# Stock Market Prediction using Long Short-Term Memory

Stylianos Gavriel
University of Twente
P.O. Box 26, 7523SB Enschede
The Netherlands
s.gavriel@student.utwente.nl

## ABSTRACT

Strategies of the stock market are widely complex and rely on an enormous amount of data. Hence, predicting stock prices has always been a challenge for many researchers and investors. Much research has been done, and many machine learning techniques have been developed to solve complex computational problems and improve predictive capabilities without being explicitly programmed. This research attempts to explore the capabilities of Long Short-Term Memory a type of Recurrent Neural Networks in the prediction of future stock prices. Long Short-Term Memory variations with single and multiple feature models are created to predict the value of S&P 500 based on the earnings per share and price to earnings ratio.

## Keywords

Long Short-Term Memory, Market Prediction, Recurrent Neural Networks, Root Mean Square Error.

## 1. INTRODUCTION

The stock market can be seen as the public marketplace, where shares and other financial instruments are being sold and bought everyday. Each share represents a portion of a company's ownership, and S&P 500 constitutes shares of the five hundred most important United States companies [19].

From the appearance of markets, investors explored ways to acquire more knowledge of the companies listed in the market, and further tried to keep up with the enormous number of news feed in the world. With the increase of market size and the speed at which trades are executed investors became less capable on relying on personal experience to identify market patterns. As technology progressed, investors and researches have developed many techniques and various models to solve problems that arise. Examples of those techniques are statistical models [3, 26], machine learning methods [18], artificial neural networks [31] and many more. The first generated trade procedures used historical data and can be traced back to the early 1990s, focused on achieving positive returns with minimal risk [10]. In the 2000s major advances in deep learning and reinforcement learning allowed for the creation of many hybrid algorithms using core principles for the prediction of stock value [10]. However, models at the time period of the stock market crash of 2008 often referred as the depression of 2008, demonstrated limitations at their abilities to forecast during periods of rapid changing prices [25].

Furthermore, most studies are conducted using a single time scale feature of the stock market index, it is therefore reasonable for studying multiple time scale features to determine a more accurate model outcome. It is important to note that markets are affected by many elements such as political, industrial development, market news, social media and economic environments. One reason for the luck of predictability is that appropriate variables to model are unknown and hard to acquire.

### 1.1 Research Question

This research attempts to answer the following questions:

RQ: To what extend can one find a more accurate Long Short-Term Memory (LSTM) based method for stock market prediction?

In order to answer this question an analysis of the input data selection and prediction methods will be made. Consequently, the following two questions will be addressed.

- RQ1: Can the prediction performance increase by selecting a different combination of variables?
- RQ2: Can the prediction performance increase by using other LSTM based features?

### 1.2 Main Contribution

This research attempts to analyse the capabilities of Recurrent Neural Networks using LSTM to predict future stock prices. A popular data-set from finance.yahoo.com, will be compared with an alternative data-set from multpl.com. Variations of LSTM based models will be trained and evaluated. There are two main contributions of this research:

- A deep understanding of the S&P 500 databases.
- Customized deep learning methods based on LSTM, both for single and multiple features, aiming to obtain a more accurate prediction model.

In the remaining of this paper included in Section 2 is the background giving more insight into the stock markets and recurrent neural networks used in this research. Further, some related researches are elaborated in Section 3, the approach of the research is being explained in Section 4, data used is analysed in Section 5 and experiments for the LSTM variants are being presented in Section 6. Finally a discussion and future work are being presented in Section 7.


*$34^{th}$ Twente Student Conference on IT* Jun. $29^{th}$, 2021, Enschede, The Netherlands.

## 2. BACKGROUND

In order to perform research in the field of predicting stock prices it is important to understand the features found in the market, and the machine learning techniques that will be used. In this section, firstly an indication of quantitative data about S&P 500, and secondly recurrent neural networks are being elaborated.

### 2.1 Stock Market

The S&P 500 is a stock market index which measures the performance of the five hundred largest companies in the United States, such companies include Apple Inc., Microsoft, Amazon.com and many more. A share is characterized by a price which is available on the S&P 500 index [1]. Stock markets usually open during weekdays at nine-thirty a.m. and close at four p.m eastern time. Many data-sets used for the prediction of prices include features such as the Open, Close, High, Low, Adjusted Closing price and Volume [17]. High and Low refer to the prices of a given stock at its maximum and minimum of a day, respectively. Adjusted Closing refers to the closing price taking into account any corporate actions, which differs from the raw closing price. Finally, Volume characterises the amount of stocks sold and bought each day. Earnings per share (EPS) is an important measure, which indicates how profitable a company is [14]. Price to Earnings ratio (P/E) refers to the ration of the current stock price to their EPS [13].

### 2.2 Recurrent Neural Networks

Recurrent Neural Networks (RNN) are a class of neural networks specifically designed to handle sequential data. There are two types of RNNs, the discrete-time RNNs and continue-time RNNs [35]. They are designed with a cyclic connection architecture, which allows them to update their current state given the previous states and current input data. RNNs are usually artificial neural networks that consist of standard recurrent cells. These types of neural networks are known to be very accurate for solving problems. It is specialized in processing a sequence of values $\chi^1..\chi^n$ where $n$ is the total number of features, and $\chi$ are the features, such as time-series data. Scaling of images with large width and height, and processing images of variable size is also feasible to a large extent. Furthermore, most RNNs are capable of processing sequences of variable length. However, RNNs are lacking the ability to learn long-term dependencies as it is illustrated in a research contacted by Yashoua et al [8]. Therefore, in order to handle these long-term dependencies, in 1997 Hochreiter and Schmidhuber proposed a solution called Long Short-Term Memory (LSTM) [16].

## 3. RELATED WORK

Since the evolution of artificial intelligence many attempted to combine deep learning and machine learning using core principles. Artificial intelligence methods include convolutional neural network, multi-layer perceptron, naive Bayes network, back propagation network, recurrent neural network, single-layer LSTM, support vector machine and many more [12]. A study in 2018 by Sima et al. [30] has shown a comparison between LSTM and ARIMA [4] a model used for analysing time series data. This study focused on implementing and applying financial data on an LSTM which was superior to the ARIMA model. Further, a study by Khaled A. Althelaya et al. in 2018 [5], evaluated the performance of bidirectional and stacked LSTM for stock market prediction. The performance of the tuned models where also compared with a shallow and an unidirectional LSTM. The study concluded that the bidirectional and stacked LSTMs had better performance for short term prices opposed to the long term prediction results. Further, the results have shown that deep architecture outperformed their shallow counter parts. Another example is a study made in 2015 by Roondiwala et al. [27], which attempted to create a model based on LSTM for an accurate read of the future market indices. The researchers further analysed various hyperparameters of their model and mainly the number of epochs used with various variable combinations of the market. At the end they concluded that using multiple variables (High/Low/Open/Close) resulted to the least errors. Latter on, in 2020 Hao et al. [34], proposed a hybrid model based on LSTM and multiple time scale feature learning and compared it to other existing models. The study also compared models based on single time scale feature learning. Furthermore, design was made to combine the output representation from three LSTM based architectures. A study by David G. McMillan [24] attempts to understand the variables proxy for changes in the expected future cash flow. It was concluded that forecasting combinations outperform single future models.

Looking into the combinations of the features, the hyperparameters and different LSTM variations would allow to better understand LSTMs and expand on the research of this topic in general. From the related studies we expect that deep architectures would outperform their shallow counterparts and multiple feature combinations would perform generally better than single features.

## 4. METHODOLOGY

In this section included is an introduction with all the major steps which will be considered in this research project: (i) data acquisition, (ii) data prepossessing, (iii) details about the RNN-based models, and (iv) the evaluation metrics.

(i) Data used for this research will be extracted from multpl.com [2] and finance.yahoo.com [1].

(ii) Data will be normalized using a python library sklearn. Feature scaling is a method to normalize the range of independent feature variables. Data is then split into a training and a testing set.

(iii) Data will be used to train the single time scale feature models over many iterations to predict each variable independently. As many traditional predicting models Closing Price will be used as a feature to train the control model. Another model will be then trained using the price from multpl.com as feature. After evaluating these methods, multiple time scale feature models are created and trained. First a control model will be trained using the best possible features of the standard data-set, which will be selected by running tests for each combination of features and comparing their losses, and the label will be set to the Close price. The stock price can be calculated with the equation below (2) using EPS and PE creating a new feature Calculated Price. Multiple feature models will be then trained using the EPS, PE and Calculated Price as features and the Price as the label, and further compared with the control model. Finally a comparison of the traditional models with the proposed multiple feature models will be made. A standard dropout LSTM model will be optimized though experimentation of hyperparameters and compared with other variants of LSTM.

(iv) Evaluation of the methods will be made using root mean squared error (1) and visuals will be created de-

picting predicted and real values, where N are the total number of values, $Y_i$ is the predicted price value and $\hat{Y}_i$ is the real price.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{Y}_i - Y_i)^2} \quad (1)$$

Figure 1 is depicting the sequence of the process that is followed for each model.


**Figure 1. Process for each model**

## 4.1 Long Short-Term Memory

As mentioned in Section 2.2 Long Short-Term Memory (LSTM) was introduced by Hochreiter and Schmidhuber in 1997 [16] to cope with the problem of long-term dependencies. LSTM consist of a similar RNN architecture that has been shown to outperform traditional RNN on numerous tasks [16]. LSTM networks work extremely well with sequence data and long-term dependencies due to their powerful learning capabilities and memory mechanisms. By introducing gates they were able to improve memory capacity and control the memory cell. One gate is dedicated for reading out the entries from the cell, the output gate. Another gate is needed to decide when data should be read into the cell, this is called the input gate. Finally a forget gate which resets the content of the cell. This design was used in order to decide when to remember and ignore inputs at the hidden state. A sigmoid activation function computes the values of the three gates, these values belong in the range of (0, 1), and represent the current time step and hidden state of the previous time step. The hidden states values are then calculated with a gated version of the tangent activation function of the memory cell which take values in the range of (-1, 1) [37].

## 4.2 Stacked Long Short-Term Memory

Stacked LSTMs are now a stable technique for challenging different sequential problems introduced by Graves et al. in the paper of speech recognition in 2013 [15]. Existing studies [22] have shown that LSTM architectures with several hidden layers can build up higher level of representation of sequential data, therefore working more effectively and with higher accuracy. Its architecture comprises of multiple stacked LSTM layers, where the output of a model's hidden layer will be fed directly at the input of the subsequent hidden layer. Instead of the traditional multilayer LSTM architecture where a single layer provides a single output value to the next layer, stacked LSTM provides a sequence of values.

## 4.3 Bidirectional Long Short-Term Memory

A bidirectional LSTM (BiLSTM) invented in 1997 by Schuster and Paliwal [29], is capable of getting trained with the sequence of data both forwards and backwards into two separate recurrent networks which are connected into the same output layer [29, 6]. The idea is that you split the state of neurons of a network in a part that is responsible for the forward states starting from a date frame of t=1 and a part for the backwards direction starting from t=T.

# 5. DATA


**Figure 2. Real Price and Calculated Price (PriceCal) using formula (2) as an introduced feature.**

This section focuses on describing the original data collected, the processing steps, and the feature selection.

The first data-set is collected from finance.yahoo.com [1]. Yahoo is one of the best resources for stock research because it is freely available and provides stock data from around the world. Yahoo provides approximately 1,822,800 records of the S&P 500 index from 1927 to 2020. For the purposes of this research, data of ten years is used from 2010 to 2020, with a total number of 19,600 approximately records. The second data-set is collected from multpl.com [2]. This website provides S&P 500 data not only of the price index but of the price to earnings ratio, earnings per share and dividend yield, to name a few. There are approximately 5,400 records of monthly data, from April 1st, 1871 to January 28, 2021. Moreover, data of the last 120 years is used, with approximately 4,350 records. Needless to say that calculating the price gives values very similar to the real price. The formula (2) can be used to introduce another feature to the data-set. The graph in Figure 2 depicts the real price of S&P 500 from multpl.com and the calculated price.

$$EPS \times P/E = StockPrice \quad (2)$$

## 5.1 Datasets Basic Statistics

In order to understanding this data, numpy from python was used to calculate the mean and standard deviation for each feature. Table 1 shows some statistics, Figure 3 shows some box plots with the data collected from finance.yahoo.com and data from multpl.com, with an extra calculated price using formula (2). Data in the boxplots and for the rest of the research will be scaled between zero and one. Figure 3 would allow to understand the distribution of numerical data and skewness through displaying the data quartiles and averages. We can hence observe that Open, High, Low, Close and Adjusted Close follow a very similar trend with the mean being almost identical. Moreover, Volume has a huge number of outliers, that differ significantly from other observations or overall Volume data has huge variations of numbers. The same can be said for PE, EPS, Price and the price calculated with formula (2). Machine learning are generally sensitive to the distribution and range of values. Therefore, outliers may mislead and spoil the training process resulting in more losses and longer training times. A paper by Kai Zhang et al. in 2015 has concluded that outliers played a huge role at the performance of Extreme Learning machines (ELM) [38].

In order to make Open, High, Low and Close more clear

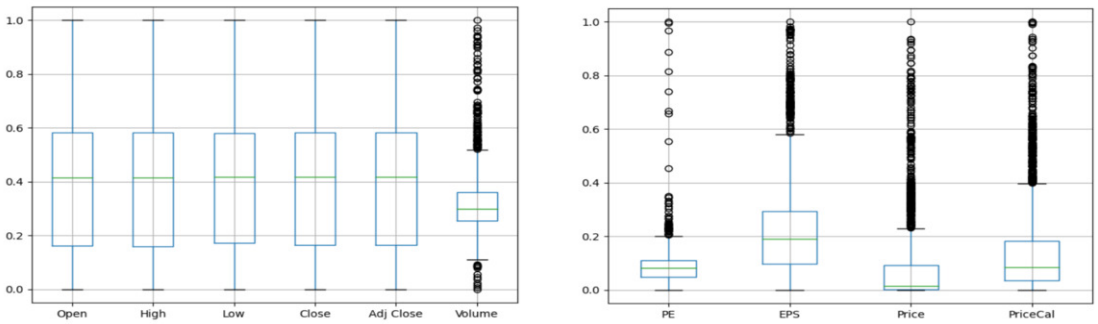



**Figure 3. A box plot for, Open, High, Low, Close, Adjusted Close and Volume**

**Table 1. Data-set statistics, in terms of mean ($\mu$) and standard deviation ($\sigma^2$). The first six rows are data from finance.yahoo.com, while the next three rows are data from multpl.com.**

| Features | Mean | Standard Deviation |
|---|---|---|
| Open | 2570.66 | 432.82 |
| High | 2583.49 | 435.41 |
| Low | 2556.43 | 430.15 |
| Close | 2570.89 | 432.77 |
| Adj. Close | 2570.89 | 432.77 |
| Volume [$\times 10^7$] | 383.20 | 95.51 |
| PE | 16.13 | 9.25 |
| EPS | 38.29 | 28.42 |
| Price | 379.47 | 676.48 |
| Calculated Price | 678.29 | 704.70 |

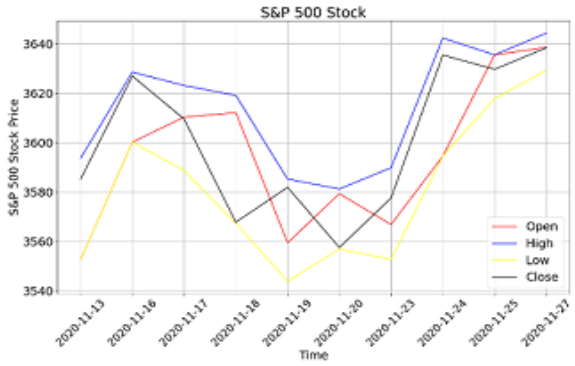



**Figure 4. Ten days of S&P 500, Open, High, Low and Close.**

Figure 4 is included. It is observed that Open and Close fluctuate between High and Low, and the overall data is following the same trend hence the high correlation levels observed. The Calculated Price can be used as an extra feature for prediction purposes.

While Open, High, Low, Close and Adjusted Close price are almost identical they present some very minor differences, which should in theory pose no huge effects for the selection process of the model. Adjusted Close price is identical to the feature Close, therefore for the purpose of this research Adjusted Close will not be used. Figure 5 shows the correlation between features in heat maps. It is noted that Volume has the least correlation between features. Further, Close and Adjusted Close have 100% correlation supporting the previous statement of being identical. Price is highly dependent on EPS, but surprisingly less on P/E ratio. The Calculated Price as expected has high correlation levels with the real Price, and should allow for overall good results when used.

Statistics should be able to give us more insight into the data and is generally considered an indispensable piece to the field of machine learning. Understanding the data and the characteristics of it is really important to finally come to a conclusion about certain results found in the subsequent sections. In the next section I will execute some experiments in order to select the best features that could be applied on LSTMs.

# 6. EXPERIMENTS AND RESULTS

In this section a model is constructed as a basis of testing features, their combinations and model parameters.

## 6.1 LSTM Model Details

LSTMs in general are capable of coping with historical data, hence they are really good candidates for stock prediction. LSTMs can learn the order dependence between items in a sequence and are known for their good performance on sequence data-sets. For the purposes of selecting the best combination of features, a dropout based LSTM model (DrLSTM) with four hidden LSTM layers and 50 units per hidden layer is trained and evaluated. Each hidden LSTM layer has a subsequent dropout layer and finally a dense layer is used to connect all the neurons followed by the last dropout. Dropout is a technique which selects neurons that will be ignored during training, this means that their contribution to the activation of downstream neurons is temporarily removed. The structure of the DrLSTM is found in Figure 7 of the appendix. The DrLSTM is trained with windows of 60 previous days predicting the next day. Table 2 show the windows of days, where X are the input arrays for the 60 days of data and y are the predicted prices per day, the outcomes of the model for each array X and finally n is the total amount of days in the data-set.

**Table 2. Sliding window input (X, blue), the outcomes (y, red), and n the number of days in total.**

| Days | 1 | 2 | 3 | ... | 60 | 61 | 62 | 63 | ... | d |
|---|---|---|---|---|---|---|---|---|---|---|
| X1 | | | | | | y1 | | | | |
| X2 | | | | | | | y2 | | | |
| X3 | | | | | | | | y3 | | |
| ... | | | | | | | | | | |

## 6.2 Feature Selection

In order to perform the feature selection step I have done a grid-search using all future combinations. There are $\frac{d!}{(r!(d-r)!)}$ of possible combinations for each data-set, where $d$ is the total number of possibilities and r is the number

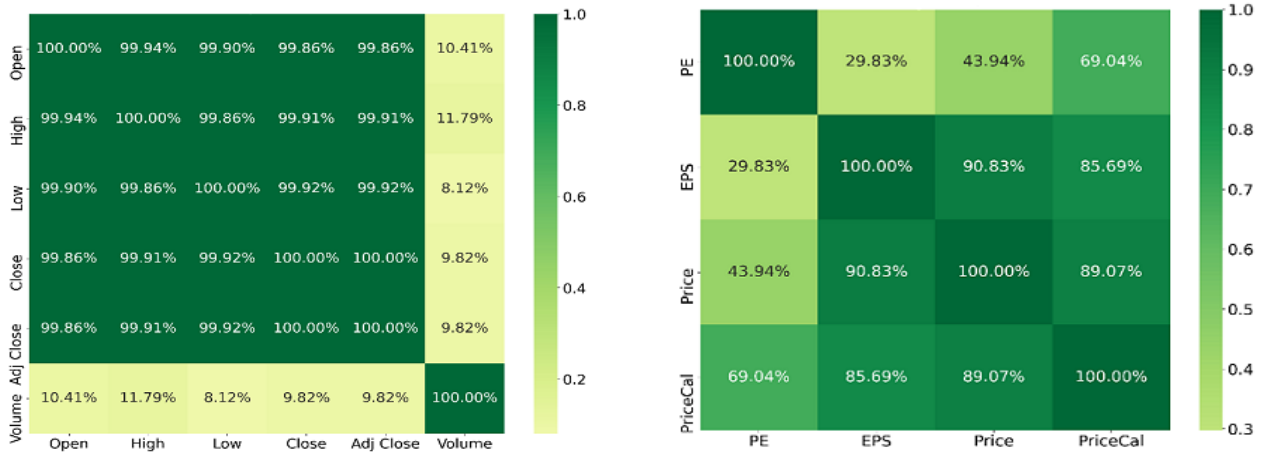



**Figure 5. Correlation Coefficient for the data-set of finance.yahoo.com.**

of selections made. For each of the r values between two to five selected and a total of five features, there are 27 total combinations of the data-set from finance.yahoo.com including the single feature Close price. For the data-set from multpl.com, there are a total of 12 combinations taking into account the Calculated Price as a feature. Results of the trained DrLSTM model for each combination of the data-sets are found in Table 4 and Table 5 of the appendix. Based on this results, we can therefore conclude that a single feature DrLSTM is capable of performing surprisingly better than multiple feature combinations. For the selection of two features, $\{High, Volume\}$ and $\{EPS, Price\}$ had the best results throughout the combinations. For a selection of three features $\{High, Low, Close\}$ as well as $\{PE, Price, CalculatedPrice\}$ had low loss values. For a selection of four features $\{High, Low, Close, Volume\}$ had performed well while the rest of the data from multpl- .com did not perform equally or better than the rest of the combinations. Since $\{Close\}$ had good results during feature selection it is therefore used to run the rest of the tests to optimize DrLSTM.

## 6.3 LSTM Model Hyperparameters

In this section I will attempt to make a selection of parameters opposing to the model's loss.

Model parameters such as neuron weights, are the fitted parameters of a model that are being estimated and learned from the training set. On the other hand, hyperparameters which are adjustable and must be tuned in order to obtain a model with optimal performance. Therefore, I will run some experiments to determine the optimal number of nodes, dropout probability and optimizers for the best adequate performance of DrLSTM. Figure 6 shows the results for the number of nodes per layer with a static dropout probability of 0.2 for the DrLSTM model introduced in the previous section. Secondly Figure 6 also depicts the results for a dropout probability range of 0.05 to 0.3 with 50 nodes per layer. From Figure 6, we can conclude that adding more nodes would lead to better results in some cases. However since the time required to train the DrLSTM with 150 nodes exceeding by far the process for 50 nodes, and the results show insignificant difference we will proceed the research with a selection of 50 nodes totaling 200 throughout. As with the dropout probability we can observe that a decreased number of ignored nodes can potentially lead to better results. With this in mind we expect that a stacked LSTM (StLSTM) architecture where the dropout layers are skipped, can lead to better results.

I will further investigate some types of optimizers which can contribute to the DrLSTM's optimization process. Optimizers are algorithms used to change parameters of neural networks such as weights and learning rate in order to reduce losses [28]. Keras from python is used to create and train the DrLSTM where optimizers are one of the two parameters required for compiling a model. Therefore, analysing the performance of optimizers in this scenario could potentially prove worthy. In order to run these tests the same DrLSTM is used as a basis of the comparison. Figure 6 shows the results from: Adam [20], RMSprop [32], SGD [33], Adadelta [36] and Adamax [20] optimizers. In conclusion there is a remarkable difference of the Adam and the rest of the optimizers, hence for the rest of this research the Adam optimizer will be used. Adam was firstly introduced in 2014 by Kingma and Ba [20]. It is an adaptive learning rate optimization algorithm generally performing well in a vast array of problems.

## 6.4 Model Variants

Now that we have established some features and parameters, we can proceed into testing different variants of the LSTM model. This comparison would allow us to find the best LSTM variant throughout the models analysed in this research paper.

We start with the DrLST model introduced in Section 6.1, we then proceed with a StLSTM and a shallow LSTM model (ShLSTM) consisting of one LSTM hidden layer with 200 nodes and finally a bidirectional LSTM (BiLSTM) model consisting of the same number of nodes. Architectures of the model variants are included in Figure 7 of the appendix. The tests are completed using the best features which provided the least losses for every number of combinations. Finally the optimizer used for testing is set to Adam. Table 3 depicts the losses from the tests that have been performed, and Figure 8 of the appendix shows the graphs plotted using pyplot of python for the best result of each model.

From Table 3 it is depicted that the DrLSTM had the least performance throughout the models. A stacked LSTM with a loss of 0.0247 has proven to perform better than the model with dropouts, this is mainly caused by the absent of dropout layers. Surprisingly ShLSTM seems to be slightly better than the previous models. This result seems out of order since many researches have shown that deep recurrent networks usually outperform their shallow coun-

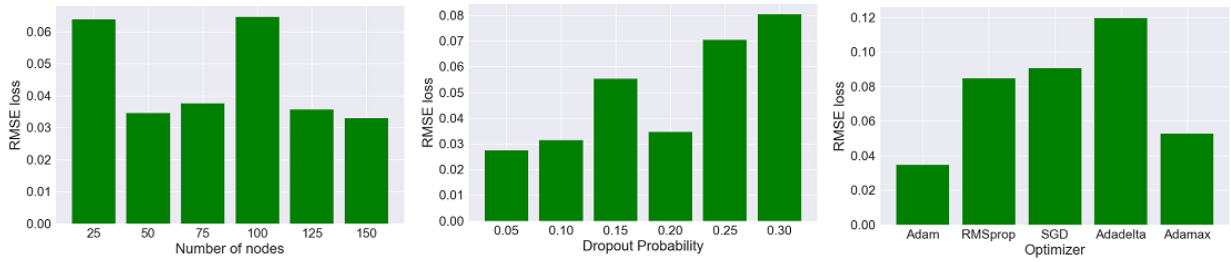



**Figure 6. Losses for number of nodes, dropout probability and optimizers used respectively.**

**Table 3. RMSE losses for LSTM four layered model with Dropouts (DrLSTM), stacked LSTM (StLSTM), shallow LSTM (ShLSTM) and bidirectional LSTM (BiLSTM).**

| Features/Models | DrLSTM | StLSTM | ShLSTM | BiLSTM |
|---|---|---|---|---|
| Close | **0.0346** | **0.0247** | **0.0230** | 0.0224 |
| High, Vol. | 0.0408 | 0.0275 | 0.0238 | 0.0233 |
| High, Low, Vol. | 0.0356 | 0.0297 | 0.0231 | **0.0219** |
| High, Low, Close, Vol. | 0.0389 | 0.0574 | 0.0233 | 0.0252 |
| Price | 0.0552 | 0.0454 | 0.0346 | 0.0712 |
| EPS, Price | 0.0411 | 0.0682 | 0.0535 | 0.0651 |
| PE, Price, Calc.Price | 0.0507 | 0.0818 | 0.0374 | 0.1197 |

terparts [23]. Finally, the best performing model was the BiLSTM which had a loss of 0.0219 with the use of multiple features. In order to understand the representative losses with real prices, it would mean that at the average closing price of 2570.89 a model with a loss of 0.0346 has a deviation of 124.73 dollars, and the best performing with a loss of 0.0219 has a deviation of 56.18 dollars. In Section 7, I will be discussing the possible reasons for the behaviours observed.

## 7. DISCUSSION AND FUTURE WORK

While experimenting with DrLSTM, we have observed that dropouts introduce a bottleneck in the adjustment of the model's parameters. In many machine learning processes it is useful to know how certain the output of a model is. For example, a prediction is more likely to be closer to the actual price when an input is very similar to elements of the training set [21]. The outputs of a dropout layer are randomly ignored, therefore having the effect of reducing the capacity of a network during training. Requiring more nodes in the context of dropout could potentially remove this bottleneck. In Figure 6 we have observed that increasing the nodes gives more positive outcomes. The StLSTM supports this argument since dropout layers are absence hence the better performance. A ShLSTM was the second most performing model, contrary to what I was expecting. A good reason is that the 200 nodes used to train the ShLSTM in one layer was a much better fit for the data used contrary to the 50 nodes per layer of deep counterparts. A book by Andrew R. Barron in 1993 [7] gives more insight into the size of a single-layer neural network needed to approximate various tasks. Furthermore, the BiLSTM had performed the best throughout the experiments and could potentially be used for long term transactions in the stock market, however it leaves much room for improvement. Since the BiLSTM passes the data-set twice it makes certain trends more visible, adding more weight to certain neurons and extending data usage [11]. LSTM architecture is mainly used for long term dependencies, so it is generally good to have more and more contextual information.

In this research the default optimizer parameters where used and seemed to perform generally good. However, running more experiments while adjusting the Adam's parameters accordingly can provide improvements. More improvement can be achieved also by looking into the depth (number of layers) and width (number of nodes) for each variant. The window span used to create the input data for the models could be tested for values more or less than 60 days. Finding a more suitable window could also fix the lag observer of the predicted price from the real values. A study by Salah Bouktif et al. [9], tried solving the lag arising from time features, with a selection of appropriate lag length using a genetic algorithm (GA). A deviation of 56.18 dollars for short term transactions could seem high since a stock index in general requires more than a couple of days to deviate significantly in order to minimize trading losses, therefore even the BiLSTM leaves much room for improvement.

## 8. CONCLUSION

This research paper attempts to forecast the S&P 500 index using multiple LSTM variants while performing several experiments for optimization purposes. I trained the models with a popular data-set from finance.yahoo.com and a data-set from multpl.com. This paper has proven that a single feature selection has performed better in some instances while multiple features have proven advantageous in BiLSTMs. The testing results conform that the LSTM variants are capable of tracing the evolution of closing price for long term transactions leaving much room for improvement of daily transactions. This study gave insight into two different data-sets and analysed the results of different variants of LSTM, which should allow researches and investors to use and expand upon in the future. Although one of the many machine learning techniques has been used in this research, there are many more methods that can be broken down into two categories (statistical techniques and artificial intelligence).

## 9. ACKNOWLEDGMENT

I would like to thank Dr. Elena Mocanu for helping me during the execution of this research. I also appreciate the time she spend directing me to the right sources.

# 10. REFERENCES


[1] SNP, Dec 10, 2020: S&P 500 (^GSPC), retrieved from: https://yhoo.it/3ikW3DM.

[2] multpl https://www.multpl.com/s-p-500-pe-ratio.

[3] A. Adebiyi, A. Adewumi, and C. Ayo. Comparison of arima and artificial neural networks models for stock price prediction. J. Appl. Math., 1–7. 2014.

[4] R. Adhikari and R. K. Agrawal. An introductory study on time series modeling and forecasting, 2013.

[5] K. A. Althelaya, E. M. El-Alfy, and S. Mohammed. Evaluation of bidirectional lstm for short-and long-term stock market prediction. In *2018 9th International Conference on Information and Communication Systems (ICICS)*, pages 151–156, 2018.

[6] P. Baldi, S. Brunak, P. Frasconi, G. Soda, and G. Pollastri. Exploiting the past and the future in protein secondary structure prediction BIOINF: Bioinformatics, p. 15. 1999.

[7] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

[8] Y. Bengio, P. Frasconi, and P. Simard. The problem of learning long-term dependencies in recurrent networks. In *IEEE International Conference on Neural Networks*, pages 1183–1188 vol.3, 1993.

[9] S. Bouktif, A. Fiaz, A. Ouni, and M. Serhani. Optimal deep learning lstm model for electric load forecasting using feature selection and genetic algorithm: Comparison with machine learning approaches †. *Energies*, 11(7):1636, Jun 2018.

[10] S. Chakraborty. Capturing financial markets to apply deep reinforcement learning, 2019.

[11] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, and I. Androutsopoulos. Neural contract element extraction revisited. In *Workshop on Document Intelligence at NeurIPS 2019*, 2019.

[12] G. Ding and L. Qin. Study on the prediction of stock price based on the associated network model of lstm. *International Journal of Machine Learning and Cybernetics*, 11, 06 2020.

[13] J. Fernando. Price-to-Earnings Ratio – P/E Ratio https://www.investopedia.com/terms/p/price-earningsratio.asp. Nov 13, 2020.

[14] J. Fernando. Earnings Per Share – EPS Definition https://www.investopedia.com/terms/e/eps.asp. Nov 17, 2020.

[15] A. Graves, N. Jaitly, and A. r. Mohamed. "Hybrid speech recognition with deep bidirectional lstm," in Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on. IEEE, pp. 273–278. 2013.

[16] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997.

[17] J. Jagwani, M. Gupta, H. Sachdeva, and A. Singhal. Stock price forecasting using data from yahoo finance and analysing seasonal and nonseasonal trend. pages 462–467, 06 2018.

[18] Y. Kara, M. Acar, and Baykan. Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock Exchange. Expert Syst. Appl.,38, 5311–5319. 2011.

[19] W. Kenton. SP 500 Index – Standard Poor's 500 Index, https://bit.ly/2LWBUYO, Dec 22, 2020.

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[21] A. Labach, H. Salehinejad, and S. Valaee. Survey of Dropout Methods for Deep Neural Networks. 25 Octomber 2019.

[22] Y. LeCun, Y. Bengio, and G. Hinton. "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444. 2015.

[23] Y. Levine, O. Sharir, and A. Shashua. Benefits of Depth for Long-Term Memory of Recurrent Networks. 15 February 2018.

[24] D. G. McMillan. Which variables predict and forecast stock market returns? 2016.

[25] T. Moyaert and M. Petitjean. The performance of popular stochastic volatility option pricing models during the subprime crisis. Applied Financial Economics. 21(14). 2011.

[26] P.-F. Pai and C.-S. Lin. A hybrid arima and support vector machines model in stock price forecasting. Omega, 33, 497–505. 2005.

[27] M. Roondiwala, H. Patel, and S. Varma. Predicting stock prices using lstm. *International Journal of Science and Research (IJSR)*, 6, 04 2017.

[28] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[29] M. Schuster and K. Paliwal. Bidirectional recurrent neural networks IEEE Transactions on Signal Processing, 45, pp. 2673-2681. 1997.

[30] S. Siami-Namini and A. S. Namin. Forecasting economics and financial time series: Arima vs. lstm, 2018.

[31] T. J. Strader, J. J. Rozycki, T. H. Root, and Y. J. Huang. Machine learning stock market prediction studies: Review and research directions, Journal of International Technology and Information Management, 28(3). 2020.

[32] T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

[33] J. Yang and G. Yang. Modified convolutional neural network based on dropout and the stochastic gradient descent optimizer. *Algorithms*, 11(3), 2018.

[34] H. Yaping and G. Qiang. Predicting the trend of stock market index using the hybrid neural network based on multiple time scale feature learning. *Applied Sciences*, 10(11), 2020.

[35] Y. Yu, X. Si, C. Hu, and J. Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural Computation*, 31(7):1235–1270, 2019.

[36] M. D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[37] A. Zhang, Z. Lipton, M. Li, and A. Smola;. *Dive into Deep Learning*. 2020.

[38] K. Zhang and M. Luo. Outlier-robust extreme learning machine for regression problems, volume 151, part 3. pages 1519–1527, March 5 2015.

# APPENDIX

# A. EXPERIMENTS AND RESULTS

## A.1 Feature combinations

**Table 4. RMSE losses from feature combinations. (to predict Close values)**

| Features | RMSE Loss |
|---|---|
| Close | **0.0346** |
| Open - High | 0.0446 |
| Open - Low | 0.0573 |
| Open - Close | 0.0616 |
| Open - Volume | 0.0578 |
| High - Low | 0.0513 |
| High - Close | 0.0505 |
| High - Volume | **0.0408** |
| Low - Close | 0.0675 |
| Low - Volume | 0.0412 |
| Close - Volume | 0.0479 |
| Open - High - Low | 0.0497 |
| Open - High - Close | 0.0666 |
| Open - High - Volume | 0.0506 |
| Open - Low - Close | 0.0755 |
| Open - Low - Volume | 0.0623 |
| Open - Close - Volume | 0.0607 |
| High - Low - Close | 0.0367 |
| High - Low - Volume | **0.0356** |
| High - Close - Volume | 0.0711 |
| Low - Close - Volume | 0.0683 |
| Open - High - Low - Close | 0.0432 |
| Open - High - Low - Volume | 0.0455 |
| Open - High - Close - Volume | 0.0697 |
| Open - Low - Close - Volume | 0.0588 |
| High - Low - Close - Volume | **0.0389** |
| Open - High - Low - Close -Volume | **0.0548** |

Table 4 shows the results from the combination of features collected from finance.yahoo.com. Even though many machine learning models have better results with a selection of multiple features, in this research it was proven that a single feature was capable of performing the better.

**Table 5. RMSE losses from feature combinations(to predict Price).**

| Features | RMSE Loss |
|---|---|
| Price | **0.0552** |
| EPS - PE | 0.5294 |
| EPS - Price | **0.0411** |
| EPS - Calculated Price | 0.3440 |
| PE - Price | 0.1212 |
| PE - Calculated Price | 0.5054 |
| Price - Calculated Price | 0.0935 |
| EPS - PE - Price | 0.0968 |
| EPS - PE - Calculated Price | 0.5305 |
| EPS - Price - Calculated Price | 0.0916 |
| PE - Price - Calculated Price | **0.0507** |
| EPS - PE - Price - Calculated Price | **0.0953** |

Table 5 shows the results from the combination of features collected from multpl.com. In contrast to Table 4 a combination of two features performed the best.

## A.2 LSTM Model Parameters

**Table 6. RMSE for a number of nodes per layer.**

| Nodes no. | RMSE (Dropout 0.2) |
|---|---|
| 25 | 0.0639 |
| 50 | **0.0346** |
| 75 | 0.0376 |
| 100 | 0.0647 |
| 125 | 0.0356 |
| 150 | **0.0329** |

Table 6 shows the results of the tests performed to optimize the DrLSTM for the number of nodes per layer. It was observed that 150 nodes have performed the best however the time required to train the DrLSTM was significantly higher than 50 nodes. In the discussion Section 7 I expand on this observation.

**Table 7. RMSE for a number of nodes per layer.**

| Dropout Probability | RMSE (Nodes 50) |
|---|---|
| 0.05 | **0.0274** |
| 0.1 | 0.0314 |
| 0.15 | 0.0554 |
| 0.20 | 0.0346 |
| 0.25 | 0.0705 |
| 0.30 | 0.0805 |

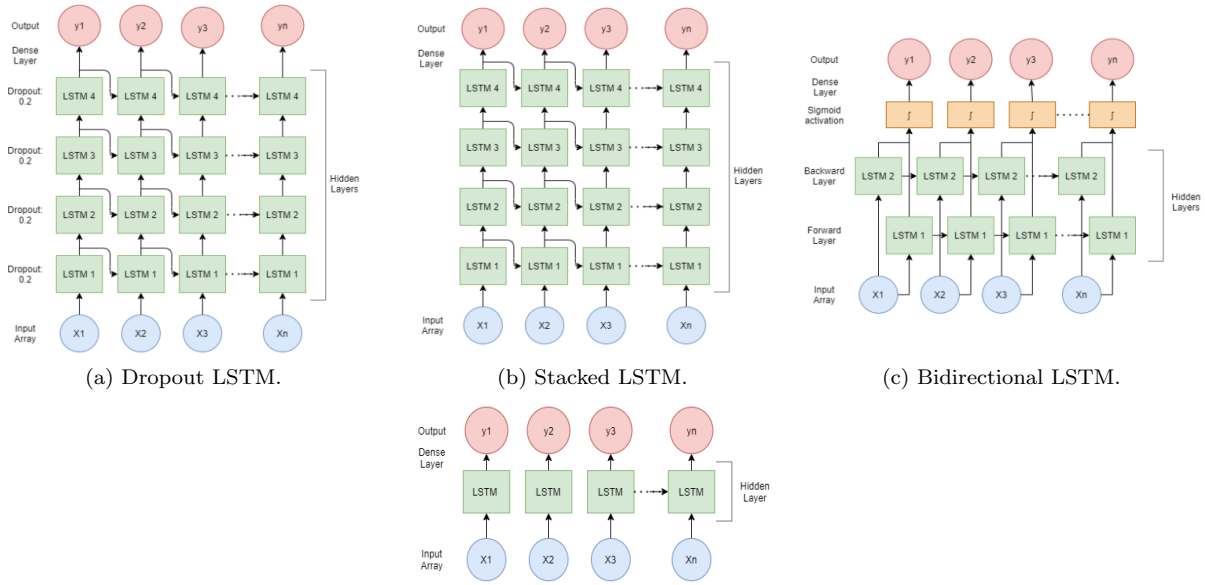
Table 7 shows the results of the tests that have performed in order to optimize the DrLSTM model, respectively with the dropout probability of the layers. You can find more information about the structure of the DrLSTM model in Figure 7. It is observed that decreasing the dropout probability would give better results. Therefore, the dropout layers are creating a barrier to the DrLSTM's process of adjusting parameters during training.

**Table 8. RMSE for a number of nodes per layer.**

| Optimizers | RMSE (50 nodes) (0.2 dropout) |
|---|---|
| Adam | 0.0346 |
| RMSprop | 0.0847 |
| SGD | 0.0905 |
| Adadelta | 0.1198 |
| Adamax | 0.0528 |

Table 8 shows the results of the tests that have performed in order to optimize the DrLSTM model, respectively with the optimizers that the DrLSTM uses to adjust the parameters of the model during training. It was observed that the Adam optimizer is performing the best for the purpose of this research. In Section 7, I discuss the possibility of adjusting the parameters of the optimizer, for simplicity purposes this research used the default parameters.

## A.3 Model Variants



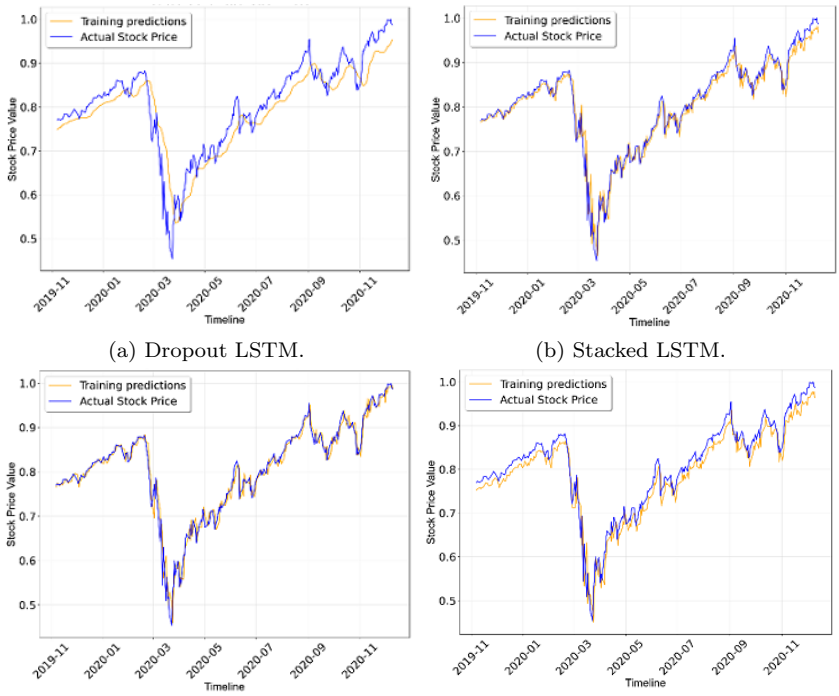

(a) Dropout LSTM.

(b) Stacked LSTM.

(c) Bidirectional LSTM.

(d) Shallow LSTM.

**Figure 7. Model variant architecture, (a) is for the dropout LSTM model (DrLSTM) which consists of 4 LSTM layers with a dropout layer each. A stacked LSTM (StLSTM) (b) consists of the same four layers LSTM excluding the dropout layers. The bidirectional LSTM (BiLSTM) (c) consists of a single forward and backward layer. The shallow LSTM (ShLSTM) in graph (d) has a single LSTM 200 node layer.**




(a) Dropout LSTM.

(b) Stacked LSTM.

(c) Bidirectional LSTM.

(d) Shallow LSTM.

**Figure 8. Best results for each model depicting the actual price and the predicted price, (a) is for dropout LSTM model, (b) is for stacked LSTM, (c) is for bidirectional LSTM and (d) is for shallow LSTM. In graph (a) it is noticeable that the predicted values (in orange) and the real price (in blue) deviate and have a noticeable lag which is touched upon in Section 7. This lag is most noticeable in (a) but can be found in the rest of the graphs as well. In graph (c) it is noticeable that the bidirectional LSTM had good performance, hence the darker color of the line.**