**CE807-7-SP: Text Analytics**

**Assignment 2**

**Task 4**: Report

**Professor**: Dr. Habil Ansgar Scherp

**Student Name**:

Atiwat Onsuwan 1802514

Fazal Jamadar 1802447

**1.Introduction**

In this first section of this report will explain the description of the implementation of sentiment analysis on the movie reviews system using supervised learning and evaluate the performance by uploading the prediction of test data to Kaggle competition to get the public score.

For the second part of the report will compare and contrast the state-of-art with what we found in the literature review in task 1.

The last section of the report will explain the lesson learned from this assignment.

**2. Materials used**

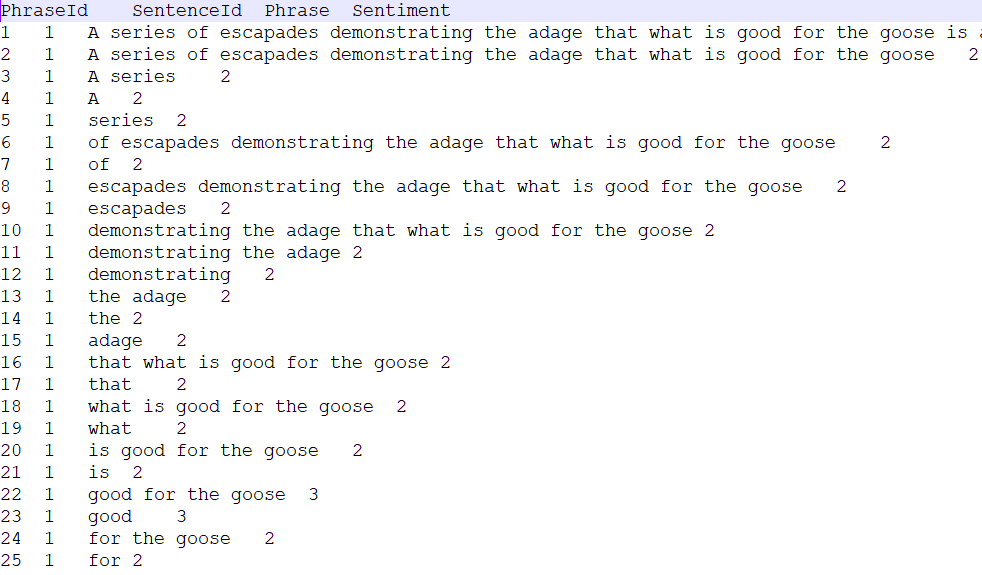
First of all, we will give a definition of all the materials used in this sentiment analysis system as the followings:

**2.1) Training set**: **train.tsv**, the given data from Rotten Tomatoes movie reviews data set along with the classes, use to train the model

**2.2)** **Test set**: **test.tsv** the given test data from Rotten Tomatoes movie reviews used to evaluate the model performance

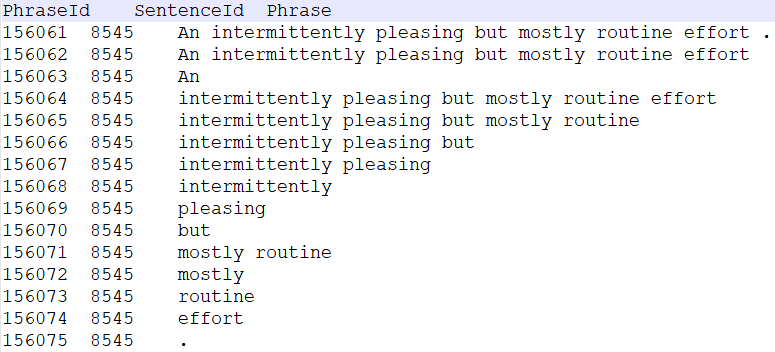
**3. Description of Implementation**

**3.1)** **Select the features**: before training the model, we have to decide which feature from the training set we will use to train our model, so we looked into the **train.tsv** file and found that the training set has 4 columns as shown in figure 1 below.



**Figure 1**: train.tsv

The first column **PhraseId** is the running number which is definitely not useful for the training process. The second column is **SentenceId** which can be useful for the training process, but we dropped it out because after we had tried to put this column as one of the features, we did not get higher accuracy. The third column is **Phrase;** this column is what we used for training our model. Last column **Sentiment** is the class of training set.

Also, we checked the **test.tsv** to make sure that it contains the same feature (Phrase column) see figure 2.

**Figure 2**: test.tsv

Therefore, in our system, we used **Phrase** and **Sentiment** from the training set to train the model and **Phrase** from the test set to have our model predict the class.

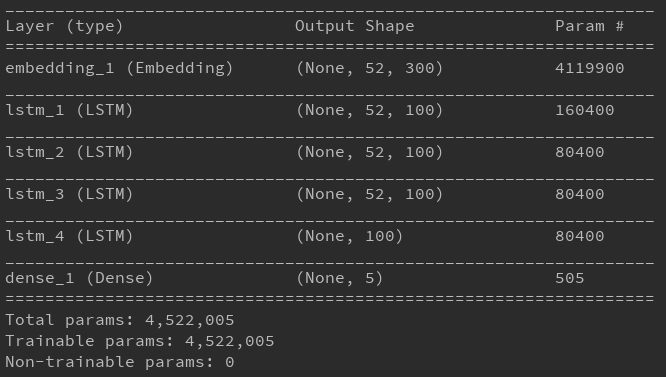
Next, we had to decide how could we make maximum use of the feature by doing the pre-processing.

**3.2)** **Choose a machine learning framework**: we chose Long Short Term Memory networks for this system because we read the paper and found that this **LSTM** is very powerful for text analytics.

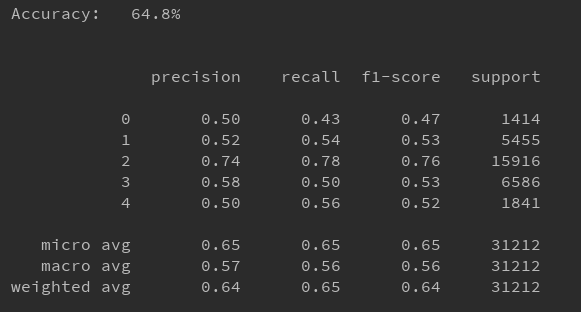
**3.3)** **Split training set**: we divided the training set into 2 parts, 80% for training and 20% for our evaluation so we can get classification report to see how our system was working before using it for the actual test set.

**3.4)** **Pre-processing**: in this step, first we lower case the words in the features and did lemmatization. We decided not to remove **stopwords** out from the feature because some of the stopwords are useful for this sentiment analysis system. Then we turned the feature from word to numerical form using package named **sequence** from **kera.preprocessing**, so our classifier can understand.

**3.5)** **Model setting**: In the model setting, we have added 6 hidden layers which included an embedded layer. Set at 10 epochs. See model summary figure 3 below.

****

**Figure 3**: model summary

**3.6)** **Train model**: then we trained the model using the pre-processed divided training set (80% of the training set) and evaluated the trained model over 20% of validation set and got this classification report as well as the accuracy. See figure 4 below.

**Figure 4**: classification report on the validation set

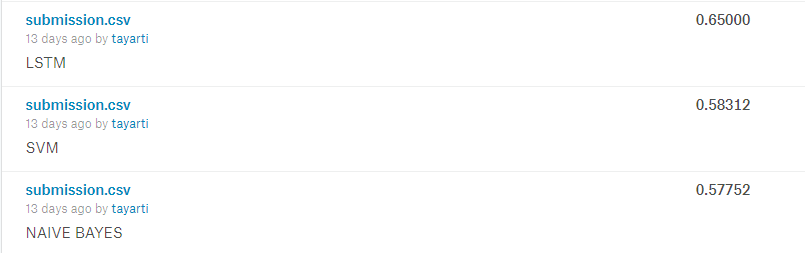
**3.7)** **Evaluate the system**: in this step, we used our model to predict on the test set and then saved the result to CSV file then we submitted to Kaggle competition and got a score of 0.65000 from Kaggle. See figure 5.



**Figure 5**: Kaggle public score for LSTM

**4. Comparison with the state-of-art**

We have tried 3 states of art as we have discussed in task 1 which is LSTM, SVM, and Naïve Bayes to predict on the test set then submitted to Kaggle competition. From the below figure is it seen that LSTM gives the best accuracy that is 65% whereas SVM and Naïve Bayes give 58 and 57 percent respectively.

****

**Figure 6**: Kaggle public score for a different state of the art

**5. Lessons learned**

We have learned that in this sentiment analysis problem some of the stopwords in the feature are useful, it can be the point where the classifier learns which class of each set should belong to. Also, we have learned that in the sentiment analysis it is difficult to make the system achieve high accuracy like the other problem because sentiment analysis is about an expression which is complicated for a machine we have to careful while making the feature selection and pre-processing. We also learned how to use the LSTM network which is very powerful for many types of machine learning problem.