Stock Market Forcasting Using LSTM

Aliakbar Mehdizadeh
(Dated: December 14, 2023)

Investing in a portfolio of assets has always presented challenges, as the unpredictability of the financial market complicates the use of straightforward models for accurately forecasting future asset values. This project endeavors to construct a predictive model for sequential data using the Long-Short Term Memory model (LSTM), to anticipate future stock market values. The primary aim of this project is to evaluate the precision with which this neural network architecture can make predictions and to assess how the manipulation of epochs, batch size, learning rate, can enhance our model performance.

I. INTRODUCTION

Numerous studies have delved into the application of machine learning in quantitative finance, particularly in predicting prices, managing portfolios, and optimizing investment processes [1, 2]. Machine learning algorithms prove versatile in handling various financial operations, relying on data rather than explicit programming instructions to unveil patterns [3]. Specifically within quantitative finance and asset selection, a plethora of models offers diverse methods leveraging machine learning to forecast future asset values. These models amalgamate disparate information sources, forming a potent tool for efficient utilization [4–6].

The fluctuation in stock prices is influenced by various factors within the current industry and the overall stock market. Two primary factors come into play: 1. The impact of stock prices on other companies, illustrating how the combined stock prices of multiple firms can affect the stock price of a specific company. 2. Historical performance and data related to stock price forecasts for the company, involving the application of diverse techniques to refine the historical trends in stock market trading and predict future outcomes. The stock market is characterized as dynamic, unpredictable, and non-linear.

In this context, I employ LSTM, and Moving Average to analyze the future behavior of stocks, examining their historical patterns. In particular, I aim to employ LSTM model, ARIMA model and Moving Average to predict adjusted closing prices.

II. DATASET

In this dataset that is obtained from yahoo finance API, information spans 15 years, encompassing the years from 2008 to 2023. The dataset comprises six columns, identified as `Date`, `Open`, `High`, `Low`, `Close`, and `Volume`.

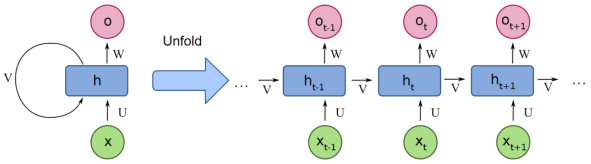

FIG. 1. Schematic Representation of a Simple Recurrent Network (SRNs).

The dataset includes various features that provide essential information for stock market analysis. The 'Date' feature serves as an index, providing a chronological order to the data. 'High' corresponds to the highest recorded stock value within a given day, while 'Low' represents the lowest observed stock value. 'Open' signifies the opening price at the beginning of the day, and 'Close' indicates the closing price at the end of the trading day. Additionally, the 'Volume' feature reflects the quantity of stock traded on a particular day.

While, all these information can be helpful in forcasting tasks our focus will be on the '`Adj Close`' price of each stock as the pertinent attribute for individual assessment in our target stock.

III. LSTM

Long Short-Term Memory (LSTM) represents a specialized category within Recurrent Neural Networks (RNNs, FIG. 1), distinguished by its unique ability to capture and learn long-term dependencies. An LSTM unit, a fundamental component of an RNN with LSTM architecture, consists of crucial elements such as a cell, an input gate, an output gate, and a forget gate. This combination enables the cell to retain information over arbitrary time intervals, while the three gates regulate the inflow and outflow of information.

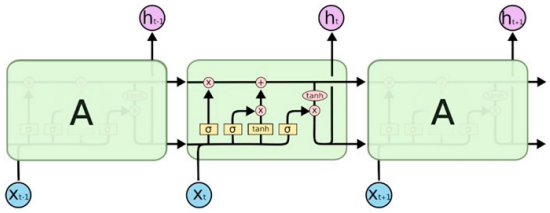

FIG. 2. The Schematic Representation of Internal Structure of an LSTM

LSTM, as a deep learning model, is particularly effective in analyzing sequential data with long-range dependencies, notably for time series data prediction. Its versatile applications span various domains, including language modeling, neural machine translation, music generation, time series prediction, financial forecasting, and more.

A. LTSM Architecture

The key components of an LSTM unit include:

`Cell State` (C_t): The cell state is the memory of the LSTM unit, allowing it to retain information over arbitrary time intervals. This feature enables the model to learn and remember relevant patterns over extended sequences.

`Input Gate` (i_t): The input gate determines how much of the new information should be added to the cell state. It regulates the flow of information from the current input.

`Forget Gate` (f_t): The forget gate decides what information from the cell state should be discarded or forgotten. It helps the LSTM unit filter out irrelevant information.

`Output Gate` (o_t): The output gate controls the information that is output from the cell state. It regulates the flow of information from the cell state to the output of the LSTM unit.

The LSTM architecture addresses the vanishing gradient problem [7] encountered by traditional RNNs, which hinders the effective learning of long-term dependencies.

IV. MODEL & METHODOLOGY

The data analyzed in this paper comprises the daily Adj Close prices of Apple Inc (Apple) on the New York Stock Exchange (NYSE) sourced from Yahoo Finance API. Check FIG. 3 and FIG. 4 for details.

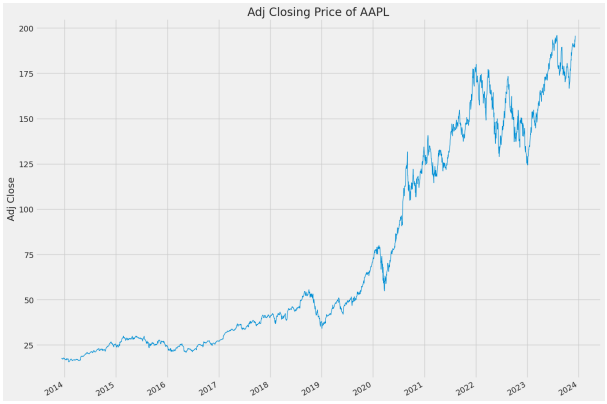

FIG. 3. Pattern of data: APPLE Adj Close Value since 2014.

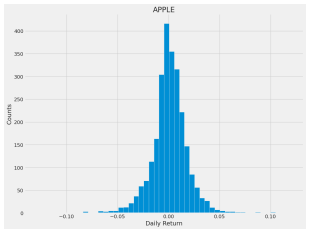

FIG. 4. One-Step Ahead Return of Apple Stock Distribution. It resembles a normal distribution which is a good sign for training our model.

The model construction involves the utilization of Long Short-Term Memory (LSTM) units. The dataset is partitioned, allocating 90% of the data for training and validation purposes and reserving the remaining 10% for testing. Also, Mean Absolute Error is employed during training to optimize the model and its internal parameter's. Various epochs, batch sizes, learning rates, memory window are employed for training data.

The structural composition of the model can be seen in FIG. 5. We have used four layers of LSTM, each with a dropout layer in between to avoid overfitting.

V. RESULTS

Following the training of the neural network, the testing results showed variations influenced by the number of epochs, batch sizes, learning rates, and the window length of the historical data (process memory).

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 30, 50)	10400
dropout (Dropout)	(None, 30, 50)	0
lstm_1 (LSTM)	(None, 30, 50)	20200
dropout_1 (Dropout)	(None, 30, 50)	0
lstm_2 (LSTM)	(None, 30, 50)	20200
dropout_2 (Dropout)	(None, 30, 50)	0
lstm_3 (LSTM)	(None, 50)	20200
dropout_3 (Dropout)	(None, 50)	0
dense (Dense)	(None, 1)	51

Total params: 71051 (277.54 KB)
Trainable params: 71051 (277.54 KB)
Non-trainable params: 0 (0.00 Byte)

FIG. 5. Architecture of the LSTM model for 30 days historical data. Its include 4 layers of LSTM with dropout later between them in order to avoid over-fitting

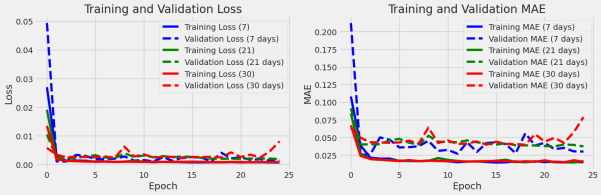

FIG. 6. Loss and MAE for Different Historical Window $[7, 21, 30]$ days.

A. Prediction Memory

The optimal history window (memory) for an LSTM model refers to the ideal sequence length of past data that the model should consider when making predictions. If the window is too short, the model may overlook important context, while an excessively long window may introduce noise and hinder the learning process. Finding the right balance is essential for achieving optimal performance.

In FIG. 6, one can observe the result of varying the history window size from 7 days to 30 days and its impact on the model's accuracy or loss. Typically, there may be a sweet spot where the model demonstrates the best generalization to unseen data, highlighting the importance of selecting an appropriate history window for effective LSTM modeling. A decreasing training loss and a relatively stable validation loss indicate that the model is learning well without overfitting. However, if the validation loss starts increasing such as what we see for the case of 31 days at the end, it might suggest that the model is overfitting to the training data. Model Predictions are shown in Fig. 7

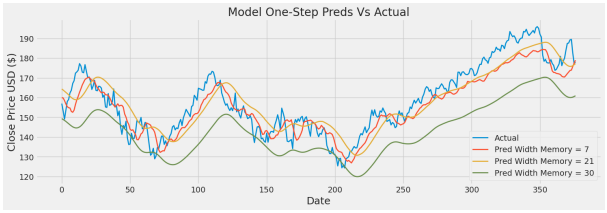

FIG. 7. Prediction with Different Memory Window

FIG. 8. Loss and MAE for Multiple Batch Sizes: [64,128,256].

B. Optimal Batch Size

A key focus was placed on determining the appropriate batch size during the training process. The figure provided illustrates the evolution of the loss and mean absolute error (MAE) over epochs for various batch sizes, FIG. 8.

It becomes evident from the plot that the choice of batch size significantly influences the convergence and stability of the model. Smaller batch sizes tended to exhibit faster convergence, yet with increased noise, while larger batch sizes demonstrated smoother curves but slower convergence. Validation loss increases at the end for *batch size* $= 256$, it may suggest overfitting.

C. Learning Rate

A critical aspect of investigation centered on identifying the optimal learning rate during the train-

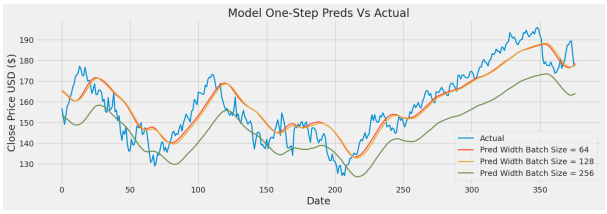

FIG. 9. Prediction with Different Batch Size

FIG. 10. Loss and MAE for Multiple Learning Rates

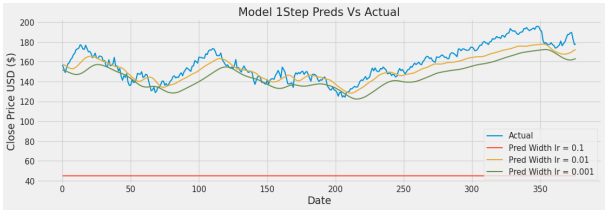

FIG. 11. Prediction with Different Learning Rates

ing process. The figure provided serves as a visual representation, depicting the historical progression of both loss and mean absolute error (MAE) across epochs for various learning rates, FIG. 10.

Too high a learning rate lead stop network from learning, while an excessively low rate may result in slow or incomplete convergence, FIG. 11.

D. One-Step aHead Prediction with Optimal Model

This is the general form of our forecasting model

$$X_{n+1} = f(X_n, X_{n-1}, \cdots, X_{n-i}) \tag{1}$$

We found out the best performance will achieve with *learning rate* = 0.01, *batch size* = 128, and *memory* = 21 days. FIG. 12 Compare model next day stock price prediction with its actual value over the test dataset that consist of around 300 datapoints.

Based on the previous hypertuning steps, we set learning rate(lr) = 0.001, *batch size* = 128, and *historical window* = 21 days. It can be seen that the model correctly predict the short trend but it is not capable to exactly predict the actual value of the stock.

For a more comprehensive comparison, we also plot predictions compared to the process moving average, which is basically the average stock price over the past 21 days as the next day prediction, see FIG. 13.

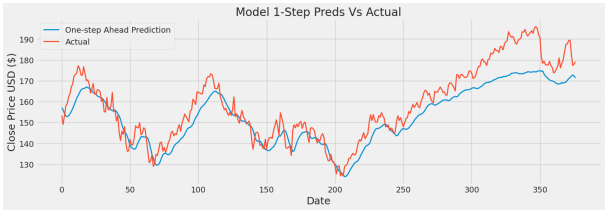

FIG. 12. Model One-Step Ahead Prediction Vs Actual Stock Value. $lr = 0.01$, *batch size* = 128, *history* = 21 days, epoch = 100

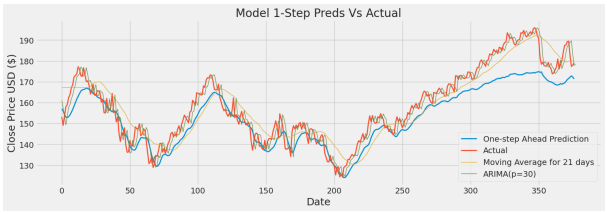

FIG. 13. Prediction compare to moving average (MA), and ARIMA models

The mean absolute percentage error (MAPE) is a measure of prediction accuracy of a forecasting method in statistics. It is used for comparing our models:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{A_i - P_i}{A_i}\right| \times 100 \tag{2}$$

Here, n is the number of observations, P_i is prediction value and A_i is the actual value.

Method	MAPE(%)
Moving Average	6.01
LSTM	4.02
ARIMA(p=7)	1.50

TABLE I. Comparison of Methods: 21-days Moving Average and LSTM, and ARIMA(6,1,0)

In conclusion, the application of Long Short-Term Memory models in stock market prediction holds significant promise and has shown noteworthy results. The LSTM model demonstrates a capacity to discern intricate patterns and trends that may elude traditional analytical methods, conditioned on a comprehensive hypertuning. Although, LSTM perform better than to moving averages in general, it potentially requires more epochs and cross-validations in order to always predict better than the moving average.

It is also noteworthy that we can use larger training datasets and employ more epochs to achieve a more accurate and stable prediction model. Additionally, deep learning models can incorporate additional features, such as stock opening and closing prices, volume, other related stock prices such as its competitors, news, and trend change recognition, to further enhance the model's performance.

[1] T. J. Strader, J. J. Rozycki, T. H. Root, and Y.-H. J. Huang, Machine learning stock market prediction studies: review and research directions, Journal of International Technology and Information Management **28**, 63 (2020).

[2] N. Rouf, M. B. Malik, T. Arif, S. Sharma, S. Singh, S. Aich, and H.-C. Kim, Stock market prediction using machine learning techniques: a decade survey on methodologies, recent developments, and future directions, Electronics **10**, 2717 (2021).

[3] D. Kumar, P. K. Sarangi, and R. Verma, A systematic review of stock market prediction using machine learning and statistical techniques, Materials Today: Proceedings **49**, 3187 (2022).

[4] A. Garlapati, D. R. Krishna, K. Garlapati, U. Rahul, G. Narayanan, *et al.*, Stock price prediction using facebook prophet and arima models, in *2021 6th International Conference for Convergence in Technology (I2CT)* (IEEE, 2021) pp. 1–7.

[5] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, N-beats: Neural basis expansion analysis for interpretable time series forecasting, arXiv preprint arXiv:1905.10437 (2019).

[6] S. J. Taylor and B. Letham, Forecasting at scale, The American Statistician **72**, 37 (2018).

[7] S.-H. Noh, Analysis of gradient vanishing of rnns and performance comparison, Information **12**, 442 (2021).