Research of RNN Models Performance on New York Stock

Zhentong Fan[1], Yang Wang[2a]*, Cheng Ma[3]

[1]*Valley Christian High School, San Jose, California, United States, 95111*
[2]*Monash University, Melbourne, 3168, Australia*
[3]*Nanjing Foreign Language School, Nanjing, 210008, China*
[a]*ywan0440@gmail.com*

ABSTRACT
Stock market forecasts have become a popular topic for researchers and investors. Stock market forecasting methods range from traditional analysis based on statistics to machine learning models such as decision trees, SVM, and neural networks. In this project, we decided to use Recurrent Neural Network (RNN) models to make stock price forecasts due to the time series nature of stock prices. From simple RNN models to more complex models such as the GRU and LSTM, three different RNN models have been used to compare error values and the performance of each. Based on the results, we found that the LSTM was taking a longer time to train but better performance compared to the other two simple models. This RNN stock forecast study lays the foundation for the future use of RNN models in economic markets.

Keywords: *Artificial intelligence, Recurrent neural networks, Stock market prediction*

1.INTRODUCTION

We plan to use three different Recurrent Neural Network algorithms to analyze the New York Stock Exchange, which is one of the largest stock exchanges all over the world. The stock market is affected by various factors including ones that are extremely hard to notice [1]. At the same time, we believe that the post-pandemic situation is very unstable and unpredictable. Therefore, it is necessary to develop machine learning algorithms to predict future stock market trends to maximize profit and reduce risks. The results between different algorithms are compared and recommendations are given based on these results.

The stock price is an important criterion for investors' decision-making strategy [2]. Moreover, stock prices enable company executives to determine the direction in which brand value will increase or decrease. However, stock prices are affected by many factors such as economic crisis, public relations, internal management, and so on. [3]. In addition, the stock price fell sharply after the plague. There is an enormous gap between the market's demand for reliable forecasting tools and the currently existing research for it.

Since the advent of the Recurrent Neural Network in the last century, it has been gradually modified and developed into multiple variants. It has been proved that these algorithms have high recognition and prediction accuracy especially for Natural Language Processing (NLP) [4]. In this article, we use simple Recurrent Neural Networks, Long Short-term Memory (LSTM), and Gated Recurrent Units (GRU) to compare stock price predictions.

The data is based on the historical prices of the S&P 500 index. As a manufacturer, 3M has been steadily increasing its stock value for many years [5].

2.THEORY

2.1 RNN

A time series is a continuous and consistent sequence of data points [6]. Time series, like stock prices, can track the movement of passive quantities. When investing, it is vital to analyze the initial stage and historical data in order to understand future business trends in the eyes of investors [7]. We use Recurrent Neural Network models that allow us to process sequential data in time. The RNN model is capable of memorizing the historical information by recalling the prior data when calculating

for the future data. The RNN model has three parameters: the input vector, the weight vector, and the output vector. The input vector is the share price of our algorithm. This will calculate and generate the hidden weight vector. The output depends on the weight vector and the input vector.

2.2 Simple RNN

Simple RNN is based on a basic Recurrent Neural Network algorithm that utilizes past information to predict future trends. The information from the previous time step returns to the next step in the hidden layer [8]. We use the hyperbolic tangent (tanh) as the activation function in the simple RNN model. The activation function determines the output of the node and sets the input packet from -1 to 1. The advantage of using tanh is to map input. In the next step, we will use this function to calculate the weight scale. The weighted and updated matrix can be used to learn the input. The matrices will use backpropagation to tune themselves during training.

2.3 GRU

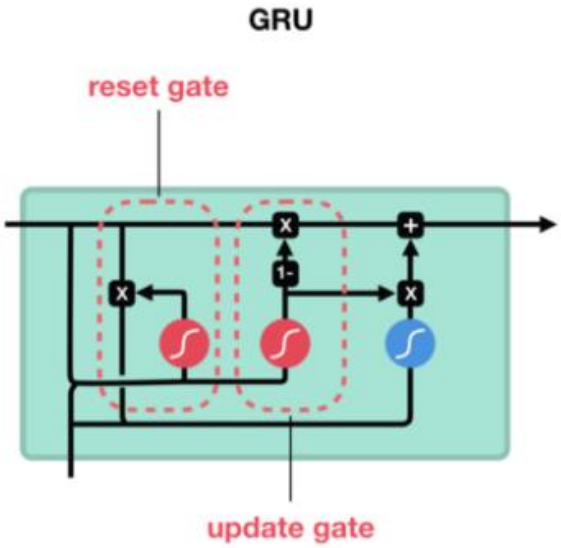


Figure 1 GRU Retrieved from [9] https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

The Gated Recurrent Unit, also called the GRU, is a RNN model that is implemented widely nowadays. To improve the performance of simple RNN algorithms, the GRU provides a "gate" aimed to describe and recall long-term memory. Each GRU cell takes the value of the new entry's hidden layer for both the previous and the current time step. As shown in Figure 1, there are two separate parameters being used to process the two input data. After applying the sigmoid function, we can have two different gates: the reset gate determines how much information from the last hidden layer should be entered into the new data to be processed, and the update gate determines how much information can be retrieved from the last layer. Once we have the two gates, we first get a new input and a common data containing the last part of the hidden layer (the rate is determined by the reset gate). Then, we use the tanh function to compress the data to prevent it from exceeding the threshold. Finally, we use the update gate to select a particular piece of information from the last hidden layer and add it to the new hidden layer. For balance, we also use the "forgotten door" to determine how much new information needs to be selected to enter a new hidden layer. Realistically, a forgotten door is always mathematically equivalent to "one minus update gate", and the overall ratio will always be the same.

2.4 LSTM

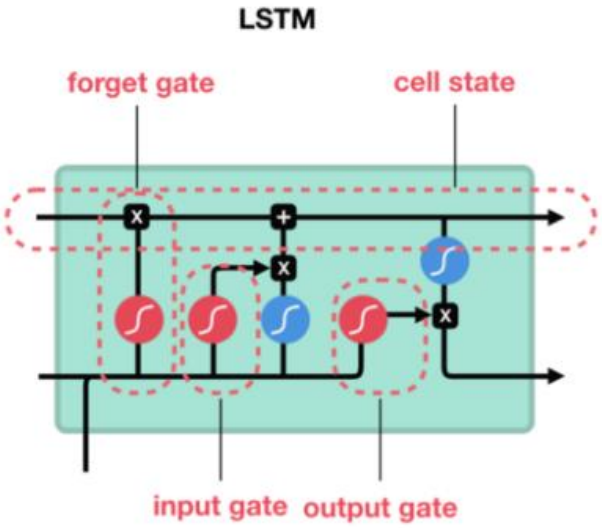
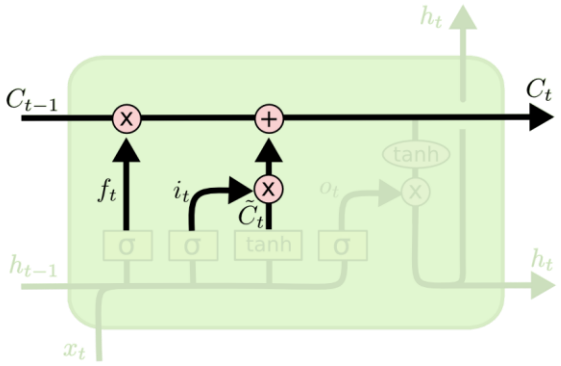


Figure 2 LSTM Retrieved from [9] https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

We use LSTM to solve the gradient vanishing problem in simple RNN. The LSTM model is more meticulous than a simple RNN model. The first major difference of LSTM is the perception of the state of the cell [9]. As shown in Figure 2, each step shows the state of the cell to update the output based on the previous calculation. Introducing the concept of cell state allows the gradient to be calculated freely over time, making LSTM more resistant to gradient loss [10]. The first step in updating a cell state begins with deciding what information to store from the previous cell state. The prediction vector from the last step and the new information from this step are related to the forgotten gate, and the output will be the weight vector, which will then be used to update the state of cells. The next step is to decide what needs to be done with the new

information, putting the same paired vector into two different cells, which determines what the vector-based prediction is and how many vectors we should spend to update the state of our cells. Add all the weights and information needed to update the cell, as shown in Figure 3, in particular, multiply the forgotten weight by the previous cell state, add it to the result of the input weight, and add the calculations obtained at this stage. [11].



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 3 LSTM Retrieved from [6] https://www.quora.com/How-is-LSTM-different-from-RNN-In-a-layman-explanation

After updating the state of the cells, the vector of the same sequence is returned to the generated output neuron to determine which part of the state of the cell we are using to make the next iteration hypothesis.

3.APPLICATION

3.1 Testing & training data

Neural networks learn from existing data. Recurrent neural networks use supervised learning to classify sequential sequences [12]. The output tag should be included in each step for the prediction. For example, LSTM outperforms other RNNs in various controlled sequences. The data is divided into three different sections: the training dataset, the validation dataset, and the testing dataset.

The training dataset is a dataset used in the model for training purposes. They were mined and modeled during training. The so-called validation dataset or development dataset is a dataset used to match bridge parameters. The training dataset and the validation dataset follow the same probability distribution. By using the validation dataset in the model, the model can avoid over-matching problems. The statistical model is appropriate when it exactly matches the training dataset. On the other hand, an over-matched model makes it impossible to predict the actual data. Therefore, it is important to use the validation dataset for model building and avoid unnecessary matching [13]. Experimental data is a standalone dataset, unlike a training dataset. In theory, the experimental database uses mathematical performance measures such as MSE to evaluate the performance of the model.

We created a database based on original data and checked the database. The data is well organized with no unlimited data or blank data. As a result of the 2008 financial crisis, 3M's share price, which was launched in September 2008, reached a record low in the global financial crisis. On the other hand, the global coronavirus pandemic is having a huge impact on global stock markets. Then, we decided to use the daily stock price as a database for training and testing data from April 1, 2010, to December 30, 2016. The first 80% of the data is shared in the training database and the last 20% in the experimental database.

3.2 Hyperparameter

Hyperparameters aim to control the learning process and the design structure. For instance, the number of layers is used to control the complexity of a design [14]. We must decide on the appropriate number of layers so as not to overload. In this paper, we decided to use one layer in all three models. Step by step, our model is used to control how far we go from the error gradient. The speed of learning is too slow, and it takes a very long time for the error to reach the local minimum. On the other hand, a very high level of learning rate will bring the error closer to the local minimum, and we will get a lower limit that is very far from the actual local limit. In order to reduce the error to an acceptable level within a reasonable time, we decided to use 0.05 as the training average. To avoid overfitting, the dropout rate was set at 0.2. This means that at each step, 20% of the neural network cells are accidentally discarded during training.

3.3 Comparison

The Mean Square Error, called the MSE, is a measure of the risk function. This is the abstract demonstration of errors. In our model, MSE is calculated by experimenting with data and using real data to manage accuracy.

As shown in Figure 4, it is clear to observe that a simple RNN has a relatively high training loss compared to the GRU and LSTM. The difference between the training loss in GRU and the loss in LSTM is not very big. The validation loss of each neural network is

unstable; Of the three methods, the greatest losses occur when using a simple RNN. On average, it is clear that a simple RNN has the largest validation loss. LSTM has less validation loss than GRU.

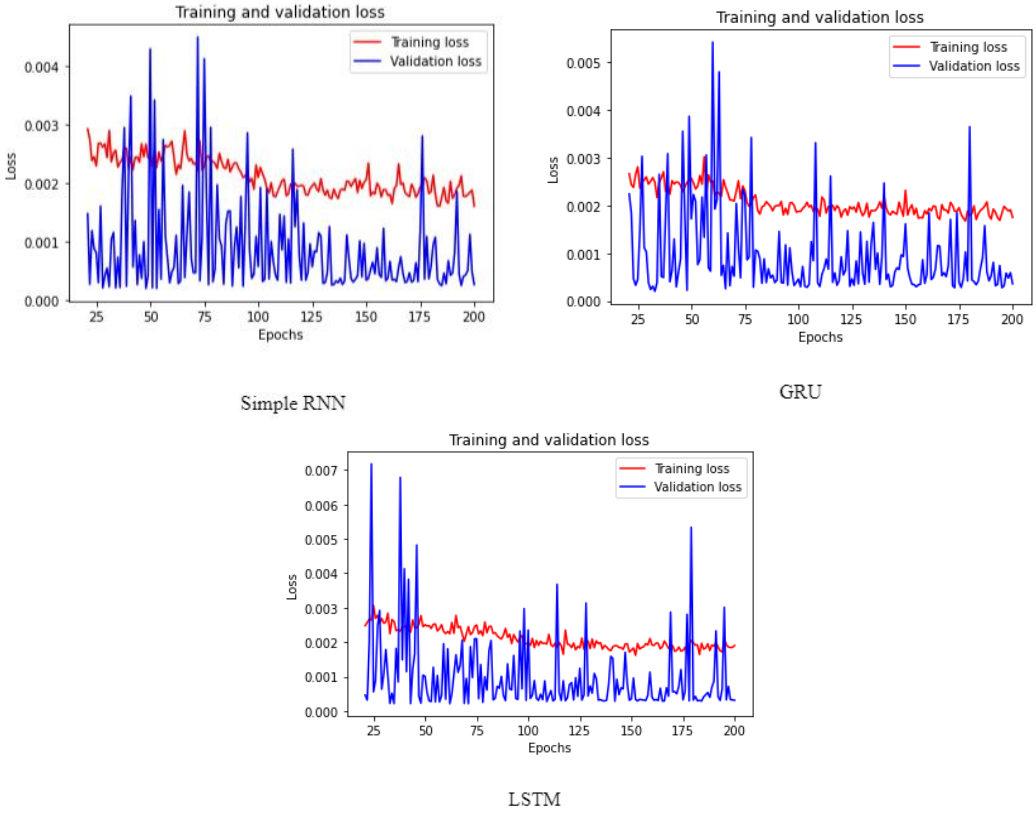


Figure 4 Comparison

3.4 Result

Based on the above comparison, it can be concluded that the accuracy of GRU and the accuracy of LSTM are close to each other. Both are slightly more accurate than simple RNNs. Also, a simple RNN takes the least amount of time to produce results, and a GRU takes almost twice as long as a regular RNN. LSTM spends 30% more time than GRU.

The conclusion that LSTM and GRU perform better than normal RNN is evident in the end times. However, a simple RNN may be related to the simplest mathematical structure. In theory, if we had large enough data and time, the LSTM could perform better. The difference between LSTM and GRU in our project is not abstract, but it can be assumed that we will see the difference if we have more data.

In conclusion, if consumers want to predict stock prices over a longer period, we provide a simple RNN for forecasting stock prices. The users are advised to use GRU or LSTM if they want to achieve the most accurate results in a limited time and era.

3.5 Verification

We changed the number of echos and hidden layers and tested them with different stock. The stock we look at is insurance broker Willis Tower Watson. The results found at 3M are still valid for this new model, as shown in Figure 5. However, all three models have excellent long-term accuracy.

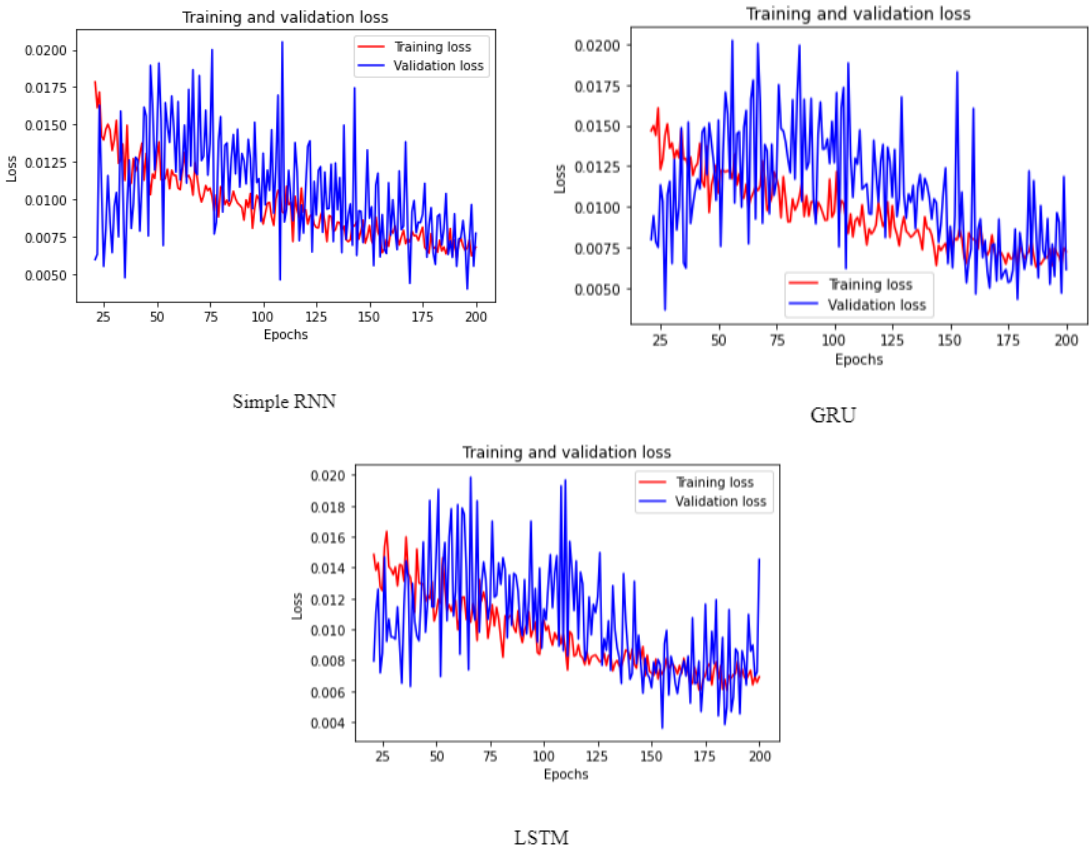


Figure 5 Verification of WTW

4.PREDICTION

Orange - Train stock price (training data)

Blue - Historical stock price (actual data)

Green - Stock price prediction (test data)

X-axis: days since 2010 - 01 - 04

Y-axis: Close price of the day

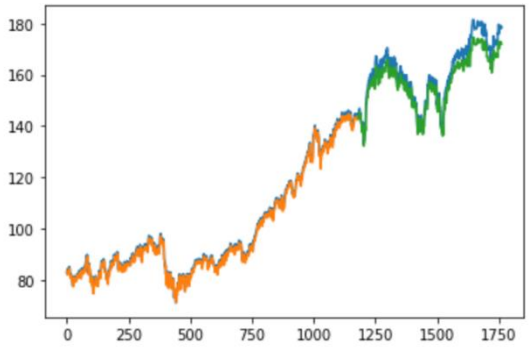


Figure 6 LSTM Prediction of 3M

Figure 6 shows the training data, test data, and actual data. The green line represents the experimental data generated by our LSTM model, while the blue line represents the actual data obtained from the stock market. According to Section 3.3, the LSTM has the smallest MSE. We only show LSTM assumptions in Figure 6. In addition, the results of all three algorithms are very similar.

5.CONCLUSION

In general, three-link models are all suitable for making predictions and modeling accuracy. Simple RNN models consume the shortest time and the LSTM models require the longest time. All three models can be well implemented if time is unlimited, but LSTM achieves better results faster due to its algorithm design. Depending on previous sections, RNN models are a possible solution from hedge managers, schools, and governments to forecast the market to raise futures contracts.

Joerg Sixt <joerg.sixt@springernature.com>

[1] Kearns, M. (1996). A bound on the error of cross validation using the approximation and estimation rates, with consequences for the training-test split. Advances in Neural Information Processing Systems, 183-189.

[2] Zhang, J., & Man, K. F. (1998, October). Time series prediction using RNN in multi-dimension embedding phase space. In SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218) (Vol. 2, pp. 1868-1873). IEEE.

[3] Olah, C. (2015, August 27). Understanding LSTM Networks. Understanding LSTM Networks -- colah's blog. Retrieved from https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[4] Fu, R., Zhang, Z., & Li, L. (2016, November). Using LSTM and GRU neural network methods for traffic flow prediction. In 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC) (pp. 324-328). IEEE.

[5] Yuniningsih, Y., Widodo, S., & Wajdi, M. B. N. (2017). An analysis of decision making in the stock investment. Economic: Journal of Economic and Islamic Law, 8(2), 122-128.

[6] Jiao, J. (2018). How is LSTM different from RNN? In a layman explanation. Quora. Retrieved from https://www.quora.com/How-is-LSTM-different-from-RNN-In-a-layman-explanation.

[7] Islam, M. S., Salam, M. A., & Hasan, M. M. (2015). Factors affecting the stock price movement: a case study on dhaka stock exchange. International Journal of Business and Management, 10(10), 253.

[8] Prabowo, Y. D., Warnars, H. L. H. S., Budiharto, W., Kistijantoro, A. I., & Heryadi, Y. (2018, September). Lstm and simple rnn comparison in the problem of sequence to sequence on conversation data using bahasa indonesia. In 2018 Indonesian Association for Pattern Recognition International Conference (INAPR) (pp. 51-56). IEEE.

[9] Phi, M. (2020, June 28). Illustrated Guide to LSTM's and GRU's: A step by step explanation. Medium. Retrieved from https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21.

[10] Alldredge, M., Johnson, C., Stoltzfus, J., & Vicere, A. (2003). Leadership development at 3M: New process, new techniques, new growth. People and Strategy, 26(3), 45.

[11] Ding, X., Zhang, Y., Liu, T., & Duan, J. (2015, June). Deep learning for event-driven stock prediction. In Twenty-fourth international joint conference on artificial intelligence.

[12] Sobel, R. (2000). The Big Board: a history of the New York stock market. Beard Books. New York.

[13] Claesen, M., & De Moor, B. (2015). Hyperparameter search in machine learning. arXiv preprint arXiv:1502.02127.

[14] Kozma, R., Alippi, C., Choe, Y., & Morabito, F. C. (Eds.). (2018). Artificial Intelligence in the Age of Neural networks and Brain computing. Academic Press. Boston.