Predicting Stock Prices Using Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs)

Conor Dempsey and Barbara Schmitz

Master of Data Science Students, University of Memphis.

Abstract

Accurate predictions of stock prices can result in large financial gain. The goal of this paper was to make S&P 500 stock price predictions using two deep learning methods, namely, Long Short-Term Memories (LSTMs) and Gated Recurrent Units (GRUs). Predictions were also made using a linear time series model and results were compared to our deep learning models. We opted to evaluate our model's performances using root mean square error (RMSE) which can be interpreted as the average difference between the models' predicted stock price to the actual stock price in USD. Test RMSE for the LSTM, GRU and linear time series models were 7.09, 5.97 and 24.93 respectively. Training times were 10.32, 8.3 and 0.002 seconds respectively. Training times were significantly higher for the deep learning models which may pose an issue when dealing with larger amounts of data, however, the deep learning models appear to be significantly more accurate.

1 Introduction

Accurate predictions of stock prices can result in large financial gain. During times of recession, individuals or companies may be motivated to make more informed investing decision. Overall, the goal is to buy stocks at a low price and then later sell them at a higher price than which you bought them. The challenge lies in knowing whether you should buy a particular stock and when the right time to sell it is.

For example, if you knew that a given stock's price was going to rise, you would buy that stock knowing that its value was about to increase. Likewise, if you knew a stock's price was about to fall, you may decide to sell that stock before it loses its value. Knowing for certain whether a stock's price will rise, or fall is unfortunately not possible but, given the profit that there is to be made with even just relatively accurate predictions, stock market forecasting has been a major research topic in the financial area.

2 Related Work

Traditionally, linear models were used for stock forecasting, however, stock price behaviour over time is noisy and non-linear (Grudnitski & Osburn, 1993) and therefore, linear models do not have the capabilities to follow their trends. Abhyankar et al. (1997) suggested applying non-linear methods to obtain more accurate predictions and with advancements in the field of machine learning, applying these non-linear methods have been possible for quite some time.

Donaldson and Kamstra (1996) applied neural networks to stock market forecasting and found that they outperformed the traditional linear models. More recent studies have found that deep learning-based methods tend to outperform non-deep learning-based methods in stock price prediction (Hu et al., 2021) (Nikou et al., 2019). As a result, research as a of late has looked to deep learning to provide state of the art stock price predictions. Some studies have found success in using convolutional neural networks (CNN) to predict stock prices (Chen et al., 2016) (Tsantekidis

et al., 2017). As stock data can be seen as sequential, recurrent neural networks (RNN) have been a popular deep learning method of choice for stock price prediction. Long Short-Term Memory (LSTM) is a special type of RNN which allows for the learning of long-term dependencies. LSTMs have also seen success in stock price prediction due to their ability to make predictions based off long sequences of previous inputs (Chen et al., 2015) (Nelson et al., 2017).

3 Data

The dataset we used contains five years' worth of Apple stock data and was retrieved from Kaggle (Nugent, 2018). The dataset contains 1,259 rows and seven variables. Each row refers to stock data for a particular day. Variable descriptions are detailed in table 1 below (note that all prices are in USD). Note, there are 252 trading days in a year.

Date	In format: yy-mm-dd
Open	Price of the stock at market open
High	Highest price reached in the day
Low	Lowest price reached in the day
Close	Price of stock at market close
Volume	Number of shares traded
Name	Stock ticker's name

Table 1. Variable Description

	open	**high**	**low**	**close**	**volume**
count	1259	1259	1259	1259	1259
mean	109.05	109.95	108.14	109.06	54047899.7
std	30.54	30.68	30.37	30.55	33468353.3
min	55.42	57.08	55.01	55.79	11475922
25%	84.64	85.33	84.25	84.83	29694376.5
50%	108.97	110.03	108.05	109.01	45668931
75%	127.33	128.1	126.29	127.12	68708720
max	179.37	180.1	178.25	179.26	266833581

Table 2. Descriptive Statistics

4 Methods

We opted to train two specialized types of Recurrent Neural Networks (RNNs), namely, a Long Short-Term Memory (LSTM) network and Gated Recurrent Unit (GRU) network to predict the highest price reached in each day. RNNs work with sequence data to use past elements of a given sequence (or past inputs) to predict the next element of that sequence (i.e., the output) (Bengio, Simard, & Frasconi, 1994). RNNs pass both the input at a given time step and the hidden state from the previous time step/element of the sequence through an activation layer to obtain a new hidden state. At the last time step, a final hidden state is calculated (using information from both the input at that time step as well as all previous hidden states) and used to compute an output.

As the number of time steps increases, traditional RNNs fail to use information held in the earlier time steps in the computation of an output due to the vanishing gradient problem (Gers, 1999). This is where the signal (gradient) sent to update the weight at a given layer during back propagation decreases across layers. These small gradient values essentially prevent the weight from being updated. To solve this problem, Schmidhuber and Hochreiter (1997) introduced LSTMs. LSTMs use a gating mechanism to control the flow of information. At a given time step in RNNs, the network receives information from the current input and the previous time step's hidden state. At a given time step in LSTMs, the network receives information from the current input as well as the previous time step's short-term memory and long-term memory. An input gate decides what information should be stored in the long-term memory. A forget gate decides what information should be discarded from the long-term memory and an output gate produces a new short term memory.

GRUs were introduced by Cho et al. (2014) and are similar to LSTMs except that they have two gates. An update gate decides how much previous information should be passed on to the next state and a reset gate decides how much previous information can be disregarded. A major benefit of this alternative gating mechanism is that there is less computational

cost resulting in faster processing times (Yang et al., 2020).

5 Experiment

We performed traditional linear regression using time steps as the sole predictor with an 80/20 train/test split. This was done in order to have a control group to compare the results of the LSTM and GRU against. The PyTorch framework was used to train and test the LSTM and GRU models. Data was normalized and the sliding window method was used to create all possible sequences of a given length in the data. The data was split into training and testing (the 80/20 was used again). The training and testing data were transformed into tensors. The hyper parameter values that were used for both models can be seen below.

Input dimension = 1

Hidden dimension = 32

Number of layers = 2

Output dimension = 1

Number of epochs = 100

Both models used the adam optimizer and Mean Squared Error (MSE) loss function. All models were evaluated using Root Mean Square Error (RMSE).

6 Results

	Linear Regression	**LSTM**	**GRU**
Train RMSE	13.93	2.14	1.37
Test RMSE	24.93	7.09	5.97
Training Time	0.002s	10.32s	8.30s

Table 3. Comparison of RMSE and Training Time Among Models

As seen in table 3 above, the GRU and LSTM far outperformed the linear regression model based off both train and test RMSE values. The linear regression model took far less time to train however this won't matter in cases like this whereby the amount of data does not impose heavy computational cost, even when training more complex deep learning models. As expected, training time for the GRU was lower than the LSTM. Again, this was not an issue for this experiment given the size of the data but would be important to consider should one want to train an LSTM or GRU on large amounts of data. Figures 1, 2 and 3 below provide a visual aid on how well the models predicted the stock prices.

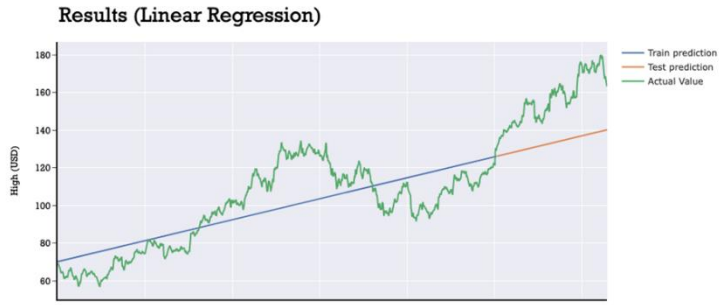

Figure 1. Linear Regression Predictions vs. Actuals

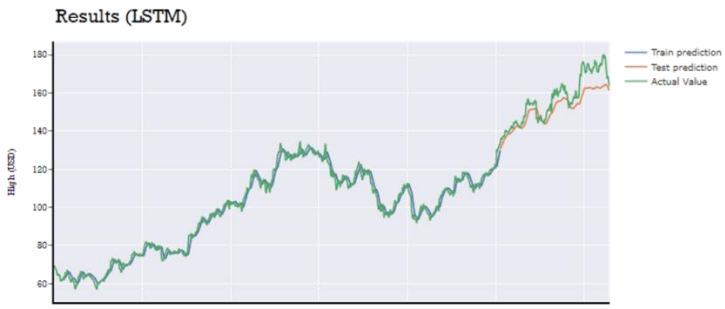

Figure 2. LSTM Predictions vs. Actuals

Figure 3. GRU Predictions vs. Actuals

The above figures clearly illustrate the LSTM and GRU's superior predictive ability as is evident by the orange test prediction lines versus the green actual values. What is also interesting to see is that the GRU's test predictions better picked up on the non-linear behaviour of the stock price compared to the

LSTM. While their test RMSE were similar, this may point to the GRU being a better model of choice when predicting stock price.

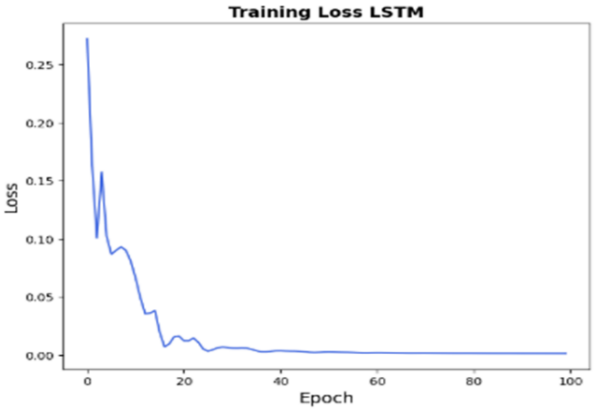

Figure 4. LSTM Training Loss Across Epochs

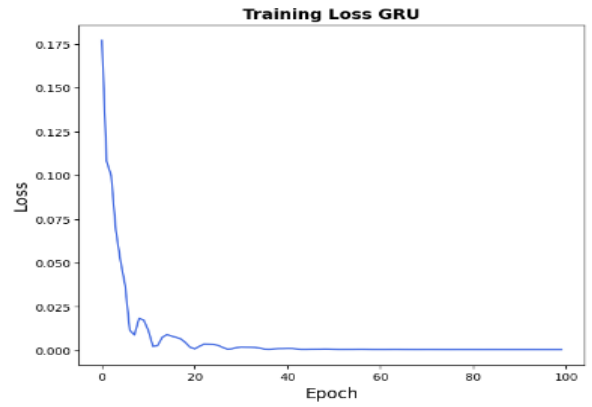

Figure 5. GRU Training Loss Across Epochs

Figures 4 and 5 respectively show that the default 100 epochs were not necessary as model convergence occurred after approximately thirty to forty epochs.

7 Conclusion

GRU outperformed LSTM in terms of time complexity and loss as well as captured nuances in data better. However, both GRU and LSTM outperformed using a traditional linear regression for forecasting. The GRU model is more sensitive to noise than the LSTM, and thus, captures the volatility more accurately than the LSTM model. The LSTM model's prediction of stock behavior tended to smooth out the noise. In this economic domain, we did not want a smoothed trend. Instead, a model that can capture market volatility is much better in this context. For the deep learning models, thirty to forty epochs were sufficient to train the data.

For future improvements, now that we know the data is sufficiently trained at thirty to forty epochs and does not need the full 100, we could include a validation set for the data and train the LSTM and GRU models at forty epochs. In addition, more data could have used more years of data and more stocks added into the mix to evaluate the whole of the S&P 500 portfolio. However, we were intentional in choosing five years (2013-2018) as they were not recession or COVID-afflicted years. There is opportunity to experiment with different hyper parameters such as the input dimensions, hidden dimensions, number of layers, a different optimizer, etc.

One of the drawbacks with deep learning methods, however, is that it is not intuitive which parameters to initialize the architecture with. This rings particularly true when the user has little to no domain knowledge or is not a subject matter expert.

Despite the training times for these models being relatively low, for a more robust analysis and data, this would have benefitted from a CUDA enabled GPU or using Google Colab to achieve the same goal.

To take this experiment even further, this could be a robust analysis of three traditional forecasting models versus three deep learning forecasting models. The three traditional models could include the linear regression, moving day average, and a statistically advanced model i.e., the ARIMA model. The three deep learning forecasting methods could include the LSTM and GRU models along with the newer and rising in popularity architecture, the transformer model.

8 Contributions

Data Collection – Barbara Schmitz
Data Processing – Barbara Schmitz and Conor Dempsey
Mid Term Write Up – Conor Dempsey
Poster – Barbara Schmitz

Final Report Write Up – Barbara Schmitz and Conor Dempsey

9 References

Abhyankar, A., Copeland, L. S., & Wong, W. (1997). Uncovering nonlinear structure in real-time stock-market indexes: The S&P 500, the Dax, the nikkei 225, and the FTSE-100. *Journal of Business &*

Economic Statistics, *15*(1), 1. https://doi.org/10.2307/1392068

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166. https://doi.org/10.1109/72.279181

Chen, J.-F., Chen, W.-L., Huang, C.-P., Huang, S.-H., & Chen, A.-P. (2016). Financial time-series data analysis using deep convolutional neural networks. *2016 7th International Conference on Cloud Computing and Big Data (CCBD)*. https://doi.org/10.1109/ccbd.2016.027

Chen, K., Zhou, Y., & Dai, F. (2015). A LSTM-based method for stock returns prediction: A case study of china stock market. *2015 IEEE International Conference on Big Data (Big Data)*. https://doi.org/10.1109/bigdata.2015.7364089

Cho, K., van Merrienboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder–decoder approaches. *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. https://doi.org/10.3115/v1/w14-4012

Donaldson, R. G., & Kamstra, M. (1996). Forecast combining with Neural Networks. *Journal of Forecasting*, *15*(1), 49–61. https://doi.org/10.1002/(sici)1099-131x(199601)15:1<49::aid-for604>3.0.co;2-2

Gers, F. A. (1999). Learning to forget: Continual prediction with LSTM. *9th International Conference on Artificial Neural Networks: ICANN '99*. https://doi.org/10.1049/cp:19991218

Grudnitski, G., & Osburn, L. (1993). Forecasting S&P and Gold Futures Prices: An application of Neural Networks. *Journal of Futures Markets*, *13*(6), 631–643. https://doi.org/10.1002/fut.3990130605

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hu, Z., Zhao, Y., & Khushi, M. (2021). A survey of Forex and Stock Price Prediction using Deep learning. *Applied System Innovation*, *4*(1), 9. https://doi.org/10.3390/asi4010009

Nelson, D. M., Pereira, A. C., & de Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. *2017 International Joint Conference on Neural Networks (IJCNN)*. https://doi.org/10.1109/ijcnn.2017.7966019

Nikou, M., Mansourfar, G., & Bagherzadeh, J. (2019). Stock price prediction using deep learning algorithm and its comparison with machine learning algorithms. *Intelligent Systems in Accounting, Finance and Management*, *26*(4), 164–174. https://doi.org/10.1002/isaf.1459

Nugent, C. (2018, February 10). S&P 500 stock data. Retrieved November 7, 2022, from https://www.kaggle.com/datasets/camnugent/sandp500

Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2017). Forecasting stock prices from the limit order book using Convolutional Neural Networks. *2017 IEEE 19th Conference on Business Informatics (CBI)*. https://doi.org/10.1109/cbi.2017.23

Yang, S., Yu, X., & Zhou, Y. (2020). LSTM and GRU neural network performance comparison study: Taking Yelp Review dataset as an example. *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*. https://doi.org/10.1109/iwecai50956.2020.00027