CS583: Data Mining Text Mining
Research Report

Chieh-Hsi Lin 670777349, Charlie Wang 656530083

I. Introduction

This project aims to find the best model which can classify the sentiment or opinion expressed based on the given tweet dataset correctly. In this dataset, there are two separate datasets, one is tweets for Obama, and the other is tweets for Romney. For each dataset, every tweet is classified into four classes: positive(1), negative(-1), neutral(0), and mixed(2) class, where neutral(0) means no sentiment, while mixed(2) represents the tweet expresses both positive and negative sentiments. In this report, we did not use the tweet classified as mixed for both training and testing, and we provided and implemented classification and deep learning algorithms to conduct sentiment analysis on the given dataset.

II. Data Preparation & Preprocessing

For both sheets of Obama and Romney, we extracted the data with the sentiment (positive, negative, neutral) classes that are going to use and replaced the column name 'Annotated Tweet' with 'Tweets' for easy use. With these two datasets, we did some data exploratory. Since the original tweet data contains unarranged elements like URL, hashtag, userID, etc. inside, therefore, we applied some data preprocessed methods on column 'Tweets' converting them to a suitable format to fit the models:

1. Lowercase: each tweet is converted to lowercase

2. Remove Special Characters: convert HTML tags to characters, replace URL, Username, Hashtag, Non-Alphabets, and Digits

3. Replace Abbreviated Words: set up the commonly abbreviated dictionary to replace the words in each tweet.

4. Remove Stopwords: stopwords are English words that do not have meanings to a sentence, which can be ignored without harm (eg. "the", "he", "have")

5. Stemming: reduce inflected words to their word stem, base, or root form (eg. "automate", "automatic", "automation" -> automa)

III. Data Modeling

A. Data Split

After preprocessing with the methods mentioned above, we divided this dataset into two parts for both Obama and Romney, training (80%) and validation(20%), and the test data will be provided at demo time.

In this report, we fit the model with three different datasets: Obama's tweets only,

Romney's tweets only and Combine Obama and Romney's tweets with shuffle ordering called 'All data'.

B. Classification Model

Before fitting data to classification models, we used n-grams feature representation consisting of CountVectorizer and TF-IDF to extract one feature for each word. We ignored words that appear in less than 3 documents, and those terms appear more than 90% of the document to filter out too infrequent and too frequent words to improve the performance.

After we got the TF-IDF value, we roughly implemented different kinds of classification with these three datasets and, in the end, we chose 4 models, SVC, LogisticRegression, RandomForest, and Multinomial Naive Bayes, that have better results. And we also tried ensemble models consisting of these four classification models. In our case, the best classification model happened using the SVC model with 0.6 accuracy.

C. Deep Learning Model

CNN+LSTM:

After surveying the SOTA for SemEval 2017 task 4, which is very similar to our task, we decided to try the CNN+LSTM model. The model includes the word embedding layer, 1 CNN layer, and 1 dropout layer, and 1 LSTM layer, and using zero-padding with 100 dimensions. After the training, we got 0.55 accuracy for Obama tweets. Since we did not find significant improvement in accuracy compared to the Classification Model, we decided to try other solutions.

BERT(Bidirectional Encoder Representations from Transformers):

We observed that BERT, especially RoBERTa model, became popular in the recent Kaggle competition about Twitter Sentiment Analysis. To implement it on this task, we used the pretrained model bert_uncased_L-12_H-768_A-12, which has been trained on English Wikipedia and the BookCorpus. We used the model's tokenizer to transfer inputs to a compatible tensorflow format (tokenized text, segment embedding, position embedding) and added it as a downstream model to fine-tune by updating the attention when masking different positions and use cross-entropy as the loss function. The accuracy result is around 0.6, which is almost the same as the traditional bag of words + classifier model.

IV. Experiment Results

In the previous section, we mentioned different models, and we did lots of experiments and parameters tuning. In this section, we provide only the best results found from those models mentioned earlier.

A. Train with Obama's Data Only

	Accuracy	Positive			Neutral			Negative		
		Precision	Recall	f1-score	Precision	Recall	f1-score	Precision	Recall	f1-score
Classification										
SVC	0.6	0.62	0.64	0.63	0.53	0.55	0.54	0.65	0.61	0.63
LogisticRegression	0.59	0.6	0.6	0.6	0.53	0.54	0.53	0.64	0.62	0.63
RandomForest	0.57	0.61	0.53	0.57	0.48	0.54	0.5	0.62	0.63	0.63
Multinomial NaïveBayes	0.57	0.48	0.62	0.54	0.51	0.56	0.53	0.72	0.56	0.63
Ensemble Model										
LogReg + RandForest + SVC	0.58	0.65	0.55	0.6	0.49	0.55	0.51	0.61	0.63	0.62
LogReg + SVC + SVC	0.58	0.66	0.55	0.6	0.49	0.55	0.52	0.61	0.64	0.62
Deep Learning										
CNN + LSTM	0.56	0.57	0.66	0.61	0.53	0.46	0.49	0.58	0.58	0.58
BERT	**0.6036**	0.57	0.63	0.6	0.58	0.55	0.57	0.65	0.64	0.65

B. Train with Romney's Data Only

	Accuracy	Positive			Neutral			Negative		
		Precision	Recall	f1-score	Precision	Recall	f1-score	Precision	Recall	f1-score
Classification										
SVC	0.6	0.28	0.66	0.39	0.27	0.51	0.35	0.9	0.61	0.72
LogisticRegression	0.59	0.32	0.57	0.41	0.3	0.48	0.37	0.85	0.62	0.72
RandomForest	0.58	0.3	0.6	0.4	0.27	0.46	0.34	0.86	0.6	0.7
Multinomial NaïveBayes	0.58	0.43	0.51	0.47	0.34	0.45	0.39	0.77	0.64	0.7
Ensemble Model										
LogReg + RandForest + SVC	0.57	0.28	0.57	0.38	0.3	0.47	0.37	0.83	0.6	0.69
LogReg + SVC + SVC	0.57	0.28	0.56	0.37	0.3	0.47	0.37	0.83	0.6	0.69
Deep Learning										
CNN + LSTM	0.57	0.45	0.35	0.39	0.45	0.45	0.45	0.67	0.72	0.7
BERT	**0.62832**	0.46	0.56	0.51	0.55	0.47	0.51	0.72	0.73	0.72

C. Train with All Data (Obama + Romney)

	Accuracy	Positive			Neutral			Negative		
		Precision	Recall	f1-score	Precision	Recall	f1-score	Precision	Recall	f1-score
Classification										
SVC	0.58	0.44	0.62	0.51	0.43	0.58	0.49	0.8	0.57	0.67
LogisticRegression	0.57	0.46	0.56	0.5	0.46	0.54	0.49	0.74	0.59	0.66
RandomForest	0.57	0.42	0.59	0.49	0.41	0.56	0.48	0.78	0.56	0.65
Multinomial NaïveBayes	0.57	0.48	0.54	0.51	0.4	0.59	0.47	0.76	0.57	0.65
Ensemble Model										
LogReg + RandForest + SVC	0.58	0.47	0.6	0.53	0.44	0.58	0.5	0.77	0.57	0.66
LogReg + SVC + SVC	0.58	0.47	0.6	0.53	0.44	0.58	0.5	0.76	0.57	0.65
Deep Learning										
CNN + LSTM	0.56	0.5	0.46	0.5	0.5	0.46	0.48	0.61	0.7	0.65
BERT	**0.600877**	0.53	0.6	0.56	0.56	0.52	0.54	0.68	0.67	0.68

From the results above, we can see that compared to classification and ensemble models, the SVC model has the best performance with an accuracy of 0.6. And among all models, BERT has the best outcome of 0.62 using Romney's dataset. If we compare each sentiment, overall, the negative class has higher results in all models, we think it is because there are more negative tweets and more representative words for the negative side within each dataset so that the model can easily learn and predict them correctly.

V. Conclusion

The best model for classifying tweets sentiments in our project is BERT. Our pre-trained, BERT, model is using google's original BERT uncased L-12 H-768 A-12 model. We split the original data to training (80%) and validation (20%)

dataset, and fine-tune the model with sequence embedding and attention mask mechanism, and used cross-entropy as the loss function. The best evaluation result of the model is 0.607 accuracy for Obama and 0.609 accuracy for Romney, with slight improvement from the classification model.

However, overall the performance of our tried models does not have reasonable results as we expected before starting experimenting with these models. Here are some reasons we think might affect and might worth improving:

1. Everything in our modeling set up is based on the random state, no matter in data split, initial model setup, parameters choosing etc., therefore, it is hard to control the performance of each model when we train every time.

2. For the dataset, because we are using tweet datasets, which contains lots of slang or phrases that are not meaningful unless people use it, and also some typo, therefore, it increases the difficulties of data preprocessing. We could not guarantee we used the 'clean' dataset since there are too many possibilities out of our range of control. Potential techniques can be used include the transformation of emoticons and continuous words (e.g. Aaaaaand for and).

3. Besides, we did a bit more on data exploration. We think that the dataset is not well separated since there is a biased amount of data between each sentiment class, not equally distributed, so it might affect the result for models to learn each class. We also tried to find out the top common words in each class to see if every sentiment has some words representative enough for models to learn and predict correctly. However, most of the common words within these three classes are similar. And they seem to be meaningless in the sentence, which is one reason we think why the model cannot learn well.

4. As for the pre-train model, besides BERT, we can also try some other pretrained models, like TweetBert trained by tweets. Since the grammar and word usage of tweets is a lot different from the wiki or book corpus, it may result better by using Tweetbert pre-trained model.

In conclusion, if we can get a more comprehensive dataset with a bigger amount of dataset for models to learn, and with enough time and right methods for tuning parameters of models then we think the performance can be improved somehow.

VI. References

- Liu, B. (2007). *Web data mining: exploring hyperlinks, contents, and usage data.* Springer Science & Business Media.
- *NLP - Twitter Sentiment Analysis Project* - Kaggle, GauravChhabra, 2018
- *Bitcoin Tweets Sentiment Analysis:cnn-lstm* - Kaggle, dundee2002, 2017
- *Step By Step Guide To Implement Multi-Class Classification With BERT & TensorFlow* - Amal Nair, 2019