**Predict Stock Prices with LSTM**

Bharath Koneti

Department of Computer Science

University of Illinois, Chicago

Bkonet2@uic.edu

Nihal Pratap Ghanathe

Department of Computer Science

University of Illinois, Chicago

nghana2@uic.edu

Sherzod Muminov

Department of Computer Science

University of Illinois, Chicago

smumin2@uic.edu

Project Link: https://github.com/BharathKoneti/AIMACS594

1. **Abstract**

There are a lot of complicated financial indicators and the fluctuation of the stock market is highly volatile. However, as the technology is getting advanced, the opportunity to gain a steady fortune from the stock market is increased and it also helps experts to find out the most informative indicators to make a better prediction. The prediction of the market value is of great importance to help in maximizing the profit of stock trade while keeping the risk low.

Long Short-Term memory is one of the most successful RNNs architectures. LSTM introduces the memory cell, a unit of computation that replaces traditional artificial neurons in the hidden layer of the network. With these memory cells, networks are able to effectively associate memories and input remote in time, hence suit to grasp the structure of data dynamically over time with high prediction capacity.

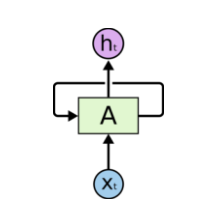
We are trying to predict stock closing price based on LSTMs. We collected a sample and used it for training and validation purposes for the model.

**Keywords:**  Long short-term memory (LSTM), Gated Recurrent Unit (GRU), recurrent neural network (RNN), NYSE, root mean square error (RMSE), prediction and stock prices.

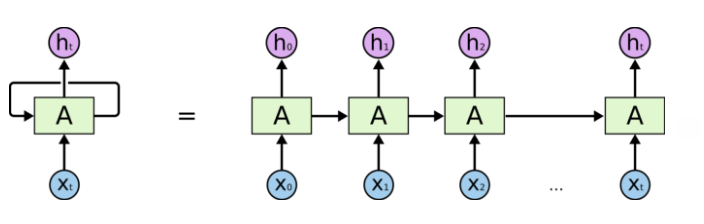
1. **Methodology**

Various types of neural networks can be developed by the combination of variegated factors such as network topology, training method etc. In this experiment we are using Long Short-Term Memory. We are also going to compare LSTM to other popular methodology – GRU (Gated Recurrent Unit).

LSTM is a neural network which can be used to predict and process at any point, which the traditional neural network cannot handle. They are networks with loops:

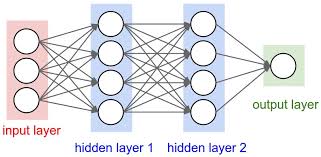


In this diagram, neural network A takes as input ‘xt’ and produces output ‘ht’. Loop makes it possible to pass information from one state of the network to the next step. Next steps are actually similar to their predecessors, and you can assume that you receive input ‘xt’ and producing output ‘ht’ at every step:



Above diagram represents unrolled version of a particular neural network A.

**Neural Network architecture:**

****

The architecture of our solution looks like as show in the above figure. There is an input layer, two hidden layers each with 50 and 100 LSTM cells respectively and an output layer. The hyperparameters here are the number of hidden units, the drop-out rate, number of epochs, batch size. We found that for the below hyperparameters, we got the best results.

Drop-out rate – 20%

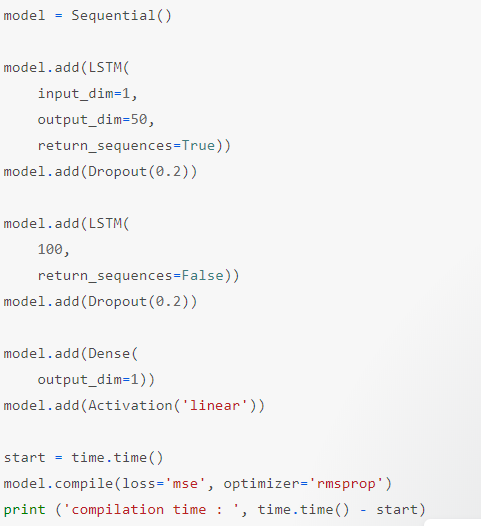
Batch-size- 100

Number of Epochs = 50

Number of hidden units in layer1 = 50

Number of hidden units in layer2 = 100

The code snippet of the model being initialized is as shown in the below image with the above parameters. Here we use a **Linear Activation function**, a **mean squared error** loss function and a **gradient descent optimizer**.

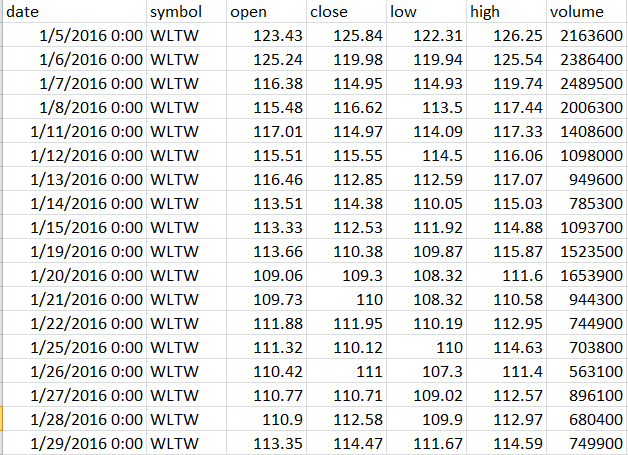


1. **Data**

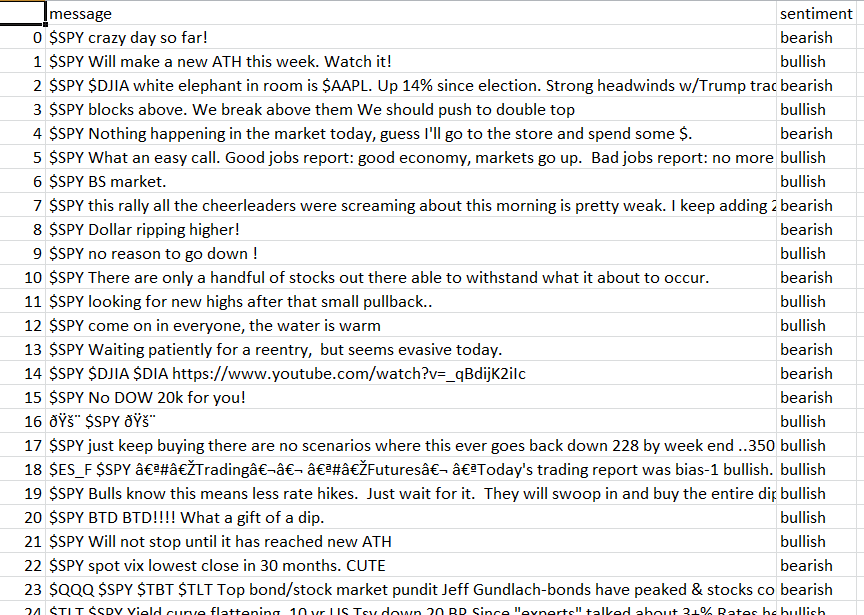
Stage 1: Raw Data:

In this stage, the dataset consisting of S&P 500 companies is used for prediction of future stock prices

Ex1: The date from S&P500 index is shown below:



Ex2: The data for stock twits is shown below:



Stage 2: Data Preprocessing:

The pre-processing stage involves

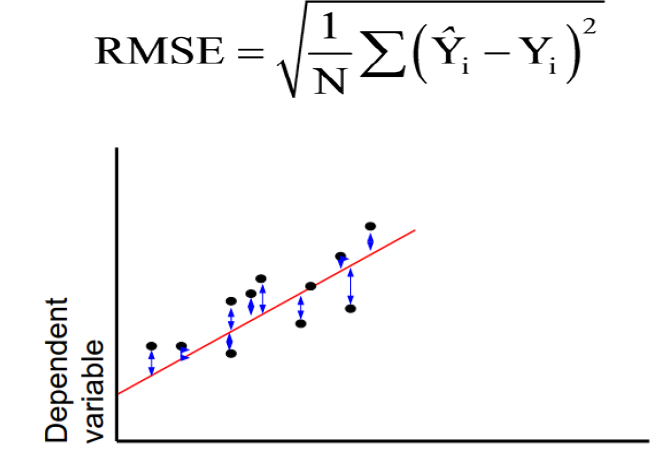
1. Data discretization: Part of the data reduction but with importance, especially for numerical data.
2. Data transformation: Normalization
3. Data cleaning: Fill in missing values.
4. Data integration: Integration of data files.

Stage 3: Feature Extraction

In this layer, only the features which are to be fed to the neural network are chosen. We will choose the feature from Date, open, high, low, close and volume.

1. **Analysis**

For analyzing the efficiency of the system, we have made use of the Root Mean Square Error(RMSE). The error or the difference between the target and the obtained output value is minimized by using RMSE value. The RMSE is the square root of the mean/average of the square of all the error. The use of RMSE is highly common and it makes an excellent general-purpose error metric for numerical predictions. RMSE amplifies and severely punishes large error.



1. **Experiment**

For our project we have followed these steps:

1. Downloaded daily closing prices for stocks both form NYSE and NASDAQ for the period between 1/1/2000 and 12/31/2016.
2. We have implemented LSTM and GRU neural networks in order to predict stock prices
3. We have also used sentiment analysis of stocks based on TwitStocks
4. Other supporting functions in Python like plotting results were also implemented
5. Watson Conversation Services were used in order to interact with users with the help of Python SDK
6. All interaction with code and users is held in Jupyter Notebook

Data includes following fields:

1. Date
2. Ticker (name of the stock)
3. Closing price

We have preprocessed the data based on user input. For example, if user wants to make prediction for MSFT (Microsoft) stock, we have parsed data to include only MSFT results. Neural Networks are sensitive to the input data. So, we had to rescale data to achieve reasonable output.

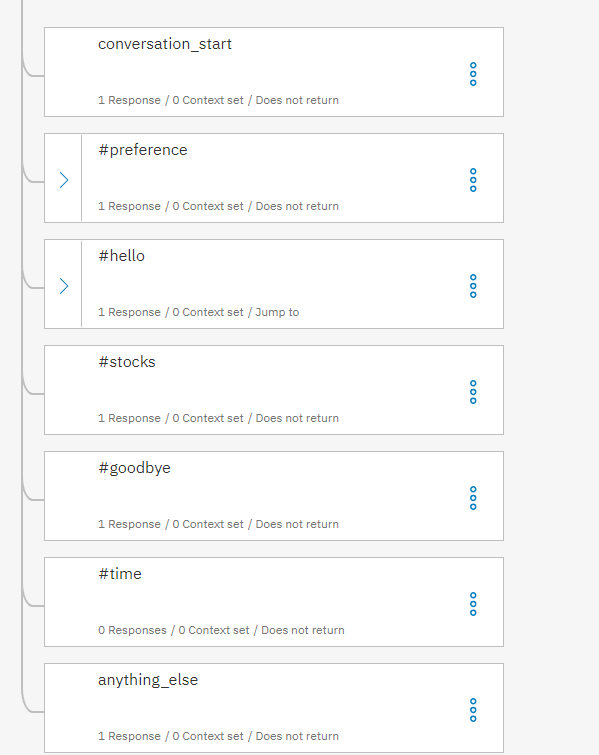
**Watson**

For the purpose of this project we have used Watson conversation services to interact with users. This makes usage of our project smoother and more enjoyable. Moreover, we have incorporated Watson API into our Jupyter solution, so that everything can be done directly from there.

*Structure*. We have created several intents for our conversation services:

* #hello –greets the user
* #methodused – recognizes method user wants to implement, this would be either LSTM or GRU
* #number – recognizes number of days user wants to predict stock prices
* #preference – recognizes what user wants to do. Currently, only stock prediction is implemented. But in the future other functionality may also be implemented
* #stocks – list of stocks that are supported by our service
* #goodbye –recognizes farewell messages, say bye to user and shuts down the program

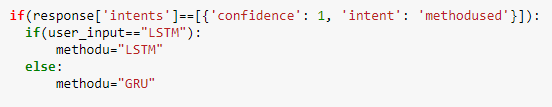
Dialog consists of implementation of above intents



Further we have downloaded Python API for Watson Conversation Services and connected to our service directly from Jupyter:



We manipulated data in Jupyter, aka executed appropriated functions based on recognized intents by simple “if” statements. For example:

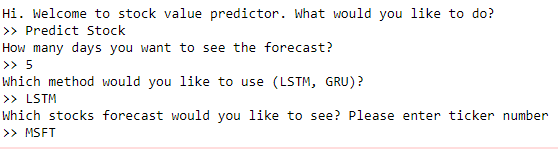


In above code chunk, based on user input, we recognize #methodused intent and assign user\_input into methodu variable, which is used as an argument in functions that has to be executed.

If we recognize #goodbye intent, we shut down all connections to Watson Conversation Services.

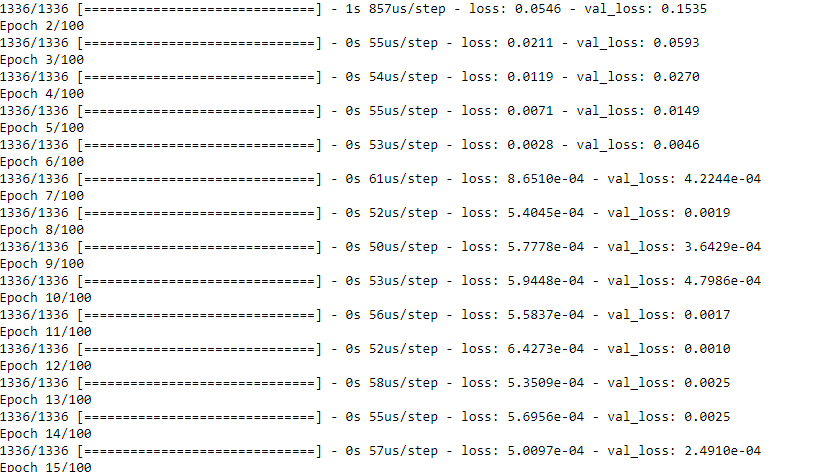
Output

Let’s review demo. When we run last code group in Jupyter we come across into dialog:

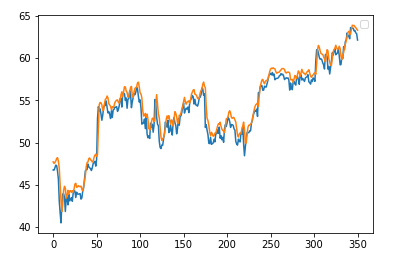


To first question we should specify our intent to predict stock value. Next, service asks us for how many days we want to make a prediction. Next, we choose method to use: either LSTM or GRU. And finally we specify stock ticker to predict.

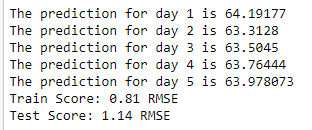
Then, model trains and for each epoch it shows us loss value, which decreases over iterations:



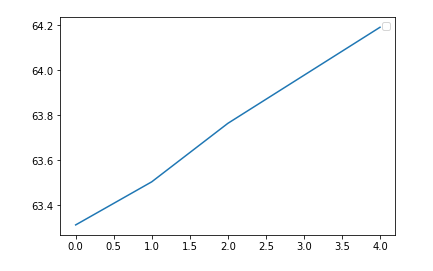
Finally, we receive an output with test (blue) and predicted (orange) values.



We also receive predictions for each particular day in the future. As well as error value, based on parameters entered previously:



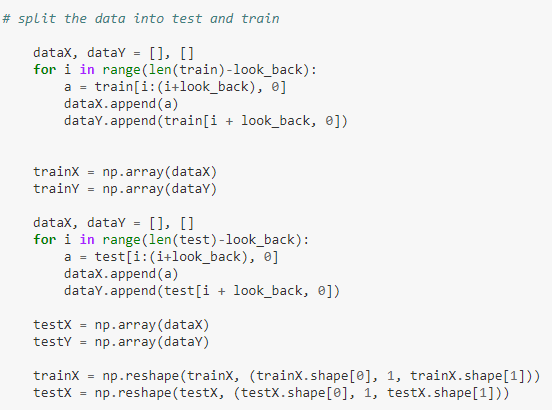
Future price plot is also showed



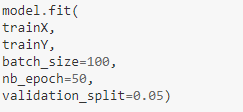
At the end, if we type bye, program quits.

1. **Code Flow:**

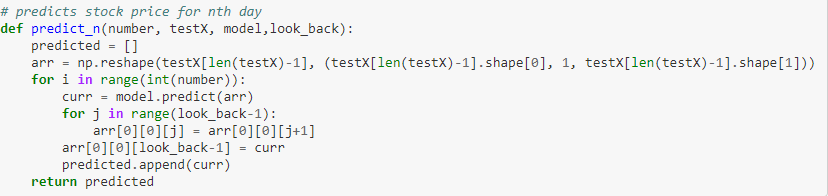
First we have the **initialize** method that takes in the name of company whose stocks are to be predicted, number of day ahead of which the stock needs to be predicted and a parameter called “look\_back”, which basically is the number of previous stock prices to be considered to predict the next stock price.



Next based on the look\_back parameter the data is divided into features and their true labels. Later we split the data into 80% training and 20% test data. And the model is initialized as explained above. Here we have a choice of model to be used, i.e. LSTM or GRU. After the data is split into train and test, the model is trained with the above listed parameters.



Once the model is learnt, we use the function **predict\_n** to predict the stock price ‘Number’ days ahead. i.e. previous ‘look\_back’ stock prices are used to predict the stock of day 1 and next we use the look\_back as well as the day 1 predicted value to predict the value for day 2 and so on till day n and a graph is plotted to show the stock values for the N predicted days.



Along with this predicted data, we also calculate the crowd sentiment of that company using StockTwits data, Where bearish meaning the stock sentiment is negative and bullish meaning the stock sentiment is positive. Here we calculate the number of bearish and bullish and based on the majority, the sentiment for that particular company is displayed. The function used to do that is as shown below.



1. **Conclusion:**

The popularity of stock market trading is growing rapidly, which encourages researchers and stock buyers to find alternative and better methods for the prediction using new techniques. The forecasting model requires accurate prediction for the models. The work presented has one of precise calculations using Long Short Term Memory unit which helps investors or analysts to better judge buying the stock.

**References**

[1] RakhiMahant, TrilokNathPandey, Alok Kumar Jagadev, and SatchidanandaDehuri ―Optimized Radial Basis Functional Neural Network for Stock Index Prediction,‖ International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT) – 2016.

[2] Kai Chen, Yi Zhou and FangyanDai ―A LSTM-based method for stock returns prediction: A case study of China stock market,‖ IEEE International Conference on Big Data,2015.

[3] DhirajMundada, GauravChhaparwal, SachinChaudhari, and TruptiBhamare― Stock Value Prediction System,‖ International Journal on Recent and Innovation Trends in Computing and Communication, April 2015

[4] Hochreiter, Sepp, and Jürgen Schmidhuber― Long short-term memory,‖ Neural computation 9.8 (1997): 1735-1780