STOCK MARKET PRICE PREDICTION

AIML PROJECT

SIVA KISHORE-200050042

YESHASH CHANDRA-200050031

RUTHVIKA-200050018

SHIVA SAI PAMAR-200050150

→***Deciding which deep learning method to use?***

We have learned deep learning techniques like CNN and Feed forward neural networks,

But we are not going to use this method. Why?

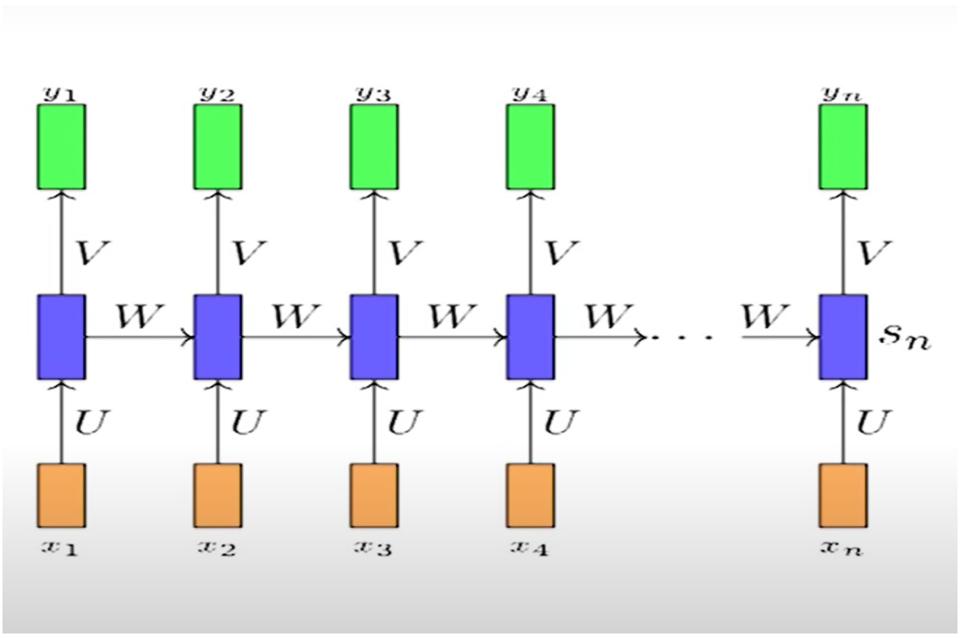
1.It takes fixed size inputs

2.The inputs are completely independent of each other.

→***Using RNN:***

But here we are predicting stock market, which depends on the previous stock trend . This is one of the sequential learning problem. So we decided to use Recurrent Neural networks which are used to solve sequential problems like Whatsapp autocorrect whose input size is not fixed and the inputs are dependent , like if 'z' is the first letter and 'e' is the second letter then most probably our next letter is b (zebra) our RNN model learns and predicts the next letter here by memorising the previous letters . Similarly in our market stock prediction we first read data of certain number of days and predict the market stock value on the next day .

So our RNN Model Diagram is as follows,



Here W is our Recurrent connection , at each stage we execute the same function and this network

Doesn't depend on the number inputs and dependence is maintained by W. The parametrs and the function notations are as follows,

$$s_i = \sigma(Ux + Ws_{i-1} + b)$$
$$y_i = \sigma(Vs_i + c)$$
$$or$$
$$y_i = f(x_i, s_i, W, U, V)$$

But still we are not going to use just RNN, because there is a problem while backpropagation

There is a term ‘ϒλ’ (gamma ,lambda) where gamma , lambda are

$$\begin{aligned}\left\|\frac{\partial s_j}{\partial s_{j-1}}\right\| &= \left\|diag(\sigma'(a_j))W\right\| \\ &\leq \left\|diag(\sigma'(a_j))\right\| \|W\|\end{aligned}$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma'(a_j)$ is bounded

$$\begin{aligned}\sigma'(a_j) &\leq \frac{1}{4} = \gamma\,[\text{if } \sigma \text{ is logistic }] \\ &\leq 1 = \gamma\,[\text{if } \sigma \text{ is tanh }] \\ \left\|\frac{\partial s_j}{\partial s_{j-1}}\right\| &\leq \gamma\,\|W\| \\ &\leq \gamma\lambda\end{aligned}$$

Here the term ‘ϒλ’ is multiplied sevaral times while calculating gradients for backpropagation

<u>Overview of what we finally did in deep learning for stock markets:</u>

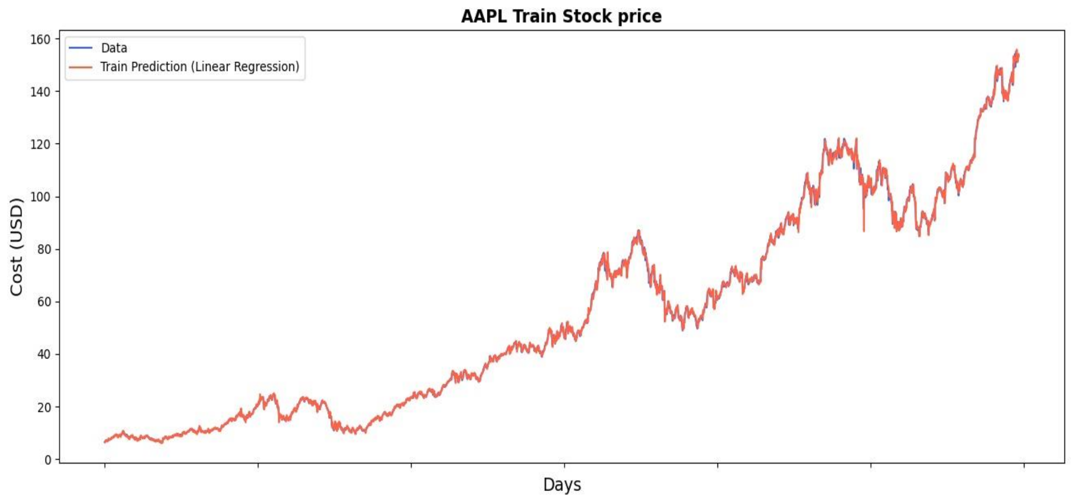
So we have this vanishing and exploding problem, hence we are going to use the concept of selective read , selective write and selective forget through which the algorithms LSTM and GRU have come up . we analysed both LSTM and GRU and have seen the differences through plots , though datas from various companies like Apple, Microsoft, Google , TATA , Bitcoin . In this comparision LSTM model conatains 2 layers of LSTM and Dropout with 32 units and then a linear layer instead dense layer . Final model contains 3 LSTM and dropout layers with appx 50 units followed by a dense layer.

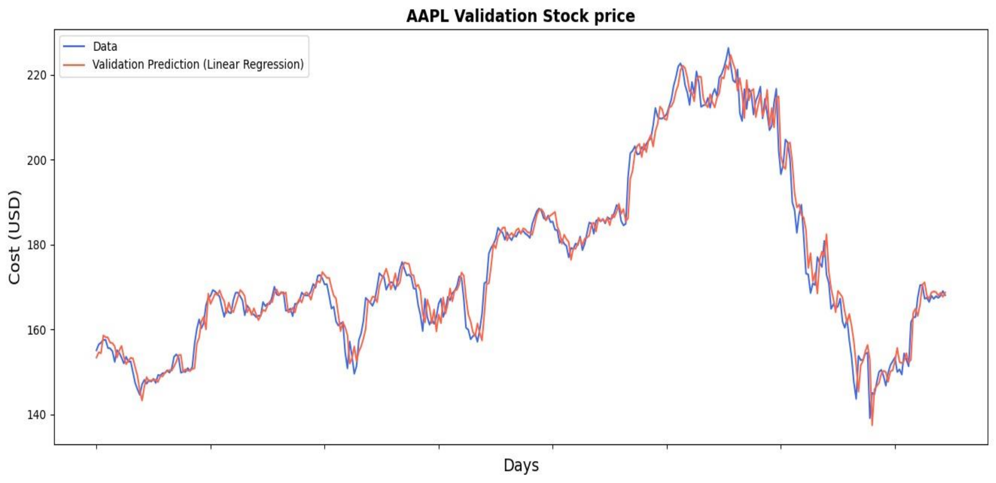
<u>Also did the prediction using linear regression :</u>

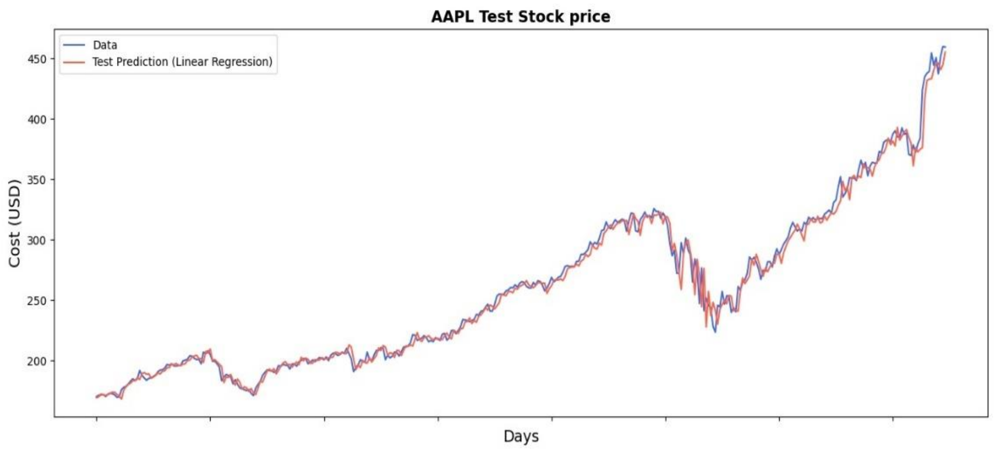
We have comapred all these deep learning results with a linear regression model also. This linear regression model takes data splits into training and validation, test datasets then calculates error on both the validation and test datasets also for visualization plots are also produced . Here in the data we have around 30 feautre values , like the data is in month end or start , quarter end or start , year ending or starting , is it a leap year? , upper band , lower band, high, low , QQQ_MA10,QQQ_MA20,QQQ_MA50,SnP_Close,SnP(t-1)),SnP(t-5),DJIA_Close,DJIA(t-1)),DJIA(t-5),etc.

Total data is split into 80% training, 10% validation , 10% testing. Obviously we will see a high accuracy in the plots of training data and a bit low accuracy in the validarion and test data.

The following the plots of the linear regression model trained over 80% of the data taken from kaggle.







COMING BACK TO DEEP LEARNING TECHNIQUES:

1. *Comaprision between LSTM and GRU .*

LSTM stands fro long short term memory and GRU stands fro gated recurrent units .

LSTM Vs GRU

Now we have seen the operation of both the layers to combat the problem of vanishing gradient. So you might wonder which one is to use? As GRU is relatively approaching its tradeoffs haven't been discussed yet.

According to empirical evaluation, there is no a clear winner. The basic idea of using a getting mechanism to learn long term dependencies is the same as in LSTM.

What is the difference between GRU & LSTM?

The few differencing points are as follows:

The GRU has two gates, LSTM has three gates

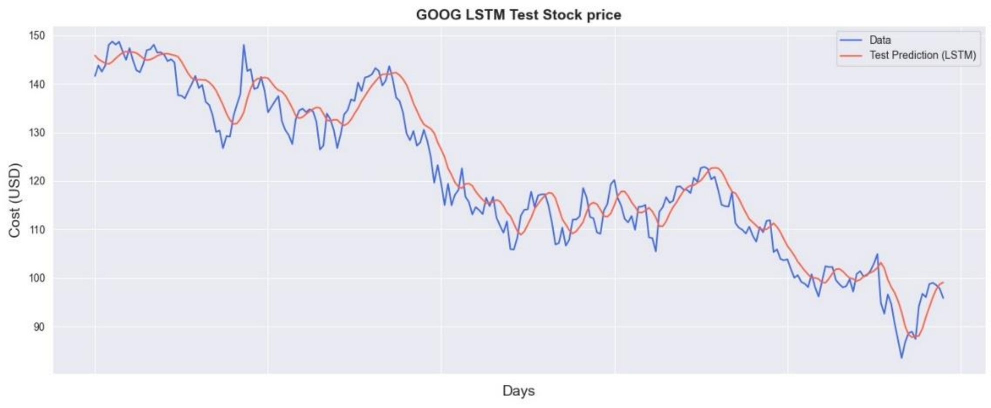
GRU does not possess any internal memory, they don't have an output gate that is present in LSTM

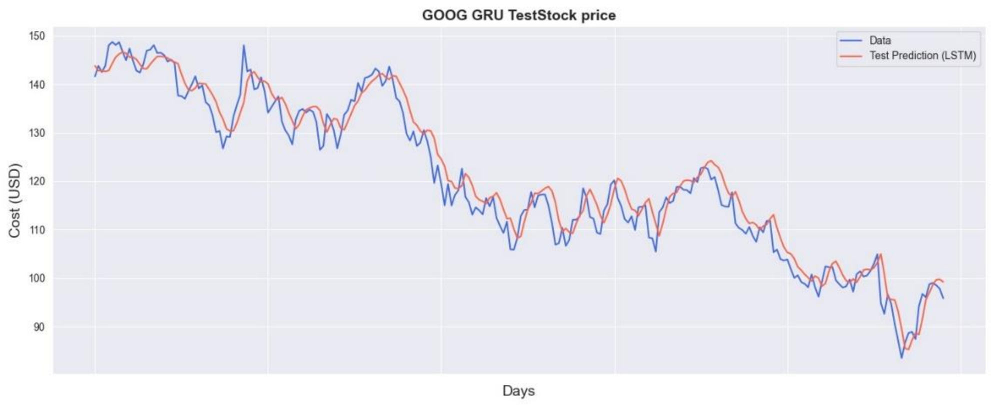
In LSTM the input gate and target gate are coupled by an update gate and in GRU reset gate is applied directly to the previous hidden state. In LSTM the responsibility of reset gate is taken by the two gates i.e., input and target.

Conclusion:

Through this project, we have understood the basic difference between the RNN, LSTM and GRU units. From working of both layers i.e., LSTM and GRU, GRU uses less training parameter and therefore uses less memory and executes faster than LSTM whereas LSTM is more accurate on a larger dataset. One can choose LSTM if you are dealing with large sequences and accuracy is concerned, GRU is used when you have less memory consumption and want faster results.

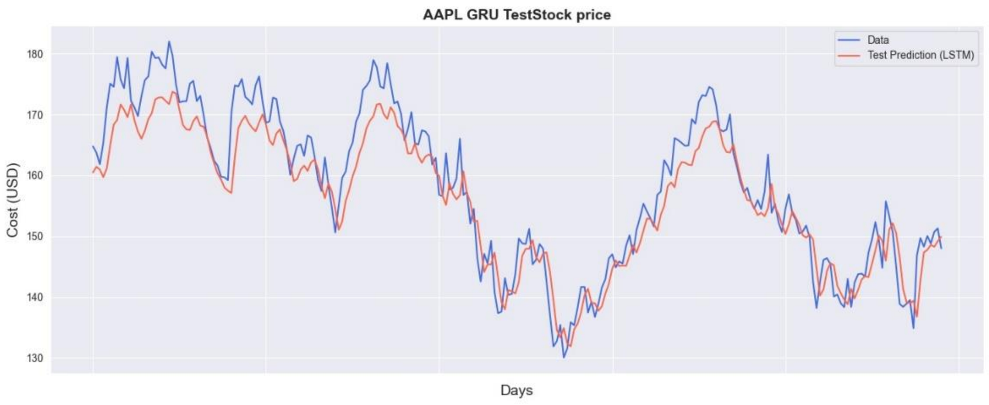
Following are the observations of performance of GRU and LSTM over APPLE and GOOGLE datasets (observations over the companies tatamotors, google, apple, microsoft,bitcoin have been uploded in google drive link [LSTM_vs_GRU - Google Drive]() .

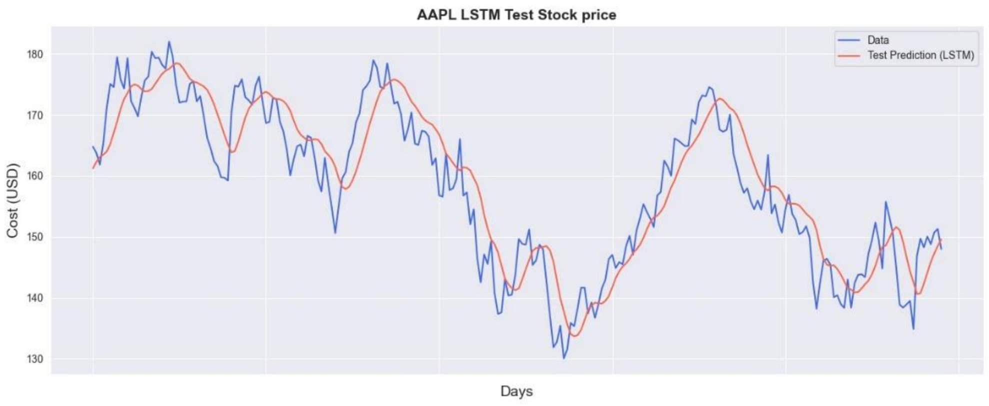


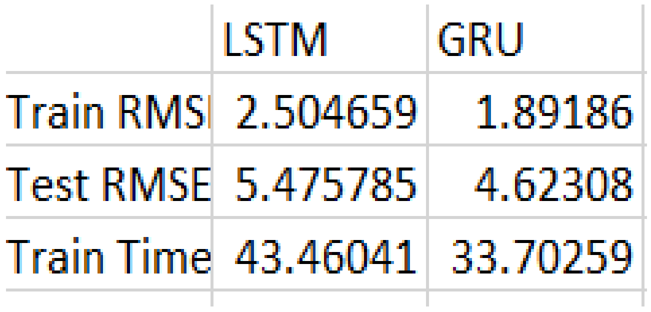


	LSTM	GRU
Train RMSI	1.851147	1.546605
Test RMSE	4.23558	3.572591
Train Time	45.372	38.0373

Above values are the calculated errors(MSE) for test data using GRU model and LSTM model for google company.also the training time is mentioned we can see that GRU performs faster than LSTM, because the number of training parametrs are less compared to LSTM.







	LSTM	GRU
Train RMSI	2.504659	1.89186
Test RMSE	5.475785	4.62308
Train Time	43.46041	33.70259

Above values are the calculated errors(MSE) for test data using GRU model and LSTM model for Apple company.also the training time is mentioned we can see that GRU performs faster than LSTM, because the number of training parametrs are less compared to LSTM .

NOTE: These LSTM model and GRU model are created using pytorch library . Final model of LSTM with 3 layers is using tensorflow.

Here the loss is less for GRU than LSTM because the our datasets are not large. Justified by the conclusion written above .

2. *Final LSTM Model.*

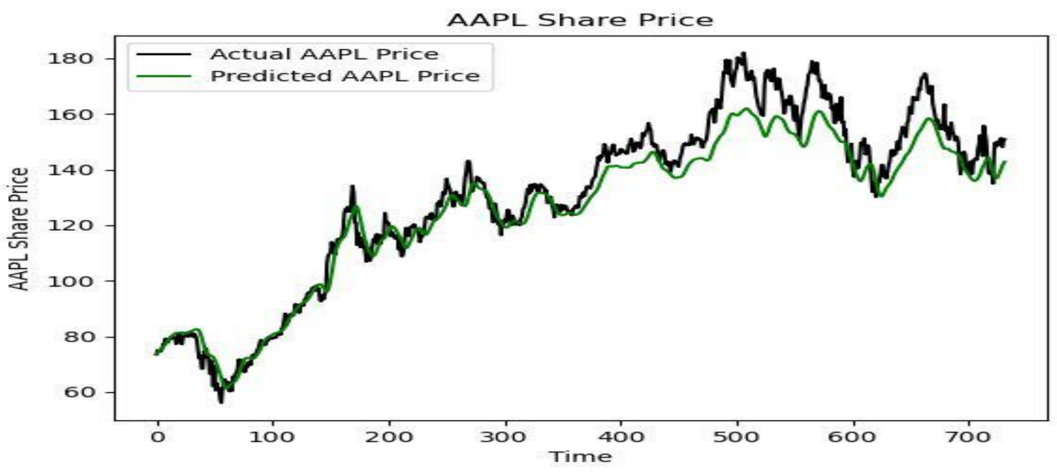
In this model we are using 3 layers of LSTM and dropout , each with units around 50.(tried units of 40, 70, 60,etc) . We take data from "finance.yahoo.com" , the training data is the stock price values from [2012 , 1st Jan] to [2020 , 1st Jan], we use

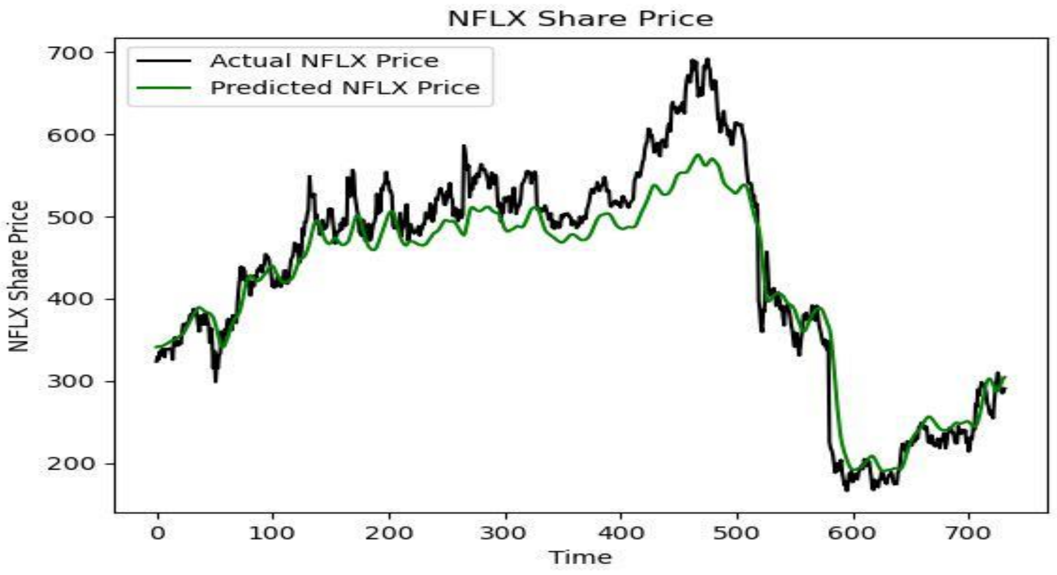
1. Sequnetial

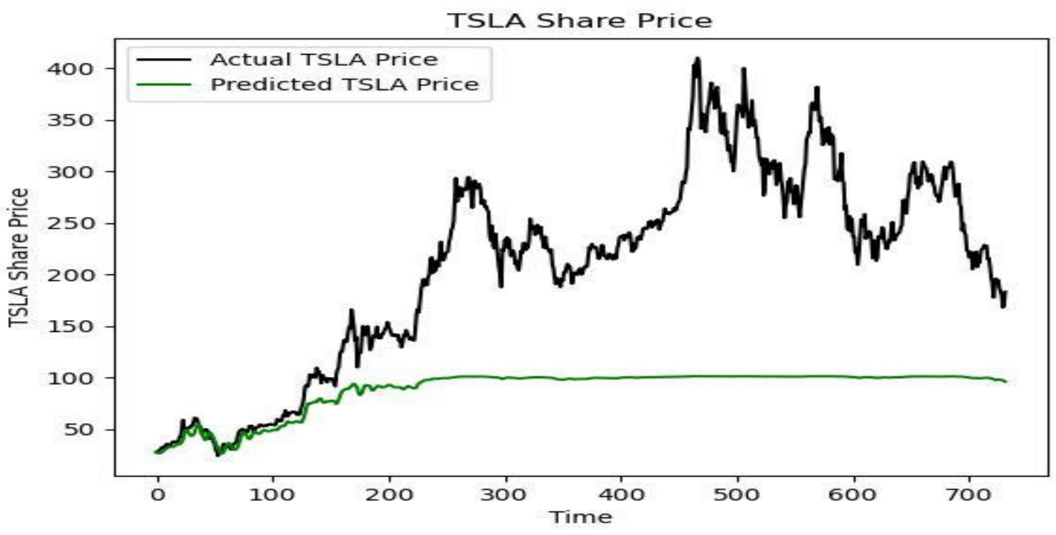
2. Dense
3. Dropout
4. LSTM

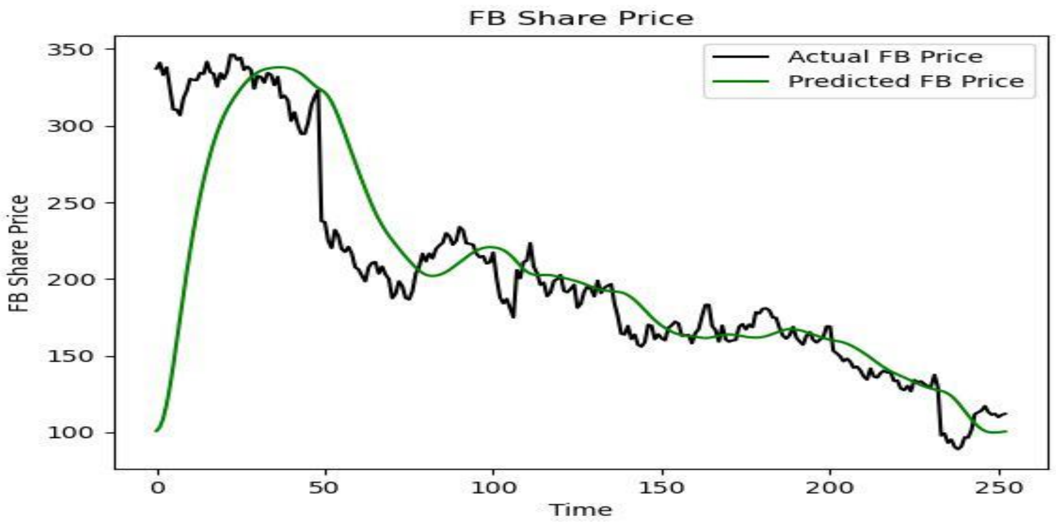
For training on this data with 3 layers as mentioned above, then we are going to test the data from [2020 , 1st Jan] to [Present Date] , the plot contains actual prices of the company in these dates and the prdicted stock price values in these dates. The prediction procedure is as follows,

Prediction_days is the number of previous days stock price values we see to predict the stock value of the day required . So model_inputs while testing will have [2020 , 1st Jan] to [Present Date] these dates and Prediction_days number of previous dates also . Here the graphs were generated by taking Prediction_days=60, so for evry prediction date we see the previous 60 days stock values in the actual data and predict the next day value using our trained LSTM model. The results of prediction and actual prices plots for various companies is as follows,

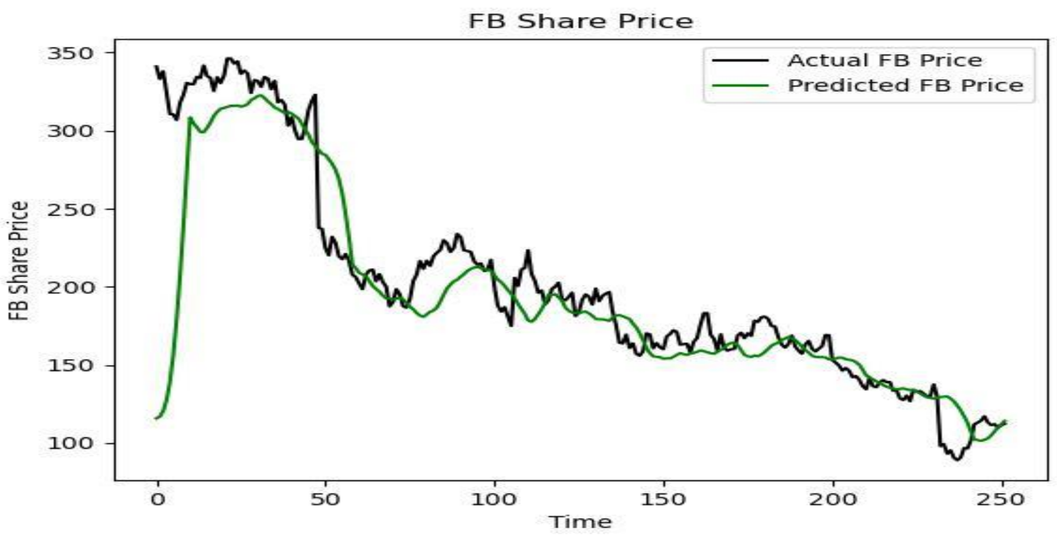








As we see the predicted curve for FB company just says whether the value is decresed or incresed in a certain period of time , but doesn't exactly predict the clusters spikes, for predicting these we need to tune variable Prediction_days and for accuracy we have to tune the number of LSTM layers and units each layer contains. We have tuned those parameters and final values are on the code.BNut for FB the more tuned value of Prediction_days is 10. The new plot for FB company is shown below,



Done with report !