Predicting Political Affiliation Based on Tweets

Molly Apsel, Marcos Catao, Alina Suarez, Alex Kumar

Abstract

Social media now allows politicians to reach more Americans than ever before. Twitter can be immensely influential in shaping public opinion. Previous research has identified linguistic differences between liberals and conservatives. If a natural language processing algorithm could distinguish between the tweets of different political parties, we could glean new insights about how different politicians appeal to their supporters and how this affects political polarization in the U.S. In this paper, we analyze the ability of recurrent neural networks and convolutional neural networks to classify a Congressperson's political affiliation based on a single tweet. We used natural language processing techniques, specifically sentiment analysis, in order to do this. The most successful algorithm was the bi-directional GRU, accurately predicting the tweeter's political affiliation 84% of the time. These results show the promising potential for use of machine learning in political analysis. Further work in this area can reveal key differences and similarities in political language on both sides of the divide.

Project Description

Language is important in politics because it performs a signaling function: certain words and phrases emotionally appeal to certain groups of people. There is some research on the use of "dog whistles" on the right (e.g. "Barack *Hussein* Obama", "states' rights"); there has been little published on whether this is a general trend across the political spectrum, or whether the left has its own code words as well. Additionally, a recent study of YouTube comments showed that liberals and conservatives often used different language to express similar ideas (e.g., "mask" and "muzzle", "liberals" and "libtards") (Knight, 2020). Further analysis of the difference in Tweets between politicians on either side will show whether this holds true across social media platforms and whether language can truly distinguish one political party from the other.

As technology and social media is prominent in our society, members of Congress are actively leveraging their Twitter platforms to expand their supporter bases. Can we predict a Congressmember's political affiliation based on their tweets? We think that there must be a way to determine what party a congress member belongs to based on social media posts. If we are able to reliably predict party membership based on tweets, this means that there is a significant difference in the way that Democratic and Republican Congressmembers use language. If we are not able to reliably predict party membership based on tweets, this would suggest that Congressmembers of different parties use language in similar ways. We hypothesize that the frequency of words and word combinations used in tweets can signal whether that person leans left or right, as defined by political party membership. If this is true, it could help people better understand how various political leaders express themselves and make informed decisions about who to support.

There have been attempts at analyzing this and similar problems in the past. One study examined various methods of predicting the political alignment of Twitter users based on both the content of their tweets and the structures of their political communication network. The researchers found that the network structure was the best predictor of political affiliation, but a model trained on hashtag data was almost as accurate (Conover et al., 2011). Another project by a group of Stanford students took a similar approach to us in using politician's tweets to train various algorithms and testing their ability to predict party affiliation based on a single tweet (Lee et al., 2018). We will use the findings from these projects to guide our decisions in our own project.

On the practical level, if our prediction is combined with an analysis of individual Congressmembers' policy positions, it can also be used to see how consistent Congressmembers are between what they say publicly (tweets) versus the actions they take. This project may also have campaign strategy implications. As social media has become a major tool to increase one's support base, our project can be used to analyze how different Congressmembers use twitter throughout their campaigns. If successful, this project can be extended to try and predict an average Twitter users' political ideology. This is an interesting problem especially in today's politically charged environment. If this project ends up being extended, we can learn more about our society by better understanding groups of people and active Twitter user's political affiliations. Essentially, this can help track a linguistic cause of polarization.

Data Description

We used tweets from members of the 115th U.S. Congress. We obtained tweet IDs from the Congressional Tweets Database (Littman, 2017), which is publically available and contains over 2,041,399 tweet IDs from Senators and Representatives collected between January 27th, 2017 and January 2nd, 2019. The data provides tweet text indexed to user IDs. We then used Twarc and Hydrator to retrieve the full tweet text from the ID. The data were composed of two subsets: Senators and Representatives. In our projects we used only the 1,058,732 tweets published by the 413 Representatives included in the database. Of the Representatives, 194 were Democrats and 219 were Republicans, and of all the tweets used in the project, 585,716 were authored by Democrats and the other 473,016 were written by republicans. We partitioned our dataset into three subsets: the training set, with 858,732 examples, the test set, with 180,000 examples and the validation set, with 20,000 examples.

Preprocessing

Since the entirety of our dataset consists of tweets, the preprocessing stage consists of hydrating the tweet IDs, reading the csv file created, cleaning the tweets, tokenizing the words, relating each word to a vector, as well as converting them to usable data frames and tensors. As our current methodology is word2vec, we decided to use **newly created word vectors**, as we would otherwise have to eliminate many words that have not been trained in pre trained word vectors. Moreover, not using pretrained embeddings allows us to have representations that are well suited for Twitter political language, while letting us have greater control over memory usage.

Next, we split each tweet into a list that consists of the words in the tweet, separated using nltk Tweet Tokenizer. This is a specialized tool for tweets, as you can not simply space separate, otherwise things such as several emoji's would be considered as one word. Since the inputs to the model all have to be the same size, we padded the tweets to all be the same length, a size of 30. In order to initialize the word embeddings, we created a dictionary of all of the unique words in our dataset. Then, we counted the frequency of each word in the entire dataset, and sorted the dictionary according to the frequencies. The word embeddings were then initialized based on the index position in the sorted dictionary. These word embeddings were then used to replace their string counterparts in the raw tweet data, creating a numpy array representation for the model to use. As the model expects a certain size input, we padded each tweet to be the same size. Within the models, we finally converted the numpy representation to a pytorch tensor representation, as the models use tensors as inputs.

As for the labels, we had to manually label the party affiliation of the Representatives in our dataset, as that was not provided. We then created a function to search for the party affiliation given a tweet, creating the label for every tweet.

Core Methods

Our goal is to classify tweets, processed into a list of numbers, into two classes that represent the political affiliation of the author. The first step, then, is to represent the list of numbers as a list of vectors. We do so using the word embeddings described in the preprocessing section.

Due to the sequential nature of text data, recurrent neural networks were obvious candidates for our classification task. In particular, we experimented with a unidirectional and a bi-directional GRU as they both achieved above 85% accuracy in a similar task on a smaller dataset (Lee et al., 2018). Convolutional Neural Networks also achieved state of the art performance on text classification tasks (Kim, 2014) so we also experimented with CNNs. More specifically we measured the performance on the validation set of a model with three and a model with four convolutional layers. We chose to have three and four convolutional layers because we expected that more layers would allow the model to capture more complex features of political language. We also tried a model that passes the same sentence input through three convolutional layers, each with a different kernel size.

We trained all the models on the training set and computed the validation accuracy for each training iteration. For each model, we then selected the weights that performed best in the validation set and computed the accuracy on the testing set.

Model	Test Accuracy
CNN-3Conv	83.64%
CNN-4Conv	83.83%
CNN-MultiKernel	82.93%
GRU	83.77%
Bidirectional GRU	83.87%

Table 1: accuracy of each model on the test set

Based on test set accuracy, we selected the bi-directional GRU, even though the accuracy of the models didn't differ by more than 1%. According to Chung et al. (2014) the GRU has one hidden state and at the t-th element, demonted x_t, of the input sequence, the GRU computes a candidate for the new hidden state:

$$\tilde{h}_t = \tanh(W_h x_t + U_h(r_t * h_{t-1})),$$

where $*$ demotes element-wise multiplication and r_t, called the reset gate, is defined as follows:

$$r_t = \sigma(W_r x_t + U_r h_{t-1}).$$

The model then computes the update gate, z_t, to determine how much of the new hidden state h_t will come from the previous hidden state and how much will come from the current input:

$$z_t = \sigma(W_z x_t + U_z h_{t-1});$$

$$h_t = (1 - z_t)h_{t-1} + z_t \tilde{h}_t.$$

In the bi-directional GRU, one GRU repeats this process from element 1 to t while the other one goes from t to 1. For classification, we concatenate the hidden state of the GRUs at the end of the sequence and send it through two fully connected layer, defined as

$$y = Ax + b,$$

where $y$is the output, A is the weight tensor and b is the bias. After the first fully connected layer we also added a dropout layer to improve models generalization.

Through random search, we tuned the hyperparameters arriving at a learning rate of 1e-3, a hidden state of length 300, embeddings of 40 dimensions and a dropout rate of 0.25

Final Results

	Predicted: Republican (0)	Predicted: Democrat (1)
Actual: Repubilcan (0)	62,720	13,767
Actual: Democrat (1)	15,260	102,020

Table 2: Confusion matrix for bi-directional GRU on the test set. Precision: 82%, recall: 80.43%

We were able to see very similar results across all of the different models that we used. Our maximum test accuracy of 83.87% was short of what has been achieved by Lee et al. (2018) on the same task using a smaller dataset. One possible explanation for this is that our dataset contains a greater proportion of tweets that do not include any political content. Another possible explanation is that our models aren't complex enough to capture some features of the data.

By classifying these tweets as only Republican or Democrat, the models are forced to guess as there is no significant information. As Table 2 shows, the false positives and false negatives occur at similar frequencies (18% for Republicans and and 15% for Democrats), while our model is able to classify tweets with similar accuracy for both Democrats and Republicans (85% and 82% respectively). In respect to the model, this signifies that both Democrats and Republicans use a similar number of words that are significant or signal to partisanship, as the model is able to correctly label tweets at similar probabilities. Below are 85 of the most weighted words, combined for both Democrats and Republicans.

🟫 : 0.7857068	text : 0.7556247	extended : 0.7556247
📺 : 0.76924163	experts : 0.7556247	fixing : 0.7556247
tax : 0.7556247	expected : 0.7556247	express : 0.7556247
next : 0.7556247	exciting : 0.7556247	experiences : 0.7556247
#taxreform : 0.7556247	expect : 0.7556247	@maxinewaters : 0.7556247
#goptaxscam : 0.7556247	expense : 0.7556247	excellence : 0.7556247
taxes : 0.7556247	excellent : 0.7556247	excuse : 0.7556247
texas : 0.7556247	extreme : 0.7556247	@dcexaminer : 0.7556247
@foxnews : 0.7556247	extend : 0.7556247	experienced : 0.7556247
sexual : 0.7556247	extremely : 0.7556247	existing : 0.7556247
pre-existing : 0.7556247	texans : 0.7556247	experiencing : 0.7556247
excited : 0.7556247	sex : 0.7556247	examine : 0.7556247
#taxcutsandjobsact : 0.7556247	preexisting : 0.7556247	#arpx : 0.7556247
executive : 0.7556247	@foxandfriends : 0.7556247	x : 0.7556247
expand : 0.7556247	complex : 0.7556247	expired : 0.7556247
@foxbusiness : 0.7556247	expanded : 0.7556247	exercise : 0.7556247
fix : 0.7556247	explain : 0.7556247	explains : 0.7556247
example : 0.7556247	#taxday : 0.7556247	#texas : 0.7556247
taxpayers : 0.7556247	exchange : 0.7556247	box : 0.7556247
thx : 0.7556247	extra : 0.7556247	exceptional : 0.7556247
taxpayer : 0.7556247	flexibility : 0.7556247	exposed : 0.7556247
experience : 0.7556247	@jacksonleetx18 : 0.7556247	#tx25 : 0.7556247
mexico : 0.7556247	extraordinary : 0.7556247	#tx23 : 0.7556247
@joaquincastrotx : 0.7556247	#goptaxbill : 0.7556247	expenses : 0.7556247
expanding : 0.7556247	exec : 0.7556247	#taxcuts : 0.7556247
tx : 0.7556247	expensive : 0.7556247	exist : 0.7556247
exactly : 0.7556247	fox : 0.7556247	rx : 0.7556247
	explore : 0.7556247	

expansion : 0.7556247 six : 0.7556247	extension : 0.7556247 toxic : 0.7556247	

Current Conclusions and Future Work

Through this project, we determined that natural language processing algorithms can be used to determine a Congressperson's political affiliation based on a single tweet with a reasonable accuracy of around 85%. While we believe this is a promising level of accuracy given the parameters of our problem, as previously mentioned, our algorithm does not reach the state of the art accuracy. Possible ways in which we could attempt to improve our accuracy include using larger word embedding sizes. This would essentially provide the algorithm with more dimensions to classify each word with, storing more information for each word. However, we do understand there is a certain size dimension where our accuracy would eventually plateau and possibly not allow us to reach the 91.6% state of the art accuracy. Another way we could try to increase the accuracy of our model would be to attempt to use character level embeddings. Using a one-dimensional CNN, these embeddings find the numeric representation of words by looking at their character-level compositions, and can therefore also help us reach a higher accuracy rate.

There are also future steps and broader applications for our algorithm. We could possibly extend this algorithm beyond just determining the political affiliation of a congress member by determining any tweeter's political affiliation. This would help us better understand different groups of people along with political polarization in the country. A second way in which we believe we can extend our algorithm is by integrating sentiment analysis into it. Tweeters could be classified into different groups and sentiment analysis could be used to better gauge public opinion and conduct nuanced research. Overall, we believe that our project has many different applications that could be explored in today's politically charged society.

References

Alexander, E. (27 December, 2019). Polarization in the Twittersphere: what 86 million tweets reveal about the political makeup of American Twitter users and how they engage with news. Knight Foundation. URL = <https://knightfoundation.org/articles/polarization-in-the-twittersphere-what-86-million-tweets-reveal-about-the-political-makeup-of-american-twitter-users-and-how-they-engage-with-news/.>

Documenting the Now. (2020). Hydrator [Computer Software]. Retrieved from https://github.com/docnow/hydrator

Documenting the Now. (2020). Twarc [Computer Software]. Retrieved from https://github.com/DocNow/twarc

Dos Santos, C., & Gatti, M. (2014, August). Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers* (pp. 69-78).

Conover, M. D., Gonçalves, B., Ratkiewicz, J., Flammini, A., & Menczer, F. (2011, October). Predicting the political alignment of twitter users. In *2011 IEEE third international conference on privacy, security, risk and trust and 2011 IEEE third international conference on social computing* (pp. 192-199). IEEE.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Kim, Y. (October 2014) Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, 2014. DOI:http://dx.doi.org/10.3115/v1/D14-1181

Knight, W. (2020, October 21). *The Left and the Right Speak Different Languages -- Literally*. https://www.wired.com/story/left-right-speak-different-languages-literally/.

Lee, C., Shiff, J., & Thatipamala, S. (2018). *Predicting U.S. Political Party Affiliation on Twitter.* Unpublished manuscript. CS 230: Deep Learning, Stanford University.

Littman, Justin, 2017, "115th U.S. Congress Tweet Ids", https://doi.org/10.7910/DVN/UIVHQR, Harvard Dataverse, V5, UNF:6:pa6q3/72341rRixB7ez15Q== [fileUNF]

Olasov, I. (November 7, 2016). Offensive political dog whistles: you know them when you hear them. Or do you? *Vox Media*. URL = <https://www.vox.com/the-big-idea/2016/11/7/13549154/dog-whistles-campaign-racism>

Velásco, Fernando L. (September, 2017) Text Classification with CNNs in PyTorch. URL=https://github.com/FernandoLpz/Text-Classification-CNN-PyTorch

Wang, J., Wang, Z., Zhang, D., & Yan, J. (2017, August). Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification. In *IJCAI* (Vol. 350).