multiple robust regularization techniques were unable to bridge the gap between training and test accuracy. This highlights the need to perform such sensitive classification tasks with a much larger text corpora than a public resource like the congressional speech data.

Deep learning is an iterative process, as we have learnt through this course. For future work, or with more time, we would seek to iterate more and improve our models through obtaining more political text data, as well as further hyperparameter tuning to increase test accuracy. We would also explore reasons for constrained performance in testing against an external dataset that arises from a distribution that differs from our primary data source - hence understanding the real-world utility of our models. Such examples might include text from social media platforms. We can then use these steps as a launching point to applying these trained models for political affiliation toward other related political phenomena classification such as political ideology.

7 Contributions

Saurabh and Miroslav collaborated to contribute to each component of the project. Saurabh took a lead on searching for pre-trained models, data wrangling, training and tuning the deep learning models, maintaining the Github repository, and literature review. Miroslav led the efforts on finding relevant training datasets, literature review, data wrangling, reporting, and video production.

References

[1] Amr Ahmed and Eric Xing. Staying informed: supervised and semi-supervised multi-view topical analysis of ideological perspective. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1140–1150, 2010.

[2] Clint Burfoot. Using multiple sources of agreement information for sentiment classification of political transcripts. In *Proceedings of the Australasian language technology association workshop 2008*, pages 11–18, 2008.

[3] Mohit Iyyer, Peter Enns, Jordan Boyd-Graber, and Philip Resnik. Political ideology detection using recursive neural networks. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1113–1122, 2014.

[4] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[5] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[6] Adithya Rao and Nemanja Spasojevic. Actionable and political text classification using word embeddings and lstm. *arXiv preprint arXiv:1607.02501*, 2016.

[7] Matt Thomas, Bo Pang, and Lillian Lee. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. *arXiv preprint cs/0607062*, 2006.

[8] Hao Yan, Allen Lavoie, and Sanmay Das. The perils of classifying political orientation from text. In *LINKDEM@ IJCAI*, 2017.