Financial market prediction using Generative Adversarial Networks

July 13, 2021

Klaudia Palak   Hossein Hashemi

Abstract

Stock price forecasting has been a very popular topic for a long time. However, due to the complexity and randomness in the stock market, it is particularly difficult to accurately predict future stock prices. Under normal circumstances, investors and researchers will use some empirical formulas to calculate a large number of more effective factors from the original stock price information such as historical stock prices, trading volume, company financial reports, etc. In this report, our aim was to predict ”Close” prices beside LSTM and GAN models and compare the results using RMSE values. In addition, we also looked at perfomance for learned models using basic features such as Low, High or Close and using additional technical analysis indicators such as CCI, STOCH RSI, MACD to understand the impact of input data on our models.

1. Introduction

Financial markets offer an interesting environment to perform sequence prediction. Stock price prediction is an challenging topic and many studies have shown that self-correcting and highly non-stationary behavior still cause many problems for state-of-art approaches. After even a short research, it is clear that many solutions are based on an classic algorithm such as Long Short-Term Memory (LSTM). Similarly, since 2014, Generative Adversarial Networks (GAN) have also been the subject of intense experimentation. The generator and discriminator in the model are adversarial, which helps increase the result's accuracy. GAN is widely used in image generating, but not in time series prediction. Since there are few studies on time series prediction using GAN, their conclusions are inconsistent according to their studies.

This paper aims to use GAN to predict the stock price and check whether the adversarial system can help improve the time series prediction. Also, it includes the comparison between GAN approach and a traditional LSTM model.

Email: Klaudia Palak <klaudia.plk@gmail.com>, Hossein Hashemi <hossein.hashemi.ir@ieee.org>.

Deep Learning and Applied AI 2021, Sapienza University of Rome, 2nd semester a.y. 2020/2021.

2. Related work

Stock market prediction is widespread via time series models: traditional LSTM models and few studies that make the prediction using GAN. Zhichao Zou and Zihao Qu in their paper (Zou & Qu, 2020) proposed three different architectures using the LSTM approach: the LSTM model, the Stacked-LSTM model and the Attention-LSTM model. In the case of the second approach, result of using GAN to make the stock prediction is inconsistent. For example, Ricardo and Carrillo in (Alberto & Romero, 2018) compared the performance of the GAN model with traditional deep learning model LSTM. They used LSTM as the generator and Convolutional Neural Network (CNN) as the discriminator. The result showed no significant differences between GAN and traditional model LSTM. What's more the performance of GAN is even a little bit worse. However, according to the study from Zhang et al. in (Zhanga et al., 2019), they proposed a GAN model with the LSTM as the generator and Multi-Layer Perceptron (MLP) as the discriminator to forecasting the one day closing price of the stock, and also compared the result with baseline LSTM. The result showed that GAN performs better than their traditional baseline model.

3. Methods

LSTM method. Long short-term memory (LSTM) is a specific recurrent neural network (RNN) architecture.The main idea of both models is to apply the sequential observations learned from the earlier stages to forecast future trends. However, LSTM model introduces the memory cell that enables long-term dependency between time lags. The memory cells filters the information through the gate structure to maintain and update the state of memory cells. The gate structure includes input gate, forget gate and output gate. The forget gate in the LSTM determines which cell state information is discarded from the model - it accepts the output from the previous time step and the new input of the current time step. Its main function is to record the

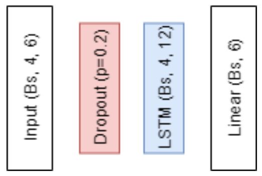

Figure 1. Basic LSTM Architecture

amount of information reserved from the previous cell state to the current cell state. The input gate determines how much the current time network input is reserved into the new cell state. It avoids feeding the unimportant information into the current memory cell. At the end the output gate controls how much the newly created cell state will be discarded (Zou & Qu, 2020).

Figure 4 indicated the architecture of LSTM which had been utilized. A single LSTM layer and dropout regularization were used in the reference model. Furthermore, Root Mean Square Error (RMSE) had been used to calculate the loss for the LSTM neural network. RMSE is defined as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}, \tag{1}$$

where y_i is the real value and $\hat{y}_i$ predicted value.

Apart from the names of the layers used, the figure 4 also shows the dimensions of the output data for each of them. In this case, Bs stands for batch size. In addition, for the Input values, we had the length of the output sequence and the number of selected features (in our case it was 6).

GAN method. In our GAN model, we set the LSTM as a generator and LSTM as the discriminator. The generator used the same architecture as our reference model described above. The discriminator in our GAN model was a LSTM that aimed to distinguish whether the input data of the discriminator was real or fake. A single LSTM layer and dropout regularization were used. In the final stage, a linear layer with a sigmoid activation function was added. The input for the discriminator was from the original data and the generated data from the generator to create a sequence of the same length as the ground truth input sequence. Figure 2 indicated the architecture of GAN architecture which had been used in experiments.

In our GAN model structure, Binary Cross Entropy (BCE) has been used to calculate the loss for both generator (G) and discriminator (D) during the training. The BCE is defined as:

$$L_{BCE} = \sum_{x^t \in X} \log(D(x^t)) + \sum_{z^t \in p(z)} \log(1 - D(G(z^t))). \tag{2}$$

Where z is the input data for generator, x is the target from the real dataset, G(z^t) is the generated data from the generator.

4. Dataset and Features

The stock price data was from Yahoo Finance. The target stock price in the model was Apple Inc. stock closing price. The statistical data were calculated using the stock closing price. There were a total of 2367 observations from 2012-01-01 to 2021-06-01. The train data, validation data and test data are split into 8:1:1. The following features were used to evaluate the models: Open, High, Low, Close, Adj Close and Volume. The listed features were defined successively as: the cash value of the first transacted in stock, the highest price of the stock in a day, the lowest price of the stock in a day, the cash value of the last transacted in stock, the raw price, which is just the cash value of the last transacted price before the market closes and the number of shares that are sold, or traded, over a day. In addition, for additional experiments, the dataset was extended with an additional 4 technical analysis indicators: Commodity Channel Index (CCI), Stochastic Relative Strength Index (STOCH RSI) and Moving Average Convergence Divergence (MACD).

Moreover, note that the normalization was necessary and a key point to achieve competitive results during the training. The final results for the test data were calculated for unscaled data.

5. Experimental results

In this paper, it evaluated the performance of each model by Root Mean Square Error defined as in the equation (1). The n is the number of the data points, y_i is actual stock price, and $\hat{y}_i$ denotes the predicted stock price. The best results were obtained for a batch size of 64 and a learning rate of 10^{-5}.

Considering the results of RMSE of close price on the validation dataset, the best performing weights for each model are appointed. The best epoch numbers of the models performed and their RMSE results for ”Close” price in the validation dataset and the test dataset are presented in Table 1.

Table 1. Overall performance results for basic dataset.

Method	Best Epoch	RMSE Validation	RMSE Test
GAN	3600	**3.3549**	**29.5047**
LSTM	**1400**	4.1865	29.6614

In addition, it was checked how additional technical analysis indicators influenced the results. The results for the best epoch with extended dataset are presented in Table 2.

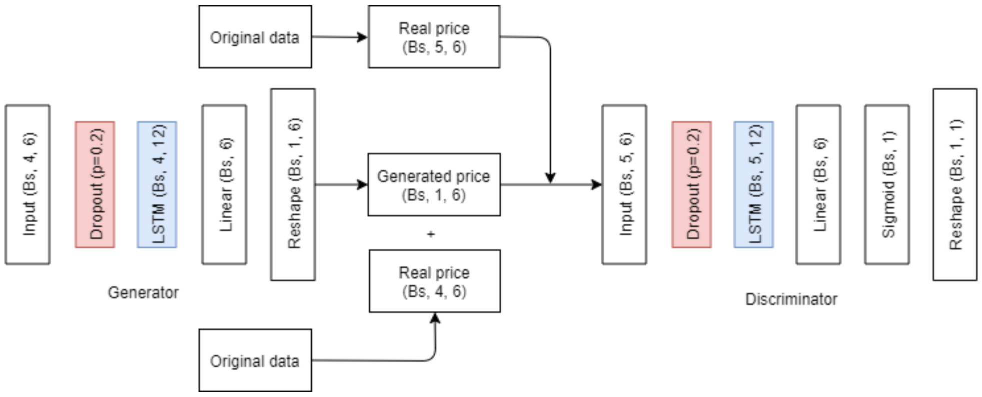

Figure 2. GAN Architecture

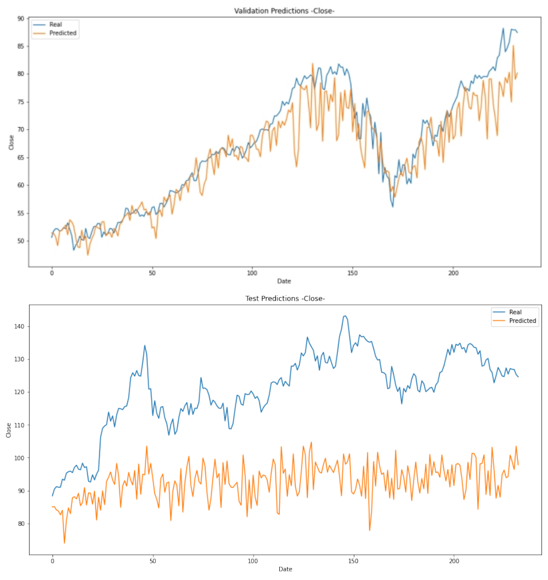

Figure 3. Prediction performance of the GAN: validation data and test data for the basic dataset.

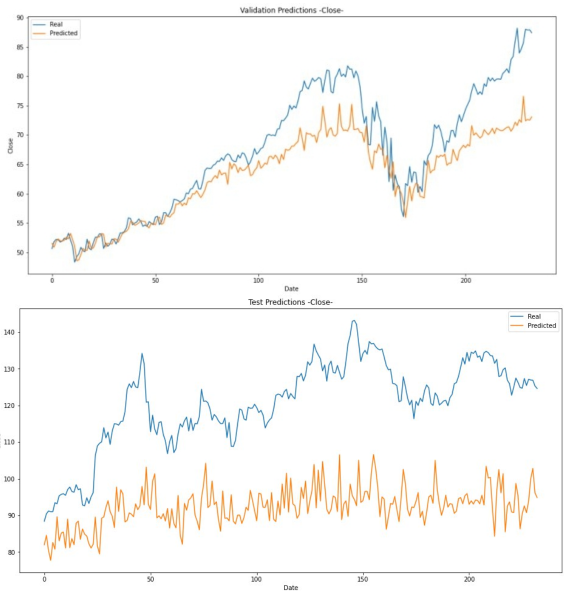

Figure 4. Prediction performance of the LSTM: validation data and test data for the basic dataset.

Table 2. Overall performance results for extended dataset.

Method	Best Epoch	RMSE Validation	RMSE Test
GAN	5300	**2.4680**	**27.0324**
LSTM	**2800**	4.6448	40.1456

6. Conclusion

This paper proposes a GAN that sets the LSTM as the generator and the LSTM as the discriminator. According to the results of the experiment, it can be seen that the performance results for our GAN model are better than the results of the traditional LSTM model. Nevertheless, the difference is insignificant, and the training time of the GAN model was three to four times longer.In an additional experiment using the extended dataset, worse results for the LSTM model could be noticed, but for the GAN model it is the lowest RMSE value in all the experiments performed. Keep in mind that both models consist of simple architectures.In the next steps, the architectures of each model could be expanded to create more deep models. Additionally, for example, we could extend the Generative Adversarial Network (GAN) with Gated Recurrent Units (GRU) - used as a generator as in article (Lin et al., 2021).

The code of the program on the basis of which this report was created can be found at the link: `https://github.com/klaudiaplk/financial_markets`.

References

Alberto, R. and Romero, C. Generative adversarial network for stock market price prediction. 2018.

Lin, H., Chen, C., Huang, G., and Jafari, A. Stock price prediction using generative adversarial networks. *Journal of Computer Science*, 17(3):188–196, 2021.

Zhanga, K., Zhonga, G., Donga, J., Wanga, S., and Wanga, Y. Stock market prediction based on generative adversarial network. *Procedia computer science*, 147:400–406, 2019.

Zou, Z. and Qu, Z. Using lstm in stock prediction and quantitative trading. 2020.