STAT 453: Stock Market Prediction using Recurrent Neural Network (LSTM) vs. Other Models

Arhum Zafar
azafar2@wisc.edu

Guk Kim
gkim95@wisc.edu

Desmond Fung
dfung2@wisc.edu

Abstract

Considering the fast growth of artificial intelligence applications, such as machine learning and deep learning, there is a strong interest towards building methods that enable us to predict trends in stock market behavior. In this project, various machine and deep learning models have been implemented and tested for their efficacy to yield accurate stock market predictions, in an attempt to find whether or not predicting the stock market is a feasible task. The primary model of focus in this project is a Recurrent Neural Network (RNN) built using the Long Short-Term Memory (LSTM) architecture. The LSTM model has been given information on stock's past behavior over the years, and is asked to make a prediction on the stock's price into the future. This task is then repeated for a number of stocks, and the results are then compared. Additionally, four other simple machine learning models are also developed in order to see how effective the LSTM model is. The project then goes into an analysis of the results attained, comparing it to the related work that has been done on the topic of stock prediction. Lastly, this analysis concludes on the complexity of the stock market, and why currently don't have the means to predict the stock market.

1. Introduction

Research has shown that today, over 50% of Americans are now involved in the stock market in some way, a similar type of growth has also been seen with investors internationally. The stock market is now home to a new crowd of investors that are hungry for success; how they obtain success is up in the air.
Considering its rising popularity and substantial monetary potential, having the means to predict future trends and behavior in the stock market would be a priceless tool for anyone. Regardless if you have no financial literacy, or if you trade stocks for a living, there are millions of people worldwide interested in the stock market and are in need of financial guidance to invest properly. This "financial guidance" that so many are sought after can come in the form of predictions.

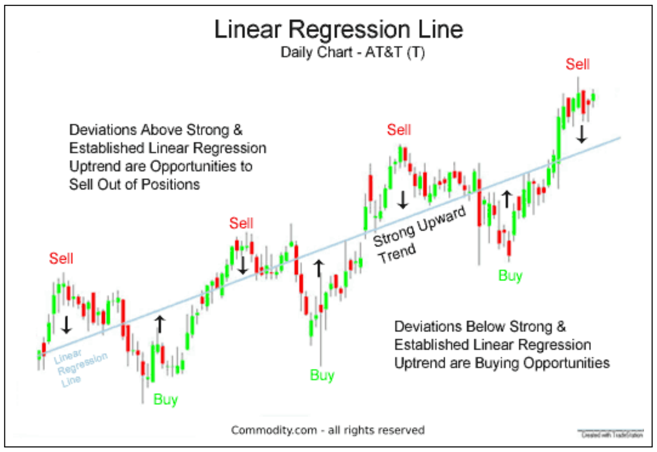


Figure 1. There have been several machine learning and deep learning applications in the financial world. Above, Linear Regression is used to let analysts know whether to hold or sell a stock based on how its price fluctuates [16].

Over the recent years, machine learning and deep learning have become an effective means to learn from data and produce data-driven analytics. Additionally, neural networks have been found as a robust method to find relations between data in order to make great predictions for future uses. There is a lot of promise at the intersection of the stock market and deep learning, and that is a subject that our project will explore.
Although there are several factors that have and will affect the way a particular stock behaves, attention is drawn as to whether or not one can use data collected on a stock's past performance and use it to predict its future behavior. In this project, our group has implemented a Recurrent Neural Network (RNN) that is built using the Long Short-Term Memory (LSTM) architecture that will take in data of a stock's behavior over the years in order to make a prediction on how the stock will behave in the future. RNNs are a class of artificial neural networks where the nodes within the network are connected along a time sequence. The network utilizes its memory to interpret inputs of various lengths. In

certain instances, storing information and utilizing the RNN memory can be done by another network/architecture; this is where LSTM comes into play. LSTMs are an RNN architecture in which feedback connections are used to process entire sequences of data [11]. A standard LSTM cell, which will be discussed in-depth later, operates by remembering information over variable time intervals. LSTMs have been seen as a reliable tool for making predictions based on time sequence data, which is why our group decided to use it as our primary model for this project.

2. Related Work

Even before the emergence of artificial intelligence, there have been various attempts at predicting the stock market through other means; all to which no success has been found. Now, with the fast growth of machine learning and deep learning, along with how accessible financial information is, the search to find a robust method that can predict financial trends continues. There have been countless attempts to utilize machine learning for financial prediction; our group's purpose to pursue such a project was to explore the "hype" that exists at the intersection of machine learning and the finance world, all while applying the concepts we have learned in class this semester.
The first piece of related work comes from a study from the University of California, San Diego, titled, *A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction* [1]. Simply stated, the Dual-Stage Attention Based RNN (DA-RNN) belongs to a class of models that predict the current value of a time series based on historical values of the respective series plus the historical values of a different time series that is related [1]. From a broad overview, the DA-RNN model includes two LSTM networks with attention mechanisms, components that allow the decoder within the network to address different parts of the sequence at each part of the output generation [2]. Considering the LSTM networks along with their attention mechanism components, the research group at UC-San Diego strongly believed that their proposed model would not only make accurate predictions, but would also be easy to follow. The study goes on to show that the DA-RNN model produces mixed results, as it yielded promising results over time series data on a NASDAQ 100 Stock dataset with model accuracies ranging over 76%, but failed to bring the desired accurate predictions when it came to long term forecasts [3]. Nonetheless, the DA-RNN model developed at UC-San Diego served as a step forward in the path to achieve a reliable predictive model.

Additionally, another study done at the University of Economics and Technology in Ankara, Turkey proposed utilizing a Convolutional Neural Network (CNN) to assist with financial trading. Dr. Omer Berat Sezer, the author of the study, proposes a model using a 2-D CNN based on image processing techniques [4]. The proposed model takes a financial dataset of 15 technical indicators with different parameters. The 15 indicators then produce data for a 15 day period, creating a 15x15 sized 2-D image. Each image is given a label of "Buy, Sell, or Hold" depending on the hills and valleys of the original time series [4]. Using the assigned labels and the visual characteristics of the respective images, the model is able to distinguish when a stock will increase, decrease, or stay put, based on the stock's previous data that it has trained on. This study goes on to show that although the proposed model does do a good job at short term predictions, it has an overwhelming tendency to label large amounts of data as "Hold", the neutral choice, as it encounters trouble when predicting a stock's price due to the volatility of the market. The 2-D CNN model, although coming up short, takes a different approach by using image processing in an attempt to find patterns in financial data. Overall, there are numerous highly advanced efforts underway to build a predictive financial model; building a successful model that yields accurate and repeatable results would definitely be a grand milestone in the world of artificial intelligence.

3. Proposed Method

In this project, our group proposed to build a RNN using the LSTM architecture in order to make predictions on a stock's behavior, given a dataset containing the respective stock's history. Our group decided to focus on LSTMs for several reasons, primarily for their ability to depend on the historical context on inputs, as well as how they manage to loop information from one step to the next [11]. These characteristics allow our model to not only reference stock prices in the days prior, but also in the previous months, as well as in the corresponding month(s) in the years past.
The acquired datasets, which contain several years of data on stock behavior, will be split into training and testing datasets based on their size as well as the time period over which we would like our model to produce predictions for (i.e. a day, month, year). In the training process, our LSTM model uses the loss function shown below. After each step, the loss function is calculated, based on the loss at every time step.

$$L(\hat{y}, y) = \sum_{t=1}^{T_y} L(\hat{y}^{<t>}, y^{<t>})$$

After training, the LSTM model will then make a prediction on stock prices for the entirety of the following year. This prediction is then compared to the testing dataset, which holds the true values of the respective stock over that corresponding year. To compute accuracy, our group decided to use the R^2 score, also known as the coefficient of determination, which is shown below.

$$R^2 = 1 - \frac{SS_{regression}}{SS_{total}}$$

Where SS$_{regression}$ is the sum of squares due to regression and SS$_{total}$ is the total sum of squares. This process was then completed for other remaining stocks we analyzed in this project.

Additionally, we also implemented some smaller simpler machine learning models that also are capable of producing predictions such as: Linear Regression, Support Vector Regression (SVR), and Random Forest (RF). Each of these simpler models were also implemented in Python3, by using the sklearn machine learning library [12].
For the simple Linear Regression model, our group attempted to have the model learn the relationship between price and time, to create a trendline that would serve as a basis to make predictions. In the simple Support Vector Regression model, the objective was to implement a regression method to predict stock prices while maintaining the features of a default SVM, which is to individualize the hyperplane which maximizes the margin. Lastly, the primary objective of the simple Random Forest model was to have a collection of decision trees make a prediction on a stock's price; whether or not that prediction is correct would decide whether a split would take place. In decision trees, a *split* is based on some criteria that helps attain the best possible division of data in a way that optimizes that criteria, in our case, minimizing the cost function. These simple models were implemented for the sole purpose of seeing whether they could show promise for stock prediction.

4. Experiments

The LSTM, Linear Regression (LR), Support Vector Regression (SVR), and Random Forest (RF) models were all tested in a similar manner to see their abilities on making accurate predictions on stock behavior.
Using quandl and stockAI, which are discussed later, we trained each model on a dataset for a respective stock's history. For this project, our group's analysis focuses on predicting the stock's *Adjusted Closing Price*, which is defined as the true value of the stock at the end of the trading day after accounting for any corporate actions [19]. After being trained, we had each model produce a stock price prediction into the future. In some models, this prediction would just be a day into the future; this was the case for the SVR model. Other models such as Random Forest and Linear Regression, predictions were made over 30 days. The LSTM made a prediction over an entire year. These produced predictions were then compared to the real stock prices that were stored in the testing datasets. Predictions were then plotted against the real stock prices and appropriate accuracy scores were calculated for each model.
This process was then repeated again, for each model, for a different stock. For this project we decided to produce stock forecasts for the Tesla (TSLA), Twitter (TWTR), Microsoft (MSFT), and NVDIA (NVDA) stocks. Our group specifically chose tech stocks due to the massive growth that the tech industry has seen over the past decade, as well as their resiliency to periods of economic downfall [5] [6]. Following this experimental procedure, our group was able to analyze each of the models over all applicable stocks, being able to see what models were and were not practical for stock prediction.

4.1. Dataset

Our group gathered all of our datasets holding stock data using the quandl and stockAI python packages. Quandl is a Python package which is highly used in the financial sector, granting access to stock data, which are updated daily [8]. Using quandl in a program results in the package outputting a stock dataset into a Pandas dataframe, ready for immediate use [9]. StockAI, is a similar python package, which utilizes web-scraping to gather data from Yahoo Finance on a daily routine [10]. The datasets gathered using these two packages give our group access to several years worth of stock behavior on every stock we plan to investigate. The datasets themselves offer data on the stock's open, closing, adjusted closing, minimum, and maximum price for a given day. As stated earlier, our focus will be on predicting the Adjusted Closing Price, as it is the best representation of a stock's price for an arbitrary day.

4.2. Software and Hardware

The models implemented in this project were implemented using Jupyter Notebooks running Python 3.7.5 (or later) on the laptops of each group computer. The main Python packages that our group used are PyTorch and sklearn. Our group collected data on stock history using the quandl and stockAI packages, which are discussed above. Additionally, our group enabled GPU utilization on our LSTM model for the case in which future users want to run our model on their device; they can use their respective GPUs if needed.

5. Results and Discussion

As the primary implementation for this project, the LSTM model was trained, tested, and analyzed for Twitter, Microsoft, Tesla, and NVIDIA stocks. The LSTM model was asked to make a prediction for an entire year, that prediction would then be compared to the testing set in order to find a model accuracy. The LSTM testing accuracies over all 4 stocks floated around 54-73%, as shown below in Table

Stock (Symbol)	LSTM Testing Accuracy
Tesla (TSLA)	57.1709 %
NVIDIA (NVDA)	73.5984 %
Twitter (TWTR)	54.0381 %
Microsoft (MSFT)	71.7533 %

Table 1. LSTM testing accuracy scores.

Stock (Symbol)	LR Testing Accuracy
Tesla (TSLA)	94.512 %
NVIDIA (NVDA)	98.060 %
Twitter (TWTR)	81.126 %
Microsoft (MSFT)	98.573 %

Table 2. Linear Regression testing accuracy scores.

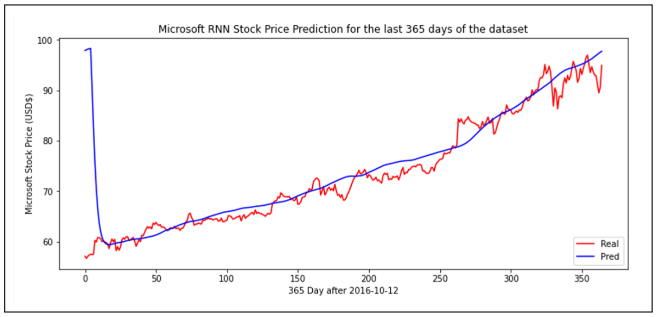


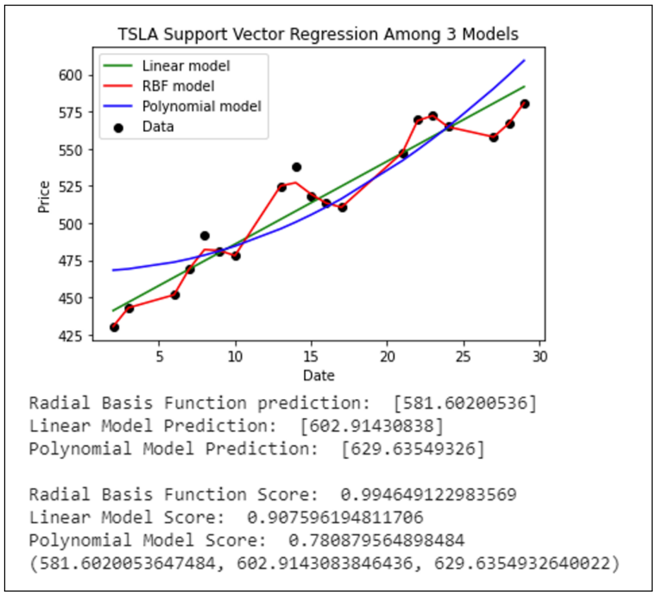
Figure 2. LSTM model predictions plotted against true prices for Microsoft stock for the year of 2017.

1, far lower than what our group anticipated. When plotting the predictions against the stock's true values, as shown in Figure 2, it's clear to say that our LSTM was able to grasp the behavior of how stock changed, but nowhere near as precise as our group would like it to be.

Up next came the Linear Regression (LR) model, which was also analyzed for all four stocks. The LR model was asked to make stock price predictions for the course of a month, to be then compared to a testing dataset in which the true prices were held. Despite implementing regularization, the LR models showed tendencies to overfit, complemented with high accuracy scores, as shown in Table 2. Based on the inaccuracies produced by the model, our group concludes that stock behavior is clearly not linear, and LR is not a model sufficient to predict stocks.

The Support Vector Regression (SVR) model followed, as it was built to predict the stock price for the next possible day. In the model, our group utilized three different kernels: a linear kernel, a polynomial kernel, and a radial basis kernel, also known as the RBF kernel; which is the default kernel in SVR. The function of a kernel is to take data as input and transform it into the required form [15]. Overall, the SVR did a great job of repeatedly predicting the following days stock price, with the RBF and polynomial kernels producing the best results, as shown with the Tesla stock in Figure 3. On the contrary, the linear kernel continually produced accuracy scores of 40% or less, adding to our group's findings with the LR model, that stock behavior is obviously not a linear relationship.

The Random Forest (RF) model was the last to be tested as a part of our project. Unfortunately, our RF model was



```
Radial Basis Function prediction:  [581.60200536]
Linear Model Prediction:  [602.91430838]
Polynomial Model Prediction:  [629.63549326]

Radial Basis Function Score:  0.994649122983569
Linear Model Score:  0.907596194811706
Polynomial Model Score:  0.780879564898484
(581.6020053647484, 602.9143083846436, 629.6354932640022)
```

Figure 3. Support Vector Regression Model Results for Tesla stock, with each kernel's price prediction below the plot.

not as effective as we hoped, with testing accuracies between 45-50%, as shown in Table 3. Additionally, we also implemented our RF model to track and predict returns that one would make if one were to invest in a stock. In the financial world, a return is a profit on an investment. As shown in Figure 3, the RF model was able to grasp how big a possible return would be for a certain stock, as well as how often a potential investor could see such such a profit. Although the RF model produced feeble predictions, it could serve as a tool to see whether or not a stock is worth investing in.

Through these several attempts of using machine learning to predict the stock market, we realized how complex such a task really is. In response to this, our group asked the question: "Is using machine learning to predict the stock market a feasible task?". After researching this topic for quite some time, our group came up with two profound reasons that explain our troubles.

Firstly, the stock market is self-correcting. Unlike most prediction tasks, the stock market is constructed to correct themselves for any easy predictability that could lead to a high level of profitability. By thinking in line with the *Efficient Market Hypothesis*, which states that asset prices re-

Stock (Symbol)	RF Testing Accuracy
Tesla (TSLA)	48.1481 %
NVIDIA (NVDA)	49.3775 %
Twitter (TWTR)	46.9090 %
Microsoft (MSFT)	50.0991 %

Table 3. Random Forest testing accuracy scores.

flect all available information, we can assume that if the stock market was genuinely predictable, investors would have already exploited such patterns to a point in which predicting the stock market now becomes impossible [7]. In simpler terms, "if it was easy, everyone would be doing it".
Secondly, the stock market is incredibly volatile. After acknowledging the complex, interconnected nature of the stock market, one must consider how the rules and regulations that govern the financial world change quite frequently. Each time there is an updated regulation, rule, report, etc., the mechanisms of the market change. For example, imagine training a deep neural-network (DNN) to predict the stock market, based on stock history, as well as the rules of the market. However, in this scenario, the rules of the market change every 10,000 iterations or so; the DNN you've trained is practically useless, as it will always need to learn "more" in order to stay up to date. Additionally, the amount of influence that worldly-affairs have on the market makes stock prediction a further daunting task. Currently, it's May 2020 and our world is facing a pandemic. Several economies around the world have been impacted and have passed new regulations and stimulus packages in response to the damage caused. Although these new guidelines have been passed in the market, these are also new constraints that our machine learning model needs to become familiar with. How, and with what data will we train our models on in order to stay up to date with the market? This essentially shows why using a machine learning model for stock prediction is practically infeasible.

6. Conclusions

Overall, when looking back at the results that our group attained over our several experiments, it's clear to conclude that we had mixed results. Although all of our models were able to produce predictions, these forecasts were not accurate enough to be seen as a reliable prediction, as most test accuracies for all models fell around 60%. Prior to conducting the project, our group set a goal to obtain at least 85% testing accuracy for our LSTM model; that goal was also not achieved. To some, it may seem as if this project was "unsuccessful", however, our group is quite pleased with the findings we attained; and as discussed above, some of which go well beyond machine learning.

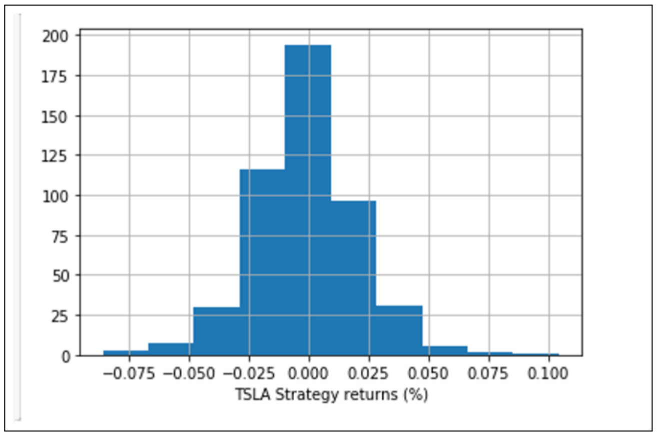


Figure 4. Using the Random Forest, we were able to see what how large of a return we would get on our investment, and how likely such a return would occur, based on how the stock has performed in the past.

In the current world of AI, there are several advanced attempts in the works to create a model that predicts the stock market; some of which are far more sophisticated than what our group implemented in this project. When even considering the works of experienced data scientists and machine learning engineers, there has yet to be a single model that predicts financial trends with efficacious results. We can conclude our efforts by stating that it is incredibly difficult to model and predict something like the stock market, a non-stationary system with an absurd amount of entropy.
Nonetheless, this project allowed our group to explore an area of personal interest, while applying several fundamental concepts that we have learned in class throughout this semester. Additionally, we were able to learn about the severe complexity of the stock market, and how we currently may not have the tools to properly predict the market in a reliable way.

7. Acknowledgements

As most of the deep learning and machine learning concepts applied in this project derive from the concepts we learned in class this semester, there were several business and finance related concepts that played a major role in the analysis of this project. Considering the fact that no member of our group has a background in finance or business, it was integral that we educated ourselves on the fundamentals of the stock market in order to make sure our conclusions were accurate. Our group would like to acknowledge *Investopedia, Zacks,* and *NerdWallet* as they were reliable, informative sources that allowed us to learn about how the stock market actually works [17] [18] [19].

8. Contributions

Overall, this project was sizable and detailed; our group would not have been able to complete this project if it were not for proper project management. All three members of our group played a big role in contributing to the creation of this project while also providing assistance to one another when any member needed help.
Guk Kim took the role of the evaluation process of the entire project. As shown in the Methods and Experiment sections, several models were implemented for this project and these models were tested on datasets for a number of stocks. Guk's main objective was not only to test and run these models, but to also ensure that the model predictions and outputs were a clear indicator of a model's efficacy/inefficacy. Additionally, Guk made the proper adjustments for a model to account for error and noise, when needed.
Arhum Zafar was primarily responsible for building the models, as well as modeling the entire project. Arhum's efforts were spent on ensuring that the data being collected was accurate, searching for simple machine models that would be compared to the RNN model, as well as putting these models to play and concluding what models were and were not effective for stock prediction. Additionally, Arhum also conducted the analysis on why the stock market may not be predictable using current machine learning methods, as discussed in the Conclusions section.
Desmond Fung was also responsible for building the models for the project. Desmond also took the lead in the debugging aspect; there were several models implemented in this project and our group faced several errors throughout this project. Desmond took charge in fixing our initial mistakes and creating models that were able to produce repeatable results. Overall, this project was completed as a result of proper project management, communication, and equal effort among all group members.

9. Code

The code and associated notebooks implemented for the project can be accessed at the following GitHub repository: `https://github.com/arhumzafar/453_group4`

References

[1] A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction
`https://chandlerzuo.github.io/blog/2017/11/darnn`

[2] Attention Mechanism in Neural Network
`https://hackernoon.com/attention-mechanism`

[3] A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction
`https://arxiv.org/pdf/1704.02971.pdf`

[4] Algorithmic Financial Trading with Deep CNNs
`https://www.researchgate.net/publication/324`

[5] Big Tech's Resilience to Coronavirus Downturn Faces First Test
`https://www.wsj.com/articles/big-techs-resilience`

[6] Types of Stocks That Will Survive Economic Collapse
`https://www.thebalance.com/types-of-stocks`

[7] Efficient Market Hypothesis (EMH)
`https://www.investopedia.com/terms/e`

[8] Quandl
`https://www.quandl.com/`

[9] Quandl Documentation
`https://docs.quandl.com/`

[10] stockAI
`https://pypi.org/project/stockai/`

[11] The Magic of LSTM Neural Networks
`https://medium.com/datathings/the-magic-of-lstm`

[12] scikit-learn
`https://scikit-learn.org/stable/`

[13] Regression Analysis: How Do I Interpret R-squared?
`https://blog.minitab.com/blog/adventures-in-stat`

[14] PyTorch RNN Documentation
`https://pytorch.org/docs/stable/nn.htmlrecurrent`

[15] Kernel Functions-Introduction to SVM Kernel Examples
`https://data-flair.training/blogs/svm-kernel`

[16] Commodity: Linear Regression Lines
`https://commodity.com/technical-analysis/lin`

[17] What Is the Stock Market and How Does It Work?
`https://www.nerdwallet.com/blog/investing/what`

[18] Impact of Techn Stock
`https://finance.zacks.com/impact-technology-stock`

[19] Investopedia: Adjusted Closing Price
`https://www.investopedia.com/terms/a/adjustedclosing`