Stock Market Prediction using Long Short Term Memory

M.Ferni Ukrit[1] , A.Saranya[2], Rallabandi Anurag[3]

[1, 2] Assistant Professor, Department of Software Engineering, SRM Institute of Science and Technology
[3]Student, Department of Software Engineering, SRM Institute of Science and Technology

Email: fernijegan@gmail.com, aksaranya@gmail.com , anuragrallabandi@gmail.com

Abstract. Stock Markets have been an integral part of our socio-economic society. People invest a lot of monetary funds into them so as to earn gains. But that is not the case every time due to the ever wavering nature of the markets. To minimize the risk of loss due to drastically changing market people have come up with many predictive models to simulate the future of stock markets. This paper presents a model that can predict the stock market. The use of Stacked Long Short Term Memory gives the model an advantage over the conventional machine learning models to provide better MSE.

Keywords: Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Stock Market, Prediction, Google, Mean Squared Error (MSE).

1 Introduction

Stock Market is a major factor that defines how successful a company is. Google, Apple and others are major players in the market and how these companies fare on the market can impact other companies. So, how these companies do on the market in the future should be of great importance. This neural network attempts to do the same for Google's stock price from beginning of 2012 to the end of 2016 and predict upon data it has not been trained upon.

A specific kind of Neural Network called the Recurrent Neural Network has been used for the purpose. The RNN is a specially designed neural network that is used for

time-series analysis viz., each layer predicts its output based on the output of the previous layer [1].
When RNNs were first developed it faced a problem. With each iteration as the gradient is passed back for updating the weights the process is slower for far off layers and the input that is transmitted for the next iteration is through untrained neurons which affects the output. This is known as the Vanishing Gradient Problem [2].
This was rectified by Long Short Term Memories that do not use the long term dependencies for computing the outputs [3][4][7]. This is the network used for this particular model. The formula for a RNN is specified in
Equation 1.

(1)

It basically says the current hidden state **h(t)** is a function **f** of the previous hidden state **h(t-1)** and the current input **x(t)**. The **theta** are the parameters of the function f.

A study of the stock market is integral for building a prediction model [5].It helps in understanding the nuances of the said market, figuring out the features that is required for the model and edging out the features that is not required [6].

2 Methodology Used

This model consist of the following steps

Step 1: **Data Preparation**

The model will predict based on date and opening value. The opening value is the value the market opens at for that particular trade day.

The data collected was normalized for values between 0 and 1 using Equation 2.

(2)

The scaled data is them recreated to a data structure with 40 and 60 time steps and 1 output which will be used as an input for the LSTM in which the output is decided by the previous 80 inputs and so on.Then the data is reshaped based on three dimensions: number of stock prices, the number of time steps and the number of indicators.

Step 2: Building the model

In the neural network model, a LSTM with 50 input neurons and add a Dropout layer with forty percent ignorance viz 20 neurons will be ignored during training for regularization.

For the second and the third layer the model mimics the input layer with the same parameters thereby creating a stacked LSTM.

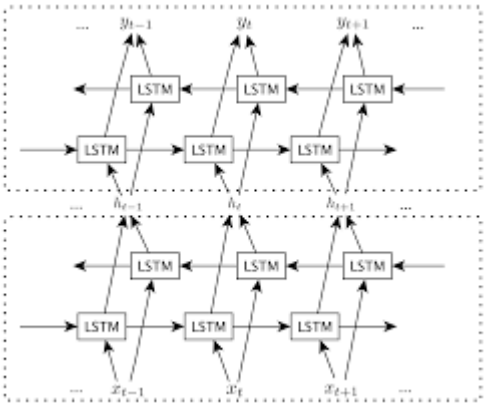

Fig. 1. Basic structure of a stacked LSTM.

The next layer is the output layer with one neuron. The output is computed based on an optimizer and loss function. The optimizer used is the ADAM or the Adaptive Moment Estimation optimizer provided in Equations 3 and 4.

(3)

(4)

where Mt is the first moment and vt is the second moment of the gradient and the loss function(error metric) used is Mean Squared Error (MSE).

Step 3: Fitting the model

In this step data is fed to the, neural network and trained for prediction assigning random biases and weights.

Step 4: Output

The output generated is compared with target values. The error obtained during each epoch is reduced through back propagation which adjusts the weights and biases of each neuron (node) present in the subsequent layers.

Table 1. Font sizes of headings. Table captions should always be positioned *above* the tables.

S.No.	Number of Epochs	MSE Achieved
1	100	0.0030
2	200	0.0019
3	300	0.0017

Lemmas, Propositions, and Theorems. The numbers accorded to lemmas, propositions, and theorems, etc. should appear in consecutive order, starting with Lemma 1, and not, for example, with Lemma 11.

3 Results and Discussion

The hardware used is Intel i5200U CPU clocked at 2.20GHz as a training platform. The model is implemented in Keras for python as front end and Tensorflow backend as coding environment.

The training data set used is Google's stock price from beginning of 2012 to the end of 2016 collected for each financial day for each month of the financial year ranging from 03-01-2012 to 16-12-2016.

The Test set used is Google’s stock price for January 2017.

The loss function Mean Square Error (MSE) or Mean Square deviation (MSD) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors or deviations. Large errors are easily identified and reduced using this metric using equation 5.

(5)

For training the optimizer used is ADAM and the data was normalized. Date and Open values are the parameters for various number of epochs and a batch size of 32 to

calculate MSE. Table 1 and 2 shows the MSE achieved for different epochs for 40 and 60 time steps.

Table 1. MSE achieved for different Epochs for 40 time steps

S.No.	Number of Epochs	MSE Achieved
1	100	0.0030
2	200	0.0019
3	300	0.0017

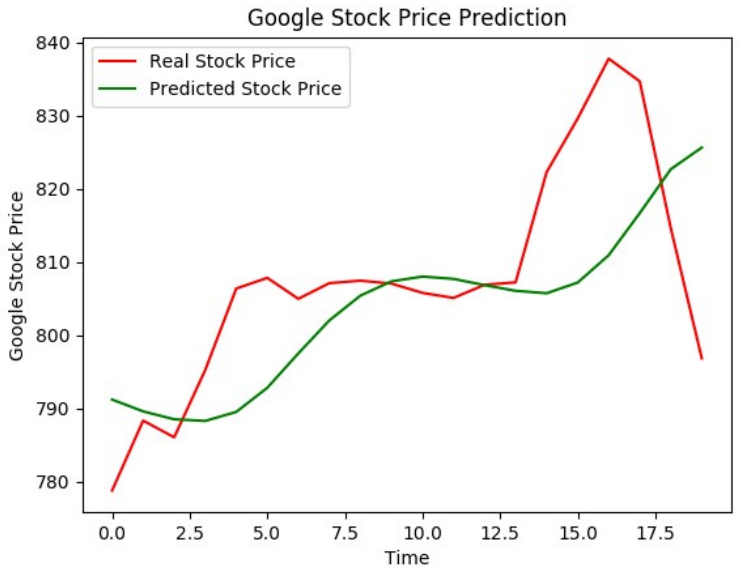

Fig. 2. Real vs Predicted Stock Price for 100 epochs for 40 time steps

Table 2. MSE achieved for different Epochs for 60 time steps

S.No.	Number of Epochs	MSE Achieved
1	100	0.0027
2	200	0.0017
3	300	0.0016

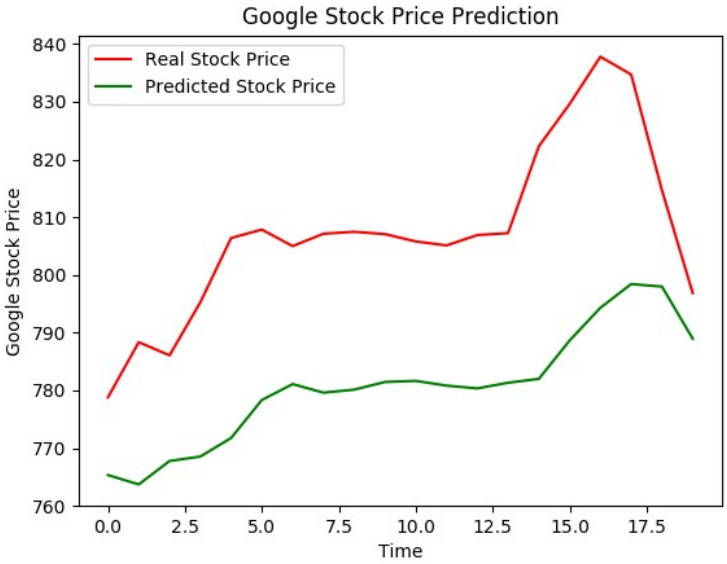

Fig. 3. Real vs Predicted Stock Price for 200 epochs for 40 time steps

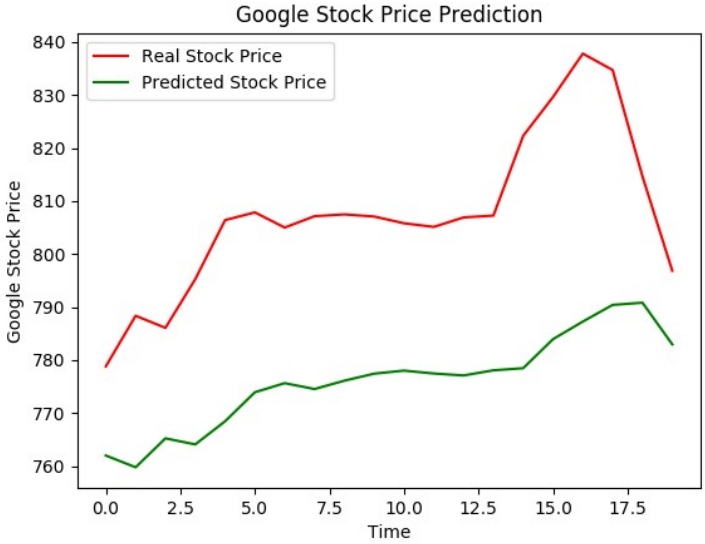

Fig. 4. Real vs Predicted Stock Price for 300 epochs for 40 time steps

Fig 2 and Fig 5 show more MSE because it was only trained for 100 Epochs and moreover it is the first training.

In Fig.3 and Fig 6 the model shows better MSE due to increase in the number of epochs

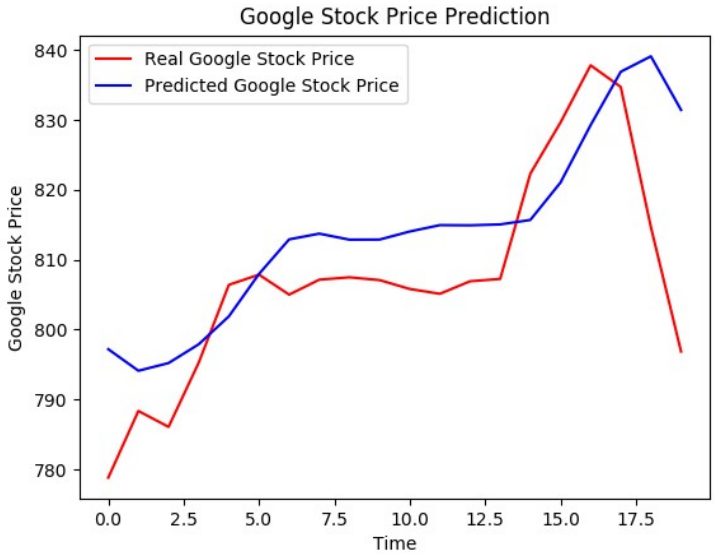

Fig. 5. Real vs Predicted Stock Price for 100 epochs for 60 time steps

In Fig.4 and Fig 7, the model predicts closest to the real value because the number of epochs is increased.

From table 1 and 2 it is inferred that the more the number of time steps the better the MSE achieved.

4 Conclusion

The increasing importance of stock markets is pushing researchers into finding newer methods for prediction which not only helps researchers but traders alike. This model

focuses on prediction using LSTM. As the number of epochs increases the model shows better Mean Square Error up to a certain number until convergence is achieved.

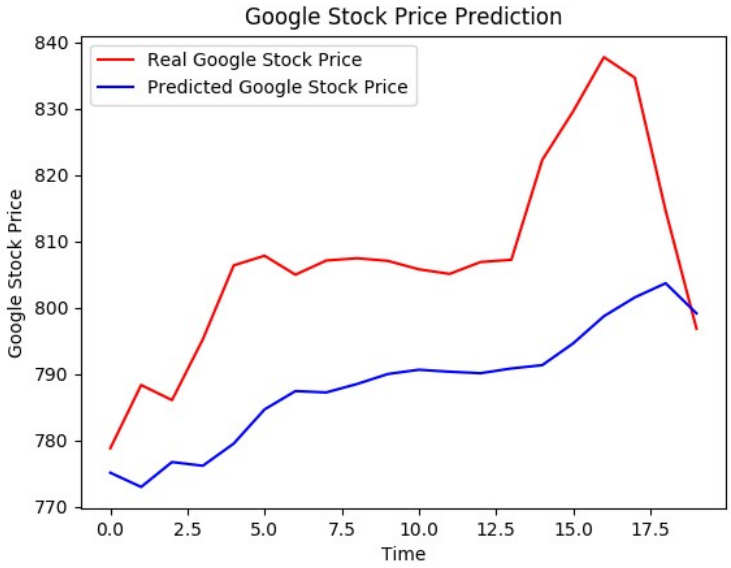

Fig. 6. Real vs Predicted Stock Price for 200 epochs for 60 time steps

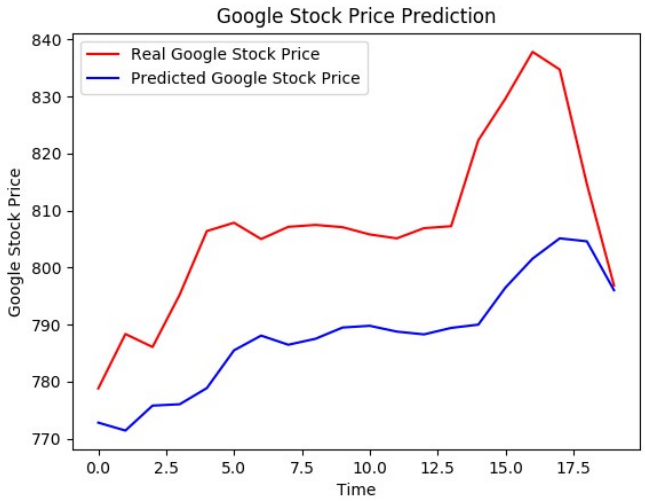

Fig. 7. Real vs Predicted Stock Price for 300 epochs for 60 time steps

References

1. Sutskever, I. (2012). Training Recurrent Neural Networks. Ph.D. thesis, University of Toronto**.**
2. Bengio et al.(1994). Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, Volume 5, Issue 2, pp. 157-166
3. Hochreiter, S. and Schmidhuber, J. (1997). Long Short Term Memory. Neural Computation, Volume **9**, No 8,pp. 1735–1780.
4. Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, Jürgen Schmidhuber (2015). LSTM-A Search Space Odyssey. IEEE Transactions on Neural Networks and Learning Systems .Volume: 28, Issue: 10, pp. 2222 – 2232.
5. Kai Chen, Yi Zhou and Fangyan Dai. (2015) A LSTM-based method for stock returns prediction: A case study of China stock market. IEEE International Conference on Big Data ISBN: 978-1-4799-9926-2.
6. Prashant S. Chavan, Prof.Dr.Shrishail. T. Patil.(2013).Parameters for Stock Market Prediction,.Int.J.ComputerTechnology&Applications, Volume 4, No.2, pp.337-340.
7. Kyoung-jae Kim, Ingoo Han. “Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index”. Expert Systems with Applications, Volume 19, Issue 2, 2000, Pages 125-132, ISSN 0957-4174