Deep Learning for Stock Price Prediction

ITCS 6156 Machine Learning

Department of Computer Science

Under the Guidance of Dr. Minwoo Jake Lee

Jayachandra Reddy Kamineni
UNC Charlotte

Surya Pavan Malireddy
UNC Charlotte

Karthikeya Vayuputra Chittuluri
UNC Charlotte

Project code: https://drive.google.com/open?id=16tXWnR_KyZriFy1PawHGvfFLN0x5Ine8

Introduction:

In this project, we build a deep learning system to predict stock prices of next day (one step time series forecast) and also for a specific period of time (multi-step time series forecast). Stock traders analyze various patterns in the stock market in order to make their investment decisions. Using this system, stock traders can automate their process of decision making. Predicting Stock prices can be achieved using deep learning models like LSTM (Long Short-Term Memory Networks), GRU (Gated Recurrent Unit), CNN (Convolutional Neural Network) plus LSTM models because of their ability to remember past information. We solved this project using two approaches.

1) **One step prediction** takes the test set until the previous day and predicts the next price.

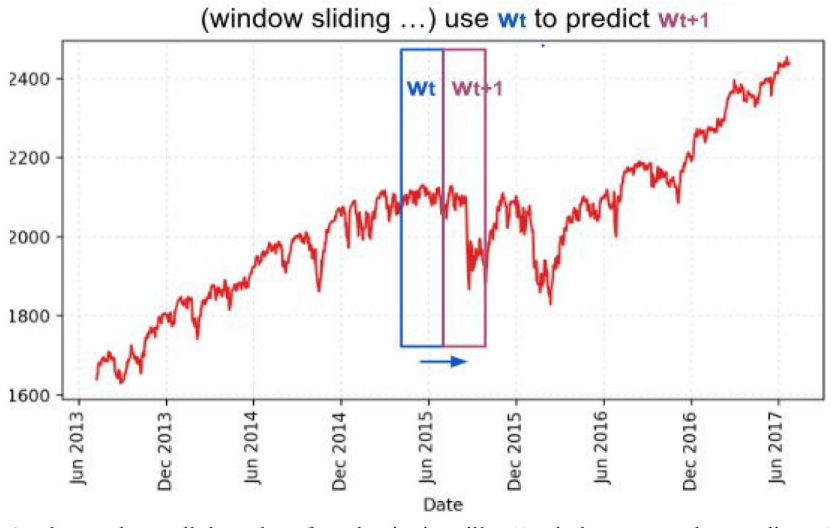
2) **Multistep prediction** starts with the first window in the test set, predicts next price, then pops out the oldest price in the window, appends the predicted price and predicts the next price on this new window for a specified period.

Related work:

We studied few research papers on time series forecasting problems using Deep learning. Their work has inspired us to take up this problem. Research work [1] by Lilian Weng implemented the RNN model with LSTM cells to predict the prices. She used Sliding window approach and used values in one window to calculate the values of the next window without any overlap.

As shown above all the values from beginning till w(t) window are used to predict w(t+1).

Research paper[2] implemented stock price prediction using RNN, LSTM, CNN - Sliding window model to understand the dynamics of data. They compared the results of three models using error percentage and found that CNN model was performing better compared to RNN, LSTM.



From this research, they proposed the reason that as CNN does not depend on any previous information for prediction and it uses only current window information, this enabled the CNN model to understand the dynamical changes and patterns occurring in the current window to accurately predict the stock. Whereas RNN, LSTM uses information from previous lags to predict the future instances. Since the stock market is a highly dynamical system, the patterns and

dynamics existing within the system will not always be the same. Above two research works has greatly inspired us to explore the research area of building CNN architecture which is proven to be best in identifying the patterns in the current window over LSTM architecture which works best in predicting future instances using previous lags.

DATASET:

We used Stock time series data from Alpha Advantage API's. It consists of High, Low, Open, Close prices and Trading Volume for each day. We pulled stock price data of Google from 8/19/2004 to 11/02/2018. It has 3579 rows and 5 columns.

Features:

1. High: Highest price reached on that day by stock
2. Low: Lowest price reached on that day by stock
3. Open: Opening price of the stock on that day
4. Close: Closing price of the stock on that day
5. Trading volume: Number of shares that changed hands on that day

Sample data:

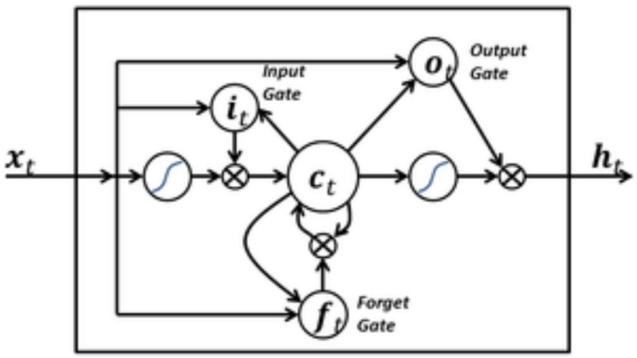
timestamp	open	high	low	close	volume
11/2/2018	1089	1098	1067.66	1071.49	2172215
11/1/2018	1091.4	1099.9	1077.82	1085.98	2006575
10/31/2018	1068.2	1108	1068.2	1090.58	3545821
10/30/2018	1020.01	1050.9	1013.97	1049.51	2988418
10/29/2018	1096.54	1108.83	1007.2	1034.73	4064452
10/26/2018	1048.33	1117	1042.23	1083.75	5321883

Out of these five features, we have used open price feature to predict future prices. We normalized our data using min-max normalization so that the mean of the data is 0 and variance is 1. Finally, the company's stock prices are split into 60-day windows for training and testing.

Methods:

1. LSTM (Long Short-Term Memory Networks)**:**

Structure of a single LSTM node :



LSTM Unit

Each LSTM node has a cell state which stores the information. The information stored in it is controlled by three gates :

a. Forget gate: The first state in the LSTM is to identify that information that is not required and will be thrown away from the cell state. This decision is made by a sigmoid layer called as a forget gate layer.
b. Input gate: The next step is to decide, what new information we are going to store in the cell state. A sigmoid layer called the 'Input gate' layer decides which values

be updated. A 'tanh ' layer creates a vector of new candidate values, that could be added to the state.

c. Output gate: the output gate is a sigmoid layer which decides which parts of the cell state need to be included in the output of the LSTM node.

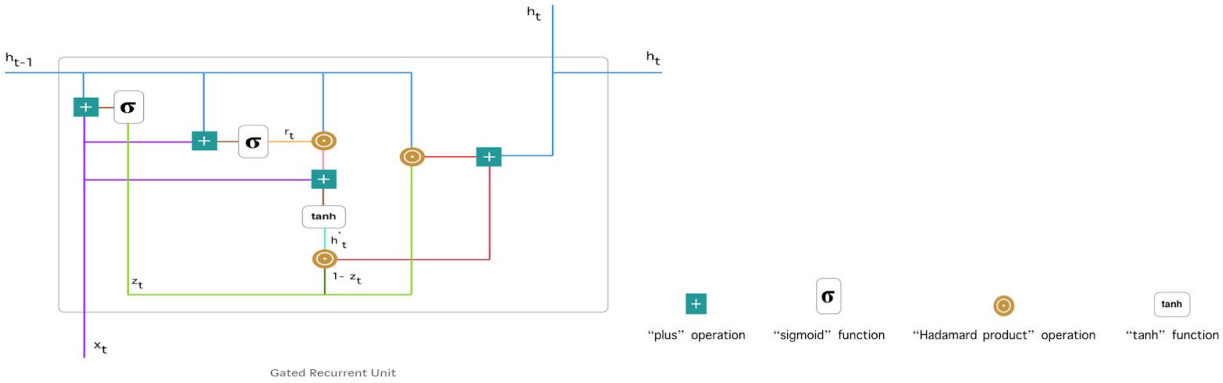
The LSTM Architecture which we have implemented is as follows :

LSTM Architecture

As RNNs are successful in dealing with sequential data, we implemented the LSTM model in our predictions. In our model, we used 3 LSTM layers each with 50 units and connected to 2 dense layers. We used 100 epochs with MeanSquaredError as our loss function and Adam as the optimizer.

2. GRU

Structure of Single GRU node:



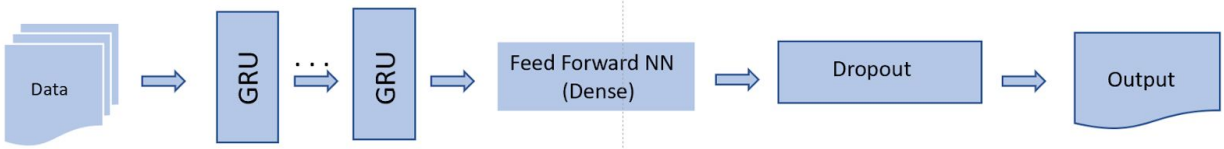
Gated Recurrent Unit

To solve the vanishing gradient problem of a standard RNN, GRU uses, so-called, update gate and reset gate. Basically, these are two vectors which decide what information should be passed to the output. The special thing about them is that they can be trained to keep information from long ago, without washing it through time or remove information which is irrelevant to the prediction.

a.Update gate: The update gate helps the model to determine how much of the past information (from previous time steps) needs to be passed along to the future.

b. Reset gate: this gate is used from the model to decide how much of the past information to forget.

The GRU Architecture we have used is as follows:

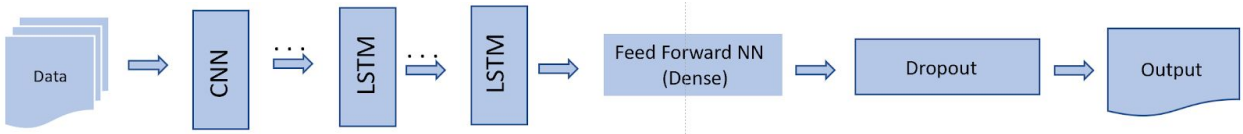


GRU Architecture

As RNNs are successful in dealing with sequential data, we implemented the GRU model in our predictions. In our model, we used 2 GRU layers each with 50 units and connected to 2 dense layers. We used 100 epochs with MeanSquaredError as our loss function and Adam as the optimizer.

3. CNN - LSTM:

The CNN LSTM Architecture which we have implemented is as follows :



CNN - LSTM Architecture

In our model, we used 3 Conv 1D layers each with 16 filters, 60 filter length, connected to 2 LSTM layers each with 150 units and connected to 4 dense layers. We used MeanSquaredError as our loss function and Adam as the optimizer.

Experiments/Results:

Hyper-Parameter tuning:

We performed Hyperparameter tuning by changing the number of layers, dense layers, epochs, batch size, units in each cell(LSTM/GRU), optimizer, activation functions in each model.

LSTM:

Below are the hyperparameters we have tried changing to obtain the optimum model. We experimented with different number of LSTM layers(n =1,2,3,4,5), number of hidden units as

50, number of epochs as (40,60,100), batch size as 256. Below are the results we got for each parameter during tuning. The number of LSTM layers had a lot of impact on the results while other parameters did not affect the performance much.

GRU:

To optimize our GRU model, we experimented with different number of GRU layers(n =1,2,3,4,5), number of hidden units as 50, number of epochs as (40,60,100), batch size as 256. Below are the results of the model for number layers we changed. Other parameters did not have much effect on performance.

CNN LSTM:

To optimize the CNN - LSTM model, we experimented with different combinations of Conv 1D, LSTM layers with a number of epochs as (40,60,100,200), batch size as 50, 100, 256. We also performed Hyperparameter tuning by changing the filter length, number of filters in CNN layers.

Hyper-parameter results:

LSTM - Epochs:100, Batch size:256, Optimizer: Adam,Activation : Relu ,Loss: MSE, Units:50
CNN LSTM – Epochs: 200, Batch size:50, Optimizer: Adam, Activation : Relu,Loss: MSE, Units:50,150,250.
GRU - Epochs: 90, Batch 256 , Optimizer :Adam ,Activation : Relu,Loss:MSE, Units: 50

LSTM

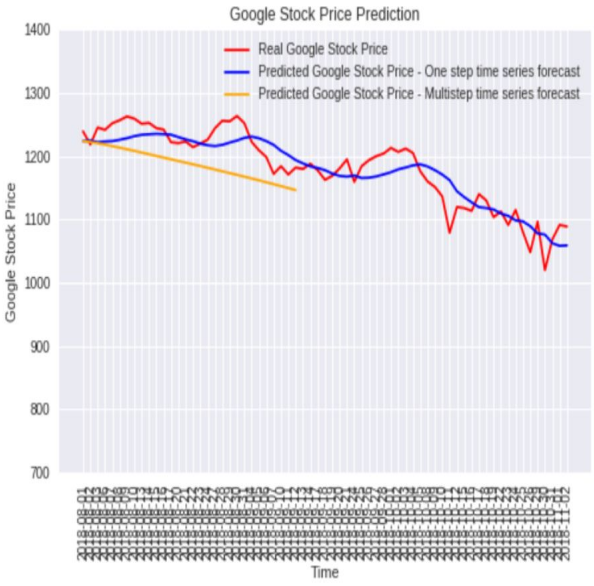
Layers	1	2	3	4	5
RMSE	29.56	30.806	28.38	32.38	35.068
F1 Score	0.407	0.333	0.415	0.392	0.392
Accuracy	0.515	0.515	0.53	0.53	0.53

CNN LSTM

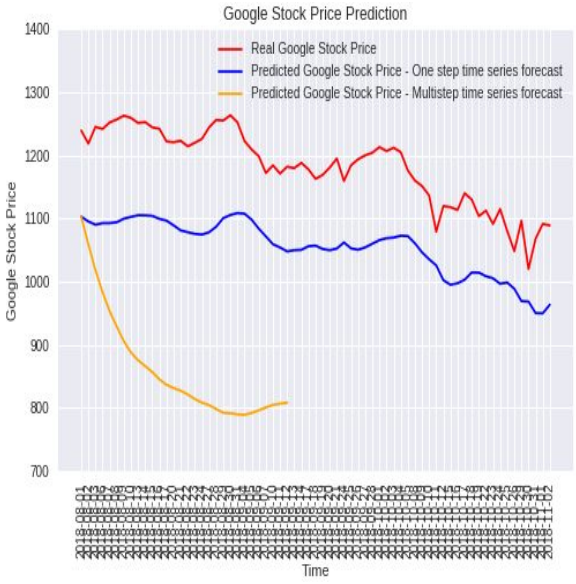
Layers	3 Conv1D, 1 LSTM	4 Conv1D, 3 LSTM	3 Conv1D, 2 LSTM	4 Conv1D, 2 LSTM	3 Conv1D, 3 LSTM
RMSE	89.33	131.41	65.56	116.4	102
F1 Score	0.515	0.514	0.483	0.532	0.516
Accuracy	0.447	0.386	0.418	0.459	0.463

GRU

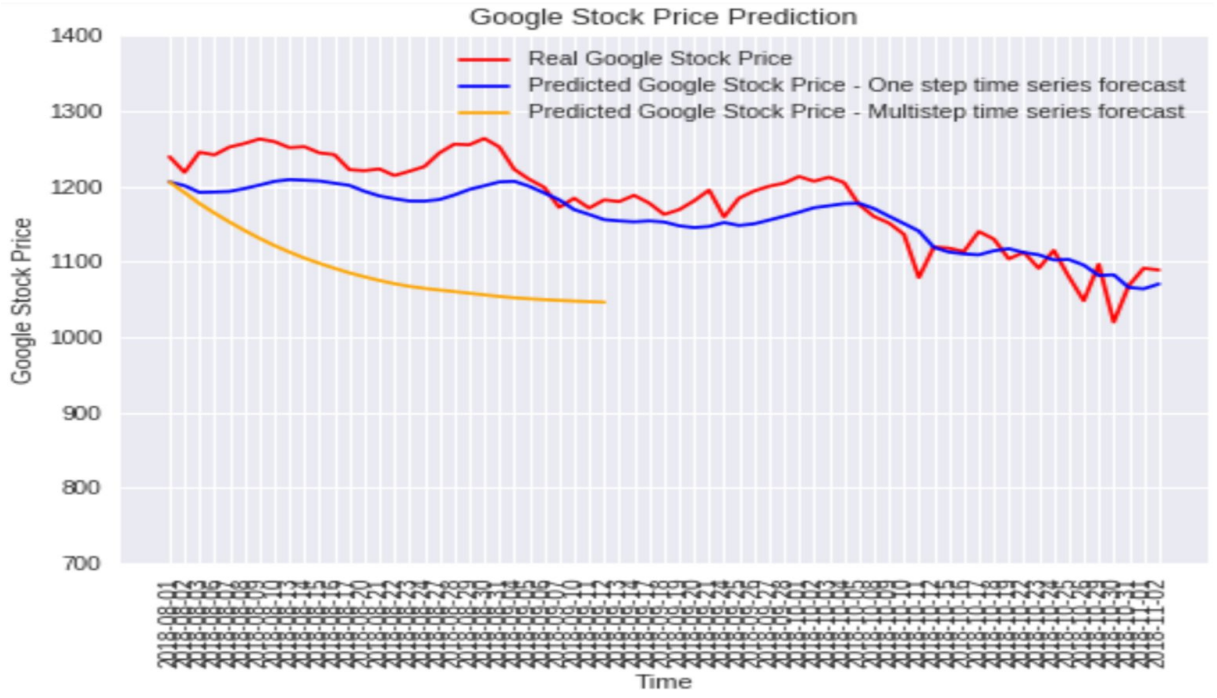
Layers	1	2	3	4	5
RMSE	123.82	22.632	30.459	27.84	34.25
F1 Score	0.516	0.413	0.436	0.436	0.436
Accuracy	0.545	0.484	0.530	0.530	0.530



Prediction using LSTM



Prediction using CNN-LSTM



Prediction using GRU

Analysis:

Overall, analyzing the performance of the above three models Day by Day prediction seems to work better. Sequence predictions did not work well in predicting the pattern. It showed downward movement even there is an upward trend.

Increasing number of layers by a large value did not improve the model performance.

Batch size did not have much effect on RMSE so we choose 256 to reduce training time.

Similarly, we choose the number of epochs as 100 to balance RMSE and training time.

Comparison between the model used by research papers we followed and model implemented in this project:

We combined the ideas from 3 research papers and implemented in this project. We developed one step and multistep(30 days) time series prediction models using 3 different models whereas In research paper[1] they predicted prices of next window using the previous window using LSTM model. We used the idea of a sliding window approach from their paper. In the research paper[2] they have built 3 models LSTM, RNN, and CNN without using a sliding window approach in understanding the dynamics of data and reported CNN worked best by comparing error percentages. The research paper[3] implemented LSTM and GRU neural network methods for traffic flow prediction. That was the first time GRU is applied to traffic

flow prediction. From the above two papers. We combined the ideas of the above three papers in building three models and comparing the results. We reported RMSE value of 22.632 for GRU model and CNN LSTM was not able to perform as expected.

Conclusion and Future work:

We worked on predicting stock price patterns using 3 different approaches. GRU worked better in predicting day by day stock prices and LSTM worked better in sequence stock price movements.

For future work, we are planning to make a data stationary before feeding into the deep learning algorithms that may improve the results of multi-step prediction. We are also looking forward to including Text analytics strategies of Topic modeling, Sentiment analysis on News data related to Google for predicting better future because News, textual data on the web plays a crucial role for changes in stocks. We also wanted to make changes in our sliding window approach as stock prices maintain long-term dependencies sliding window of 30 days may not be sufficient in predicting 30 days into future.

Throughout this project, we learned various aspects of how time series data can be handled using Deep learning models. We learned to build Deep learning models and how each factor/ parameter affects the performance of models.

Response to feedback:

Exploratory Data Analysis has been performed to understand the trend followed by Google stock prices. We checked for the seasonality present in the data. We were not able to present actual dates on the x-axis as dates were overlapping with each other. EDA has also been performed to check if the data is stationary or not. Yes, the approach we have followed is Seq2Seq prediction model. Each window in our model is a sequence with 60-time steps and output is a sequence with one-time step.

From our research, we found that everyone has tried solving this problem using LSTM, CNN alone and they have made a comparison between them. But in our model, we have combined both the architectures to see if it can improve the results and also built models of LSTM, GRU separately. We compared the results of the 3 models we built and also compared with the results of the research papers we have followed. Another unique problem we have tried solving is predicting multi-steps into future and see which model has better scope in predicting 30 days into future. From the plots above we can see that LSTM model has better scope for predicting multisteps into future. Also from our findings, we have not seen anyone implementing GRU architecture in predicting stock prices or time series financial data. GRU architecture worked best in predicting one step ahead among the 3 models we built.

Team Roles and Contributions:

	Team Members	Responsible For
1.	Jayachandra Reddy Kamineni 801046554	- Project research, documentation - EDA and pre-processing - Modeling using LSTM, CNN -LSTM, GRU - Evaluation Metrics and Tuning the Models
2.	Surya Pavan Malireddy 801042753	- Project research, documentation - EDA and pre-processing - Modeling using CNN -LSTM, GRU - Evaluation Metrics and Tuning the Models
3.	Karthikeya Vayuputra Chittuluri 801046118	- Project research, documentation - EDA and pre-processing - Modeling using LSTM, GRU - Evaluation Metrics and Tuning the Models

References:

1. https://lilianweng.github.io/lil-log/2017/07/08/predict-stock-prices-using-RNN-part-1.html
2. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8126078
3. https://ieeexplore.ieee.org/abstract/document/7804912
4. http://cs230.stanford.edu/files_winter_2018/projects/6906443.pdf
5. http://cs230.stanford.edu/files_winter_2018/projects/6940337.pdf