

# Recurrent Neural Network (Time Series - Stock)

Haeun Kim

School of Natural Sciences and Mathematics  
University of Texas at Dallas Richardson, United\_States  
hxx210027@utdallas.edu

**Abstract**— In this report, it contains the prediction of the real stock price by using the Recurrent Neural Network(RNN) with Long Short Term Memory(LSTM). The purpose of this project is to find the optimal parameters (e.g. iterations, number of layers, learning rate) in order to minimize the errors and loss. This data consists of 248 instances and 7 attributes, but I focused on the Open. The Open's optimized parameters are "activation: 'tanh', learning rate: '0.01', and iteration: 1000."

**Keyword**—Recurrent Neural Network, Stock Market, Machine Learning, Deep Learning, LSTM, Prediction, Loss, Activation Functions ('sigmoid', 'relu', 'tanh')

## I. INTRODUCTION

Have you ever experience of buying stock or selling stocks before? The stock market is the system in which the individual or groups invest some money in a company with Blue chip stock or Growth potential. After that, the individuals or groups receive some proportionate profit or lose their money as much as proportionate loss. When you own stocks of certain company, you are essentially owning a portion of the shares issued by the company. As a shareholder, you have the right to claim profits from the company in the form of dividends. However, for the majority of people, investing in stocks is a means of making money through the buying and selling of stocks, and profiting from the difference between the purchase and sale prices. It would be good to predict the price of stock to gain the profits, but predicting the value of stocks is very difficult due to the variability of stock prices, this is because the value of stocks can fluctuate every hour or every second due to various

factors such as exchange rates and the company's situation, making it challenging to predict. However, while predicting accurate data is difficult, it is said that using the algorithm techniques of machine learning and deep learning, specifically Recurrent Neural Network, can enable us to predict stocks to some extent.

## II. THEORY OF RECURRENT NEURAL NETWORK

### A. Time Series

First, before knowing the algorithm or starting to predict, we have to understand the type of data first. Time series data means that data are arranged at regular time intervals. The example of time series analysis are stock price, waterfall measurement, electrocardiogram, and so on. In order to know the future data, we can identify characteristics of the data by using the past and current data, and this is called time series analysis. Time series prediction means creating a model of the characteristics based on time series analysis data, and using this model to predict future data.

### B. Recurrent Neural Network

RNN(Recurrent Neural Network) is the one of the neural network model that the previous step's output becomes the new input for the current step, so it influence the current step and it goes on and on. RNN is used widely for language translation, natural language processing such as speech recognition, and image captioning. It is also used to predict the time series data such as stock market, weather (forecasting), hurricane intensity, etc. To predict the future with the current dataset, this RNN is the suitable algorithm. So, I'll try to predict the Samsung Electronic Stock price by using this algorithm. One of the important feature of RNN is that the RNN has the hidden state.

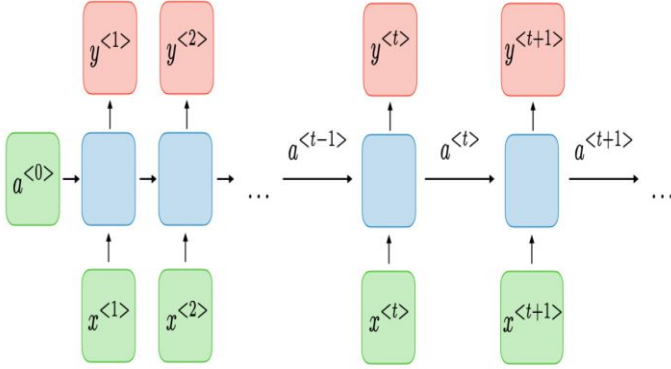
$$a_t = f(a_{t-1}, x_t) ; y_t = h(a_t)$$

The above equation is the basic of RNN. It consists of input ( $x_t$ ), state ( $a_t$ ), and output ( $y_t$ ). After applying some activations, it becomes (for example, using sigmoid)

$$a_t = \text{sigmoid}(W_{aa}a_{t-1}, W_{xa}x_t)$$

where  $W_{aa}$  (which is the weight at the recurrent) and  $W_{xa}$  (which is the weight at the input) are the weight. After calculating the state, we could get the output

$$y_t = W_{ay}a_t$$



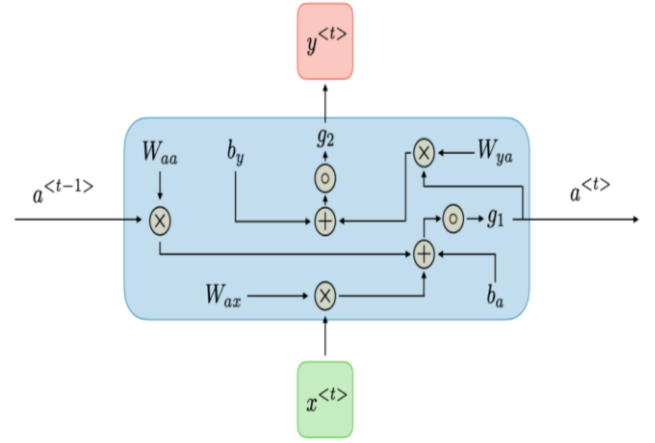
The above figure is the picture of the recurrent neuron network process.  $x^{<1>}$ ,  $x^{<2>}$ , ...,  $x^{<t>}$ , those are the input of data, and the blue box is the hidden layers which occurs recurrent training, and  $y^{<1>}$ ,  $y^{<2>}$ , ...,  $y^{<t>}$  are the output after the hidden state, and  $a^{<0>}$ ,  $a^{<1>}$ , ...,  $a^{<t>}$  are the result of hidden layers, called hidden state.

However, RNN has drawbacks: One is exploding gradient and the other is vanishing gradient. For the exploding gradient, it happens when weight greater than '1', so the term goes to infinity or the loss goes to infinity. The solution for the exploding gradient is 'Gradient Clipping'. For the vanishing gradient it happens when weight less than '1', so the term goes to 'zero', but it is difficult to figure out in the middle of training. The solution for the vanishing gradient is 'Gated RNNs' such as 'LSTM' or 'GRU'

$$1. \text{ Vanishing gradient } \left\| \frac{\partial h_i}{\partial h_{i-1}} \right\|_2 < 1$$

$$2. \text{ Exploding gradient } \left\| \frac{\partial h_i}{\partial h_{i-1}} \right\|_2 > 1$$

So, to avoid the Vanishing gradient, I'll use the LSTM for the Samsung Electronics to overcome RNN's drawback. Long-Short Term Memory (LSTM) has 3 gates (Forget gate, Input gate, and Output gate), and 2 state (Cell state and Hidden state). LSTM is useful in order to train long input sequence and is widely used model in various fields.



### III. APPLY THE SAMSUNG ELECTRONIC DATA TO RNN

To briefly introduce why I chose the Samsung Electronics company to predict the stock price, this is one of the big companies, which makes some electronic goods like TV or Samsung Galaxy phone, in South Korea. So, most Korean are shareholders of this company, but those who have invested in this company for the purpose of investing lost their money because the stock price has been volatile. So, if I train this stock by using the algorithm, it might be fun to predict because it is dynamic, not monotone, and I could get some accuracy from this model.

This data consists of a total of 248 data (from 2022-04-07 to 2023-04-06), and attributes consist of 'Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', and 'Volume'. I want to focus on the value of 'Open' and 'Close' because the stock market starts with its initial price and it closes with its final price of the day. And I need to do normalization the dataset because this price is indicated in the Korean currency because it is the Korean company.

I did this algorithm process twice: Open and Close respectively. To start this RNN training, I tried to look over the dataset. When we look into the details, there are the differences in aspect of the shape and normalized values. However, I could notice that there are no big differences between Open and Close, So I decided to focus only on the Open dataset.

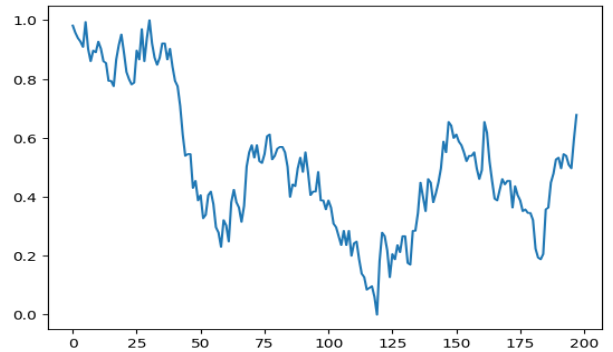


Figure 1 Samsung Electronics Open Values

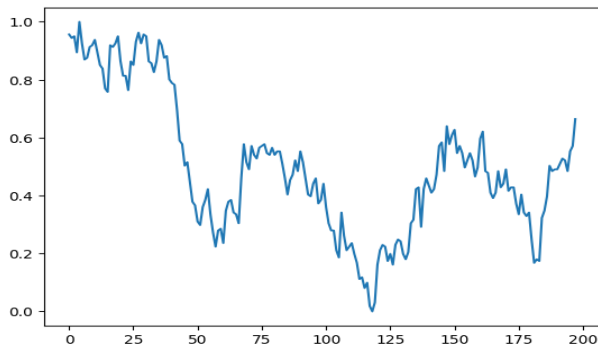
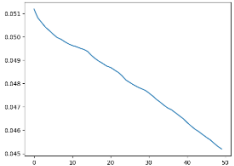
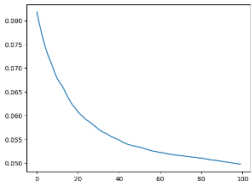
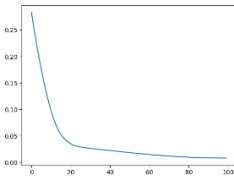
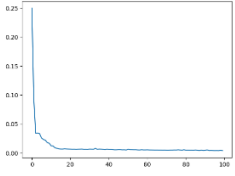
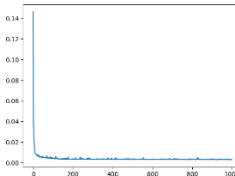
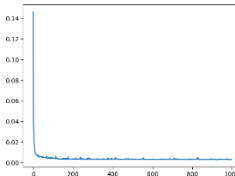


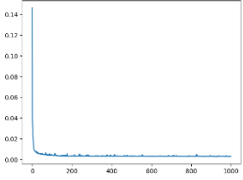
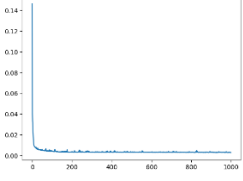
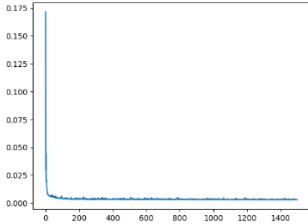
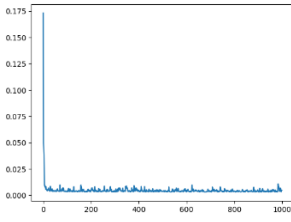
Figure 2 Samsung Electronics Close Values

#### IV. RESULT AND ANALYSIS

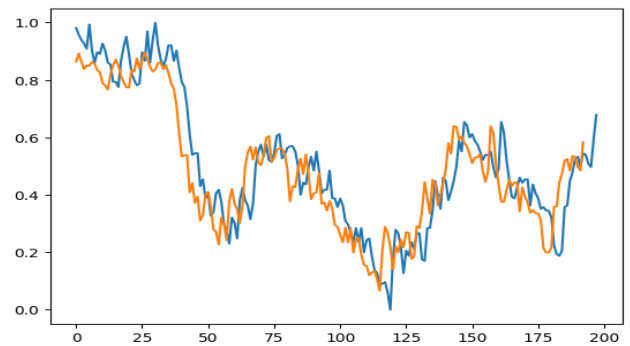
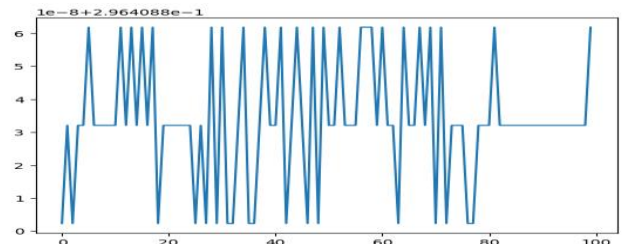
And I tried to do modeling by using RNN and LSTM. The below table is log table to find the optimized parameters.

Experiment Number	Parameters Chosen	Results <OPEN>
1	Size: 5 Activation 'sigmoid' Learning Rate: 0.001 Iteration: 50 Split Size: 0.8	Loss:  Training MSE: 0.04490 Test MSE: 0.01441
2	Size: 5 Activation 'sigmoid' Learning Rate: 0.001 Iteration: 100 Split Size: 0.8	Loss:  Training MSE: 0.04964 Test MSE: 0.01448
3	Size: 5 Activation 'tanh' Learning Rate:	Loss: 

	0.001 Iteration: 100 Split Size: 0.8	Training MSE: 0.00754 Test MSE: 0.00254
4	Size: 5 Activation 'tanh' Learning Rate: 0.01 Iteration: 100 Split Size: 0.8	Loss:  Training MSE: 0.00429 Test MSE: 0.00275
5	Size: 5 Activation 'tanh' Learning Rate: 0.01 Iteration: 1000 Split Size: 0.8	Loss:  Training MSE: 0.00307 Test MSE: 0.00207
6	Size: 3 Activation 'tanh' Learning Rate: 0.01 Iteration: 1000 Split Size: 0.8	Loss:  Training MSE: 0.00308 Test MSE: 0.00241
7	Size: 8 Activation 'tanh'	Loss:

	Learning Rate: 0.01 Iteration: 1000 Split Size: 0.8	 Training MSE: 0.00266 Test MSE: 0.00276
8	Size: 5 Activation 'tanh' Learning Rate: 0.01 Iteration: 1000 Split Size: 0.8	Loss:  Training MSE: 0.00302 Test MSE: 0.00233
9	Size: 5 Activation 'tanh' Learning Rate: 0.01 Iteration: 1500 Split Size: 0.8	Loss:  Training MSE: 0.00307 Test MSE: 0.00218
10	Size: 5 Activation 'tanh' Learning Rate: 0.01 Iteration: 1500 Split Size: 0.8	Loss:  Training MSE: 0.00522 Test MSE: 0.00860

The result of the optimized parameter for Open data set is activation: 'tanh', learning rate: '0.01', and iteration: 1000. For the activation, 'tanh' is the most efficient, the sigmoid is also somewhat efficient, but not much as the 'tanh' activation function, and the 'relu' is the worst one. The plot of Loss by set the 'relu' is below. For the learning rate '0.01' has the less MSEs than the '0.001'. I tried to do '0.1' because it is less than '0.01' and I thought the MSEs might decrease, but unlike my thought, the MSEs Increases. For the iteration, as the iteration increases up to 1000, the MSEs decreases, but when I did the modeling with 1500, there is not a differences with 1000. So, I thought it converged when the iteration is 1000.



The above graph is the differences between real data and the prediction data. The blue one is the real data value, and the orange one is predicted value. I can see the result is somewhat similar. If the predicted data moves right side a little bit, the result can be similar. So, I'm satisfied with the prediction and optimized parameters through this training and modeling.

## V. CONCLUSION AND FUTURE WORK

In conclusion, I could get the optimization parameters with activation: 'tanh', learning rate: '0.01', and iteration: 1000.

*Future work:* For the future work, I would like to build a model which is possible to predict today's stock price. It might need to be modeled precisely and modified. This is because it can help me to decide to sell or buy the stock on that day.

## REFERENCES

- [1] "Introduction to Recurrent Neural Network." GeeksforGeeks, GeeksforGeeks, 25 Nov. 2022, <https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/>.
- [2] "What Are Recurrent Neural Networks?" IBM, <https://www.ibm.com/topics/recurrent-neural-networks>.
- [3] "Recurrent Neural Networks Cheatsheet Star." CS 230 - Recurrent Neural Networks Cheatsheet, <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks>.
- [4] Biswal, Avijeet. "Recurrent Neural Network (RNN) Tutorial: Types and Examples [Updated]: Simplilearn." Simplilearn.com, Simplilearn, 10 Apr. 2023, <https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn>.