UNIVERSITY OF MISSOURI, KANSAS CITY

PYTHON & DEEP LEARNING – CS 5590

PART – 2 - DEEP LEARNING

LAB – 3 REPORT

BHAVYA TEJA GURIJALA
STUDENT ID: 16220446
CLASS ID: 14

TASK:

The task is to implement the Text Classification with three different models CNN, RNN and LSTM models with new dataset using in python. And also, to compare all the three models and should figure out which is the best model among the three.

1. INTRODUCTION:

The dataset chosen for performing the three different models is the IMDB movie review dataset where we would be performing the classification of the text based on the sentiment of the review whether it belongs to positive or negative. We will be applying three different models CNN, RNN and LSTM on this dataset and we calculate the test accuracy of the model. The model which gets the most is said to be the best model of all.

RNNs are specifically unfolded to a finite sequence of hidden units, and are dependent on both hidden unit and the input. LSTMs specialize the RNNs by augmenting the units with the LSTM. LSTMs are generally used for the time series which defines the past and the present data. Unlike RNNs and LSTMs, CNNs excel in the situations where the data is sampled based on the dimensions. The output parameters are confined to convolutions instead of per unit basis.

2. OBJECTIVE:

The main objective of the task is to perform the text classification on the same dataset by using three different models i.e., CNN, RNN and LSTM and should decide which is the best model of the three. We divide the dataset into test and train samples and perform the modeling on them. After training the model, we test the accuracy of the model. And based on the test accuracies we can decide that which model is the best-fitted model for the text classification.

3. APPROACH/METHOD:

The approach for deciding which performs better for the text classification is to apply the model to the same IMDB review dataset. If we use the same dataset we can easily identify the model which is the best among the three. Mapping of every movie review into a real vector domain which is called as the word embedding. This approach is followed here while building the model. These real vectors are encoded in higher dimensions and based on the similarity of the words they are arranged closer to each other.

4. WORKFLOW:

The workflow of the task is as follows.

a. The task starts with importing the classes and the functions that are used in the program.
b. IMDB dataset will be loaded and is confined to only 5000 words. After that the dataset is split into train and test sets each of 50%.
c. The sentences in the reviews are restricted to 500 words each as the inputs should be of the same length.
d. This restriction is applied to both the train and test sets.
e. Now the dataset is ready to build the model and hence we start with the embedding layer with 32 length vectors for each word.
f. The next which will be applied varies from one model to another. ‘SimpleRNN’ for RNN model, ‘Convolution2D’ for CNN model and ‘LSTM’ for LSTM model.
g. The model will be trained for 3 epochs as it leads to overfitting if we apply for more.
h. An efficient optimizing algorithm called ‘Adam’ is used for this model.
i. By this we estimate the accuracy of the model which is used to evaluate the performance of each model.
j. Based on the accuracy of the model, we can say which is the best model for the text classification.

5. DATASETS:

The dataset that is chosen is a movie review database which is associated with binary sentiment labels for the reviews. The entire dataset consists of 50K reviews which are divided equally into Train and Test sets. We train the model initially with 25K reviews and then test the model with another 25K reviews.

6. PARAMETERS:

The parameters that are used while building the model are:

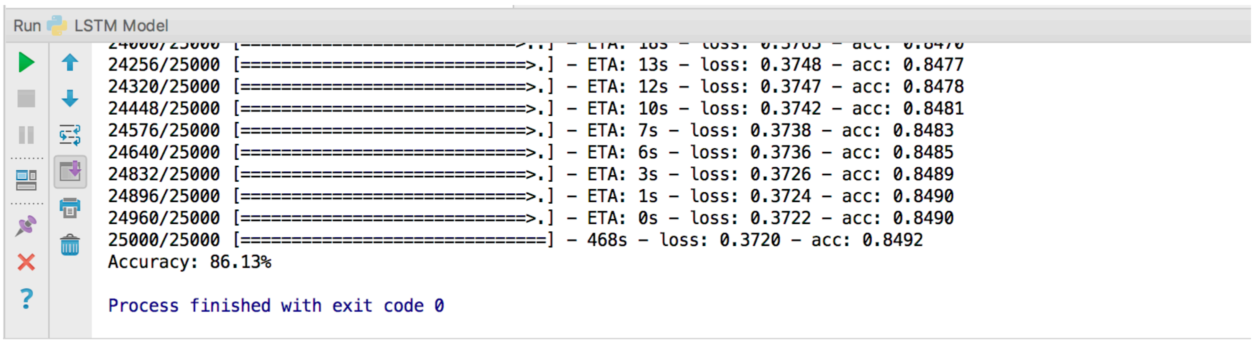
a. The first layer that is applied to the model is an embedded layer which has various parameters which make the model better. The parameters are:
    I. Restricting the words to 5000
    II. The vector length which is 32
    III. The length of the review which is 500
b. The 'sigmoid' activation is added as another layer after the model layer is added.
c. 'binary_crossentrophy' is used to calculate the loss after every iteration.
d. The optimizer that is used for the model is 'Adam'
e. All these parameters are used to calculate the accuracy of the model.

7. EVALUATION & DISCUSSION:

After training the model with three different approaches, all the three models are evaluated based on the Test accuracies.

Here are the screenshots of the accuracies of the three models.

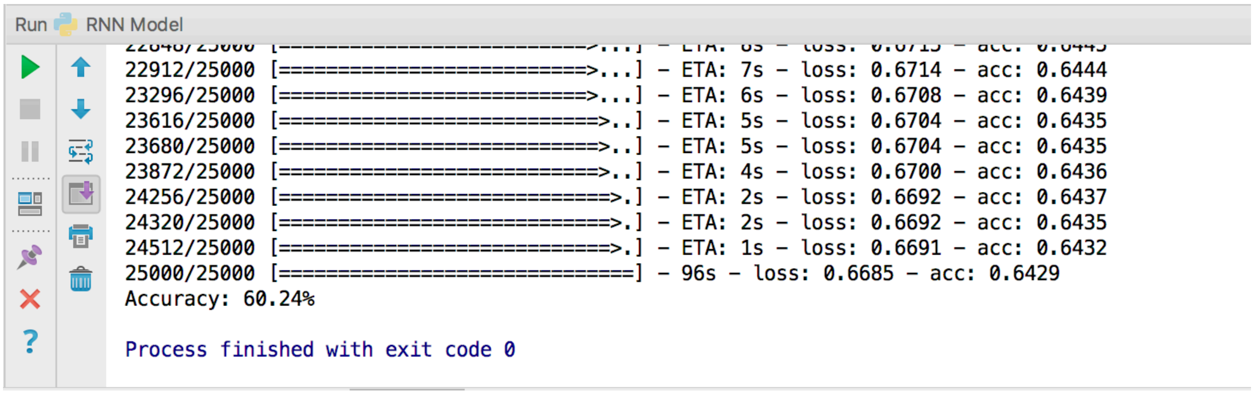
LSTM Model:



```
24256/25000 [============================>.] - ETA: 13s - loss: 0.3748 - acc: 0.8477
24320/25000 [============================>.] - ETA: 12s - loss: 0.3747 - acc: 0.8478
24448/25000 [============================>.] - ETA: 10s - loss: 0.3742 - acc: 0.8481
24576/25000 [============================>.] - ETA: 7s - loss: 0.3738 - acc: 0.8483
24640/25000 [============================>.] - ETA: 6s - loss: 0.3736 - acc: 0.8485
24832/25000 [============================>.] - ETA: 3s - loss: 0.3726 - acc: 0.8489
24896/25000 [============================>.] - ETA: 1s - loss: 0.3724 - acc: 0.8490
24960/25000 [============================>.] - ETA: 0s - loss: 0.3722 - acc: 0.8490
25000/25000 [==============================] - 468s - loss: 0.3720 - acc: 0.8492
Accuracy: 86.13%

Process finished with exit code 0
```

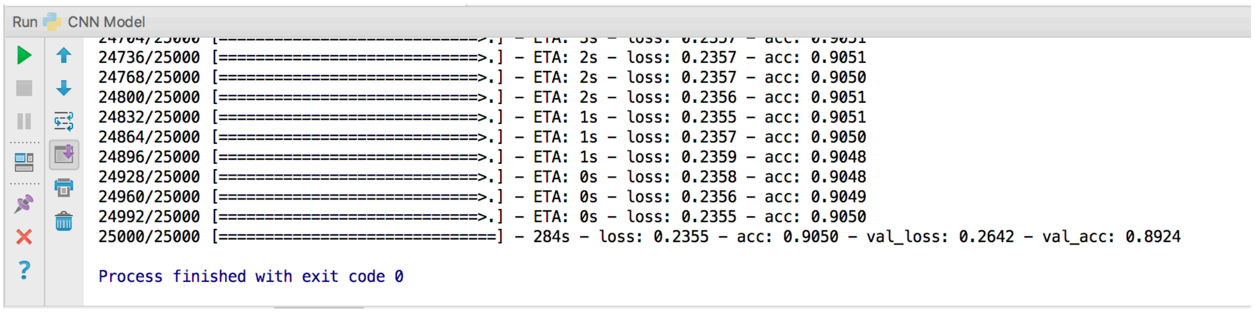
RNN Model:



```
22912/25000 [==========================>...] - ETA: 7s - loss: 0.6714 - acc: 0.6444
23296/25000 [==========================>...] - ETA: 6s - loss: 0.6708 - acc: 0.6439
23616/25000 [===========================>..] - ETA: 5s - loss: 0.6704 - acc: 0.6435
23680/25000 [===========================>..] - ETA: 5s - loss: 0.6704 - acc: 0.6435
23872/25000 [===========================>..] - ETA: 4s - loss: 0.6700 - acc: 0.6436
24256/25000 [============================>.] - ETA: 2s - loss: 0.6692 - acc: 0.6437
24320/25000 [============================>.] - ETA: 2s - loss: 0.6692 - acc: 0.6435
24512/25000 [============================>.] - ETA: 1s - loss: 0.6691 - acc: 0.6432
25000/25000 [==============================] - 96s - loss: 0.6685 - acc: 0.6429
Accuracy: 60.24%

Process finished with exit code 0
```

CNN Model:



```
24736/25000 [============================>.] - ETA: 2s - loss: 0.2357 - acc: 0.9051
24768/25000 [============================>.] - ETA: 2s - loss: 0.2357 - acc: 0.9050
24800/25000 [============================>.] - ETA: 2s - loss: 0.2356 - acc: 0.9051
24832/25000 [============================>.] - ETA: 1s - loss: 0.2355 - acc: 0.9051
24864/25000 [============================>.] - ETA: 1s - loss: 0.2357 - acc: 0.9050
24896/25000 [============================>.] - ETA: 1s - loss: 0.2359 - acc: 0.9048
24928/25000 [============================>.] - ETA: 0s - loss: 0.2358 - acc: 0.9048
24960/25000 [============================>.] - ETA: 0s - loss: 0.2356 - acc: 0.9049
24992/25000 [============================>.] - ETA: 0s - loss: 0.2355 - acc: 0.9050
25000/25000 [==============================] - 284s - loss: 0.2355 - acc: 0.9050 - val_loss: 0.2642 - val_acc: 0.8924

Process finished with exit code 0
```

By looking at the accuracies of the three models we can determine that CNN model performs better than the other two models. LSTM model stands at the second position which is somewhat closer to the CNN Model whereas RNN is way far from the top two models.

8. CONCLUSION:

Eventually, it can be concluded that the CNN model performs better than the two other models. CNN model will easily learn how to recognize the components and also how to aggregate them whereas RNN tries to translate first and then aggregate which makes it a bit slower and less accurate.

9. REFERENCES:

https://datascience.stackexchange.com/questions/11619/rnn-vs-cnn-at-a-high-level

https://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/

https://github.com/jiegzhan/multi-class-text-classification-cnn-rnn