Time Series Prediction (Stock Market Closing Prices of ADBE)

Objective

The goal of the following project is used Keras for designing a model that is capable of predicting the closing prices of the company ADBE. The following project also runs the model using CPU as well as Google Collaborator to compare the speedup factor of Google collaborator against CPU.

Introduction

Time series prediction has a large number of applications in today's world ranging from stock market prices prediction, weather forecasting, sales forecasting etc. It refers to the use of models and methods to study time series data to extract useful statistical information and patterns from the data. While models such as backpropagation are useful in data classification, models in time series prediction predicts future values based on observed values. Long Short Term Memory Networks models consists of cells that remember values over an arbitrary amount of time and three gates that regulate the flow of information in and out of cell. This nature of the LSTM model makes it a suitable model in time series prediction. In the following project I have implement a LSTM model to predict the closing prices of the company ADBE.

Methods

Long Short Term Memory Networks

In the following project I have implemented a kind of residual neural network called long short term memory networks. LSMTs are a suitable model for long term dependency problems as they are capable of retaining past information while making decisions.[1] Below is the diagram that explains the working of a LSTM model.

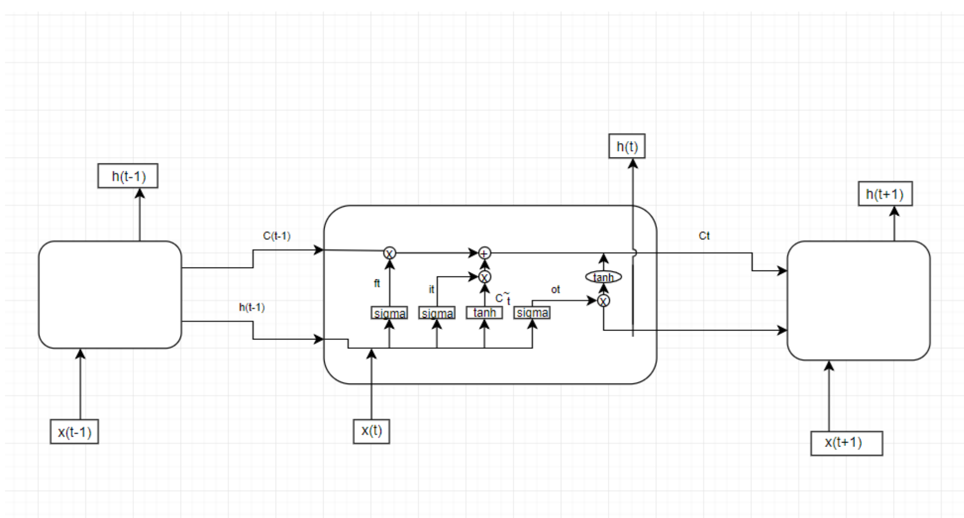

Figure 1: Overall LSTM model

The first step in the LSTM model is the value of the output f_t from the forget gate. The output from the forget gate determines what to keep or throw from the previous cell state. The formula for the output from the forget gate is given below,

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$ [1]

Next the input gate determines the new data that is going to be stored in the cell state. The output of the input gate is given by the following formula,

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$ [1]

The tanh layer then creates a vector of the new candidate values. The output from that layer is given by the following formula

$$\widetilde{C_t} = \tanh(W_c[h_{t-1}, x_t] + b_c)$$ [1]

The new state C_t is given by

$$C_t = f_t \times C_{t-1} + i_t \times \widetilde{C_t}$$ [1]

The output is then decided by the output gate given by,

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$ [1]

The final output is a filtered version of o_t, given by

$$h_t = o_t \times \tanh(C_t)$$ [1]

Thus LSTM models are very suitable for time series predictions since the output from each state is dependent on the output of the previous state and the inputs in the present state. For this reason I have chosen the LSTM model for predicting the closing prices in the project.

Cost Function and Optimization

In the following project I have used the mean squared error as the optimization. The formula for the mean squared is given by

$$MSE = \frac{1}{N}\Sigma_{i=1}^{N}(X_{actual} - X_{predicted})^2$$

The model is optimized using the adam optimizer. The adam optimizer combines the advantage of the adaptive gradient algorithm as well as the root mean squared propagation to optimize the model. [2]

Neural Network Batch Training

I have implemented batch training so that the input is send in batches during each epoch. The model is designed such that the input is send in batches of 7. The model outputs a single value for each batch of 7 input values.

Experimental model

I have implement the LSMT model using Keras in Python. Keras is high-level neural network API in Python. [3] For the following project as per requirements I have run Keras on top on Theano. Keras has a built in LSMT implementation that I have used for the following project. The LSTM model that I have created takes in inputs in batches of 7. The dimensionality of the output space of the LSTM layer is set to 256. [3] Lastly the model has a fully connected dense layer at the end that outputs a single value. 80% of the input data has been used to train the model and the remaining 20% of the input data has been used to test the model. In addition, the testing data has been used as the validation dataset during the training of the model. 20 epochs were used to train the model. I have created the model described above for predicted closing prices of ADBE using both the full and the small dataset provided. I have trained the model on both google collaborator and my personal computer.

Results

Results for model trained using personal computer (CPU)

ADBE Full Data

```
Epoch 1/20
59293/59293 [==============================] - 53s 897us/step - loss: 1.7869e-04 - val_loss: 0.0156
Epoch 2/20
59293/59293 [==============================] - 55s 932us/step - loss: 3.5606e-04 - val_loss: 0.0147
Epoch 3/20
59293/59293 [==============================] - 55s 928us/step - loss: 4.1487e-04 - val_loss: 0.0102.....
Epoch 4/20
59293/59293 [==============================] - 55s 929us/step - loss: 3.4970e-04 - val_loss: 0.0062
Epoch 5/20
59293/59293 [==============================] - 62s 1ms/step - loss: 1.6211e-04 - val_loss: 0.0028
Epoch 6/20
59293/59293 [==============================] - 64s 1ms/step - loss: 4.2678e-05 - val_loss: 7.2179e-04
Epoch 7/20
59293/59293 [==============================] - 64s 1ms/step - loss: 7.5300e-06 - val_loss: 4.9944e-04
Epoch 8/20
59293/59293 [==============================] - 63s 1ms/step - loss: 8.7112e-06 - val_loss: 0.0013
Epoch 9/20
59293/59293 [==============================] - 64s 1ms/step - loss: 1.7985e-05 - val_loss: 7.5993e-04
Epoch 10/20
59293/59293 [==============================] - 64s 1ms/step - loss: 1.0964e-05 - val_loss: 7.0158e-04...
Epoch 11/20
59293/59293 [==============================] - 65s 1ms/step - loss: 1.0073e-05 - val_loss: 7.0804e-04
Epoch 12/20
59293/59293 [==============================] - 64s 1ms/step - loss: 1.0673e-05 - val_loss: 6.2970e-04
Epoch 13/20
59293/59293 [==============================] - 66s 1ms/step - loss: 9.8894e-06 - val_loss: 5.9546e-04
Epoch 14/20
59293/59293 [==============================] - 65s 1ms/step - loss: 9.8470e-06 - val_loss: 5.5015e-04
Epoch 15/20
59293/59293 [==============================] - 53s 894us/step - loss: 9.2293e-06 - val_loss: 5.2236e-04
Epoch 16/20
59293/59293 [==============================] - 56s 944us/step - loss: 9.1368e-06 - val_loss: 5.0215e-04
Epoch 17/20
59293/59293 [==============================] - 58s 984us/step - loss: 9.3741e-06 - val_loss: 4.9307e-04
```

```
Epoch 13/20
59293/59293 [==============================] - 66s 1ms/step - loss: 9.8894e-06 - val_loss: 5.9546e-04
Epoch 14/20
59293/59293 [==============================] - 65s 1ms/step - loss: 9.8470e-06 - val_loss: 5.5015e-04
Epoch 15/20
59293/59293 [==============================] - 53s 894us/step - loss: 9.2293e-06 - val_loss: 5.2236e-04
Epoch 16/20
59293/59293 [==============================] - 56s 944us/step - loss: 9.1368e-06 - val_loss: 5.0215e-04
Epoch 17/20
59293/59293 [==============================] - 58s 984us/step - loss: 9.3741e-06 - val_loss: 4.9307e-04
Epoch 18/20
59293/59293 [==============================] - 57s 964us/step - loss: 9.2153e-06 - val_loss: 4.4263e-04
Epoch 19/20
59293/59293 [==============================] - 58s 981us/step - loss: 8.1298e-06 - val_loss: 4.4939e-04
Epoch 20/20
59293/59293 [==============================] - 61s 1ms/step - loss: 8.4430e-06 - val_loss: 4.2725e-04
```

Figure 2: Training process of the model using the ADBE full data set on CPU

Figure 3: Graph showing the actual closing prices of the company vs the prices predicted by the model during testing for the ADBE full data set on CPU

```
Mean squared error after predictions: 0.642864
```

Figure 4: Mean squared error between the actual closing prices of the company vs the prices predicted by the model during testing for the ADBE full data set on CPU

ADBE Small Dataset

```
Epoch 1/20
32441/32441 [==============================] - 33s 1ms/step - loss: 8.7516e-04 - val_loss: 0.006378e-04
Epoch 2/20
32441/32441 [==============================] - 33s 1ms/step - loss: 0.0012 - val_loss: 0.0073
Epoch 3/20
32441/32441 [==============================] - 32s 1ms/step - loss: 0.0013 - val_loss: 0.0054
Epoch 4/20
32441/32441 [==============================] - 31s 953us/step - loss: 0.0013 - val_loss: 0.0050
Epoch 5/20
32441/32441 [==============================] - 28s 873us/step - loss: 0.0011 - val_loss: 0.0043
Epoch 6/20
32441/32441 [==============================] - 29s 887us/step - loss: 9.2589e-04 - val_loss: 0.0032
Epoch 7/20
32441/32441 [==============================] - 29s 894us/step - loss: 7.0243e-04 - val_loss: 0.0022
Epoch 8/20
32441/32441 [==============================] - 28s 856us/step - loss: 4.8766e-04 - val_loss: 0.0013
Epoch 9/20
32441/32441 [==============================] - 28s 865us/step - loss: 2.9856e-04 - val_loss: 6.4080e-04
Epoch 10/20
32441/32441 [==============================] - 28s 868us/step - loss: 1.8693e-04 - val_loss: 3.2696e-04
Epoch 11/20
32441/32441 [==============================] - 28s 864us/step - loss: 1.4716e-04 - val_loss: 3.4946e-04
Epoch 12/20
32441/32441 [==============================] - 28s 868us/step - loss: 1.3375e-04 - val_loss: 3.2057e-04
Epoch 13/20
32441/32441 [==============================] - 29s 907us/step - loss: 1.2793e-04 - val_loss: 3.3614e-04
Epoch 14/20
32441/32441 [==============================] - 29s 893us/step - loss: 1.1901e-04 - val_loss: 2.7760e-04
Epoch 15/20
32441/32441 [==============================] - 29s 891us/step - loss: 1.0903e-04 - val_loss: 2.0297e-04
Epoch 16/20
32441/32441 [==============================] - 29s 904us/step - loss: 1.0040e-04 - val_loss: 1.6493e-04
Epoch 17/20
32441/32441 [==============================] - 29s 879us/step - loss: 9.3055e-05 - val_loss: 1.1457e-04
Epoch 18/20
32441/32441 [==============================] - 29s 880us/step - loss: 8.7048e-05 - val_loss: 1.0245e-04
Epoch 19/20
32441/32441 [==============================] - 29s 880us/step - loss: 8.1518e-05 - val_loss: 8.4812e-05
```


```
Epoch 19/20
32441/32441 [==============================] - 29s 880us/step - loss: 8.1518e-05 - val_loss: 8.4812e-05
```


```
Epoch 20/20
32441/32441 [==============================] - 29s 879us/step - loss: 7.7622e-05 - val_loss: 8.2664e-05
```

Figure 5: Training process of the model using the ADBE small data set on CPU

Figure 6: Graph showing the actual closing prices of the company vs the prices predicted by the model during testing for the ADBE small data set on the CPU

```
Mean squared error after predictions: 0.020100
```

Figure 7: Mean squared error between the actual closing prices of the company vs the prices predicted by the model during testing for the ADBE small data set on the CPU

Results obtained using Google Collaborator to train the model

ADBE Full Dataset

```
Epoch 1/20
59293/59293 [==============================] - 44s 743us/step - loss: 1.7803e-04 - val_loss: 0.0151
Epoch 2/20
59293/59293 [==============================] - 44s 748us/step - loss: 3.7667e-04 - val_loss: 0.0141
Epoch 3/20
59293/59293 [==============================] - 43s 732us/step - loss: 4.1907e-04 - val_loss: 0.0118
Epoch 4/20
59293/59293 [==============================] - 43s 729us/step - loss: 3.9846e-04 - val_loss: 0.0062
Epoch 5/20
59293/59293 [==============================] - 43s 729us/step - loss: 1.8621e-04 - val_loss: 0.0025
Epoch 6/20
59293/59293 [==============================] - 44s 734us/step - loss: 3.8874e-05 - val_loss: 6.3009e-04
Epoch 7/20
59293/59293 [==============================] - 46s 770us/step - loss: 6.2182e-06 - val_loss: 5.0122e-04
Epoch 8/20
59293/59293 [==============================] - 43s 723us/step - loss: 8.8210e-06 - val_loss: 0.0015
Epoch 9/20
59293/59293 [==============================] - 44s 743us/step - loss: 1.7712e-05 - val_loss: 6.8604e-04
Epoch 10/20
59293/59293 [==============================] - 43s 726us/step - loss: 7.2227e-06 - val_loss: 6.7687e-04
Epoch 11/20
59293/59293 [==============================] - 44s 741us/step - loss: 8.3693e-06 - val_loss: 8.1689e-04
Epoch 12/20
59293/59293 [==============================] - 44s 745us/step - loss: 1.0370e-05 - val_loss: 6.8509e-04
Epoch 13/20
59293/59293 [==============================] - 43s 731us/step - loss: 9.7464e-06 - val_loss: 6.3084e-04
Epoch 14/20
59293/59293 [==============================] - 45s 766us/step - loss: 9.3485e-06 - val_loss: 5.9850e-04

Epoch 15/20
59293/59293 [==============================] - 43s 730us/step - loss: 9.0470e-06 - val_loss: 5.6686e-04
Epoch 16/20
59293/59293 [==============================] - 44s 748us/step - loss: 8.8832e-06 - val_loss: 5.3886e-04
Epoch 17/20
59293/59293 [==============================] - 43s 723us/step - loss: 8.7887e-06 - val_loss: 5.0651e-04
Epoch 18/20
59293/59293 [==============================] - 43s 727us/step - loss: 8.5206e-06 - val_loss: 4.8574e-04
Epoch 19/20
59293/59293 [==============================] - 43s 726us/step - loss: 8.3885e-06 - val_loss: 4.6501e-04
Epoch 20/20
59293/59293 [==============================] - 43s 730us/step - loss: 8.2093e-06 - val_loss: 4.4616e-04
```

Figure 8: Training process of the model using the ADBE full data set on Google Collaborator

Figure 9: Graph showing the actual closing prices of the company vs the prices predicted by the model during testing for the ADBE full data set on Google Collaborator

```
Mean squared error after predictions: 0.671323
```

Figure 10: Mean squared error between the actual closing prices of the company vs the prices predicted by the model during testing for the ADBE full data set on Google Collaborator

ADBE Small Dataset

```
Epoch 1/20
32441/32441 [==============================] - 27s 827us/step - loss: 8.1123e-04 - val_loss: 0.0053
Epoch 2/20
32441/32441 [==============================] - 27s 835us/step - loss: 0.0012 - val_loss: 0.0076
Epoch 3/20
32441/32441 [==============================] - 25s 767us/step - loss: 0.0013 - val_loss: 0.0056
Epoch 4/20
32441/32441 [==============================] - 26s 786us/step - loss: 0.0013 - val_loss: 0.0053
Epoch 5/20
32441/32441 [==============================] - 25s 760us/step - loss: 0.0012 - val_loss: 0.0049
Epoch 6/20
32441/32441 [==============================] - 25s 767us/step - loss: 9.8416e-04 - val_loss: 0.0035
Epoch 7/20
32441/32441 [==============================] - 25s 756us/step - loss: 7.3704e-04 - val_loss: 0.0024
Epoch 8/20
32441/32441 [==============================] - 24s 751us/step - loss: 5.0547e-04 - val_loss: 0.0014
Epoch 9/20
32441/32441 [==============================] - 24s 743us/step - loss: 3.0594e-04 - val_loss: 6.7142e-04
Epoch 10/20
32441/32441 [==============================] - 24s 750us/step - loss: 1.8287e-04 - val_loss: 2.9941e-04
Epoch 11/20
32441/32441 [==============================] - 24s 752us/step - loss: 1.3316e-04 - val_loss: 2.9045e-04
Epoch 12/20
32441/32441 [==============================] - 24s 752us/step - loss: 1.2065e-04 - val_loss: 2.4227e-04
Epoch 13/20
32441/32441 [==============================] - 24s 746us/step - loss: 1.1055e-04 - val_loss: 1.6953e-04
Epoch 14/20
32441/32441 [==============================] - 25s 782us/step - loss: 9.6879e-05 - val_loss: 9.5693e-05
Epoch 15/20
32441/32441 [==============================] - 26s 793us/step - loss: 8.6804e-05 - val_loss: 3.7691e-05
Epoch 16/20
32441/32441 [==============================] - 24s 745us/step - loss: 8.0275e-05 - val_loss: 3.5923e-05
Epoch 17/20
32441/32441 [==============================] - 25s 775us/step - loss: 8.2553e-05 - val_loss: 3.1704e-05
```


```
Epoch 17/20
32441/32441 [==============================] - 25s 775us/step - loss: 8.2553e-05 - val_loss: 3.1704e-05
```


```
Epoch 18/20
32441/32441 [==============================] - 24s 734us/step - loss: 7.9576e-05 - val_loss: 3.0806e-05
Epoch 19/20
32441/32441 [==============================] - 24s 728us/step - loss: 7.9990e-05 - val_loss: 3.1921e-05
Epoch 20/20
32441/32441 [==============================] - 24s 732us/step - loss: 8.0194e-05 - val_loss: 2.7351e-05
```

Figure 11: Training process of the model using the ADBE small data set on Google Collaborator

Figure 12: Graph showing the actual closing prices of the company vs the prices predicted by the model during testing for the ADBE small data set on Google Collaborator

```
Mean squared error after predictions: 0.006651
```

Figure 13: Mean squared error between the actual closing prices of the company vs the prices predicted by the model during testing for the ADBE small data set on Google Collaborator

Time comparison between runtime on CPU and Google Collaborator

ADBE Full Dataset

```
Epoch 1/20
59293/59293 [==============================] - 53s 897us/step - loss: 1.7869e-04 - val_loss: 0.0156
Epoch 2/20
59293/59293 [==============================] - 55s 932us/step - loss: 3.5606e-04 - val_loss: 0.0147
Epoch 3/20
59293/59293 [==============================] - 55s 928us/step - loss: 4.1487e-04 - val_loss: 0.0102.....
Epoch 4/20
59293/59293 [==============================] - 55s 929us/step - loss: 3.4970e-04 - val_loss: 0.0062
Epoch 5/20
59293/59293 [==============================] - 62s 1ms/step - loss: 1.6211e-04 - val_loss: 0.0028
Epoch 6/20
59293/59293 [==============================] - 64s 1ms/step - loss: 4.2678e-05 - val_loss: 7.2179e-04
Epoch 7/20
59293/59293 [==============================] - 64s 1ms/step - loss: 7.5300e-06 - val_loss: 4.9944e-04
Epoch 8/20
59293/59293 [==============================] - 63s 1ms/step - loss: 8.7112e-06 - val_loss: 0.0013
Epoch 9/20
59293/59293 [==============================] - 64s 1ms/step - loss: 1.7985e-05 - val_loss: 7.5993e-04
Epoch 10/20
59293/59293 [==============================] - 64s 1ms/step - loss: 1.0964e-05 - val_loss: 7.0158e-04...
Epoch 11/20
59293/59293 [==============================] - 65s 1ms/step - loss: 1.0073e-05 - val_loss: 7.0804e-04
Epoch 12/20
59293/59293 [==============================] - 64s 1ms/step - loss: 1.0673e-05 - val_loss: 6.2970e-04
Epoch 13/20
59293/59293 [==============================] - 66s 1ms/step - loss: 9.8894e-06 - val_loss: 5.9546e-04
Epoch 14/20
59293/59293 [==============================] - 65s 1ms/step - loss: 9.8470e-06 - val_loss: 5.5015e-04
Epoch 15/20
59293/59293 [==============================] - 53s 894us/step - loss: 9.2293e-06 - val_loss: 5.2236e-04
Epoch 16/20
59293/59293 [==============================] - 56s 944us/step - loss: 9.1368e-06 - val_loss: 5.0215e-04
Epoch 17/20
59293/59293 [==============================] - 58s 984us/step - loss: 9.3741e-06 - val_loss: 4.9307e-04
```

Figure 14: Time taken for training (first 19 epochs) the model using the ADBE full data set using CPU

```
Epoch 1/20
59293/59293 [==============================] - 44s 743us/step - loss: 1.7803e-04 - val_loss: 0.0151
Epoch 2/20
59293/59293 [==============================] - 44s 748us/step - loss: 3.7667e-04 - val_loss: 0.0141
Epoch 3/20
59293/59293 [==============================] - 43s 732us/step - loss: 4.1907e-04 - val_loss: 0.0118
Epoch 4/20
59293/59293 [==============================] - 43s 729us/step - loss: 3.9846e-04 - val_loss: 0.0062
Epoch 5/20
59293/59293 [==============================] - 43s 729us/step - loss: 1.8621e-04 - val_loss: 0.0025
Epoch 6/20
59293/59293 [==============================] - 44s 734us/step - loss: 3.8874e-05 - val_loss: 6.3009e-04
Epoch 7/20
59293/59293 [==============================] - 46s 770us/step - loss: 6.2182e-06 - val_loss: 5.0122e-04
Epoch 8/20
59293/59293 [==============================] - 43s 723us/step - loss: 8.8210e-06 - val_loss: 0.0015
Epoch 9/20
59293/59293 [==============================] - 44s 743us/step - loss: 1.7712e-05 - val_loss: 6.8604e-04
Epoch 10/20
59293/59293 [==============================] - 43s 726us/step - loss: 7.2227e-06 - val_loss: 6.7687e-04
Epoch 11/20
59293/59293 [==============================] - 44s 741us/step - loss: 8.3693e-06 - val_loss: 8.1689e-04
Epoch 12/20
59293/59293 [==============================] - 44s 745us/step - loss: 1.0370e-05 - val_loss: 6.8509e-04
Epoch 13/20
59293/59293 [==============================] - 43s 731us/step - loss: 9.7464e-06 - val_loss: 6.3084e-04
Epoch 14/20
59293/59293 [==============================] - 45s 766us/step - loss: 9.3485e-06 - val_loss: 5.9850e-04
```

Figure 15: Time taken for training (first 19 epochs) the model using the ADBE full data set using Google Collaborator

ADBE Small Dataset

```
Epoch 1/20
32441/32441 [==============================] - 33s 1ms/step - loss: 8.7516e-04 - val_loss: 0.006378e-04
Epoch 2/20
32441/32441 [==============================] - 33s 1ms/step - loss: 0.0012 - val_loss: 0.0073
Epoch 3/20
32441/32441 [==============================] - 32s 1ms/step - loss: 0.0013 - val_loss: 0.0054
Epoch 4/20
32441/32441 [==============================] - 31s 953us/step - loss: 0.0013 - val_loss: 0.0050
Epoch 5/20
32441/32441 [==============================] - 28s 873us/step - loss: 0.0011 - val_loss: 0.0043
Epoch 6/20
32441/32441 [==============================] - 29s 887us/step - loss: 9.2589e-04 - val_loss: 0.0032
Epoch 7/20
32441/32441 [==============================] - 29s 894us/step - loss: 7.0243e-04 - val_loss: 0.0022
Epoch 8/20
32441/32441 [==============================] - 28s 856us/step - loss: 4.8766e-04 - val_loss: 0.0013
Epoch 9/20
32441/32441 [==============================] - 28s 865us/step - loss: 2.9856e-04 - val_loss: 6.4080e-04
Epoch 10/20
32441/32441 [==============================] - 28s 868us/step - loss: 1.8693e-04 - val_loss: 3.2696e-04
Epoch 11/20
32441/32441 [==============================] - 28s 864us/step - loss: 1.4716e-04 - val_loss: 3.4946e-04
Epoch 12/20
32441/32441 [==============================] - 28s 868us/step - loss: 1.3375e-04 - val_loss: 3.2057e-04
Epoch 13/20
32441/32441 [==============================] - 29s 907us/step - loss: 1.2793e-04 - val_loss: 3.3614e-04
Epoch 14/20
32441/32441 [==============================] - 29s 893us/step - loss: 1.1901e-04 - val_loss: 2.7760e-04
Epoch 15/20
32441/32441 [==============================] - 29s 891us/step - loss: 1.0903e-04 - val_loss: 2.0297e-04
Epoch 16/20
32441/32441 [==============================] - 29s 904us/step - loss: 1.0040e-04 - val_loss: 1.6493e-04
Epoch 17/20
32441/32441 [==============================] - 29s 879us/step - loss: 9.3055e-05 - val_loss: 1.1457e-04
Epoch 18/20
32441/32441 [==============================] - 29s 880us/step - loss: 8.7048e-05 - val_loss: 1.0245e-04
Epoch 19/20
32441/32441 [==============================] - 29s 880us/step - loss: 8.1518e-05 - val_loss: 8.4812e-05
```

Figure 16: Time taken for training (first 19 epochs) the model using the ADBE small data set using CPU

```
Epoch 1/20
32441/32441 [==============================] - 27s 827us/step - loss: 8.1123e-04 - val_loss: 0.0053
Epoch 2/20
32441/32441 [==============================] - 27s 835us/step - loss: 0.0012 - val_loss: 0.0076
Epoch 3/20
32441/32441 [==============================] - 25s 767us/step - loss: 0.0013 - val_loss: 0.0056
Epoch 4/20
32441/32441 [==============================] - 26s 786us/step - loss: 0.0013 - val_loss: 0.0053
Epoch 5/20
32441/32441 [==============================] - 25s 760us/step - loss: 0.0012 - val_loss: 0.0049
Epoch 6/20
32441/32441 [==============================] - 25s 767us/step - loss: 9.8416e-04 - val_loss: 0.0035
Epoch 7/20
32441/32441 [==============================] - 25s 756us/step - loss: 7.3704e-04 - val_loss: 0.0024
Epoch 8/20
32441/32441 [==============================] - 24s 751us/step - loss: 5.0547e-04 - val_loss: 0.0014
Epoch 9/20
32441/32441 [==============================] - 24s 743us/step - loss: 3.0594e-04 - val_loss: 6.7142e-04
Epoch 10/20
32441/32441 [==============================] - 24s 750us/step - loss: 1.8287e-04 - val_loss: 2.9941e-04
Epoch 11/20
32441/32441 [==============================] - 24s 752us/step - loss: 1.3316e-04 - val_loss: 2.9045e-04
Epoch 12/20
32441/32441 [==============================] - 24s 752us/step - loss: 1.2065e-04 - val_loss: 2.4227e-04
Epoch 13/20
32441/32441 [==============================] - 24s 746us/step - loss: 1.1055e-04 - val_loss: 1.6953e-04
Epoch 14/20
32441/32441 [==============================] - 25s 782us/step - loss: 9.6879e-05 - val_loss: 9.5693e-05
Epoch 15/20
32441/32441 [==============================] - 26s 793us/step - loss: 8.6804e-05 - val_loss: 3.7691e-05
Epoch 16/20
32441/32441 [==============================] - 24s 745us/step - loss: 8.0275e-05 - val_loss: 3.5923e-05
Epoch 17/20
32441/32441 [==============================] - 25s 775us/step - loss: 8.2553e-05 - val_loss: 3.1704e-05
```

Figure 17: Time taken for training (first 19 epochs) the model using the ADBE small data set using Google Collaborator

Conclusion and Observations

In the case of the full data set the model is seen to have a mean squared error of around 0.67 thus showing a relative high accuracy rate. The accuracy of the full dataset is seen to deteriorate after the first 4500 days. This is in line with the working of the LSTM model as the LSTM models are not seen to have very accurate rate for prolonged amounts of time.

In the case of the small dataset the model is seen to have a really low mean squared error of around 0.0066. This can be see clearly from the graphs provided above as the actual and predicted values perfectly overlap one another.

In addition to the mean squared error the trend of the actual and the predicted data was tracked. This was done so as to calculate the number of predicted data points that followed the actual data points. In case of the small dataset, well over 60% of the predicted data points was seen to follow the trend of the actual data points. In case of the full dataset, well over 50% of the predicted data points was seen to follow the trend of the actual data points. The lower value for the full data set is due to the deviation of the predicted data points from the actual data points in the later stages of testing.

As can be seen from the graphs above training the model on google collaborator gave very similar outputs to that of training the models on the CPU. In case of the full data set google collaborator is seen to have a runtime that is approximate 1.25 faster than that of the CPU for one epoch. In case of the small data set google collaborator is seen to have a runtime that is approximate 1.22 faster than that of the CPU for one epoch.

Overall the accuracy of the model could be further improved by including more features and increasing the training time. However, the mean squared error is seen to increase after around 200 epochs.

Source Code

```
import numpy as np

import pandas as pd

from keras.models import Sequential

from keras.layers import LSTM,Dense

from sklearn.preprocessing import MinMaxScaler

import matplotlib.pyplot as plt

from sklearn.metrics import mean_squared_error


#Function for processing training and testing data

def getdata(data,ln):

   X,Y = [],[]

   for i in range( len(data)-ln-1 ):

      X.append(data[ i:(i+ln),0] )

      Y.append(data[ (i+ln),0] )

   return np.array(X),np.array(Y)


#Function for loading input data

data = pd.read_csv('C:/Users/namra/Desktop/Fall 2018/ECE 629/project/nasdaq100_padding.csv')

dt = data['ADBE']

dt.dropna(inplace=True)
```

```
#Function for scaling input data
scale = MinMaxScaler()
dt = dt.values.reshape(dt.shape[0],1)
dt = scale.fit_transform(dt)
dt

#Spliting input data into training and testing data
X,y = getdata(dt,7)
X_train,X_test = X[:int(X.shape[0]*0.80)],X[int(X.shape[0]*0.80):]
y_train,y_test = y[:int(y.shape[0]*0.80)],y[int(y.shape[0]*0.80):]

#Parameters of the model
model = Sequential()
model.add(LSTM(256,input_shape=(7,1)))
model.add(Dense(1))
model.compile(optimizer='adam',loss='mse')

#Training the model
X_train = X_train.reshape((X_train.shape[0],X_train.shape[1],1))
X_test = X_test.reshape((X_test.shape[0],X_test.shape[1],1))
history = model.fit(X_train,y_train,epochs = 20,validation_data=(X_test,y_test),shuffle=False)

#Testing the model using test data
Xt = model.predict(X_test)

#Plotting actual data vs predicted data
plt.rcParams.update({'font.size': 18})
plt.plot(scale.inverse_transform(y_test.reshape(-1,1)), color='red', label='Actual value')
plt.plot(scale.inverse_transform(Xt), color='green', label='Predicted Value')
```

```
plt.legend(bbox_to_anchor=(1.05, 1), loc=1, borderaxespad=0.)
plt.xlabel('Number of days')
plt.ylabel('Closing Prices')
plt.title('Closing prices prediction for ADBE (Small dataset)')
plt.show()

#Calculating the mean squared error
xin = scale.inverse_transform(Xt)
yin = scale.inverse_transform(y_test.reshape(-1,1))
final_mse  = mean_squared_error(xin, yin)
print ("Mean squared error after predictions: %f"%(final_mse))

#Tracking the trends between actual and predicted data
Xt_int = scale.inverse_transform(Xt.reshape(-1,1))
yt_int = scale.inverse_transform(y_test.reshape(-1,1))
xt_class = [Xt_int[i+1]-Xt_int[i]for i in range (0,len(Xt_int)-1)]
yt_class = [yt_int[i+1]-yt_int[i] for i in range (0,len(yt_int)-1)]

cnt = 0
for i in range (len(yt_class)):
  if yt_class[i] <= 0 and xt_class[i] <= 0:
    cnt = cnt + 1
  if yt_class[i] > 0 and xt_class[i] > 0:
    cnt = cnt + 1


##MSE
xin = scale.inverse_transform(Xt)
yin = scale.inverse_transform(y_test.reshape(-1,1))
```

```
final_mse  = mean_squared_error(xin, yin)

print ("Mean squared error after predictions: %f"%(final_mse))
```

References

"Understanding LSTM Networks," Understanding LSTM Networks -- colah's blog. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: 06-Dec-2018].

"Gentle Introduction to the Adam Optimization Algorithm for Deep Learning," Machine Learning Mastery, 25-Nov-2018. [Online]. Available: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/. [Accessed: 06-Dec-2018].

"Keras: The Python Deep Learning library," Keras Documentation. [Online]. Available: https://keras.io/. [Accessed: 06-Dec-2018].

RSNA Pneumonia Detection Challenge | Kaggle. [Online]. Available: https://www.kaggle.com/amarpreetsingh/stock-prediction-lstm-using-keras. [Accessed: 06-Dec-2018].