Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 4907423, 11 pages
https://doi.org/10.1155/2018/4907423

Research Article

Stock Market Prediction on High-Frequency Data Using Generative Adversarial Nets

Xingyu Zhou,[1] **Zhisong Pan**,[1] **Guyu Hu**,[1] **Siqi Tang**,[1] **and Cheng Zhao**[1,2]

[1]*Army Engineering University of PLA, Nanjing 210007, China*
[2]*Anhui Provincial Key Laboratory of Network and Information Security, Anhui Normal University, Wuhu 241003, China*

Correspondence should be addressed to Zhisong Pan; hotpzs@hotmail.com

Received 6 November 2017; Revised 21 January 2018; Accepted 13 February 2018; Published 15 April 2018

Academic Editor: Qian Zhang

Stock price prediction is an important issue in the financial world, as it contributes to the development of effective strategies for stock exchange transactions. In this paper, we propose a generic framework employing Long Short-Term Memory (LSTM) and convolutional neural network (CNN) for adversarial training to forecast high-frequency stock market. This model takes the publicly available index provided by trading software as input to avoid complex financial theory research and difficult technical analysis, which provides the convenience for the ordinary trader of nonfinancial specialty. Our study simulates the trading mode of the actual trader and uses the method of rolling partition training set and testing set to analyze the effect of the model update cycle on the prediction performance. Extensive experiments show that our proposed approach can effectively improve stock price direction prediction accuracy and reduce forecast error.

1. Introduction

Predicting stock prices is an important objective in the financial world [1–3], since a reasonably accurate prediction has the possibility to yield high financial benefits and hedge against market risks. With the rapid growth of Internet and computing technologies, the frequency for performing operations on the stock market had increased to fractions of seconds [4, 5]. Since year of 2009 the BM&F Bovespa (the Brazilian stock exchange) has worked in high-frequency, and the number of high-frequency operations has grown from 2.5% in 2009 to 36.5% in 2013. Aldridge and Krawciw [6] estimate that in 2016 high-frequency trading on average initiated 10%–40% of trading volume in equities and 10%–15% of volume in foreign exchange and commodities. These percentages suggest that the high-frequency stock market is a global trend.

In most cases, the forecast results are assessed from two aspects: the first is forecast error (chiefly the RMSE (Root Mean Square Error) or RMSRE (Root Mean Square Relative Error)) between real price and forecast value; the second is direction prediction accuracy, which means the percentage of correct predictions of price series direction, as upward and downward movements are what really matters for decision-making. Even small improvements in predictive performance can be very profitable [7, 8].

However, predicting stock prices is not an easy work, due to the complexity and chaotic dynamics of the markets and the many nondecidable, nonstationary stochastic variables involved [9]. Many researchers from different areas have studied the historical patterns of financial time series and have proposed various methods for forecasting stock prices. In order to achieve promising performance, most of these ways require careful selection of input variables, establishing predictive model with professional financial knowledge, and adopting various statistical methods for arbitrage analysis, which makes it difficult for people outside the financial field to use these methods to predict stock prices [10–12].

Generative adversarial network (GAN) was introduced by Goodfellow et al. [13], where images patches are generated from random noise using two networks trained simultaneously. Specifically, in GAN a discriminative net D learns to distinguish whether a given data instance is real or not, and a generative net G learns to confuse D by generating high quality data. Although this approach has been successful and applied to a wide range of fields, such as image inpainting,

semantic segmentation, and video prediction [14–16], as far as we know, it has not been used for stock forecasting.

This work uses basic technical index data as an input variable, which can be acquired directly from trading software, so that people outside the financial field can predict stock price through our method easily. This study introduces forecast error loss and direction prediction loss and shows that generative adversarial training [13] may be successfully employed for combining these losses to produce satisfying predict results, and we call this prediction architecture GAN-FD (GAN for minimizing forecast error loss and direction prediction loss). For the purpose of conforming to the practice of actual transactions, this work carries out rolling segmentation on training set and testing set of the raw data, and we will illustrate it in detail in the experimental section.

Overall, our main contributions are twofold: (1) we adapted generative adversarial network for the purpose of price prediction, which constitutes to our knowledge the first application of adversarial training to stock market, and extensive experiments show that our prediction model can achieve remarkable results and (2) we carry out rolling segmentation on training set and testing set of the raw data to investigate the effect the of model parameter update cycle on the stock forecast performance, and the experimental results show that smaller model update cycle can advance prediction performance.

In the remainder of this paper, we begin with a review of the literature on which algorithms have been used for the financial market prediction. Then we formulate the problem and propose our general adversarial network framework. Furthermore, in the experiments section, we presented the experimental analysis with the proposed model, as well as a comparison between the obtained results with those given by classical prediction models. Finally, conclusions and possible extensions are discussed.

2. Related Work

This section introduce the related work from the stock market prediction method and the generative adversarial network.

2.1. Stock Market Prediction Method. According to the research developed in this field, we can classify the techniques used to solve the stock market prediction problems to twofold.

The first category of related work is *econometric models*, which includes classical econometric models for forecasting. Common methods are the autoregressive method (AR), the moving average model (MA), the autoregressive moving average model (ARMA), and the autoregressive integrated moving average (ARIMA) [17–19]. Roughly speaking, these models take each new signal as a noisy linear combination of the last few signals and independent noise terms. However, most of them rely on some strong assumptions with respect to the noise terms (such as i.i.d. assumption, t-distribution) and loss functions, while real financial data may not fully satisfy these assumptions. By introducing a generalized autoregressive conditional heteroscedastic (GARCH) model for conditional variances, Pellegrini et al. [20] apply ARIMA-GARCH model to the prediction of financial time series.

The second category involves *soft computing based models*. Soft computing is a term that covers artificial intelligence which mimics biological processes. These techniques include artificial neural networks (ANN) [21, 22], fuzzy logic (FL) [23], support vector machines (SVM) [24, 25], particle swarm optimization (PSO) [26], and many others. Many authors have tried to deal with fuzziness along with randomness in option pricing models [27, 28]. Carlsson and Fullér [29] were the first to study the fuzzy real options and Thavaneswaran et al. [30] demonstrated the superiority of the fuzzy forecasts and then derived the membership function for the European call price by fuzzifying the interest rate, volatility, and the initial value of the stock price. Recently there has been a resurgence of interest in deep learning, whose basic structure is best described as a multilayer neural network [31]. Some literatures have established various models based on deep neural networks to improve the prediction ability of high-frequency financial time series [32, 33]. The ability of deep neural networks to extract abstract features from data is also attractive, Chong et al. [12] applied a deep feature learning-based stock market prediction model, which extract information from the stock return time series without relying on prior knowledge of the predictors and tested it on high-frequency data from the Korean stock market. Chen et al. [34] proposed a double-layer neural network for high-frequency forecasting, with links specially designed to capture dependence structures among stock returns within different business sectors. There also exist a few studies that apply deep learning to identification of the relationship between past news events and stock market movements [35–37].

However, to our knowledge, most of these methods require expertise to impose specific restrictions on the input variables, such as combining related stocks together as entry data [12], inputting different index data to different layers of the deep neural network [34], and converting news text into structured representation as input [36]. In contrast, our proposed forecasting model directly uses the data provided by the trading software as input, which reduce the barrier for ordinary investors.

2.2. Generative Adversarial Network. Generative adversarial network (GAN) is a framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to 0.5 everywhere [13]. While G and D are defined by multilayer perceptrons in [13], most researches recently constructed G and D on the basis of Long Short-Term Memory (LSTM) [38] or convolutional neural network (CNN) [39] for a large variety of application.

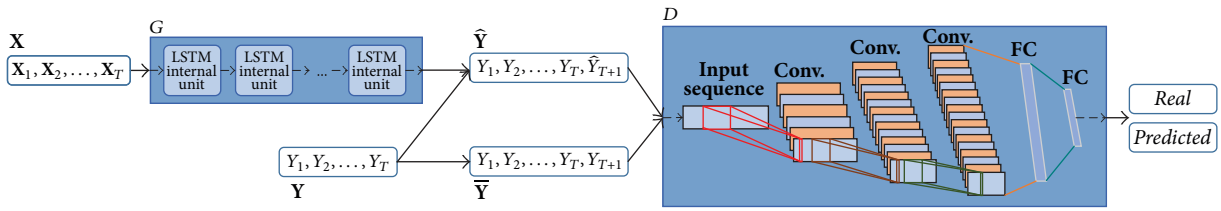

Figure 1: GAN-FD architecture. The generator (G) is founded on LSTM, which applies to predicting $\hat{Y}_{T+1}$. The discriminator (D) is based on CNN for the purpose of estimating the probability whether a sequence is real ($\overline{\mathbf{Y}}$) or being predicted ($\hat{\mathbf{Y}}$). *Conv.* means convolutional layer, *FC* is an abbreviation for fully connected layer. The structure of G and D can be adjusted according to the specific application.

LSTM is a basic deep learning model and capable of learning long-term dependencies. A LSTM internal unit is composed of a cell, an input gate, an output gate, and a forget gate. LSTM internal units have hidden state augmented with nonlinear mechanisms to allow state to propagate without modification, be updated, or be reset, using simple learned gating functions. LSTM work tremendously well on various problems, such as natural language text compression, handwriting recognition, and electric load forecasting.

CNN is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery. A CNN consists of an input layer and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers, and normalization layers. CNN also has many applications such as image and video recognition, recommender systems, and natural language processing.

Although there are a lot of literatures forecast stock price by using LSTM model, to the best of our knowledge, this paper is the first to adopt GAN to predict stock prices. The experimental part (Section 4.2) compares the prediction performances between GAN-FC and LSTM.

3. Forecasting with High-Frequency Data

In this section, we illuminate the details of the generative adversarial network framework for stock market forecasting with high-frequency data.

3.1. Problem Statement. Under the high-frequency trading environment, high-quality one-step forecasting is usually of great concern to algorithmic traders, providing significant information to market makers for risk assessment and management. In this article, we aim to forecast the price movement of individual stocks or the market index one step ahead, based solely on their historical price information. Our problem can be mathematically formalized as follows.

Let $\mathbf{X}_t$ represent a set of basic indicators and Y_t denote the closing price of one stock for a 1-minute interval at time t ($t = 1, 2, \ldots, T$), where T is the maximum lag of time. Given the historical basic indicators information $\mathbf{X}$ ($\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T\}$) and the past closing price $\mathbf{Y}$ ($\mathbf{Y} = \{Y_1, Y_2, \ldots, Y_T\}$), our goal is to predict the closing price Y_{T+1} for the next 1-minute time interval. There are literatures that examined the effects of different T [7, 12, 40], but, in this work, we just set T to 242 because each trading day contains 242-minute intervals in the China stock exchanges.

3.2. Prediction Model. The deep architecture of the proposed GAN-FD model is illustrated as in Figure 1. Since the stock data is a typical time series, we choose LSTM model, which is widely applied to time series prediction, as the generative model G to predict output $\hat{Y}_{T+1}$ based on the input data $\mathbf{X}$; that is,

$$\hat{Y}_{T+1} = G(\mathbf{X}). \tag{1}$$

The discriminative model D is based on the CNN architecture and performs convolution operations on the one-dimensional input sequence in order to estimate the probability whether a sequence comes from the dataset ($\overline{\mathbf{Y}} = \{Y_1, Y_2, \ldots, Y_T, Y_{T+1}\}$) or being produced by a generative model G ($\hat{\mathbf{Y}} = \{Y_1, Y_2, \ldots, Y_T, \hat{Y}_{T+1}\}$).

Our main intuition on why to use an adversarial loss is that it can simulate the operating habits of financial traders. An experienced trader usually predicts stock price through the available indicator data, which is the work of the generative model G, and then judges the correct probability of his own forecast with the previous stock price, as the discriminative model D does.

It is noteworthy that the structure of G and D in GAN-FD can be adjusted according to specific application, and the experimental part in this paper just proposed simple G and D framework (Section 4.2) for stock prediction. It is reasonable to believe that fine-tuning the structure of G and D can improve the predictive performance.

3.3. Adversarial Training. The training of the pair (G, D) consists of two alternated steps, described below. For the sake of clarity, we assume that we use pure SGD (minibatches of size 1), but there is no difficulty to generalize the algorithm to minibatches of size K by summing the losses over the samples.

Training G (let ($\mathbf{X}$, $\mathbf{Y}$) be a sample from the dataset). In order to make the discriminative model D as "confused" as possible, the generative model G should reduce the adversarial loss in the sense that D will not discriminate the prediction correctly. Classifying $\overline{\mathbf{Y}}$ into class 1 and $\hat{\mathbf{Y}}$ into class 0, the adversarial loss for G is

$$L_{\text{adv}}^{G}\left(\hat{\mathbf{Y}}\right) = L_{\text{sce}}\left(D\left(\hat{\mathbf{Y}}\right), 1\right), \tag{2}$$

where L_{sce} is the sigmoid cross-entropy loss, defined as

$$L_{\text{sce}}(\mathbf{A}, \mathbf{B}) = -\sum_i B_i \log\left(\text{sigmoid}\left(A_i\right)\right) + \left(1 - B_i\right) \log\left(1 - \text{sigmoid}\left(A_i\right)\right). \tag{3}$$

However, in practice, minimizing adversarial loss alone cannot guarantee satisfying predictions. Imagine that G could generate samples to "confuse" D, without being close to $\widehat{Y}_{T+1}$, and then D will learn to discriminate these samples, leading G to generate other "confusing" samples, and so on. To address this problem, the generative model G ought to decrease the forecast error loss; that is, L_p loss

$$L_p\left(\overline{\mathbf{Y}}, \widehat{\mathbf{Y}}\right) = \left\|\overline{\mathbf{Y}} - \widehat{\mathbf{Y}}\right\|_p \tag{4}$$

where $p = 1$ or $p = 2$.

Furthermore, as mentioned above, stock price direction prediction is crucial to trading, so we define direction prediction loss function L_{dpl}:

$$L_{\text{dpl}}\left(\overline{\mathbf{Y}}, \widehat{\mathbf{Y}}\right) = \left|\text{sgn}\left(\widehat{Y}_{T+1} - Y_T\right) - \text{sgn}\left(Y_{T+1} - Y_T\right)\right| \tag{5}$$

where sgn represents sign function.

Combining all these losses previously defined with different parameters λ_{adv}, λ_p, and λ_{dpl}, we achieve the final loss on G:

$$L_G(\mathbf{X}, \mathbf{Y}) = \lambda_{\text{adv}} L_{\text{adv}}^G\left(\widehat{\mathbf{Y}}\right) + \lambda_p L_p\left(\overline{\mathbf{Y}}, \widehat{\mathbf{Y}}\right) + \lambda_{\text{dpl}} L_{\text{dpl}}\left(\overline{\mathbf{Y}}, \widehat{\mathbf{Y}}\right). \tag{6}$$

Then we perform one SGD iteration on G to minimize $L_G(\mathbf{X}, \mathbf{Y})$ while keeping the weights of D fixed.

Training D (let $(\mathbf{X}, \mathbf{Y})$ be a different data sample). Since the role of D is just to determine whether the input sequence is $\mathbf{Y}$ or $\widehat{\mathbf{Y}}$, the target loss is equal to the adversarial loss on D. While keeping the weights of G fixed, we perform one SGD step on D to minimize the target loss:

$$L_D(\mathbf{X}, \mathbf{Y}) = L_{\text{adv}}^D\left(\overline{\mathbf{Y}}, \widehat{\mathbf{Y}}\right) = L_{\text{sce}}\left(D\left(\widehat{\mathbf{Y}}\right), 0\right) + L_{\text{sce}}\left(D\left(\overline{\mathbf{Y}}\right), 1\right). \tag{7}$$

We train the generator and discriminator iteratively. The entire process is summarized in Algorithm 1, with minibatches of size K.

(1) Set the learning rates ρ_D and ρ_G, and parameters λ_{adv}, λ_p, λ_{dpl};
(2) Initialize weights W_D and W_G.
(3) **while** not converged **do**
(4) **Update the generator** G:
(5) Get K new data samples $(\mathbf{X}^{(1)}, \mathbf{Y}^{(1)})$, $(\mathbf{X}^{(2)}, \mathbf{Y}^{(2)})$,..., $(\mathbf{X}^{(K)}, \mathbf{Y}^{(K)})$
(6) $W_G = W_G - \rho_G \sum_i^K \frac{\partial L_G(\mathbf{X}^{(i)}, \mathbf{Y}^{(i)})}{\partial W_G}$
(7) **Update the discriminator** D:
(8) Get K new data samples $(\mathbf{X}^{(1)}, \mathbf{Y}^{(1)})$, $(\mathbf{X}^{(2)}, \mathbf{Y}^{(2)})$,..., $(\mathbf{X}^{(K)}, \mathbf{Y}^{(K)})$
(9) $W_D = W_D - \rho_D \sum_i^K \frac{\partial L_D(\mathbf{X}^{(i)}, \mathbf{Y}^{(i)})}{\partial W_D}$
(10) **end while**

Algorithm 1: Training GAN-FD.

4. Experiments

4.1. Dataset. Next, we evaluate the performance of the proposed method based on the China stock market, ranging from January 1, 2016, to December 31, 2016. There are totally 244 trading days and each day contains 242-minute intervals, corresponding to 59048 time points. These stocks selected for the experiment should conform to three criteria: first, they should be the constituent stock of *CSI* 300 (the *CSI* 300 is a capitalization-weighted stock market index designed to replicate the performance of 300 stocks traded in the Shanghai and Shenzhen stock exchanges); second, they were not suspended during the period we just mentioned, in case accidental events bring about significant impact on their price and affect forecast results; third, their closing prices in the start time, that is, January 1, 2016, are above 30 to ensure the volatility for high-frequency exchange. This leaves 42 stocks in the sample, which are listed in Table 1. The number of increasing directions and decreasing directions for each stock's closing price per minute is also shown in Table 1, and their numbers are relatively close. The historical data was obtained from the Wind Financial Terminal, produced by Wind Information Inc. (the Wind Financial Terminal can be downloaded from http://www.wind.com.cn).

Many fund managers and investors in the stock market generally accept and use certain criteria for technical indicators as the signal of future market trends [12, 41]. This work selects 13 technical indicators as feature subsets by the review of domain experts and prior researches; that is, the input data $\mathbf{X}$ at each moment (e.g., $\mathbf{X}_T$) consists of 13 basic indicators that can be obtained directly from almost all trading software. These basic indicators are listed in Table 2, and their parameters are using the default value of the Wind Financial Terminal. As mentioned above, $\mathbf{Y}$ is defined as the closing price at each moment.

Most of the related articles use the traditional data partitioning method; that is, the entire dataset is directly split into training set and testing set [12, 22, 40, 42]. However,

Table 1: The sample stocks and their number of increasing directions and decreasing directions.

ID	Stock code	Increase	Decrease
1	000156.SZ	28927	30120
2	000432.SZ	28879	30168
3	000783.SZ	28310	30737
4	000938.SZ	28537	30510
5	000963.SZ	29192	29855
6	002007.SZ	28933	30114
7	002027.SZ	28623	30424
8	002153.SZ	28566	30481
9	002183.SZ	28795	30252
10	002195.SZ	28861	30186
11	002241.SZ	29084	29963
12	002241.SZ	28737	30310
13	002252.SZ	28696	30351
14	002292.SZ	28385	30662
15	002304.SZ	28914	30133
16	002415.SZ	29036	30011
17	002456.SZ	28671	30376
18	002475.SZ	28837	30210
19	002594.SZ	28525	30522
20	300017.SZ	28528	30519
21	300024.SZ	28411	30636
22	300072.SZ	28884	30163
23	300124.SZ	28632	30415
24	300146.SZ	29137	29910
25	600038.SH	28428	30619
26	600085.SH	28856	30191
27	600118.SH	28456	30591
28	600150.SH	28537	30510
29	600332.SH	29174	29873
30	600340.SH	28773	30274
31	600519.SH	29118	29929
32	600535.SH	29013	30034
33	600570.SH	28053	30994
34	600588.SH	28483	30564
35	600685.SH	28627	30420
36	600718.SH	28881	30166
37	600754.SH	28307	30740
38	600783.SH	28680	30367
39	601318.SH	28979	30068
40	601336.SH	28643	30404
41	601888.SH	28919	30128
42	603885.SH	28817	30230

the trading style of the stock market changes frequently; for example, investors sometimes prefer stocks with high volatility and sometimes tend to invest in technology stocks. Therefore, we should update the model parameters regularly to adapt to the change of market style. In order to make experiments closer to real transactions, we carry out rolling segmentation on training set and testing set of the experimental data. As Figure 2 shows, in the beginning, we select the first M days as training set, and the next N days play the role of testing set. After the first round of experiments, we roll forward the time window for N days, that is, choosing the $(N+1)$th day to the $(M+N)$th day as training set and the $(M+N+1)$th day to the $(M+2N)$th day as testing set. Repeat until all the data has been experimented. In other words, this N can be regarded as the model update cycle, and M is the size of the corresponding training data.

Table 2: Basic indicators for prediction.

Indicators
Opening price
Maximum price
Minimum price
Trading volume
Turnover
Bias
Bollinger bands
Directional movement index
Exponential moving averages
Stochastic index
Moving averages
MACD
Relative strength index

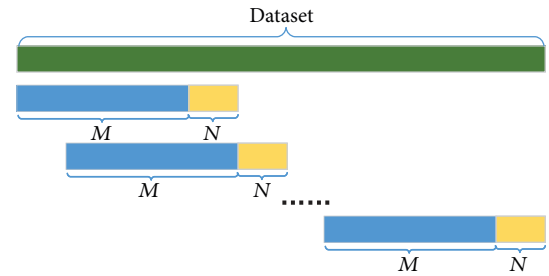

Figure 2: Rolling segmentation on training set and testing set. The green bar represents the entire dataset, the blue bar represents the training set for a round experiment, and the yellow bar represents the corresponding testing set.

4.2. Network Architecture. Given that the LSTM generator takes on the role of prediction and requires more accurate calculations of values than the CNN discriminator, we set the learning rate ρ_G to 0.0004 and ρ_D to 0.02. The LSTM cell in G contains 121 internal (hidden) units and the parameters are initialized following the normal distribution $\mathcal{N}(0,1)$. The architecture of discriminative model D is presented in Table 3. We train GAN-FD with $p = 2$ weighted by $\lambda_{\text{adv}} = \lambda_p = \lambda_{\text{dpl}} = 1$.

4.3. Benchmark Methods. To evaluate the performance of our proposed method, we include three baseline methods for comparison. The first model is ARIMA $(1,1,1)$-GARCH$(1,1)$, a fitted ARIMA model that forecasts future

Table 3: Network architecture of discriminative model D.

Layer	Configuration
Convolution 1	Filter $32 \times 4 \times 1$, strides 2, LReLU
Convolution 2	Filter $64 \times 4 \times 1$, strides 2, LReLU, BN
Convolution 3	Filter $128 \times 4 \times 1$, strides 2, LReLU, BN
FC 1	128, leaky ReLU
FC 2	2, sigmoid

Optimizer: SGD; batch size: 121; iterations: 10000; LReLU slope: 0.01.

Table 4: Summary of RMSRE with different (M, N). These figures are the average values over the 42 stocks.

	$N = 5$			$N = 10$			$N = 20$		
	$M = 10$	$M = 20$	$M = 60$	$M = 10$	$M = 20$	$M = 60$	$M = 10$	$M = 20$	$M = 60$
ARIMA-GARCH	0.0419	0.0406	0.0425	0.0529	0.0529	0.0516	0.0733	0.0657	0.0739
ANN	0.0419	0.0485	0.0531	0.0522	0.0522	0.0510	0.0739	0.0631	0.0631
SVM	0.0512	0.0450	0.0487	0.0539	0.0507	0.0527	0.0616	0.0666	0.0639
GAN-F	0.0151	0.0155	0.0157	0.0300	0.0243	0.0277	0.0326	**0.0313**	0.0299
GAN-D	0.0422	0.0304	0.0503	0.0625	0.0419	0.0405	0.0514	0.0598	0.0420
LSTM-FD	0.0200	0.0194	0.0180	0.0324	0.0230	0.0245	**0.0321**	0.0335	0.0357
GAN-FD	**0.0098**	**0.0079**	**0.0101**	**0.0218**	**0.0111**	**0.0144**	0.0333	0.0323	**0.0296**

values of stock time series and the GARCH model forecasts future volatilities [20]. The second one is artificial neural networks (ANN). The parameter optimization method and model architectural is setting as in [21], except that the input layer node is changed to 13 and the network outputs the predicted value instead of two patterns (0 or 1). The third one is support vector machines (SVM). An RBF kernel is used and the parameter is setting as in [25].

We also inspect our GAN-FD model from several ways. The GAN-F model is using a GAN architectural for minimizing forecast error loss, with $\lambda_{\text{adv}} = \lambda_p = 1$ and $\lambda_{\text{dpl}} = 0$. The GAN-D model is using a GAN architectural for minimizing direction prediction loss, with $\lambda_{\text{adv}} = \lambda_{\text{dpl}} = 1$ and $\lambda_p = 0$. The LSTM-FD model is a LSTM model aiming at minimizing forecast error loss and direction prediction loss, with 121 internal units in LSTM. Obviously, the main difference between LSTM-FD and GAN-FD is the presence of adversarial training.

4.4. Evaluation Metrics. For each stock at each time t, a prediction is made for the next time point $t + 1$ based on a specific method. Assume the total number of time points being tested is T_0; we used the following criteria to evaluate the performance of different models.

(1) Root Mean Squared Relative Error (RMSRE)

$$\text{RMSRE} = \sqrt{\frac{1}{T_0}\sum_{t=1}^{T_0}\left(\frac{\widehat{Y}_{t+1} - Y_{t+1}}{Y_{t+1}}\right)^2}. \tag{8}$$

RMSRE is employed as an indicator for the predictive power or prediction agreement. A low RMSRE indicates that the prediction agrees with the real data (the reason why this paper uses RMSRE instead of RMSE is that RMSRE facilitates a uniform comparison of the results of 42 stocks).

(2) Direction Prediction Accuracy (DPA)

$$\text{DPA} = \frac{100}{T_0}\sum_{t=1}^{T_0} I_t, \tag{9}$$

where

$$I_t = \begin{cases} 1 & \text{if } \left(Y_{t+1} - Y_t\right)\left(\widehat{Y}_{t+1} - Y_t\right) > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{10}$$

DPA measures the percentage of accuracy relating to the series trend. A high DPA promises more winning trades.

4.5. Results. In order to investigate the effect of the model update cycle on the predictive performance, let $M \in \{10, 20, 60\}$ and $N \in \{5, 10, 20\}$. In China stock exchange market, $\{5, 10, 20, 60\}$ days represent one week, two weeks, one month, and one quarter.

Tables 4 and 5 show the average values of RMSRE and DPA with different (M, N). The numbers clearly indicate that GAN-FD and its related methods perform better than three baseline methods in terms of RMSRE and DPA. This targeted method GAN-F brings some improvement in RMSRE, but it does not outperform three baseline methods in DPA. Contrary to GAN-F, GAN-D achieves better results in DPA but failed in RMSRE. LSTM-FD improves the results, since it combines forecast error loss with direction prediction loss for training. Finally the combination of the forecast error loss, direction prediction loss, and adversarial training, that is, GAN-FD, achieves the best RMSRE and DPA in the majority of scenarios.

Let us take a look at the effects of different (M, N) on the experiment. GAN-FD obtains the maximum average DPA (0.6956) and the minimum average RMSRE (0.0079) when

Table 5: Summary of DPA with different (M, N). These figures are the average values over the 42 stocks.

	$N = 5$			$N = 10$			$N = 20$		
	$M = 10$	$M = 20$	$M = 60$	$M = 10$	$M = 20$	$M = 60$	$M = 10$	$M = 20$	$M = 60$
ARIMA-GARCH	0.5464	0.5479	0.5264	0.5280	0.5315	0.5245	0.5007	0.5214	0.5296
ANN	0.5456	0.5978	0.5738	0.5473	0.5629	0.5575	0.5205	0.5280	0.5394
SVM	0.5715	0.5490	0.5839	0.5377	0.5514	0.5576	0.5147	0.5144	0.5318
GAN-F	0.5347	0.5507	0.5281	0.4930	0.5115	0.5265	0.4880	0.5008	0.5116
GAN-D	0.6220	0.6399	0.6409	0.6117	0.6245	0.6437	0.5217	0.5517	0.5498
LSTM-FD	0.6340	0.6506	0.6509	0.6124	0.6236	0.6256	**0.5546**	**0.5635**	0.5719
GAN-FD	**0.6761**	**0.6956**	**0.6793**	**0.6233**	**0.6651**	**0.6687**	0.5535	0.5583	**0.5753**

Table 6: The number of times about the minimum RMSRE.

	$N = 5$			$N = 10$			$N = 20$		
	$M = 10$	$M = 20$	$M = 60$	$M = 10$	$M = 20$	$M = 60$	$M = 10$	$M = 20$	$M = 60$
ARIMA-GARCH	0	0	0	0	0	0	0	0	0
ANN	0	0	0	0	0	0	0	0	0
SVM	0	0	0	0	0	0	0	0	0
GAN-F	11	2	4	6	6	4	13	**18**	11
GAN-D	0	0	0	0	0	0	2	0	7
LSTM-FD	0	0	5	6	3	3	**16**	10	5
GAN-FD	**31**	**40**	**33**	**30**	**33**	**35**	11	14	**19**

Table 7: The number of times about the maximum DPA.

	$N = 5$			$N = 10$			$N = 20$		
	$M = 10$	$M = 20$	$M = 60$	$M = 10$	$M = 20$	$M = 60$	$M = 10$	$M = 20$	$M = 60$
ARIMA-GARCH	0	0	0	0	0	0	0	6	5
ANN	0	0	0	0	0	0	2	0	3
SVM	0	0	0	0	0	0	3	1	1
GAN-F	0	0	0	0	0	0	0	0	0
GAN-D	0	1	6	5	1	7	2	6	2
LSTM-FD	1	3	4	11	4	0	**18**	**15**	12
GAN-FD	**41**	**38**	**32**	**26**	**37**	**35**	17	14	**19**

(M, N) is (20, 5). It is interesting to note that all these methods work better when N is 5 than when N is 10 or 20, with smaller RMSRE and higher DPA. This implies that very short-term trends are best for predicting the next minute's price. Therefore, a shorter model update cycle (e.g., N is 5) is preferred. On the other hand, for the same N, different M will bring about some changes to the prediction results. From the experimental results, we suggest that M should take the value greater than N. This makes intuitive sense. If the training sample is inadequate, it would fail to train the model, especially in the volatile stock markets. We should also notice that when the training set is small while the testing set is large (i.e., (M, N) is (10, 20)), most of these methods perform the worst, and the DPA of these methods are no better than random guessing (i.e., 50%).

Table 6 shows the number of times for each method to achieve the minimum RMSRE over the 42 stocks. It is noticeable that the results of these three baseline methods are all zero. GAN-FD with its related methods is obviously better than these three baseline methods in RMSRE. Meanwhile, GAN-FD obtains the minimum RMSRE 246 times, accounting for 65.08% in these 378 scenarios (42 stocks and 9 groups (M, N)). The best performance appeared when (M, N) is (20, 5), with 40 stocks' minimum RMSRE coming from GAN-FD.

Table 7 shows the number of times for each method to achieve the maximum DPA over the 42 stocks. Compared with the other six methods, GAN-FD achieves the maximum DPA 269 times, accounting for 71.16% in all scenarios. When (M, N) is (10, 5), the maximum DPA of 41 stocks in all 42 stocks comes from GAN-FD. Even when (M, N) is (20, 20), that is, the worst performance of GAN-FD cases, GAN-FD still obtains maximum DPA in 14 stocks. From the above analyses, the performance of the GAN-FD is significantly better than the other six ways.

The results of each representation are reported in Figures 3–11. We just focus on GAN-FD. As shown in Figures 3–5, the DPA of GAN-FD ranges around 64.59%–72.24% when N is 5, and it slumps to 52.01%–62.71% when N is 20, which is presented in Figures 9–11. When N is 5, the RMSRE of GAN-FD over the 42 stocks varies between 0.48% and 1.49%,

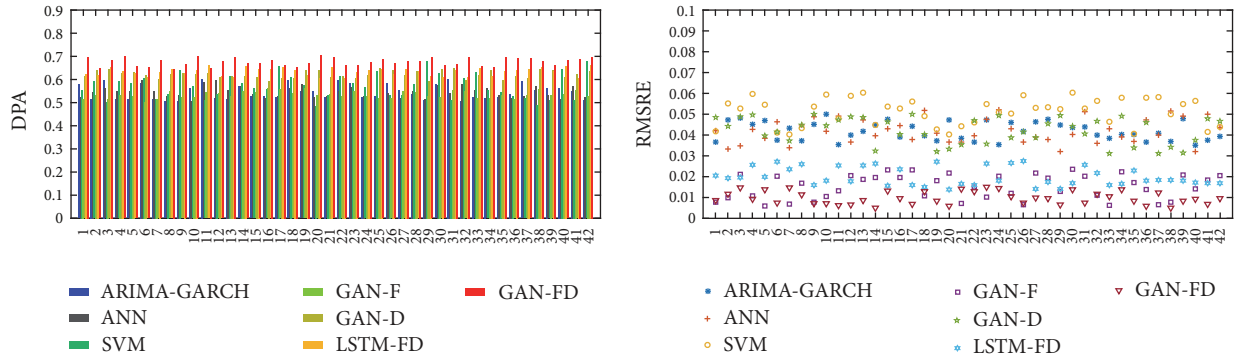

Figure 3: DPA and RMSRE of each stock when (M, N) is (10, 5) and x-axis represents the stock ID.

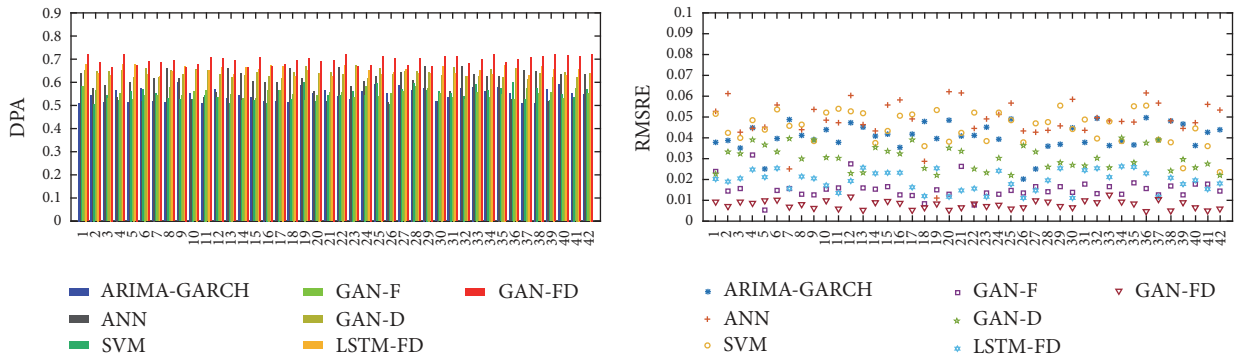

Figure 4: DPA and RMSRE of each stock when (M, N) is (20, 5) and x-axis represents the stock ID.

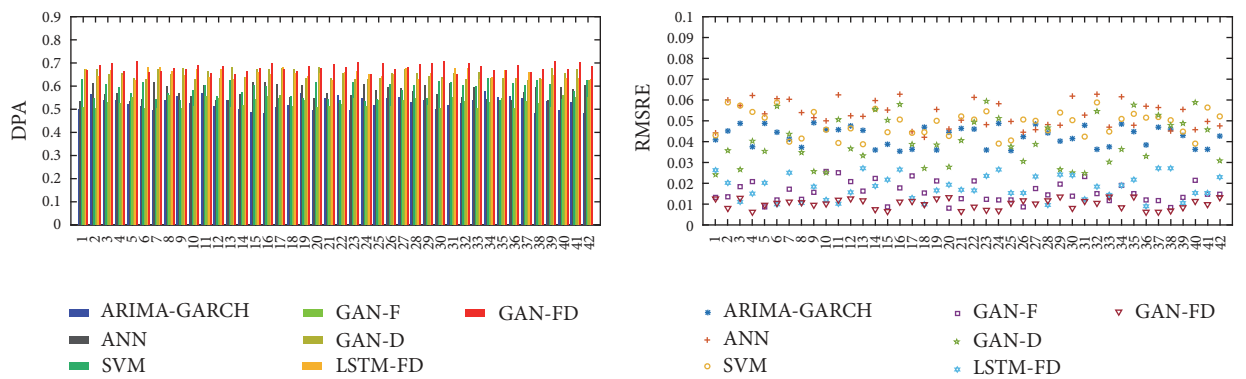

Figure 5: DPA and RMSRE of each stock when (M, N) is (60, 5) and x-axis represents the stock ID.

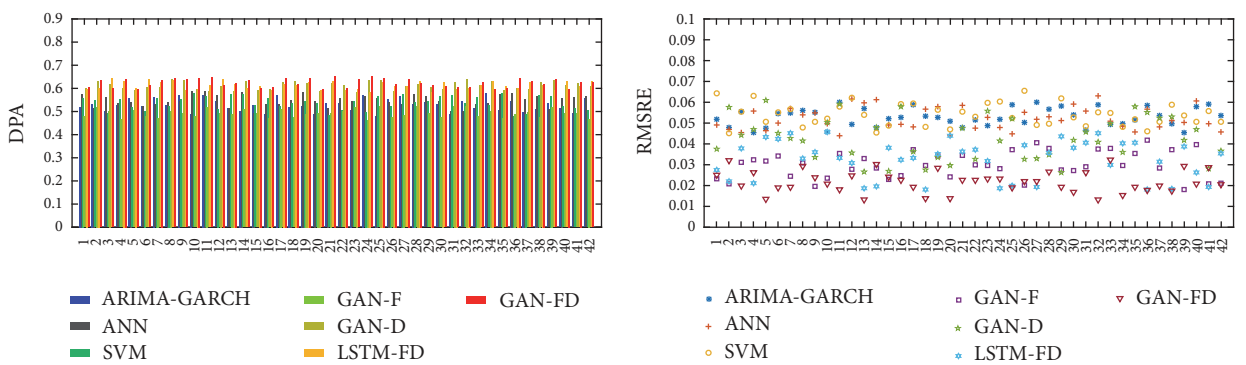

Figure 6: DPA and RMSRE of each stock when (M, N) is (10, 10) and x-axis represents the stock ID.

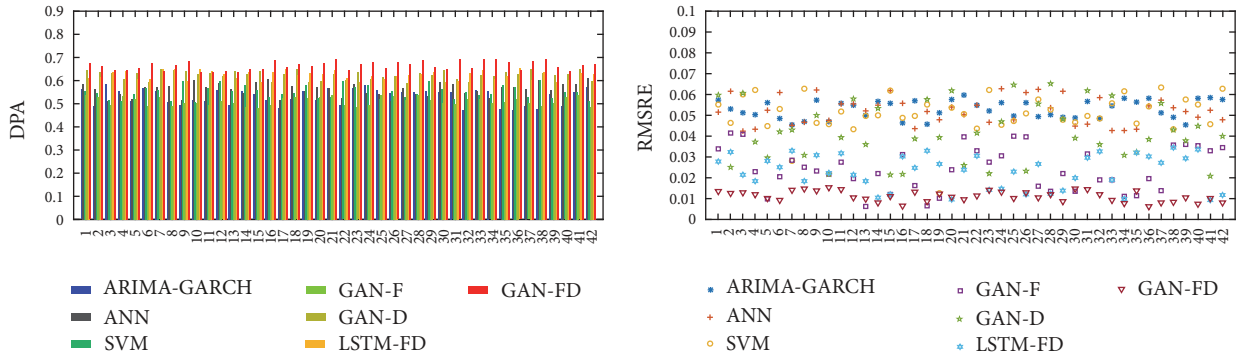

Figure 7: DPA and RMSRE of each stock when (M, N) is (20, 10) and x-axis represents the stock ID.

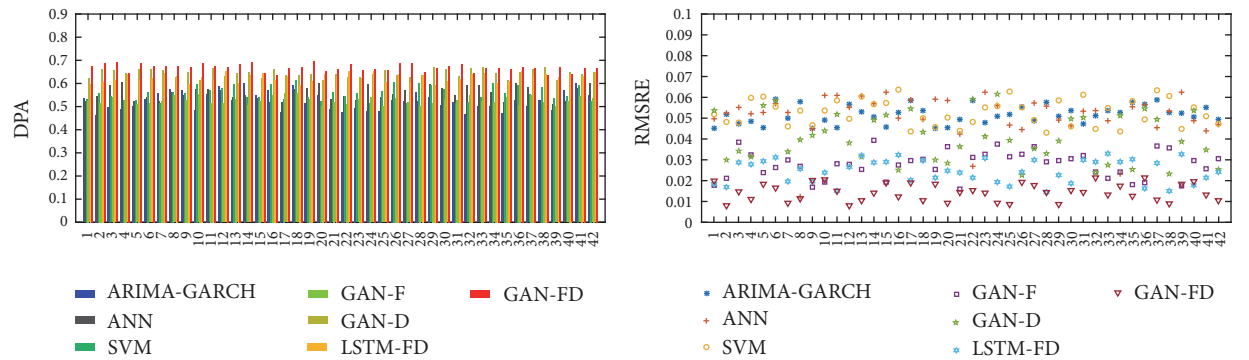

Figure 8: DPA and RMSRE of each stock when (M, N) is (60, 10) and x-axis represents the stock ID.

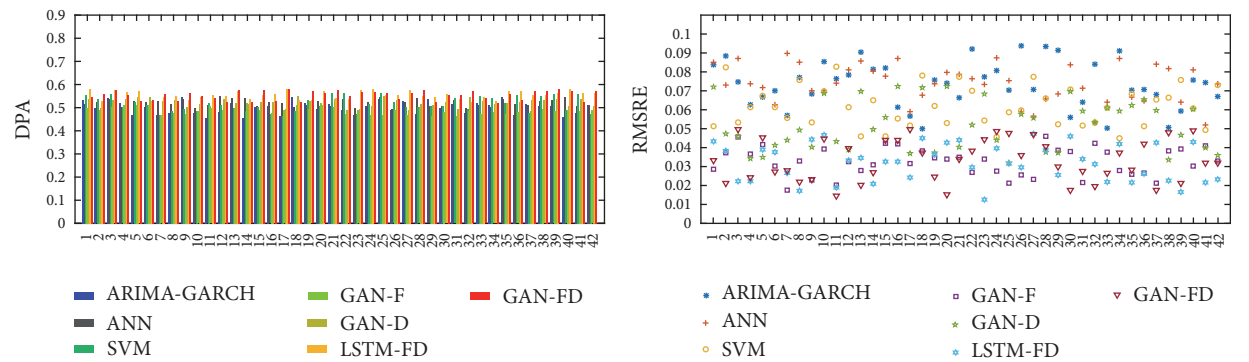

Figure 9: DPA and RMSRE of each stock when (M, N) is (10, 20) and x-axis represents the stock ID.

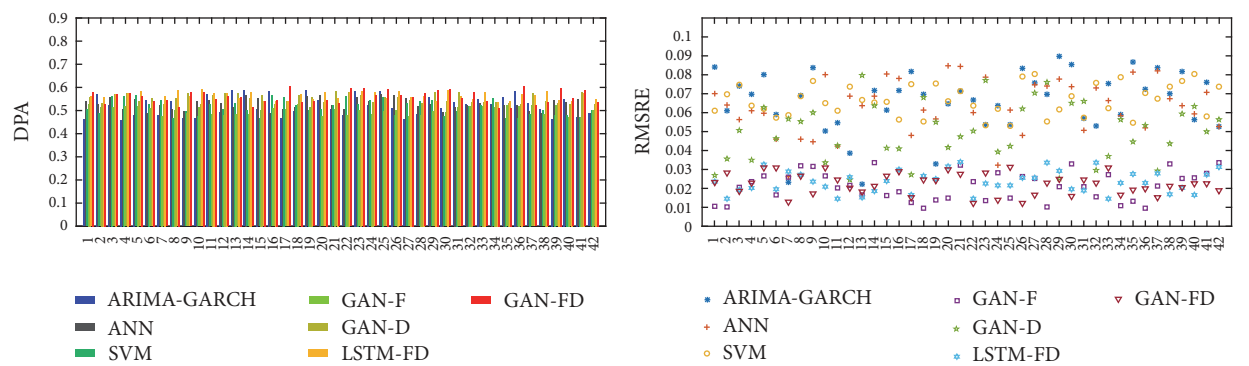

Figure 10: DPA and RMSRE of each stock when (M, N) is (20, 20) and x-axis represents the stock ID.

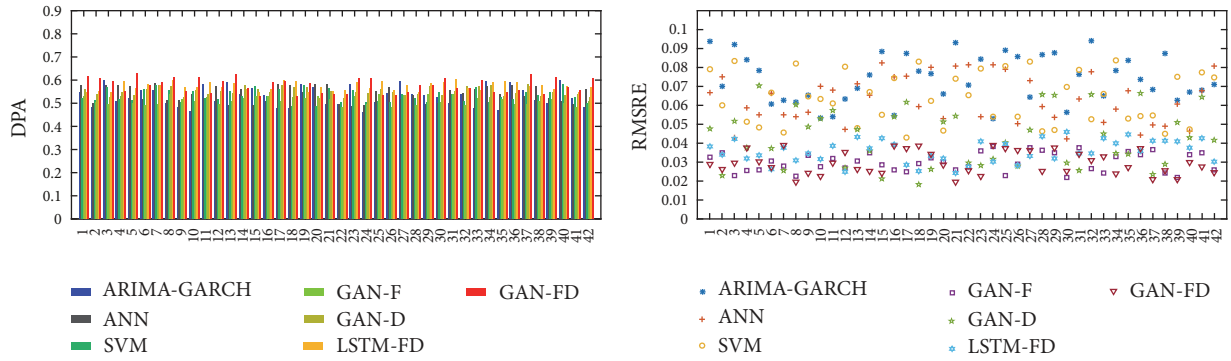

Figure 11: DPA and RMSRE of each stock when (M, N) is (60, 20) and x-axis represents the stock ID.

which is lower than other six methods in most cases, while the volatility is smaller. However, the RMSRE of GAN-FD increases dramatically and fluctuates violently when N is 20, and it varies between 1.21% and 4.96%. This further shows that we should reduce the model update cycle N and revise the model parameters regularly to adapt to the change of market style.

5. Conclusion

In this paper, we propose an easy-to-use stock forecasting model called GAN-FD, to assist more and more nonfinancial professional ordinary investors making decisions. GAN-FD adopts 13 simple technical indexes as input data to avoid complicated input data preprocessing. Based on the deep learning network, this model achieves prediction ability superior to other benchmark methods by means of adversarial training, minimizing direction prediction loss, and forecast error loss. Moreover, the effects of the model update cycles on the predictive capability are analyzed, and the experimental results show that the smaller model update cycle can obtain better prediction performance. In the future, we will attempt to integrate predictive models under multiscale conditions.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the National Key Research Development Program of China (2017YFB0802800) and the National Natural Science Foundation of China (no. 61473149).

References

[1] R. Al-Hmouz, W. Pedrycz, and A. Balamash, "Description and prediction of time series: a general framework of Granular Computing," *Expert Systems with Applications*, vol. 42, no. 10, pp. 4830–4839, 2015.

[2] S. Barak and M. Modarres, "Developing an approach to evaluate stocks by forecasting effective features with data mining methods," *Expert Systems with Applications*, vol. 42, no. 3, pp. 1325–1339, 2015.

[3] A. Booth, E. Gerding, and F. McGroarty, "Automated trading with performance weighted random forests and seasonality," *Expert Systems with Applications*, vol. 41, no. 8, pp. 3651–3661, 2014.

[4] A. Bagheri, H. Mohammadi Peyhani, and M. Akbari, "Financial forecasting using ANFIS networks with Quantum-behaved Particle Swarm Optimization," *Expert Systems with Applications*, vol. 41, no. 14, pp. 6235–6250, 2014.

[5] Y. Son, D.-J. Noh, and J. Lee, "Forecasting trends of high-frequency KOSPI200 index data using learning classifiers," *Expert Systems with Applications*, vol. 39, no. 14, pp. 11607–11615, 2012.

[6] I. Aldridge and S. Krawciw, "Real-Time Risk: What Investors Should Know About FinTech," in *High-Frequency Trading, and Flash Crashes*, John Wiley & Sons, Inc., Hoboken, NJ, USA, 2017.

[7] F. A. De Oliveira, C. N. Nobre, and L. E. Zárate, "Applying Artificial Neural Networks to prediction of stock price and improvement of the directional prediction index - Case study of PETR4, Petrobras, Brazil," *Expert Systems with Applications*, vol. 40, no. 18, pp. 7596–7606, 2013.

[8] J. H. Niño-Peña and G. J. Hernández-Pérez, "Price direction prediction on high frequency data using deep belief networks," *Communications in Computer and Information Science*, vol. 657, pp. 74–83, 2016.

[9] A. Marszałek and T. Burczyn'ski, "Modeling and forecasting financial time series with ordered fuzzy candlesticks," *Information Sciences*, vol. 273, pp. 144–155, 2014.

[10] X. Li, X. Huang, X. Deng, and S. Zhu, "Enhancing quantitative intra-day stock return prediction by integrating both market news and stock prices information," *Neurocomputing*, vol. 142, pp. 228–238, 2014.

[11] X. Wang, S. Bao, and J. Chen, "High-frequency stock linkage and multi-dimensional stationary processes," *Physica A: Statistical Mechanics and its Applications*, vol. 468, pp. 70–83, 2017.

[12] E. Chong, C. Han, and F. C. Park, "Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies," *Expert Systems with Applications*, vol. 83, pp. 187–205, 2017.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," in *Proceedings of the 28th Annual Conference*

on Neural Information Processing Systems 2014, NIPS 2014, pp. 2672–2680, can, December 2014.

[14] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Transactions on Graphics*, vol. 36, no. 4, article no. 107, 2017.

[15] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, *Semantic segmentation using adversarial networks, arXiv preprint*, arXiv, 1611.08408, 2016, arXiv:1611.08408.

[16] M. Mathieu, C. Couprie, and Y. LeCun, *Deep multi-scale video prediction beyond mean square error, arXiv preprint*, arXiv, 1511.05440, 2015, arXiv:1511.05440.

[17] J. D. Hamilton, *Time Series Analysis*, vol. 2, Princeton University Press, 1994.

[18] R. H. Shumway and D. S. Stoffer, *Time series analysis and its applications*, Springer Texts in Statistics, Springer, New York, NY, USA, 3rd edition, 2011.

[19] P. J. Brockwell and R. A Davis, *TimE Series: Theory and Methods*, Springer Science & Business Media, 2013.

[20] S. Pellegrini, E. Ruiz, and A. Espasa, "Prediction intervals in conditionally heteroscedastic time series with stochastic components," *International Journal of Forecasting*, vol. 27, no. 2, pp. 308–319, 2011.

[21] Y. Kara, M. Acar Boyacioglu, and Ö. K. Baykan, "Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5311–5319, 2011.

[22] M. Ghiassi, J. Skinner, and D. Zimbra, "Twitter brand sentiment analysis: a hybrid system using n-gram analysis and dynamic artificial neural network," *Expert Systems with Applications*, vol. 40, no. 16, pp. 6266–6282, 2013.

[23] M. R. Hassan, "A combination of hidden Markov model and fuzzy model for stock market forecasting," *Neurocomputing*, vol. 72, no. 16-18, pp. 3439–3446, 2009.

[24] W. Huang, Y. Nakamori, and S.-Y. Wang, "Forecasting stock market movement direction with support vector machine," *Computers & Operations Research*, vol. 32, no. 10, pp. 2513–2522, 2005.

[25] A. F., S. Elsir, and H. Faris, "A Comparison between Regression, Artificial Neural Networks and Support Vector Machines for Predicting Stock Market Index," *International Journal of Advanced Research in Artificial Intelligence*, vol. 4, no. 7, 2015.

[26] R. Majhi, G. Panda, G. Sahoo, A. Panda, and A. Choubey, "Prediction of S&P 500 and DJIA stock indices using particle swarm optimization technique," in *Proceedings of the Proceeding of the IEEE Congress on Evolutionary Computation (CEC '08)*, pp. 1276–1282, Hong Kong, China, June 2008.

[27] S. S. Appadoo, *Pricing Financial Derivatives with Fuzzy Algebraic Models: A Theoretical and Computational Approach [Ph.D. thesis]*, University of Manitoba, Winnipeg, 2006.

[28] A. Thavaneswaran, K. Thiagarajah, and S. S. Appadoo, "Fuzzy coefficient volatility (FCV) models with applications," *Mathematical and Computer Modelling*, vol. 45, no. 7-8, pp. 777–786, 2007.

[29] C. Carlsson and R. Fullér, "On possibilistic mean value and variance of fuzzy numbers," *Fuzzy Sets and Systems*, vol. 122, no. 2, pp. 315–326, 2001.

[30] A. Thavaneswaran, S. S. Appadoo, and A. Paseka, "Weighted possibilistic moments of fuzzy numbers with applications to GARCH modeling and option pricing," *Mathematical and Computer Modelling*, vol. 49, no. 1-2, pp. 352–368, 2009.

[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[32] A. M. Rather, A. Agarwal, and V. N. Sastry, "Recurrent neural network and a hybrid model for prediction of stock returns," *Expert Systems with Applications*, vol. 42, no. 6, pp. 3234–3241, 2015.

[33] R. D. A. Araújo, A. L. I. Oliveira, and S. Meira, "A hybrid model for high-frequency stock market forecasting," *Expert Systems with Applications*, vol. 42, no. 8, pp. 4081–4096, 2015.

[34] H. Chen, K. Xiao, J. Sun, and S. Wu, "A double-layer neural network framework for high-frequency forecasting," *ACM Transactions on Management Information Systems (TMIS)*, vol. 7, no. 4, article no. 11, 2017.

[35] A. Yoshihara, K. Fujikawa, K. Seki, and K. Uehara, "Predicting Stock Market Trends by Recurrent Deep Neural Networks," in *PRICAI 2014: Trends in Artificial Intelligence*, vol. 8862 of *Lecture Notes in Computer Science*, pp. 759–769, Springer International Publishing, Cham, 2014.

[36] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Deep learning for event-driven stock prediction," in *Proceedings of the 24th International Joint Conference on Artificial Intelligence, IJCAI 2015*, pp. 2327–2333, arg, July 2015.

[37] Z. Zhou, J. Zhao, and K. Xu, "Can online emotions predict the stock market in China?" in *Proceedings of the International Conference on Web Information Systems Engineering*, pp. 328–342, 2016.

[38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[39] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[40] A. Arévalo, J. Niño, G. Hernández, and J. Sandoval, "High-frequency trading strategy based on deep neural networks," in *Proceedings of the International conference on intelligent computing*, pp. 424–436, 2016.

[41] K.-J. Kim, "Financial time series forecasting using support vector machines," *Neurocomputing*, vol. 55, no. 1-2, pp. 307–319, 2003.

[42] S. Madge, Predicting Stock Price Direction using Support Vector Machines, Independent Work Report Spring, 2015.