Report for this NLP Assignment

A brief summary of the project layout could be found in README.md.

**Model Selection and Research**

The first time I receive this assignment, I immediately realized that it is very similar to one of my previous project about NLP. The related materials are provided in the folder “Related Past Project”. In that project, our group used Naïve Bayes as the baseline model, and used LSTM with attention as the “advanced” model. After a brief research, I recognized that LSTM is no longer considered as a state of the art technique, therefore I decided to use BERT instead. I kept using Naïve Bayes as the baseline model, but for the word embedder of the baseline model, instead of using Word2Vec as the previous project, I decide to use Doc2Vec, which is more convenient for long text word embedding. Using BERT tokenizer as the word embedder of BERT classification model is considerably a natural selection.

Most of the Naïve Bayes model codes are reused from the previous project. I referenced an article named “Sentiment Analysis Using Doc2Vec” [1] to build my Doc2Vec model. Since it is merely a baseline model, I did not perform cross-validation and select default Gaussian Naïve Bayes model from Sklearn. On the other hand, the number of epochs for training the Doc2Vec model is tuned to 100, considering both the accuracy and the processing time. The fine-tuning process is terminated when an accuracy of more than 50% has been reached, which is close to what I expected from a Naïve Bayes 5-class classifier.

As for the BERT model, I referenced an article named “BERT Fine-Tuning Tutorial with PyTorch” [2] to build both the embedding model and the classification model. For the embedding model, I used the pre-trained BERT base uncased model to perform transfer learning. Since this is a classification problem, “BertForSequenceClassification” model is chosen. Going through the referenced article, I noticed that the source code of BERT library has changed after the article was posted. As the result, most of the method calls were directly referenced from BERT’s API webpage. Several hyper-parameters were chosen due to the limit of my GPU, such as batch size and number of tokens kept for each review. Other hyper-parameters such as number of epochs and learning rates were tuned during the training process. Details of these engineering choice could be found in comments of each python files. The fine-tuning process is terminated when an accuracy of more than 80% has been reached.

[1] <http://linanqiu.github.io/2015/10/07/word2vec-sentiment/>

[2] <https://mccormickml.com/2019/07/22/BERT-fine-tuning/#12-installing-the-hugging-face-library>

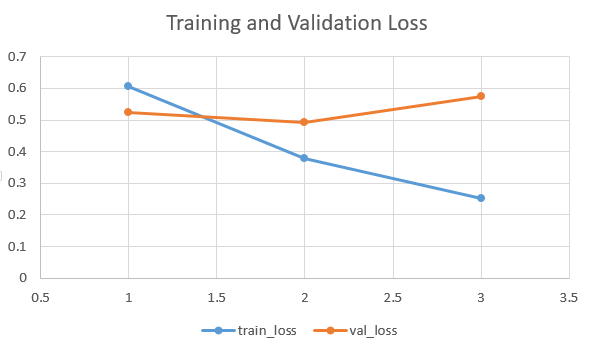
**Data Exploration and Evaluation Measures**

Since the training data is not skewed (Sample numbers are 7028, 7031, 6971, 6997, and 6977 respectfully), I choose to calculate accuracies to measure the performance of both models. In addition, processing time is also recorded. Training loss and validation loss is also observed in order to detect overfitting.

When I went through this project, I realized that one sample in the training set, ID 30944, contains invalid information. The information in the rating column is actually part of the review message. Thus I decided to delete this row in the training process, as reducing the training set by one should not make a large impact on model performance.

**Analysis and Model Comparison**

Some information could be found in the corresponding log csv files. The final validation accuracies are presented in the name of the result csv files.



The training and validation loss for the BERT model are summarized in the graph above. At epoch 3, training loss decreases while validation loss increases, which indicates an over-fitting issue. Therefore early-stopping was performed.

The accuracy on the development set for the Naïve Bayes model is 51.47%, while the accuracy for the BERT model is 81.15%. On the other hand, it takes the Naïve Bayes model less than 12 minutes to generate the results using a 12-CPU-core computer, and it takes the BERT model about 80 minutes to generate the results using an 8 GB-GPU computer.

It is no doubt that BERT classifier outperforms Naïve Bayes classifier, due to the 30% accuracy difference. Both models’ performances could be improved my further fine-tuning, which is beyond the scope of this project. At the same time, training a baseline model such as Naïve Bayes model is significantly less time-expensive. Constructing a simple model as a proof-of-concept is considerably important when starting a machine learning project.