Identifying political persuasion on Reddit

YueLan Qin
Supervised by: Frank Rudzicz
Mentor: Akshay Budhkar

I. Introduction

The increasing use of social media has introduced certain benefits as well as drawbacks. This has allowed researchers to use natural language processing (NLP) and other machine learning technologies to detect deception and cyberbullying in social media [**?**]. They have built classifiers to discriminate between neutral and invective words and found that the most invective posts have specific patterns. The volume of text available on social media attracts researchers as well. By applying NLP methods and sentiment analysis, Nathanael [**?**] et al. built three different classifiers to identify political sentiment among nation states. This reminds us that social media corpora provide new opportunities for public policy and political science, given that internet users are willing to talk about different political events. These observations make us question if the political persuasions of people can be identified through the posts they write on social media.

Our research is done on a collection of Reddit posts. The posts are assigned to four categories: to specify more clearly, we subtract topics witch tend to be alternative facts or neutral from traditional left-right distinction, and form four distinct classes – Left, Center, Right, Alt. We manually create these four classes and assign topics to these categories . We build different machine learning classifiers to do a four-way classification to identify political persuasion. A baseline is set by manually extracting 29 features based on corpus analysis, and applying scikit-learn classifiers [**?**].

Word embedding efficiently maps words to distributed and continuous vector representations that help learning algorithms achieve better performance by capturing syntax precisely and grouping similar words together [**?**]. Therefore, we use word embedding to get average word vectors as features of baseline classifiers and demonstrate that two of them perform the best among all of our algorithms, whereas the gated recurrent unit shows inferior performance [**?**].

II. Approach

A. Reddit Data

The data set we use is sampled from Reddit, which is a collection of posts commenting on various events. **Table 1** shows the four categories and their posts. We uniformly sample 20,000 posts from each category, for a total of 80,000 posts.

Each datum(post) contains several fields: 1.*ups*: the integer number of up-votes. 2. *downs*: the integer number of down-votes. 3. *score*: [ups downs] 4. *controversiality*: a combination of the popularity of a post and the ratio between ups and downs. 5. *subreddit*: the subreddit from which the post was sampled. 6. *author*: the author ID. 7. *body*: the main textual message of the post, which is our primary interest. 8. *id*: the unique identifier of the comment.

Category	Topic(Number of posts)
Left	twoXChromosomes (7, 720, 661)
	occupyWallStreet (397, 538)
	lateStageCapitalism (634, 962)
	progressive (246, 435)
	socialism (1, 082, 305)
	demsocialist (5269)
	Liberal (151, 350)
Center	news (2, 782, 9911)
	politics (60, 354, 767)
	energy (416, 926)
	canada (7, 225, 005)
	worldnews (38, 851, 904)
	law (464, 236)
Right	theNewRight (19, 466)
	whiteRights (118, 008)
	Libertarian (3, 886, 156)
	AskTrumpSupporters (1, 007, 590)
	The Donald (21, 792, 999)
	new right (25, 166)
Alt	conspiracy (6, 767, 099)
	911truth (79, 868)

TABLE I
ASSIGNMENT OF DIFFERENT TOPICS TO EACH CATEGORY WITH THE TOTAL NUMBER OF POSTS IN BRACKETS. A UNIFORM SAMPLE OF 20,000 POSTS FROM EACH SUBREDDIT CATEGORY FORMED OUR DATA SET. (E.G. RANDOMLY CHOOSE X POSTS FROM CONSPIRACY AND Y POSTS FROM 911TRUTH WHERE X+Y = 20,000, TAG THESE POSTS TO 'ALT'.)

B. Baseline

First step is to preprocess the data(e.g. remove punctuation, apply lemmatization, remove stopwords etc.). Corpus analysis is done before we decide to choose word count and other 28 features. After doing pre-processing and manually extracting features, we use five classifiers to get baseline accuracy: 1. **Support Vector Machine** [**?**]. 2. **Support Vector Machine with gamma set to 2.** 3.**Random Forest**[**?**]. 4.**MLP (multilayer perceptron)**. 5.**AdaBoost:** A boost classifier is a classifier in the form:

$$F_T(x) = \sum_{t=1}^{T} f_t(x), \tag{1}$$

where T is the number of iterations, f_t takes an object x as input and returns a value indicating the class of the object [**?**].

C. word2vec

A word embedding, trained on word co-occurence in text corpus, represents each word w as a d-dimensional *word vector*. Word2vec has gained popularity as a tool to represent words since its advent in 2013.

After training word2vec with Reddit data, each word is assigned a vector, then words containing similar semantic information are grouped together. In our case, we set the word vector dimension to be 500. We set min count to be 1, which means each word will have a representation. The model is trained for 2000 epochs.

To represent the whole post, we use the average of each word vector in the post. After getting the **average word vectors** as features (note that they are different from features used in the baseline), we apply the same five classifiers as in the baseline.

D. GRU

For the sake of training Gated Recurrent Units (GRU, gating mechanism in recurrent neural network) [**?**], we train our word2vec with 300 dimensions and minimum count as 1. The learning rate of the GRU is set to be 0.0001 and we run it for 10 epochs.

1.**Sentence-based GRU.** We use sentence vectors (the average of word vectors in a sentence) as inputs to the GRU, sending it one sentence at every time step. We use a GRU with 2 layers, hidden size 600, the softmax activation function and cross entropy loss.

2.**Word-based GRU.** For this model, instead of using sentences, we directly use word vectors as inputs. For each time step, we input one word. The GRU structure is identical to the sentence-based version.

III. Analysis

Table 2 compares all the classification methods we tried. Note that rows 1-5 are baseline values and rows 2-6 use word embedding as features. Comparing them, we find that Random Forest and AdaBoost improve with word embeddings. (**Table 3** demonstrate the confusion matrix for Random Forest and AdaBoost) On the other hand, SVC stays around 25 % accuracy and MLP performance deteriorates with word embeddings. The sentence-based GRU performs around 25% and word based GRU performs slightly better. We attribute the weak performance of the GRU models to small data set size and insufficient training, as we noted that training loss continued to decrease at the end of the 10 epochs. Training loss decreases slowly because we set learning rate to be 0.0001 and there is scope to do hyper-parameter tuning in the future.

A. AdaBoost Performance

The AdaBoost training process selects only those features known to improve the predictive power of the model, reducing dimensionality.

B. Neural network performance

The main reason may still be lack of data and lack of training epochs.

Method	**Accuracy(%)**
SVC	29.35
SVC(gam=2)	26.70
RF	37.13
MLP	30.40
AdaBoost	37.93
word embed +SVC	24.49
word embed+SVC(gam=2)	26.00
word embed + RF	**44.01**
word embed + MLP	25.68
word embed + AdaBoost	**45.55**
Sentence-based GRU	26.25
Word-based GRU	28.00

TABLE II
THE ACCURACY OBTAINED FROM DIFFERENT CLASSIFICATION METHODS. USING WORD VECTORS AS FEATURES IMPROVES BASELINE RESULTS. DUE TO INSUFFICIENT TRAINING, GRU PRODUCES INFERIOR PERFORMANCE.

	Left	**Center**	**Right**	**Alt**
Left	**2945**	406	232	473
Center	447	**2634**	349	806
Right	189	547	**2372**	646
Alt	407	1725	452	**1370**

TABLE III
CONFUSION MATRIX FOR RANDOM FOREST AND ADABOOST, WITH CLASSIFICATION COUNTS OF RANDOM FOREST ON THE HORIZONTAL AXIS AND ADABOOST ON THE VERTICAL AXIS.

IV. Conclusion

Word embedding with Random Forest or AdaBoost surpasses the performance of the other 10 tested classifiers and improves from baseline. The failure of other classifiers is probably due to feature selection. Future work in feature analysis and neural network tuning needs to be done to improve accuracy. As the classifiers mature, it would be an interesting application to identify political persuasions of people through comments they post online. It can also be extended to classify users' political attitudes when taking into account all over their posts or even interactions with other users. Moreover, instead of manual tagging, we plan to try unsupervised learning to assign topics to categories in the future.