Natural Language Processing| Assignment-03

Amit Kumar Singh
16110011

Link to GitHub repository

https://github.com/aksy1999/NLP_Assignment_3_Sentiment_Analysis

Model Details:

Various Models were tried with more focus on pre-processing the twitter data.

The Pre-processing is defined as follows:

- For each tweet (let say T), all the three parts i.e. Hindi words-T_H, English words-T_E, and Other words-T_O were taken; For each labelled sentiment- S_{-ve}, S_{+ve}, and $S_{neutral}$ all the corresponding words for each tweet were converted to embeddings using Tokenizer and Word2Vec for English, Hindi and Emoji2Vec for other class. After, this we get embeddings for each sentiment and language e.g. Negative Sentiment Embeddings for English Part, Negative Sentiment Embeddings for Hindi Part, Negative Sentiment Embeddings for Other Part.

- Finally, each embedding is averaged to get average negative sentiment English embeddings, average negative sentiment Hindi embeddings, average negative sentiment Other embeddings, and so on.

- It was found that Keras Tokenizer worked better than individual Word2Vec(English, Hindi) and Emoji2Vec. Moreover, the average embedding for Other was ignored to bad performance.

- In best Model, the Keras Tokenizer is used to get embedding for each tweet and the similarity of that embedding with average embedding for negative_english, negative_hindi, positive_eng, positive_english, neutral_eng and neutral_hindi was found giving a 6Dimensional Vector. This vector with a 300D embedding of tweet was used for training the LSTM and CNN. Furthe Decision Tree was also used.

Average Negative Tweet English Embeddings	Average Negative Tweet Hindi Embeddings	Average Negative Tweet Other (only Emoji)
Average Positive Tweet English Embeddings	Average Positive Tweet English Embeddings	Average Positive Tweet English Embeddings
Average Neutral Tweet English Embeddings	Average Neutral Tweet English Embeddings	Average Neutral Tweet Other Embeddings

Each Tweet Embeddings 300 Dimension	Cosine Similarity with red marked above embeddings 6 Dimension

- This 306-dimension vector was used for training an LSTM and CNN model. The results are reported below with model details.

Model Summary:

```
Model: "sequential_9"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_7 (Embedding)      (None, None, 400)         8000000
_________________________________________________________________
lstm_1 (LSTM)                (None, 30)                51720
_________________________________________________________________
dense_8 (Dense)              (None, 300)               9300
_________________________________________________________________
dropout_5 (Dropout)          (None, 300)               0
_________________________________________________________________
activation_7 (Activation)    (None, 300)               0
_________________________________________________________________
dense_9 (Dense)              (None, 3)                 903
_________________________________________________________________
activation_8 (Activation)    (None, 3)                 0
=================================================================
Total params: 8,061,923
Trainable params: 8,061,923
Non-trainable params: 0
_________________________________________________________________
```

Figure 1: LSTM Model Detail

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, None, 400)         8000000
_________________________________________________________________
spatial_dropout1d_2 (Spatial (None, None, 400)         0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, None, 300)         360300
_________________________________________________________________
lambda_2 (Lambda)            (None, 300)               0
_________________________________________________________________
dense_6 (Dense)              (None, 300)               90300
_________________________________________________________________
dropout_4 (Dropout)          (None, 300)               0
_________________________________________________________________
activation_5 (Activation)    (None, 300)               0
_________________________________________________________________
dense_7 (Dense)              (None, 3)                 903
_________________________________________________________________
activation_6 (Activation)    (None, 3)                 0
=================================================================
Total params: 8,451,503
Trainable params: 8,451,503
Non-trainable params: 0
_________________________________________________________________
```

Figure 2: CNN Model Detail

Performance:

Model	Type	Precision	Recall	F1
LSTM	Macro	59.83%	55.52%	56.47%
LSTM	Micro	56.98%	56.98%	56.99%
LSTM	Weighted	58.86%	56.98%	56.73%
CNN	Macro	57.61%	57.17%	57.37%
CNN	Micro	57.09%	57.09%	57.09%
CNN	Weighted	57.21%	57.09%	57.12%

References:

- **For Emoji2Vec**: Ji Ho Park, Peng Xu, and Pascale Fung. 2018. PlusEmo2Vec at SemEval-2018 Task1: Exploiting emotion knowledge from emoji and #hashtags. CoRR abs/1804.08280
- **For LSTM: https://keras.io/examples/lstm_stateful/**
- **For CNN:** Kim, Yoon Convolution Neural Network for Sentiment Analysis(10.3115/v1/D14-1181)