Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 147 (2019) 400–406

Procedia Computer Science

www.elsevier.com/locate/procedia

2018 International Conference on Identification, Information and Knowledge in the Internet of Things, IIKI 2018

Stock Market Prediction Based on Generative Adversarial Network

Kang Zhang[a], Guoqiang Zhong[a,*], Junyu Dong[a], Shengke Wang[a], Yong Wang[a]

[a]*Department of Computer Science and Technology, Ocean University of China, Qingdao, 266100, China*

Abstract

Deep learning has recently achieved great success in many areas due to its strong capacity in data process. For instance, it has been widely used in financial areas such as stock market prediction, portfolio optimization, financial information processing and trade execution strategies. Stock market prediction is one of the most popular and valuable area in finance. In this paper, we propose a novel architecture of Generative Adversarial Network (GAN) with the Multi-Layer Perceptron (MLP) as the discriminator and the Long Short-Term Memory (LSTM) as the generator for forecasting the closing price of stocks. The generator is built by LSTM to mine the data distributions of stocks from given data in stock market and generate data in the same distributions, whereas the discriminator designed by MLP aims to discriminate the real stock data and generated data. We choose the daily data on S&P 500 Index and several stocks in a wide range of trading days and try to predict the daily closing price. Experimental results show that our novel GAN can get a promising performance in the closing price prediction on the real data compared with other models in machine learning and deep learning.

Keywords: Deep Learning; Stock Prediction; Generative Adversarial Networks; Data Mining.

1. Introduction

The prediction of stock market returns is one of the most important and challenging issues in this domain. Many analyses and assumptions in financial area show that stock market is predictable. Technical analysis in stock investment theory is an analysis methodology for forecasting the direction of prices through the research on past market data. A meaningful assumption named Mean Reversion states that the stock price is temporary and tends to move to the average price over time. Moreover, this assumption has a further development called Moving Average Reversion (MAR), which supposes that the average of price is the mean of price in a past window of time, e.g. five days [7]. Based on the views mentioned above, we propose a new deep learning model to forecast the daily closing price.

The main contributions of this paper can be summarized in the followings:

* Guoqiang Zhong Tel.: +86-159-0639-8365.
E-mail address: gqzhong@ouc.edu.cn

10.1016/j.procs.2019.01.256

- A novel Generative Adversarial Network (GAN) architecture with Long-Short Term Memory (LSTM) network as the generator and Multi-Layer Perceptron (MLP) as the discriminator is proposed. The model trained in and end-to-end way to predict the daily closing price by giving the stock data in several past days.
- We try to generate the same distributions of the stock daily data through the adversarial learning system, instead of only utilizing traditional regression methods for the price forecasting.

2. Related Work

The stock market prediction can be seen as a time series forecasting issue and one of the classical algorithms is the Autoregressive Integrated Moving Average (ARIMA) [2]. ARIMA performs well in linear and stationary time series, but it doesnt perform well on the nonlinear and non-stationary data in stock market. In order to solve this problem, one approach [9] combines ARIMA with SVM. The idea is that the forecasting is constituted by a linear part and a nonlinear part, so that they can predict the linear part with ARIMA and the nonlinear part with SVM. Moreover, another approach [6] combines the wavelet basis with SVM, which decomposes the stock data with wavelet transformation and uses SVM for forecasting. Subsequently, the Artificial Neural Network (ANN) were combined with ARIMA to predict the nonlinear part of the stock price data [1]. The hybrid of wavelet transformation and ANN demonstrated that effective features should be extracted for the training of ANN [3]. Convolutional Neural Network (CNN) was also used in forecasting stock prices from the limit order book [12]. The number of orders and the price of 10 bid/ask orders were transformed into a 2D array. In addition, some designed RNNs had been applied to forecasting the stock data [11] [10]. News and events in financial area were extracted and represented as dense vectors to realize stock prediction [4]. Besides, reinforcement learning is another popular method to improve the trading strategies through fusing Q-learning and dynamic programming [8].

3. Our Methodology

3.1. Principle

GAN is a new framework which trains two models like a zero-sum game [5]. In the adversarial process, the generator can be seen as a cheater to generate the similar data as the real data, while the discriminator plays the role of judge to distinguish the real data and generated data. They can reach an ideal point that the discriminator is unable to differentiate the two types of data. At this point, the generator can capture the data distributions from this game. Based on this principle, we propose our GAN architecture for the prediction of stock closing price.

3.2. The Generator

The generator of our model is designed by LSTM with its strong ability in processing time series data. We choose the daily data in the last 20 years with 7 financial factors to predict the future closing price. The 7 factors of the stock data in one day are High Price, Low Price, Open Price, Close Price, Volume, Turnover Rate and Ma5 (the average of closing price in past 5 days). The 7 factors are valuable and significant in price prediction with the theory of technical analysis, Mean Reversion, or MAR. Therefore, these factors can be used as 7 features of the stock data for the price prediction. Suppose our input is $\mathbf{X} = \{\mathbf{x_1}, ..., \mathbf{x_t}\}$, which consists of the daily stock data of t days. Each $\mathbf{x_k}$ in $\mathbf{X}$ is a vector, which is composed of 7 features as follows:

$$[\mathrm{x_{k,i}}]_{\mathrm{i=1}}^{7} = [\mathrm{x_{k,High}}, \mathrm{x_{k,Low}}, \mathrm{x_{k,Open}}, \mathrm{x_{k,Close}}, \mathrm{x_{k,TurnoverRate}}, \mathrm{x_{k,Volume}}, \mathrm{x_{k,Ma5}}]. \tag{1}$$

The architecture of the generator is shown in Fig. 1. For simplicity, we have omitted the details of the LSTM. With the generator, we extract the output $\mathbf{h_t}$ of the LSTM and put it into a fully connected layer with 7 neurons to generate the $\hat{\mathbf{x}}_{\mathbf{t+1}}$. $\hat{\mathbf{x}}_{\mathbf{t+1}}$ aims to approximate $\mathbf{x_{t+1}}$ and we can get $\hat{\mathrm{x}}_{\mathrm{t+1,Close}}$ from $\hat{\mathbf{x}}_{\mathbf{t+1}}$ as the prediction of closing price on the $t+1$ day.

The output of generator $\mathrm{G}(\mathbf{X})$ is defined as follows:

$$\mathbf{h_t} = \mathrm{g}(\mathbf{X}), \tag{2}$$

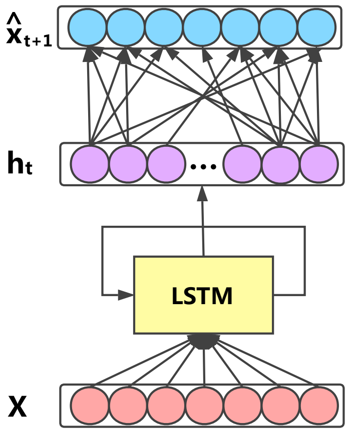

Fig. 1. The generator designed with an LSTM.

$$G(\mathbf{X}) = \hat{\mathbf{x}}_{\mathbf{t+1}} = \delta(\mathbf{W}_{\mathbf{h}}^{\mathbf{T}}\mathbf{h}_{\mathbf{t}} + \mathbf{b}_{\mathbf{h}}), \tag{3}$$

where g(·) denotes the output of LSTM and $\mathbf{h_t}$ is the output of the LSTM with $\mathbf{X} = \{\mathbf{x_1}, ..., \mathbf{x_t}\}$ as the input. δ stands for the Leaky Rectified Linear Unit (ReLU) activation function. $\mathbf{W_h}$ and $\mathbf{b_h}$ denote the weight and bias in the fully connected layer. We also use dropout as a regularization method to avoid overfitting. Furthermore, we can continue to predict $\hat{\mathbf{x}}_{\mathbf{t+2}}$ with $\hat{\mathbf{x}}_{\mathbf{t+1}}$ and $\mathbf{X}$.

3.3. The Discriminator

The purpose of the discriminator is to constitute a differentiable function D to classify the input data. The discriminator is expected to output 0 when inputting a fake data and output 1 when inputting a real data. Here, we choose an MLP as our discriminator with three hidden layers h1,h2,h3 including 72, 100, 10 neurons, respectively. The Leaky ReLU is used as the activation function among the hidden layers and the sigmoid function is used in the output layer. In addition, the cross entropy loss is chosen as the loss function to optimize the MLP. In particular, we concatenate the $\mathbf{X} = \{\mathbf{x_1}, ..., \mathbf{x_t}\}$ and $\hat{\mathbf{x}}_{\mathbf{t+1}}$ to get $\{\mathbf{x_1}, ..., \mathbf{x_t}, \hat{\mathbf{x}}_{\mathbf{t+1}}\}$ as the fake data $\mathbf{X_{fake}}$. Similarly, we concatenate the $\mathbf{X} = \{\mathbf{x_1}, ..., \mathbf{x_t}\}$ and $\mathbf{x_{t+1}}$ to get $\{\mathbf{x_1}, ..., \mathbf{x_t}, \mathbf{x_{t+1}}\}$ as the real data $\mathbf{X_{real}}$. The output of the discriminator is defined as follows:

$$D(\mathbf{X_{fake}}) = \sigma(d(\mathbf{X_{fake}})), \tag{4}$$

$$D(\mathbf{X_{real}}) = \sigma(d(\mathbf{X_{real}})), \tag{5}$$

where d(·) denotes the output of MLP and denotes the sigmoid activation function. Both $\mathbf{X_{fake}}$ and $\mathbf{X_{real}}$ output a single scalar. Fig. 2 shows the architecture of the discriminator.

3.4. The Architecture of GAN

With the two models mentioned above, we propose our GAN architecture. According to [5], in the two-player minimax game, both G and D try to optimize a value function. Similarly, we can define the optimization of our value function V(G,D) as follows:

$$\min_G \max_D V(G, D) = E\left[\log D\left(\mathbf{X_{real}}\right)\right] + E\left[\log\left(1 - D\left(\mathbf{X_{fake}}\right)\right)\right]. \tag{6}$$

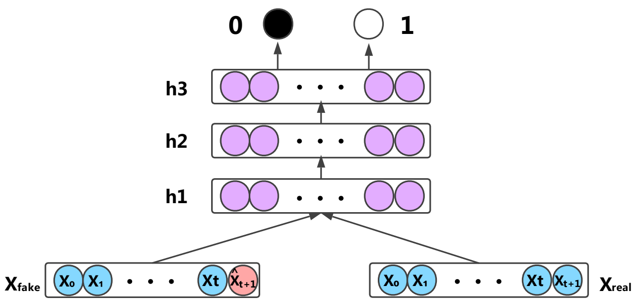

Fig. 2. Discriminator designed using an MLP with $\mathbf{X_{fake}}$ and $\mathbf{X_{real}}$ as the inputs.

We define the generator loss G_{loss} and discriminator loss D_{loss} to optimize the value function. Particularly, we combine the Mean Square Error (MSE) with the generator loss of a classical GAN to constitute the G_{loss} of our architecture. The G_{loss} and D_{loss} are as follows:

$$D_{loss} = -\frac{1}{m}\sum_{i=1}^{m} \log D(\mathbf{X^i_{real}}) - \frac{1}{m}\sum_{i=1}^{m} \log(1 - D(\mathbf{X^i_{fake}})), \tag{7}$$

$$g_{MSE} = \frac{1}{m}\sum_{i=1}^{m} (\hat{\mathbf{x}}^i_{\mathbf{t+1}} - \mathbf{x^i_{t+1}})^2, \tag{8}$$

$$g_{loss} = \frac{1}{m}\sum_{i=1}^{m} \log(1 - D(\mathbf{X^i_{fake}})), \tag{9}$$

$$G_{loss} = \lambda_1 g_{MSE} + \lambda_2 g_{loss}. \tag{10}$$

The loss function G_{loss} is composed by g_{MSE} and g_{loss} with λ_1 and λ_2, respectively. λ_1 and λ_2 are hyper-parameters that we set manually. Fig. 3 shows the architecture of our GAN. The reason why we put $\mathbf{X_{fake}}$ and $\mathbf{X_{real}}$ rather than $\hat{\mathbf{x}}_{\mathbf{t+1}}$ and $\mathbf{x_{t+1}}$ in the discriminator is that we expect the discriminator to capture the correlation and time series information between $\mathbf{x_{t+1}}$ and $\mathbf{X}$.

4. Experiments

4.1. Dataset Descriptions

We evaluate our model on the real stock data, including the Standard & Poor's 500 (S&P 500 Index), Shanghai Composite Index in China, International Business Machine (IBM) from New York Stock Exchange (NYSE), Microsoft Corporation (MSFT) from National Association of Securities Dealers Automated Quotation (NASDAQ), Ping An Insurance Company of China (PAICC). All the stock data can be downloaded in Yahoo Finance. We select the trade date within the last 20 years (almost 5000 pieces of data in each stock). For instance, some examples of the stock features are shown in Tab. 1 . The trade date is not continuous due to the limitation of trading on weekends and holidays.

Note that the normalization is necessary and a key point to achieve competitive results. With the assumption of MAR mentioned above, we normalize the data as follows:

$$\mathbf{x_i} = \frac{x_i - \mu^t}{\tau^t}, \tag{11}$$

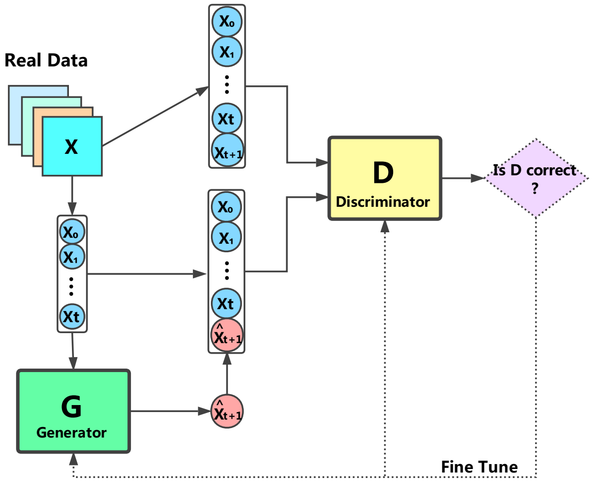

Fig. 3. The architecture of our GAN.

Table 1. Raw stock data from PAICC.

Name	Trade Date	Open Price	Highest Price	Lowest Price	Close Price	Turnover Volume	Turnover Rate	Ma5
PAICC	2017/12/20	73.49	74.59	73.08	73.94	93938288	0.0087	71.792
PAICC	2017/12/21	73.64	75.58	73.21	74.91	93470766	0.0086	72.608
PAICC	2017/12/22	74.8	75.2	73.63	74.02	7195658	0.0066	73.482
PAICC	2017/12/25	74.06	76.17	73.37	74.16	110997896	0.0102	74.201
PAICC	2017/12/26	74.11	74.56	72.88	73.94	83228950	0.0077	74.136

where μ^t and τ^t are the mean and standard deviation of $\mathbf{X}$. We select $t = 5$ empirically because we attempt to predict the data in the next day by data in the past one week (trade is limited on weekends). For instance, we compute the mean and standard deviation of the data of 5 days to normalize the data. Afterwards, the normalized data are used to predict the data on 6th day. The data in both training and testing periods are processed in the same way.

4.2. Training of our model

Our purpose is to predict these 7 factors and get the closing price in the next day through the data in the past t days. The reason why predicting 7 factors in the next day is that the generator aims to mining the distributions of the real data and we can get the closing price from the generated data. The data are separated into two parts for training and testing. We choose the first 90%-95% of the stock data for training and the remaining 5%-10% (about 250-500 pieces of data) for testing.

The loss in the training period can be seen in Fig. 4. There is a significant adversarial process between the discriminator and generator during training. Both the discriminator and generator have been optimized during the adversarial process.

4.3. Experimental results

We evaluate the forecasting performance of our model by the following statistical indicators: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE) and Average Return (AR).

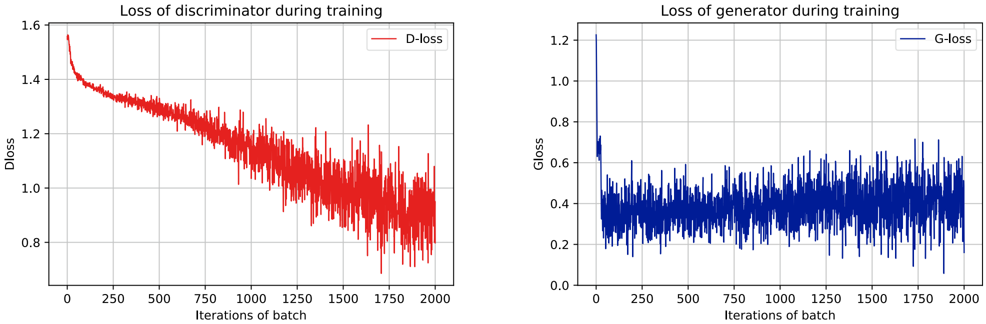

Fig. 4. Losses of the discriminator and generator during training.

Suppose the real closing price and the prediction of closing price on the k-th day as y_k and $\hat{y}_k$. Then the indicators are given as follows:

$$\mathrm{MAE} = \frac{1}{N}\sum_{k=1}^{N}\left|\hat{y}_k - y_k\right|, \tag{12}$$

$$\mathrm{RMAE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(\hat{y}_k - y_k)^2}, \tag{13}$$

$$\mathrm{MAE} = \frac{1}{N}\sum_{k=1}^{N}\frac{\left|\hat{y}_k - y_k\right|}{y_k}, \tag{14}$$

$$\mathrm{AR} = \frac{1}{N-1}\sum_{k=1}^{N-1}(y_{k+1} - y_k)\,, \text{if } \hat{y}_{k+1} > \hat{y}_k. \tag{15}$$

We compute the mean of RMSE on our five datasets as the average evaluation. MAE and HR are also calculated in this way. Support Vector Regression (SVR), ANN and LSTM are classical methods for stock market prediction and we choose them as the baselines to compare with our model. The prediction results are shown in Tab. 2 with the boldface as the best results. Low MAE, RMSE and MAPE indicate that the prediction of closing price is approximate to the real data. AR shows the daily average return of these stocks based on four prediction methods. We can see our method achieves a competitive performance compared with other methods.

Table 2. The average evaluation on five stock data sets.

Method	MAE	RMSE	MAPE	AR
Our GAN	**3.0401**	**4.1026**	**0.0137**	**0.7554**
LSTM	4.1228	5.4131	0.0145	0.6859
ANN	7.3029	9.1757	0.0808	0.5249
SVR	4.9285	8.2261	0.0452	0.7266

Fig. 5 shows an example of prediction by four methods on PAICC with the same training steps. From Fig. 5 we can see that the best performance in matching the trend line of the real price is achieved by our method.

5. Conclusion

We have made an exploration in stock market prediction and attempt to catch the distributions of the real stock data by our proposed GAN. For the future work, we plan to explore how to extract more valuable and influential financial

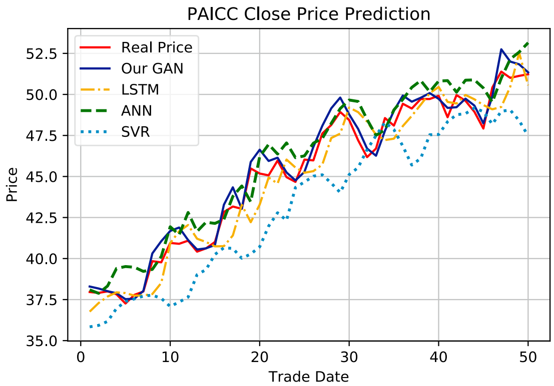

Fig. 5. Illustration of price prediction by our GAN and some compared models on PAICC.

factors from stock markets and optimize our model to learn the data distributions more accurately, so that we can obtain a higher precision of trend or price prediction in stock market by our method.

Acknowledgements

This work was supported by the National Key R&D Program of China under Grant 2016YFC1401004, the National Natural Science Foundation of China (NSFC) under Grant No. 61170312 and 61633021, the Science and Technology Program of Qingdao under Grant No. 17-3-3-20-nsh, the CERNET Innovation Project under Grant No. NGII20170416, the State Key Laboratory of Software Engineering under Grant No. SKLSE2012-09-14, and the Fundamental Research Funds for the Central Universities of China.

References

[1] Areekul, P., Senjyu, T., Toyama, H., Yona, A., 2010. A hybrid arima and neural network model for short-term price forecasting in deregulated market. IEEE Transactions on Power Systems Pwrs .
[2] Box, G.E.P., Jenkins, G.M., 1976. Time series analysis: Forecasting and control. Journal of Time 31, 238–242.
[3] Chandar, S.K., Sumathi, M., Sivanandam, S.N., 2016. Prediction of stock market price using hybrid of wavelet transform and artificial neural network. Indian Journal of Science & Technology 9.
[4] Ding, X., Zhang, Y., Liu, T., Duan, J., 2015. Deep learning for event-driven stock prediction, in: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015, pp. 2327–2333.
[5] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A.C., Bengio, Y., 2014. Generative adversarial nets, in: Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada, pp. 2672–2680.
[6] Huang, S., Wang, H., 2006. Combining time-scale feature extractions with svms for stock index forecasting, in: Neural Information Processing, 13th International Conference, ICONIP 2006, Hong Kong, China, October 3-6, 2006, Proceedings, Part III, pp. 390–399.
[7] Li, B., Hoi, S.C.H., 2012. On-line portfolio selection with moving average reversion, in: Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012.
[8] Nevmyvaka, Y., Feng, Y., Kearns, M.J., 2006. Reinforcement learning for optimized trade execution, in: Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006, pp. 673–680.
[9] Pai, P.F., Lin, C.S., 2005. A hybrid arima and support vector machines model in stock price forecasting. Omega 33, 497–505.
[10] Rather, A.M., Agarwal, A., Sastry, V.N., 2015. Recurrent neural network and a hybrid model for prediction of stock returns. Expert Syst. Appl. 42, 3234–3241.
[11] Saad, E.W., Prokhorov, D.V., II, D.C.W., 1998. Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. IEEE Trans. Neural Networks 9, 1456–1470.
[12] Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., Iosifidis, A., 2017. Forecasting stock prices from the limit order book using convolutional neural networks, in: 19th IEEE Conference on Business Informatics, CBI 2017, Thessaloniki, Greece, July 24-27, 2017, Volume 1: Conference Papers, pp. 7–12.