CAIS++ Pre-Semester Project

Aryan Gulati, aryangul@usc.edu

Task

Exploratory data analysis showed that although the 'valence' variable was designed to contain negative, neutral and positive tweets it only takes values corresponding to positive and negative. Therefore, the task was treated as classifying tweets as positive or negative.

Preprocessing

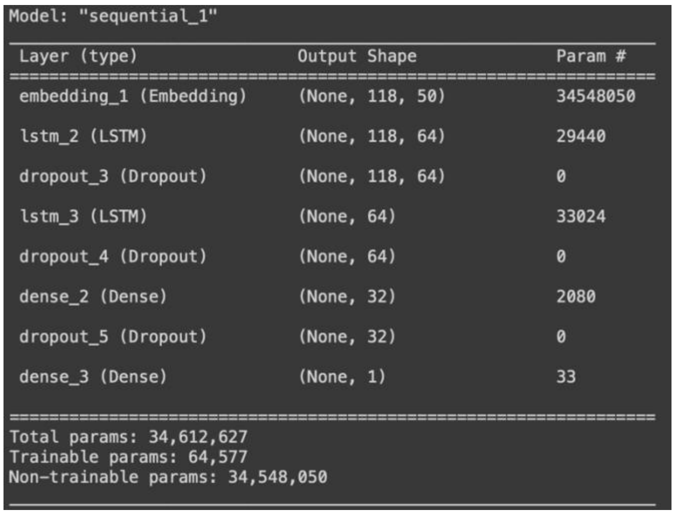
A GloVe embedding was used for this classification task. The steps were as follows:

1. The raw CSV file was loaded into a Pandas dataframe
2. This was converted to a numpy array
3. Keras was used to tokenize and pad the tweets
4. The Twitter GloVe embedding was used to generate the word embeddings for each of the words in each tweet
5. Labels were converted from {0,4} to {0,1}

Model Inputs:

Feature	**Name**	**Type**	**Shape**
Features	X_train	Numpy array of int32	(1280000, 118)
Targets	y_train	Numpy array of int64	(1280000,)

Model Architecture



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 118, 50)           34548050

 lstm_2 (LSTM)               (None, 118, 64)           29440

 dropout_3 (Dropout)         (None, 118, 64)           0

 lstm_3 (LSTM)               (None, 64)                33024

 dropout_4 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 32)                2080

 dropout_5 (Dropout)         (None, 32)                0

 dense_3 (Dense)             (None, 1)                 33

=================================================================
Total params: 34,612,627
Trainable params: 64,577
Non-trainable params: 34,548,050
_________________________________________________________________
```

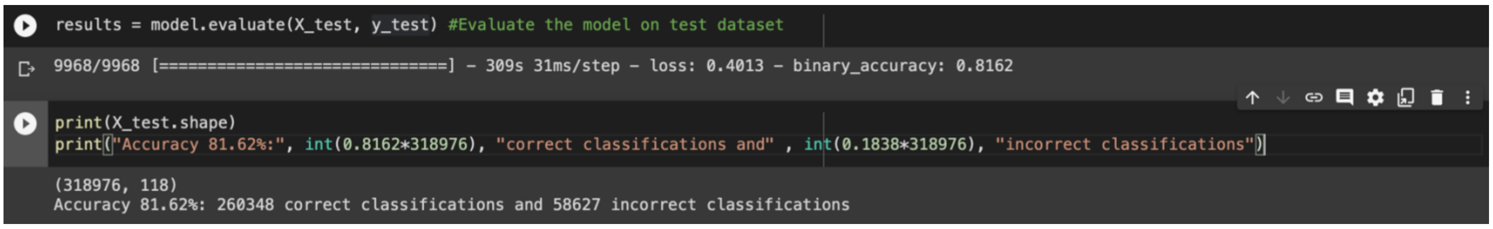
The LSTM layers used a tanh activation, but the dense_2 layer uses a relu activation to reduce computation time (relu activation may be faster as it is a simpler computation). The final dense layer (classification layer) uses a sigmoid activation function. The dropout layers had a d318,976ropout rate of 0.2

The loss function used was binary cross entropy as the problem was treated as binary classification between positive and negative (based on the dataset provided). An Adam optimizer was used because it is commonly used on NLP tasks.

Results

This model was trained for 5 epochs, with a batch size of 256. The final accuracy was 81.62% with a binary cross entropy loss of 0.4013 on the test dataset (consisting of 318,976 examples).

The train/test/validation split used was: 1,280,000/318,976/1024



```
results = model.evaluate(X_test, y_test) #Evaluate the model on test dataset
```

```
9968/9968 [==============================] - 309s 31ms/step - loss: 0.4013 - binary_accuracy: 0.8162
```

```
print(X_test.shape)
print("Accuracy 81.62%:", int(0.8162*318976), "correct classifications and" , int(0.1838*318976), "incorrect classifications")
```

```
(318976, 118)
Accuracy 81.62%: 260348 correct classifications and 58627 incorrect classifications
```