Recurrent Neural Networks for Stock Market Prediction

Yi Zheng
Computer science
University of Texas at Dallas
Richardson,US
yxz180009@utdallas.edu

Qiang Yao
Computer science
University of Texas at Dallas
Richardson,US
qxy180009@utdallas.edu

Feifei Guo
Computer science
University of Texas at Dallas
Richardson,US
fxg180009@utdallas.edu

Xinyue Yu
Computer science
University of Texas at Dallas
Richardson,US
xxy170004@utdallas.edu

***Abstract*—Stock, as one of the most common types of investment nowadays, draws the attraction of many researchers due to its high risk. Analyzing and predicting the future trends of certain stock features given historical data become a hot topic in economic sphere. In our project, we decide to apply Long Short-Term Memory (LSTM) model, a popular RNN method for processing time sequential data, to predict the highest stock price in the future based on previous data sets including factors such as lowest price, highest price, opening price, closing price, trading volume, increase or decrease. LSTM has the advantages over traditional neural network model because its output for each time step is not only decided by the input at the current time instant, but also the states of previous time steps, which means LSTM can memorize previous input information and system state information, and figure out new outputs based on current network state[1].**

Keywords— lstm, stock, predict, neural network

I. Introduction and Background work

Predicting the stock market is of great value nowadays because stock is an important way of investment. It is valuable if we can make an accurate prediction in stock market using machine learning. We try to build a model based on RNN (recurrent neural network) to do some prediction of stock market. There are several ways to predict a stock market from our background study. One way is to use a technique called Averaging mechanisms[2]. We predict the stock price of Day t+1 by calculating average price from a time window t to t-N. It is sensible because the stock cannot change a lot from day t to day t+1. On the other hand, RNN has a great value of predict the time-series model like stock market, especially LSTM, which allow us to fit the long-term and short-term data.

II. Technique and algorithm

A. Data Representation

If we set our time_step = 3 and input_size=1(Input features is "close price" only)

Input1=[[p_0],[p_1],[p_2]] Label1=[[p_3]]
Input2=[[p_1],[p_2],[p_3]] Label2=[[p_4]]
Input3=[[p_2],[p_3],[p_4]] Label2=[[p_5]]
……
Input$_n$=[[p_{n-1}],[p_n],[p_{n+1}]] Label$_t$=[[p_{n+2}])]]

We just predict one step forward at each input and we gather all of our output label and we are able to predict the change of the stock market.

B. Normalization

Since our data goes from 1100 to 3100 which has a large range of number. If we use the original dataset, we will come to out of scale issue and make pool prediction for our test data. Then we use the normalization technique. We use z-score normalization. We subtract mean of the price from the price and then divided by the standard deviation.

$$x' = \frac{x - x_{mean}}{\sigma}$$

Equation. 1. Equation of Normalization.

C. LSTM Structure

RNN (Recurrent neural network), derived from neural networks, is one of the most useful deep learning technologies to process time sequential data. RNN cells have the capability to memorize previous information which could influence the ongoing states, which provides them the advantages to handle the problems such as sentence prediction [3] or speech recognition.[4][5].

In standard RNNs, it has the form of a chained cells of simple neural network structure with a single *tanh* output layer. (shown in Fig. 1)

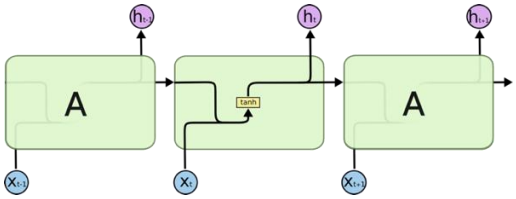

Fig 1. Chained cells of basic RNN with one output gate [6].

In time sequential data, information from previous time series may affect the current and future states. However, in traditional neural network models, with the increase of the hidden layers, the influence of previous states decreases a lot due to the vanishing gradient problem. RNN model is created to deal with this issue and is popular in the analysis of time sequential data.

Basic RNN networks usually focus on the states at the current and last one-time instant which will lose the long-term influence. To address this limitation, "Long-short term memory (LSTM)" model is introduced to take the long-time dependencies into consideration [7]. LSTM has good property in choose the degree on what to memorize or forget from old information. This make it can combine the new input data and the old memory. Due to this property, LSTM can perform well to predict stock price.

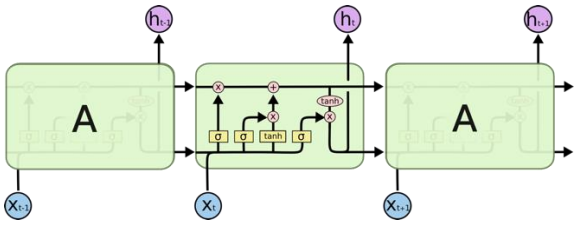

Fig 2. The repeating module in an LSTM contains four interacting layers[6].

LSTMs also have similar structure with RNN, but the repeating module is quite different. The repeating unit has four, interacting inputs in a very special way. (shown in the middle unit in Fig. 2). This unit is called LSTM cell.

To learn long-term dependencies, we need the information not only at the current time instant, but also from previous cells. LSTM model uses the term *Ct* to transport the dependency from the past. The forget gate layer, f_t, is used to forget certain amount of information from the previous cell state with the formula:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (1)$$

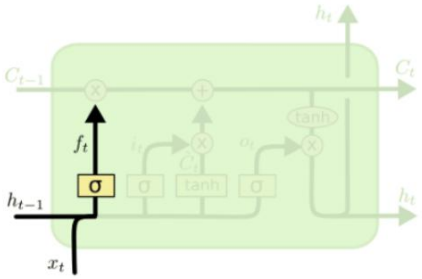

Fig.3. Forget gate, where h_{t-1} is the hidden state from previous layer and xt is the input at the current time instant [6].

Through the sigmoid activation function the output of the forget gate layer will be a real number between 0 and 1, where 0 means totally forget the previous dependencies and 1 means totally remember previous dependencies.

The input gate layer is used to decide how much information to include from the input t and the previous hidden layer h_{t-1}. Similarly, a sigmoid function is used to produce a value between 0 and 1 to determine the proportion the remember from the input[6].

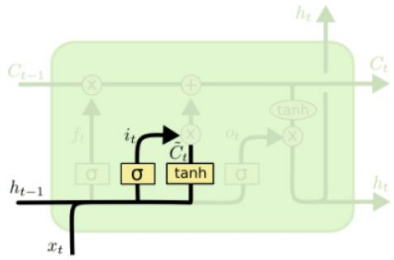

Fig.4. Input gate [6].

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2)$$

To update the overall cell state, we also need to calculate the cell state portion Ct from the current layer by applying a *tanh* function on the current input [6].

$$\widetilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (3)$$

Finally, we can update the overall cell state C_t by multiplying the previous cell state C_{t-1} with the forget gate ratio ft and multiplying the current cell state with the input gate ratio and add them together [6].

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C} \quad (4)$$

The output information is controlled by the output gate layer and hidden layer information ht is calculated by multiplying the output gate and the tanh result of the overall cell state [6].

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t * \tanh(C_t) \quad (6)$$

The above process is just the basic approach for the LSTM algorithm and people may use slightly different version of it. For the simplicity of our program, we will use the above algorithm to implement our functions.

In this project, we will use the LSTM networks for the prediction of stock price based on the prices of certain sequence of days in the past.

The loss function is the commonly used Minimum Square Error (MSE). The train data set and the test data are normalized using the same normalization process. We will explore the influence of the sequence length, learning rate and other parameters on the performance of our model.

III. Result and Analysis

A. Analysis Rule

We use our model to predict the price of the stock in the next time sequence, we have no definition of train or test

accuracy. we use the MSE for just test data to judge our model. And we show the graph for train loss and test dataset in every experiment.

B. Hyperparameters Tuning

Experiment Number	Parameters Chosen	Results
1	Hidden_size=50 Epoch=50 Time_squence= 3 Learning_rate = 0.1	Train/Test Split = 0.8/0.2 Size of dataset = 2517 Train MSE = 0.00064 Test MSE = 0.162
2	Hidden size =100 Epoch = 50 Time_squence=3 Learning_rate = 0.1	Train/Test Split = 0.8/0.2 Size of dataset = 2517 Train MSE =0.00067 Test MSE = 0.179
3	Hidden size =50 Epoch = 50 Time_squence=10 Learning_rate = 0.1	Train/Test Split = 0.8/0.2 Size of dataset = 2517 Train MSE =0.00060 Test MSE = 0.016
4	Hidden_size=50 Epoch=100 Time_squence= 3 Learning_rate = 0.1	Train/Test Split = 0.8/0.2 Size of dataset = 2517 Train MSE = 0.000632 Test MSE = 0.135
5	Hidden_size=50 Epoch=50 Time_squence= 3 Learning_rate = 0.01	Train/Test Split = 0.8/0.2 Size of dataset = 2517 Train MSE = 0.000722 Test MSE = 0.011

C. Visualization

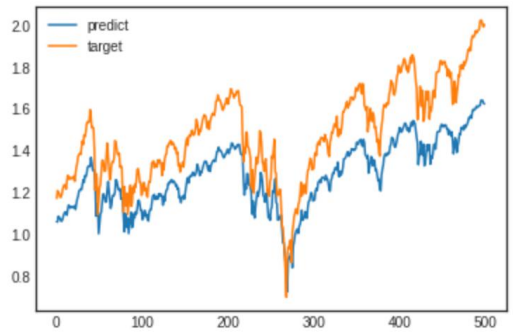

Fig.5. Comparison of Predict and Target in Experiment 1

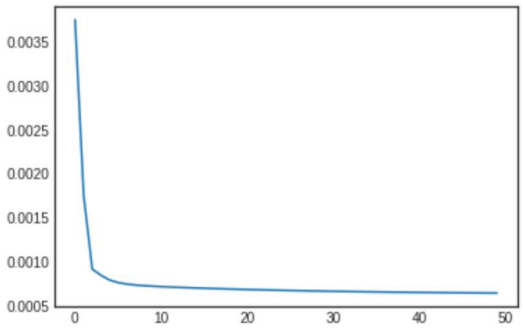
Fig.6. Training loss in Experiment 1

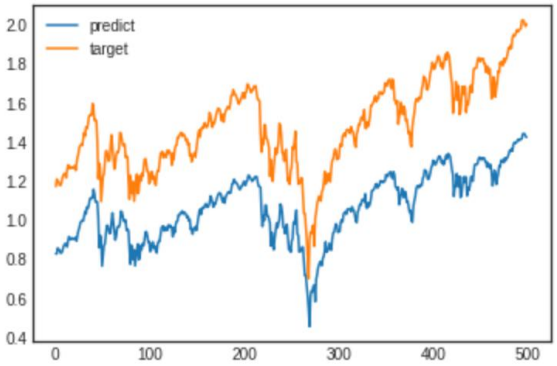

Fig.7. Comparisons of Predict and Target in Experiment 2

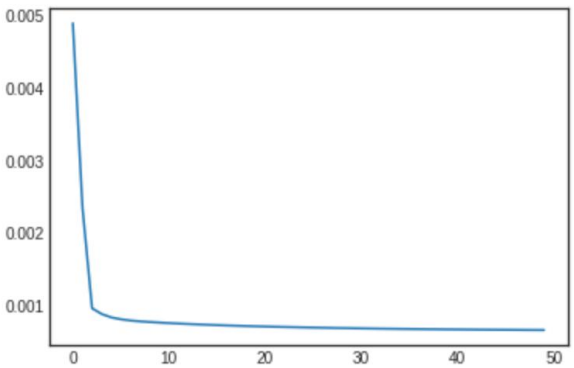
Fig.8. Training loss in Experiment 2

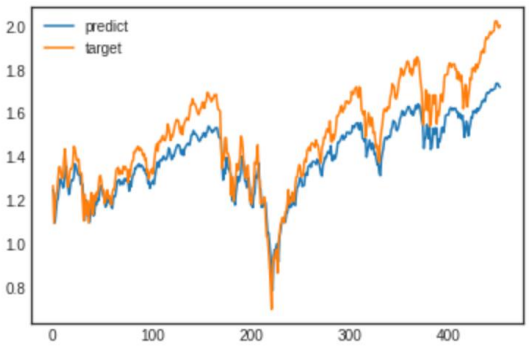

Fig.9. Comparisons of Predict and Target in Experiment 3

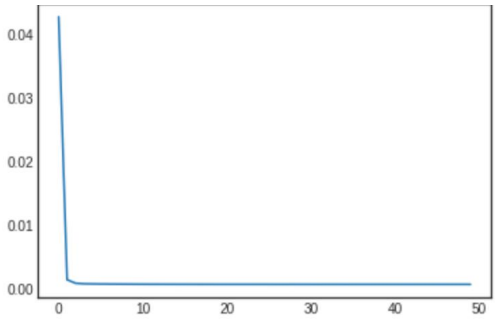

Fig.10. Training loss in Experiment 3

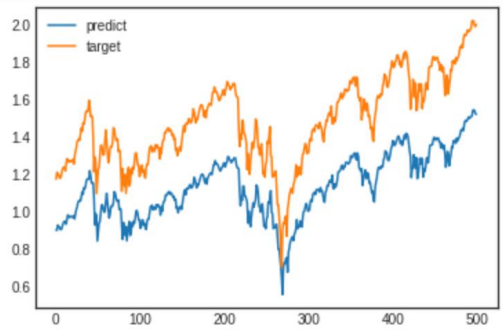

Fig.11. Comparisons of Predict and Target in Experiment 4

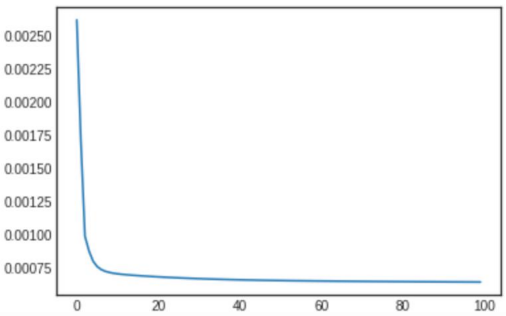

Fig.12. Training loss in Experiment 4

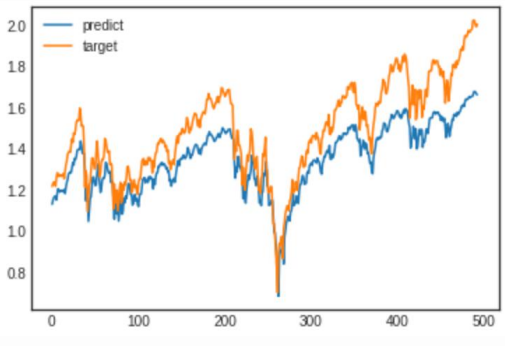

Fig.13. Comparisons of Predict and Target in Experiment 5

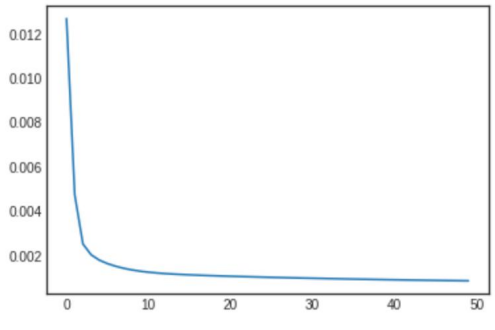

Fig.14. Training loss in Experiment 5

D. Effect of the hyperparameters

According to the figure above, we can see that the epoch and the hidden_size do not affect our train and test MSE too much. However, the increase of the time_sequence does reduce the MSE of the test dataset. It is reasonable because now we input more data (numbers of time_sequence) to get the one more step data. And at the same time, the change of learning_rate will affect our model because an appropriate learning_rate can prevent the model from underfitting and overfitting.

E. Result

According to the graph shown above, we can see that the predictions and targets are similar to each other and they almost have the same trend. i.e. It is reasonable to use LSTM model to do time series prediction in stock market.

IV. Conclusion and Future Work

By using LSTM model, it is able to memorize the historical data and make some prediction for the future stock price. It provides a good reference for analysts and investor or any people interested in stock market. However, it is not always stable because the stock market is affected by a lot of information like the consensus and confidence from people.

The future work will focus on the following two aspects:

Figure out more features about the stock market which may helper to predict the stock market. And at the same time, it would be better if we can predict some more steps forward.

Use the regularization (or dropout) to the neural network.

REFERENCES

[1] CSLT Group. [n. d.]. Historical Volatility Prediction based on LSTM. http://cslt.riit.tsinghua.edu.cn/mediawiki

[2] https://www.datacamp.com/community/tutorials/lstm-python-stock-market

[3] Graves, A.; Liwicki, M.; Fernandez, S.; Bertolami, R.; Bunke, H.; *Schmidhuber, J.* (2009). "A Novel Connectionist System for Improved Unconstrained Handwriting Recognition" (PDF). IEEE Transactions on Pattern Analysis and Machine Intelligence. 31 (5)

[4] Sak, Hasim; Senior, Andrew; Beaufays, Francoise (2014). "Long Short-Term Memory recurrent neural network architectures for large scale acoustic modeling" (PDF).

[5] Li, Xiangang; Wu, Xihong (2014-10-15). "Constructing Long Short-Term Memory based Deep Recurrent Neural Networks for Large Vocabulary Speech Recognition". *arXiv:1410.4281* [*cs.CL*].

[6] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[7] Hochreiter, Sepp; Schmidhuber, Jürgen *(1997-11-01).* "Long Short-Term Memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735.