Hindawi
Scientific Programming
Volume 2021, Article ID 7321671, 12 pages
https://doi.org/10.1155/2021/7321671

Research Article

Research on the Application of Generative Adversarial Networks in the Generation of Stock Market Forecast Trend Images

Daiyou Xiao

School of Finance, Central University of Finance and Economics, Beijing 100081, China

Correspondence should be addressed to Daiyou Xiao; 2019110026@email.cufe.edu.cn

Received 20 October 2021; Revised 9 November 2021; Accepted 16 November 2021; Published 17 December 2021

Academic Editor: Bai Yuan Ding

Investors make capital investment by buying stocks and expect to get a certain income from the stock market. When buying stocks, they need to draw up investment plans based on various information such as stock market historical transaction data and related news data of listed companies and collect and analyze these data. The data are relatively cumbersome and require a lot of time and effort. If you only rely on subjective analysis, the reference factors are often not comprehensive enough. At the same time, Internet social media, such as the speech in stock forums, also affect the judgment and behavior of investors, and investor sentiment will have a positive or negative effect on the stock market. This has an impact on the trend of stock prices. Therefore, this article proposes a stock market prediction model that uses data preprocessing technology based on past stock market transaction data to establish a stock market prediction model, and secondly, an image description generation model based on a generative confrontation network is designed. The model includes a generator and a discriminator. A time-varying preattention mechanism is proposed in the generator. This mechanism allows each image feature to pay attention to the image features of other stock markets to predict stock market trends so that the decoder can better understand the relational information in the image. The discriminator is based on the recurrent neural network and considers the degree of matching between the input sentence and the 4 reference sentences and the image features. Experiments show that the accuracy of the model is higher than that of the stock pretrend forecast model based on historical data, which proves the effectiveness of the data used in this paper in the stock price trend forecast.

1. Introduction

Generative Adversarial Network (GAN) is a new neural network architecture born in recent years [1]. A traditional neural network is usually composed of an input layer, several hidden layers, and an output layer. In the context of the generation task, the output layer directly outputs the generated results. For the image description generation task of stock market prediction trend, the goal of generation is a highly flexible language description. Compared with the maximum likelihood estimation, the structure of the generation confrontation network can provide more flexible and diverse results [2]. In addition to the generation confrontation, the network can overcome the exposure bias problem caused by the maximum likelihood estimation in the sequence generation task. Therefore, the stock market trend prediction image description model based on the generation of the confrontation network has a good research value.

2. Related Work

Most of the current research papers on text-generated images are based on Generative Adversarial Networks. In 2016, Reed [3, 4] and others first proposed the problem of generating images based on GAN text descriptions. The article proposed to extract effective semantic information from text descriptions, and the method of letting the computer recognize, by generating an adversarial neural network, produces an image that is more consistent with the text content. Subsequently, Reed [5] and others proposed a Generative Adversarial What-Where Network (GAWWN). The article proposed a Generative Adversarial What-Where Network model that takes text information, position, and posture as conditions. This method enriches the details of image generation but increases the difficulty of the training process. Zhang [6] and others proposed the Stacked Generative Adversarial Network Architecture (StackGAN). The article proposed the use of two Generative Adversarial Networks to achieve text-to-image generation because when the generator is just a simple upsampling method, it cannot improve the quality of the generated samples, so StackGAN's task of generating text images is divided into two stages. The first stage generates images with lower accuracy. This stage is mainly used to generate basic information such as the outline, background, and color of the image. In the second stage, the output samples of the first stage are used as input to generate high-quality high-definition images, and the detailed information lost in the generated samples is complemented. Zhang [7–9] and others proposed the StackGAN++ model on the basis of previous research. In the paper, the Generative Adversarial Network was designed to be similar to a tree structure, and multiple generative neural networks and adversarial neural networks were trained in parallel, which effectively reduced the model's training time.

This paper designs a stock market trend prediction stock market trend image description generation model based on a generative confrontation network. The model includes a generator and a discriminator. The generator proposes a time-varying preattention mechanism, which allows each stock market trend to predict the stock market trend. Image features pay attention to other image features to introduce the relationship between different image regions so that the decoder can better understand the relationship information in the stock market trend prediction stock market trend image. The discriminator takes the recurrent neural network as the main body and considers the input sentence and the degree of matching between the reference sentences and the image features of the stock market trend.

3. Related Theoretical Methods for Generating Confrontation Networks

Generative Adversarial Network is a neural network that uses unsupervised learning methods. It does not require complex processing of input data and is very suitable for processing data with large amounts of data and incomplete information. The basic structure of the generative confrontation network includes a generator and a discriminator. The generator is generally composed of a multilayer neural network, which reconstructs the data obtained by random sampling in a certain way to be consistent with the sample dimension. When the characteristics of the generated stock market trend prediction image need to be restricted, a control vector is added to the data and input to the judgment. In the device, finally update the network with the judgment result, as shown in Figure 1 [10–12].

GAWWN uses a pair of generators and discriminators to build a model, introduces bounding box conditions and key point information to control the pose and position of the subject in the composite image, and generates a 128×128 resolution image. Figure 2 shows a typical GAN framework in the field of image generation.

4. Generating the Model of Stock Market Forecast Trend Image Based on the Generative Confrontation Network

This section discusses the specific structure of the stock market prediction trend image generation model based on the generative confrontation network and the training method of the network. Figure 3 shows the overall architecture of the model. This section is divided into three parts: generating network, discriminating network, and training method.

4.1. Generating Network

4.1.1. Encoder. In the model, the encoder uses a deep residual neural network (Resnet). The residual neural network is an improvement of the traditional neural network. After the traditional neural network reaches a certain depth, it is difficult to increase the depth to obtain performance improvement. The residual neural network greatly deepens the depth of the network by increasing the residual connection. For the encoder, in terms of tasks, the residual network can also better extract feature vectors from the data. The principle and structure of the residual network are explained as follows [13–15].

The starting point of the residual network is to solve the problem that the single-layer neural network is difficult to fit the unit mapping. Assuming that the input of the network is X and the fitting target is $H(X)$, experiments show that, under the condition of $H(X) = X$, fitting is very difficult, so the residual network introduces a new fitting target $F(X) = H(X) - X$. At this time, the fitting goal of $H(X) = X$ is equivalent to that of $F(X) = 0$. This fitting is easy to achieve, so the fitting goal can be obtained as $F(X) = H(X) + X$, as shown in Figure 4(a).

In the figure, the first linear transformation and activation function is equivalent to a layer of the neural network, and the function of the subsequent linear transformation is to transform the dimension to be the same as the input. The main contribution of the residual network is to increase a path from input to output. The learning of the original objective function is transformed into the learning of the residual. The addition of residuals allows the neural network to train normally after reaching a depth of tens of layers and achieve better performance than networks with fewer layers. In practical applications, in order to further deepen the network depth, Resnet also uses a structure called a bottleneck building block [16], which is shown in Figure 4(b).

Each box in the figure represents a convolution operation. The input of the block is a piece of data with 256 channels, after a convolution operation with a convolution window of 1×1 and a convolution kernel of 64, the function of this step is to reduce the dimensionality of the data, and the data continues to pass through a convolution window to 3×3. The convolution kernel is a convolution operation of

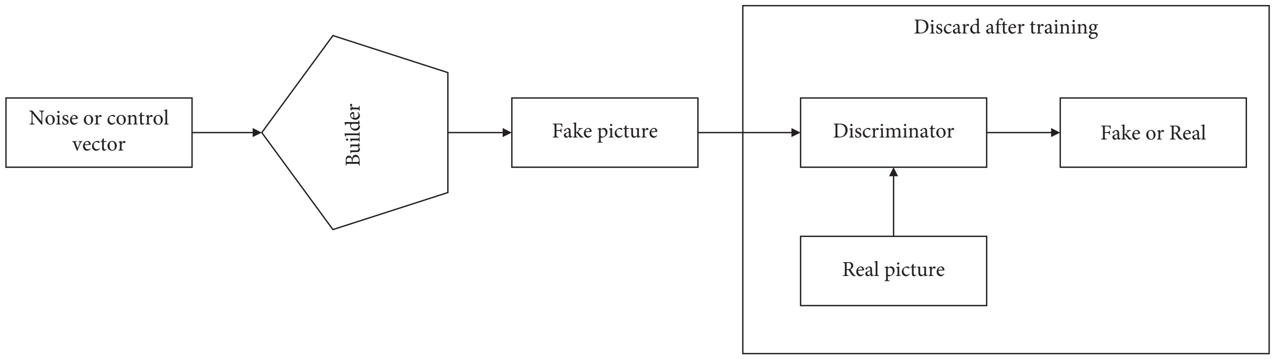

Figure 1: The structure diagram of the generated confrontation network.

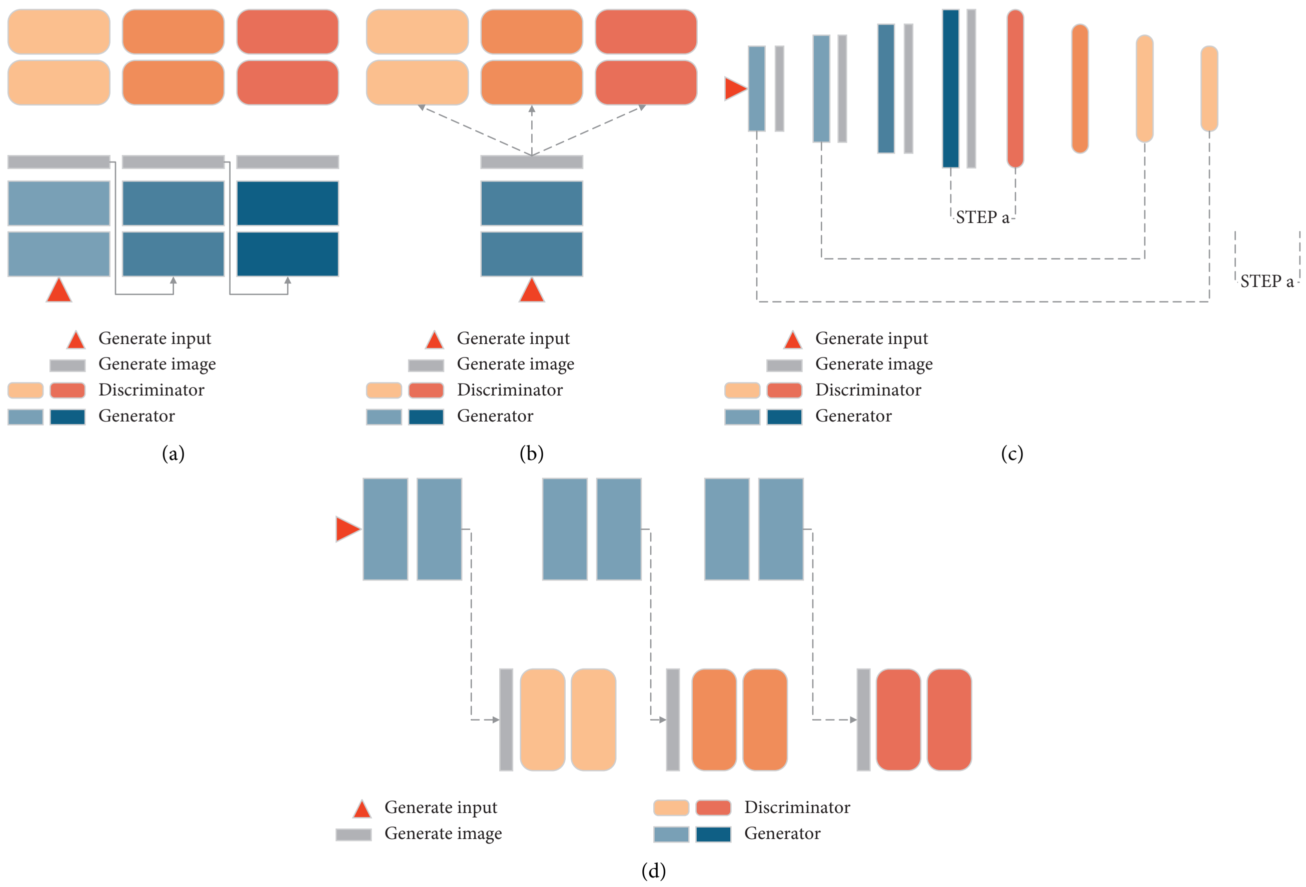

Figure 2: A typical GAN framework in the field of image generation. (a) Stacked. (b) Multidiscriminator. (c) Progressive. (d) Nested.

64, and finally, after a convolution operation with a convolution window of 1×1 and a convolution kernel of 256, the data dimension is restored to 256 channels, and the data after the restored dimension is the same as the original data. Add the restored dimension data to the original data, that is, convert the original pixel feature data to more specific and physical features and then solve the classification This block structure is the basic structure in Resnet, and extremely deep networks are stacked by this block. The experiment in this paper uses the 101-layer Resnet, denoted as Resnet-101, and its complete structure is shown in Table 1.

First, input the historical raw stock market data to do preliminary convolution and pooling. After the data passes through multiple bottleneck building blocks, each matrix in the table corresponds to the above basic block structure. Except for the number and step size of the convolution kernel, the others are consistent with the above example. After the matrix, $\times N$ represents stacking N of the same structure. The output after all block operations is $7 \times 7 \times 2048$, as 49 2048-dimensional stock market trend image spatial feature vectors, and each feature vector is linearly transformed into D-dimensional feature vectors used by the model [17–19].

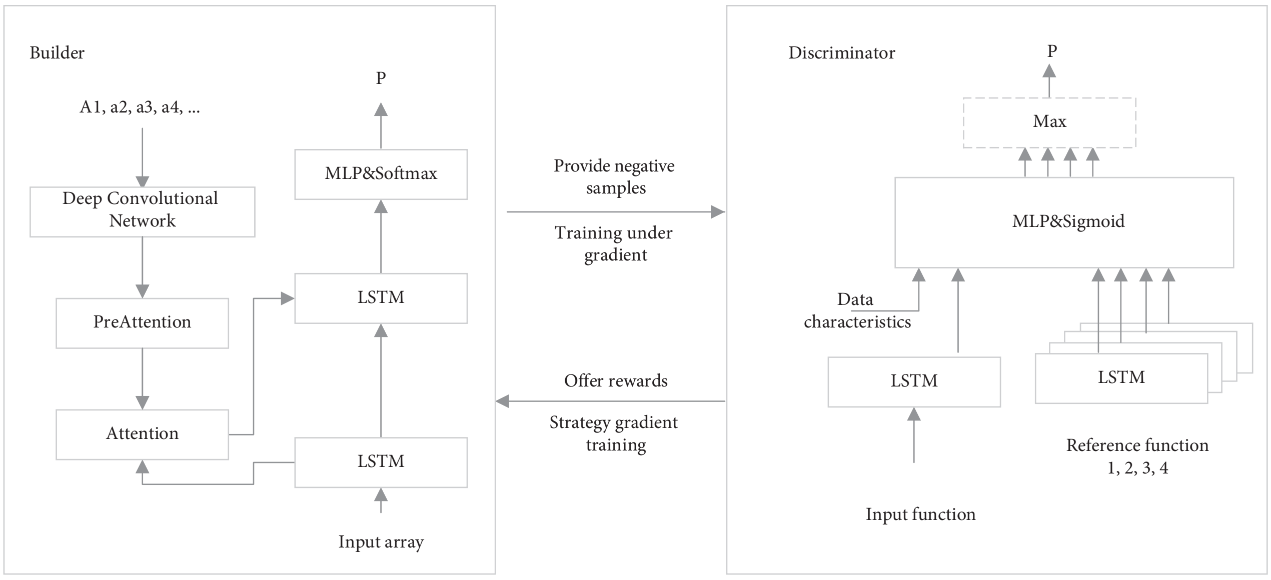

FIGURE 3: The overall architecture of the stock market prediction trend image generation model based on the generative confrontation network.

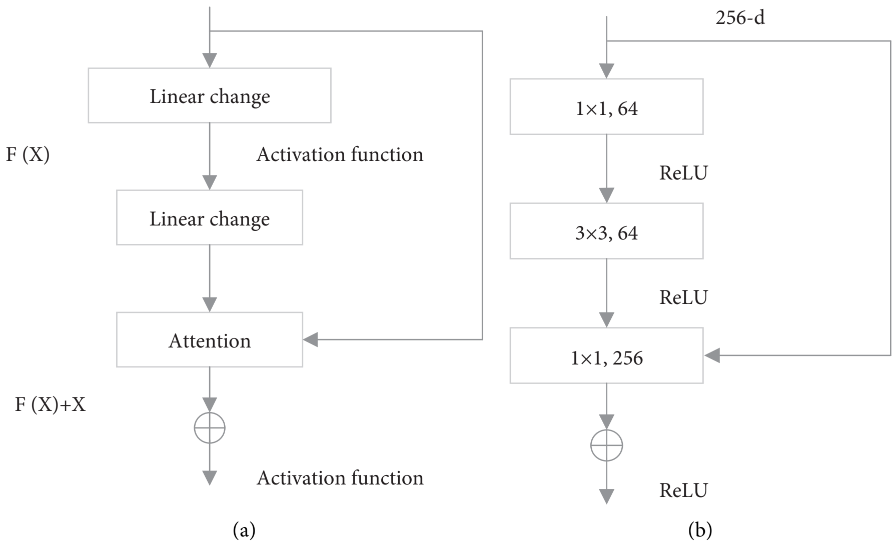

FIGURE 4: Schematic diagram of residual neural network. (a) Schematic diagram of residual learning. (b) Bottleneck building block structure.

4.1.2. Time-Varying Preattention Mechanism. The attention mechanisms used in previous research work, such as commonly used soft attention and top-down attention, directly use the data spatial features extracted from the deep convolutional network, and the final attention vector is expressed in the form of the weighted sum of all feature vectors. However, some studies have pointed out that although convolutional neural networks and fully connected networks have achieved excellent results in many tasks of predicting stock market trend images, even simple reasoning about relationships in stock market trend images is difficult for them. Relational reasoning ability is very important in the image description task of predicting stock market trends. A simple image description of predicting stock market trends may only contain entities and attributes, but a slightly more complicated description often involves the association between entities or attributes. The lack of relational reasoning capabilities of convolutional neural networks means that the encoder only extracts the local spatial information of the data but does not include the correlation between these pieces of information. In this case, if the attention mechanism is only a simple image feature weighted sum, it is difficult to fully express the relationship in the image, and considering that the image information predicting the stock market trend that the decoder can access only comes from the attention mechanism, the decoder may not be able to accurately understand the relationship in the image predicting the stock market trend.

There are rich relationships in predicting stock market trend images. The relationship between any two data trends is intricate. Such a large number of relationships lead to a large number of relationships in the aggregate feature vector,

Table 1: Complete structure of Resnet-101.

Operate	Output dimension
Resize and crop	$224 \times 224 \times 3$
7×7 convolution, convolution kernel 64, step size 2	$112 \times 112 \times 64$
3×3 max pooling, step size 2	$56 \times 56 \times 64$
$\begin{bmatrix} 1 \times 1 & 64 \\ 3 \times 3 & 64 \\ 1 \times 1 & 256 \end{bmatrix} \times 3$	$56 \times 56 \times 256$
$\begin{bmatrix} 1 \times 1 & 128 \\ 3 \times 3 & 128 \\ 1 \times 1 & 512 \end{bmatrix} \times 4$	$28 \times 28 \times 512$
$\begin{bmatrix} 1 \times 1 & 256 \\ 3 \times 3 & 256 \\ 1 \times 1 & 1024 \end{bmatrix} \times 23$	$14 \times 14 \times 1024$
$\begin{bmatrix} 1 \times 1 & 512 \\ 3 \times 3 & 512 \\ 1 \times 1 & 2048 \end{bmatrix} \times 3$	$7 \times 7 \times 2048$

Average pooling, 1000-dimensional fully connected network, softmax 1000.

which makes it difficult for the decoder to determine if the required relationship is not limited. What relationship is important, so it is necessary to limit the relationship considered in the preattention mechanism. What kind of relationship is required cannot be determined in advance. It should be determined by the sentence that you want to describe or has been described. That is, a semantic context is needed to help the preattention mechanism find the required relationship. Here, the semantic context vector is represented as h^2_{t-1}, and its source will be introduced in the next section. This context vector changes with time because every moment a new word is generated, which affects the content to be described next and the corresponding preattention mechanism aggregate feature vector [20, 21].

It is also time-varying, so it is called time-dependent preattention (TDPA). After introducing the semantic context h^2_{i-1}, the formula is as follows:

$$\begin{aligned} a_{ti} &= w_p^T \tanh\left(W_{vp} + \left(W_{sp} v_i + W_{hp} h^2_{i-1}\right) 1^T\right), \\ \alpha_{ti} &= \operatorname{softmax}\left(a_{ti}\right), \\ v'_{ti} &= \sum_{j=1}^{L} \alpha_{tij} v_j, \end{aligned} \tag{1}$$

where $1 \in R^L$ is an L-dimensional all-one vector and $W_{vp} \in R^{L\times D}$, $W_{sp} \in R^{L\times D}$, $w_{hp} \in R^{L\times D}$, and $w_p \in R^L$ are parameters to be learned.

v'_{ti} contains the relationship information between v_j and other feature vectors. Connect it with the image feature vector v_i and describe the features of this area at the current moment through a fully connected network:

$$v_{ti} = \operatorname{MLP}\left(\left[v'_{ti};\ v_i\right]\right). \tag{2}$$

Figure 5 shows the above process.

4.1.3. Decoder. Inspired by some previous research work using hierarchical LSTM (hierarchical LSTM), this paper designs a hierarchical LSTM combined with a time-varying

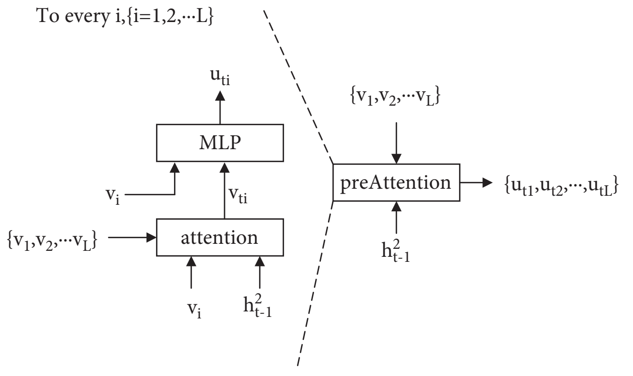

Figure 5: The structure of the time-varying preattention mechanism.

preattention mechanism as a decoder. The advantage of hierarchical LSTM is that it has multiple LSTM units, and the special LSTM units can be used to encode the information required by the attention mechanism, which helps to better play the role of the time-varying preattention mechanism. In the overall framework, the decoder contains the following parts: a preattention module, which is responsible for generating aggregate feature vectors; an attention module, which helps LSTM pay attention to the aggregate feature vectors; 2 LSTM units, one of which is used to encode the attention module , the required information, one for text generation; MLP and softmax layers Figure 6 shows the complete decoder framework.

The following describes the specific work of each module one by one.

The bottom LSTM: the function of this LSTM unit is to encode the information required by the attention mechanism from the memory of the network, the current input, and the global image features, so its input is the hidden state of the top LSTM at the previous moment h^2_{t-1}, the currently input word vector x_t, and the average image feature $\overline{v} = 1/L\Sigma_{i=1}^{L} v_i$, these vectors are connected and input into the

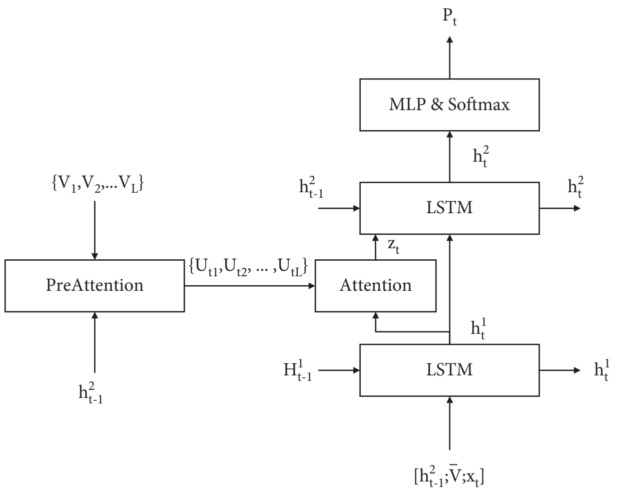

Figure 6: Decoding architecture.

bottom LSTM, and the bottom LSTM combines its own memory to get the output.

Preattention layer: this layer includes the use of preattention mechanism to generate aggregated and predicted stock market trend image features and the use of the underlying LSTM output information to guide the generation of attention to aggregate predicted stock market trend image features. The hidden state h_{t-1}^2 of the top LSTM at the previous moment encodes all the semantic information generated before, which is used in preattention, that is, in the previous section:

$$U_t = \text{preattention}\left(V, h_{i-1}^2\right). \tag{3}$$

The output U_t contains the relational information and the image features of aggregated and predicted stock market trends. Then another attention module will notice the local special image feature U_t according to the hidden state h_t^1 of the underlying LSTM, that is, do the following calculation:

$$\begin{aligned} a_i &= w_a^T \tanh\left(W_{ua}U_t + \left(W_{ha}h_t^2\right)1^T\right),\\ \alpha_t &= \text{softmax}\left(a_t\right),\\ z_t &= \sum_{i=1}^{L} \alpha_{ti}U_{ti}. \end{aligned} \tag{4}$$

Among them, $1 \in R^L$ is an L-dimensional all-one vector, $W_{ua} \in R^{L\times D}$, $W_{ha} \in R^{L\times M}$, and $w_a \in R^L$ are the parameters to be learned, and z_t is on U_t attention vector.

The top LSTM: the function of this LSTM unit is to integrate the attention vector z_t and the output information h_i^1 of the bottom LSTM and combine its own memory to generate the information of the next word:

$$h_t^2 = \text{LSTM}\left(\left[h_t^1; z_t\right], h_{t-1}^2\right), \tag{5}$$

where $h_t^2 \in R^M$ represents the hidden state of the top LSTM at time t.

MLP and softmax layers: the hidden state h_t^2 of the top LSTM becomes a probability distribution through a single-layer fully connected network and softmax function.

$$P_t = \text{softmax}\left(\text{MLP}\left(h_t^2\right)\right). \tag{6}$$

Interpret this probability distribution as the distribution on the dictionary, and then you can get the generated words at the current moment by sampling.

4.2. *Discriminating the Network.*

The discriminator is used to distinguish whether a sentence is a real data description or generated by a model. Therefore, the input of the discriminator is a sentence and a set of data, and the output is the probability P that the sentence is a true description of the predicted data image. Each group of data in the training data has 5 description sentences, 4 of which are randomly selected as reference sentences and 1 is used as a positive sample to train the discriminator.

Figure 7 shows the architecture of the discriminator. The input sentence (denoted as in) will first be mapped to the word vector sequence for neural network processing and then input into an LSTM word by word, and the hidden state of the LSTM at the last moment will be taken out as the sentence encoding vector of the input sentence, denoted as

$$h_{\text{in}} = \text{LSTM}\left(s_{\text{in}}\right). \tag{7}$$

For the 4 reference sentences (denoted as $s_{\text{ref}}^i = 1, 2, 3, 4$), the same LSTM module is used to do the same processing, and they are, respectively, turned into corresponding sentence encoding vectors:

$$h_{\text{ref}}^i = \text{LSTM}\left(s_{\text{ref}}^i\right). \tag{8}$$

Each reference sentence encoding vector is connected to the input sentence encoding vector and the data global feature vector and then input to a fully connected network. The data global feature vector is the average of all the feature vectors. This fully connected network contains 3 layers, and each layer is used. In addition to the residual connection, the activation function of the last layer is the sigmoid function, which is used to map the input to a value between 0 and 1. It is interpreted as the probability of the input sentence and the image under the auxiliary reference of the reference sentence:

$$P^i = \sigma\left(\text{MLP}\left[\bar{v}; h_{\text{in}}; h_{\text{ref}}^i\right]\right). \tag{9}$$

Then take the maximum of the 4 probabilities as the probability that the input sentence is the true description of the image, namely,

$$P = \max_i P^i. \tag{10}$$

4.3. *Training Method.*

This section discusses how to train the generative network and the discriminant network because when training the generative network, you will encounter a discrete problem, which makes the gradient unable to be

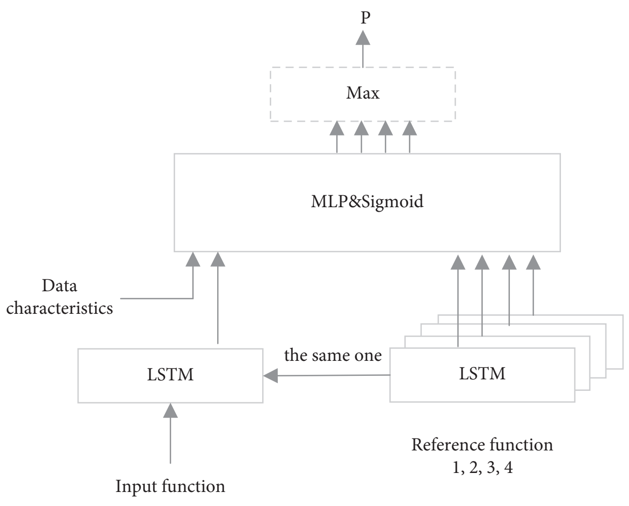

FIGURE 7: Discriminator architecture.

backpropagated. This section introduces a reinforcement learning framework and uses the policy gradient method in reinforcement learning to solve this problem. Discuss the training objectives and training algorithms of the generated network and the discriminant network through the reinforcement learning side.

4.3.1. Reinforcement Learning. Reinforcement learning is a decision-making model that describes how to seek the largest long-term reward in the interaction between the subject and the environment. There are several key concepts in reinforcement learning. The agent in reinforcement learning has the ability to act, where the action is the interaction with the environment, and the action will get new observations from the environment and change the subject's state (state, a specific state corresponds to a specific reward (reward) obtained from the environment). Use the symbols *a*, *o*, *s*, and *r* to represent actions, observations, states, and rewards; then, a series of behavior history can be expressed as

$$o_1, r_1, a_1, \cdots, a_{t-1}, o_t, r_t. \tag{11}$$

The subscript represents time. This sequence represents the observation O_1, the reward r_1 is obtained, and the action a_1 is performed, the observation is changed to O_2, the reward r_2 is received, and the action a_2 is performed, and the observation is changed to o_3, the process is repeated until the action at -1 is observed and the observation is changed to o_t, and reward r_t is obtained. State *s* is a summary of the entire behavior history, namely,

$$s_t = f(o_1, r_1, a_1, \cdots, a_{t-1}, o_t, r_t). \tag{12}$$

In many reinforcement learning tasks, it is assumed that the environment is fully observable (full observability); that is, the subject can observe the complete current environmental state, and the state can be determined by current observations, namely,

$$s_t = f(o_t). \tag{13}$$

In reinforcement learning, the main body contains one or more parts of policy, value function, and model.

Strategy refers to the method of determining action from the state, which is divided into deterministic strategy and random strategy. The mathematical representation of a deterministic strategy is

$$a = \pi(s). \tag{14}$$

The mathematical representation of randomness strategy is

$$\pi(a|s) = P(a_t = a|s_t = s). \tag{15}$$

Here, π represents the strategy.

The value function is used to estimate the value of a specific action in a certain state. Unlike rewards, rewards are obtained with the state and are short-term benefits at the current moment, while the value function focuses on the fact that an action will bring how many long-term rewards. The value function can be defined as

$$Q^{\pi}(s,a) = E\left(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots |s,a\right). \tag{16}$$

Among them, γ is a conversion factor between 0 and 1 because future rewards are uncertain, and discounts need to be discounted until now. The longer the reward, the greater the uncertainty and the greater the discount.

The model is the prediction of the state and reward of the environment at the next moment under the current state and the selected action. The mathematical description is

$$P_{ss'}^{a} = P(s_{t+1} = s'|s_t = s, a_t = a), \tag{17}$$

$$R_s^a = E(r_{t+1}|s_t = s, a_t = a). \tag{18}$$

Formula (17) expresses the probability of transition to state s' after taking action a in state s, and formula (18) expresses the expected value of environmental rewards after taking action a in state s.

Subjects can be roughly divided into three categories according to their composition: (1) policy based, such as policy gradient and actor-critic algorithm; (2) value based, such as Q-learning; (3) model based.

4.3.2. Train Discriminant Grid and Generating Grid. Use θ and w for all the parameters contained in the generator and discriminator, respectively. To represent, the generator is represented by the symbol G_θ, and the discriminator is represented by the symbol D_w. Then, discuss how to train these two networks.

In the traditional training method, the goal of the training generator G_θ is to maximize the likelihood of generating the true description sentence $y = \{y_1, y_2, \ldots, y_T\}$ corresponding to the predicted data image in a given set of data. The objective function at this time is

$$L_{XE}(\theta) = \frac{1}{N}\sum_{y}\sum_{i=1}^{T}\log P(y_i|y_{1:\,i-1}, I, \theta). \tag{19}$$

Likelihood here is equivalent to an artificially set model evaluation index, but as described in Section 1, this index has obvious flaws. The design of the generative confrontation network introduces the idea of confrontation in the neural network by introducing a discriminator D_w. To determine whether a sample is real or generated by the generator, the discriminator D_w in fact plays a role in evaluating the quality of the samples generated by the generator. It can be seen as an evaluation method for generators, scoring the samples generated by the generator to help improve the generator. This evaluation method is not obtained by setting rules but has the ability to self-learn and progress. A simple strategy is used when training the discriminator: the real sample is good, give 1 point; the generated sample is bad, give 0 points; this paper uses the least square loss function to train the discriminator:

$$\min_{w} \frac{1}{2}E_{y\sim P_{\text{real}}}\left[\left(D_w(y,I)-1\right)^2\right]+\frac{1}{2}E_{y\sim P_{\text{fake}}}\left[\left(D_w(y,I)\right)^2\right], \tag{20}$$

where P_{real} is the distribution of the real language description corresponding to image I and P_{fake} is the distribution of the language description of the generator corresponding to the image I. In the image description task, the work of the discriminator includes two aspects. On the one hand, it judges whether the generated sentence meets the natural language specification, and on the other hand, it judges whether the generated sentence semantically matches the image. Therefore, in the above formula, introduce additional items to require the discriminator to learn the matching relationship between sentence semantics and images. The final optimization goal is

$$\min_{w} \frac{1}{2}E_{y\sim P_{\text{real}}}\left[\left(D_w(y,I)-1\right)^2\right]+\frac{\alpha}{2}E_{y\sim P_{\text{fake}}}\left[\left(D_w(y,I)\right)^2\right] + \frac{\beta}{2}E_{y\sim P_{unm}}\left[\left(D_w(y,I)\right)^2\right]. \tag{21}$$

Among them, P_{unm} represents the distribution of image I that does not match but comes from the real language description, and α and β are hyperparameters. This objective function can be optimized by the backpropagation algorithm.

For the generator, the goal of optimization is to make the samples generated by themselves get high scores under the evaluation index of the discriminator. According to the original design of the adversarial network, the goal of optimization should be

$$\min_{\theta} \frac{1}{2}E_I\left[\left(D_w(G_\theta(I),I)-1\right)^2\right]. \tag{22}$$

However, this optimization goal cannot be trained by backpropagation. This is because the generator $G_\theta(I)$ (approximately needs to sample the output distribution when generating words). This process is not differentiable, causing the gradient signal of the loss function to fail to generate. The output of the generator continues to propagate back, so the parameters of the generator cannot be updated through backpropagation. In order to solve this problem, this paper introduces a reinforcement learning method in the training.

First, explain the data description model as a reinforcement learning problem. Because every word generated is known, the problem of data description is a completely observable problem, so no distinction is made between observation and state. The main body in reinforcement learning is the generator in the confrontation network. The action a_t is the word y_t generated by the generator at each moment, and the state s_t is the generated word sequence $y_{1:t-1}$. The intermediate state is not obtained. Only when the complete sentence is generated, the reward is obtained, and the reward for the complete sentence is the score given by the discriminator.

Then determine the strategy, value estimation function, and model included in the subject. The meaning of the strategy is how to generate the distribution of the next word according to the state (the generated word sequence), that is, the output distribution of the generator model $G_\theta(y_{t+1}|y_{1:t},I)$ approximately. The value estimation function is represented by $Q^{G_\theta}(s_t,y_{t+1},I)$ approximately, which means that starting from the state s_t, select the action y_{t+1}, and follow the strategy G_θ. Generate words, the long-term cumulative rewards are obtained, and the calculation of this reward depends on the data I. In the model, under the current state $y_{1:t-1}$ and the selected action y_t, the state and reward at the next moment are determined. In this way, the original Generative Adversarial Network structure is interpreted as a reinforcement learning model. Then, set the generator's optimization goal to start from the initial state, 0, and follow the generator's strategy to generate the sentence with the largest expected cumulative reward, namely,

$$L(\theta)=\sum_{y_1} G_\theta(y_1|s_0,I)Q^{G_e}(s_0,y_1,I). \tag{23}$$

In order to train the generator, it is necessary to obtain the gradient of the optimization target to the network parameters and use the policy gradient to solve this problem.

Introduce the symbol $V^{G_e}s_t$, I to represent the expected cumulative reward of the state s_t under the strategy G_e, namely,

$$V^{G_e}s_t, I=\sum_{y_{t+1}} G_\theta(y_{t+1}|s_t,I)Q^{G_e}(s_t,y_{t+1},I)=Q^{G_e}(s_{t-1},y_t,I). \tag{24}$$

Derivation of the objective function is as follows:

$$\begin{aligned}\nabla_\theta L(\theta) &= \nabla_\theta\left[\sum_{y_1} G_\theta(y_1|s_0,I)Q^{G_e}(s_0,y_1,I)\right]\\ &= E_{y_{1:t-1}\sim G_\theta}\sum_{y_t}\nabla_\theta G_\theta(y_t|s_{t-1},I)Q^{G_e}(s_{t-1},y_t,I).\end{aligned} \tag{25}$$

Formula (25) gives an estimate of the target gradient, where the expectation can be approximated by sampling, and the gradient descent method can be used to train the generator after the gradient is obtained. G_θ in the formula only defines complete sentences. For incomplete sentences, let it share the same value of G_θ with complete sentences. At this time,

$$\nabla_\theta L(\theta) = \frac{1}{T}\sum_{t=1}^{T} E_{y_t \sim G_\theta(y_t|s_{t-1},I)} \left[D_w(y_{1:T}, I)\nabla_\theta \log G_\theta(y_t|s_{t-1}, I)\right]. \tag{26}$$

In order to reduce the variance of the estimated gradient, consider not adding the reference term b at the end of the gradient estimation, as long as b has nothing to do with W: T; then,

$$\nabla_\theta L(\theta) = \frac{1}{T}\sum_{t=1}^{T} E_{y_t \sim G_\theta(y_t|s_{t-1},I)} \left[D_w(y_{1:T}, I) - b\right]\nabla_\theta \log G_\theta(y_t|s_{t-1}, I). \tag{27}$$

For the selection of b, this paper uses the self-critical benchmark $D_w(\hat{y}_{1:T}, I)$ proposed in," where $\hat{y}_{1:T}$ is the sentence obtained by the current model using the inference method. The inference method can choose greedy decoding, beam search, and so on. The greedy decoding method is selected in the paper. From this formula, if the quality of the sampled sentence is better than the quality of the greedy decoding during the training process, then the probability of generating such a sentence is increased. If the sentence is sampled, the quality is worse than greedy decoding, thus reducing the probability of generating such a sentence. If the generator is well trained at this time, the quality of the sampled sentence and the sentence of greedy decoding should be better. If the generator is poorly trained at this time, then the quality of the sampled sentence and the sentence of greedy decoding should be relatively poor, so the self-criticism benchmark can always be maintained at the level near the sampled sentence, so the variance of the estimated gradient can be effectively reduced.

Now it is possible to train the entire adversarial network, but after starting the adversarial training, the generator and the discriminator need to be pretrained because if the pretraining is not done, the quality of the samples generated by the generator is very poor, and the discriminator is easy to distinguish. Given a very low score, the generator can only learn which samples are bad but cannot learn good samples, so the maximum likelihood estimation is used to pretrain the generator and the discriminator, respectively. After the pretraining is completed, the generator and the discriminator are trained alternately until the two converge. Algorithm 1 shows the complete training process.

5. Model Application and Result Analysis

5.1. Experimental Environment and Data Set Research Object. Randomly select a number of stocks in the A-share market to carry out the price prediction experiment, verify the basic data generation through the database, and finally select all the daily data of the stock market since 2020, including the opening price, closing price, highest price, and lowest price. The selected indicators mainly include opening price, closing price, highest price, lowest price, trading volume, total trading value, etc. Research methods: according to the key indicators of investors' attention, the 14-day RSI, 10-day CR, 10-day BR, WR difference from the previous day, 10-day BIAS, MACD value, 9-day KDJ value, ROC difference, price change, and price, economic indicators such as the range of change and the range of change in trading volume are used as data characteristics inspection criteria to simulate the psychological state of investors. At the same time, the selection of technical indicators is constantly evaluated and adjusted to ensure the validity of data features. Using the LSTM-GAN network, adjust the network structure through continuous experimentation (number of hidden layers, number of hidden layer neurons, weight initialization method of each layer, excitation function used by each network layer, dropout function setting, model training iteration number, iteration Batch_size, and other parameters) to predict the increase in the closing price of the stock market. Today's closing price increase = (today's closing price-yesterday's closing price)/ yesterday's closing price. How many days' data are used as a window to predict the next day's stock price increase data requires careful consideration. After a large number of experimental tests, the final model uses the data of the previous 14 days to predict the closing price increase on the 15th day, and the positive or negative increase value information represents the rise or fall of the closing price of the stock market. Through this method, the stock price rise and fall predictions are made, and at the same time, a rise value is obtained as a reference. n represents the number of data records in the training set.

5.2. Using the Model to Conduct Experiments on Important A-Share Indexes. Among the A shares, 30 stock markets were randomly selected. Each stock market selected about 700 data, which is the historical stock market data since 2020. Among them, 80% of the data are used as training data to train the model, and the remaining 20% are used as test data to evaluate the model. The BP neural network is used to predict 30 stocks.

The forecast accuracy rates of each stock market are 51.33%, 52.94%, 54.19%, 58.21%, 51.20%, 52.94%, 56.39%, 54.55%, 55.56%, 51.47%, 55.15%, 53.91%, 50.86%, 52.10%, 51.89%, 53.23%, 53.79, 54.41%, 55.56%, 54.81%, 53.68%, 56.81%, 51.13%, 57.01%, 52.59%, 55.45%, 51.47%, 52.94%, 51.47%, and 52.94%, the highest of which is 58.21% and the minimum is 50.8%. The average forecast accuracy rate of all stock markets is 53.67%, as shown in Figure 8.

The same stock market data directly use the LSTM-GAN model; after the tuning process, the average accuracy rate obtained is 54.03%, as shown in Figure 9. The accuracy of the stock price fluctuations predicted by the two is roughly the same, with a slight difference. Comparison of results obtained by LSTM reveals that the BP network has a weak advantage, which also verifies the potential of LSTM in stock market forecasting.

```
     Input: generator $G_e$, discriminator $D_w$. Training data <stock market, real trend> collection;
     Output: Trained generator $G_e$, discriminator $D_w$
(1)  Randomly initialize generator $G_e$, discriminator $D_w$
(2)  Use the objective function announcement (19) to pretrain the generator $G_e$;
(3)  Use the generator $G_e$ to generate the stock market trend prediction graph, and use three kinds of data <stock market, real trend>,
     <stock market, generated trend>, and <stock market, unmatched real trend> to train the discriminator $D_w$ according to the
     objective function (21)
(4)  **repeat**
(5)  for g-turn times do
(6)  Use generator $G_e$ to generate sentence $y_{1:r}$
(7)  Estimate the gradient according to formulas 3–45 and update the parameters of the generator $G_e$
(8)  **end for**
(9)  for g-turn times do
(10) Use the generator $G_e$ to generate the image description, use three kinds of data <stock market, real trend>, <stock market,
     generated trend>, and <stock market, unmatched real trend> to train the discriminator $D_w$ according to the objective function
     formula (25)
(11) **end for**
(12) until generator $G_e$, discriminator $D_w$ converges
```

Algorithm 1: A network training algorithm for stock prediction trend image generation based on a generative confrontation network.

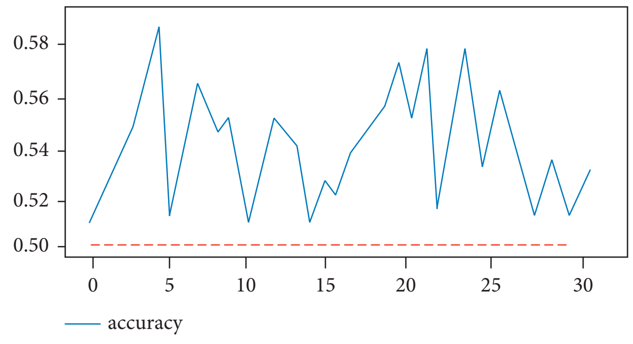

Figure 8: BP network closing price forecast results.

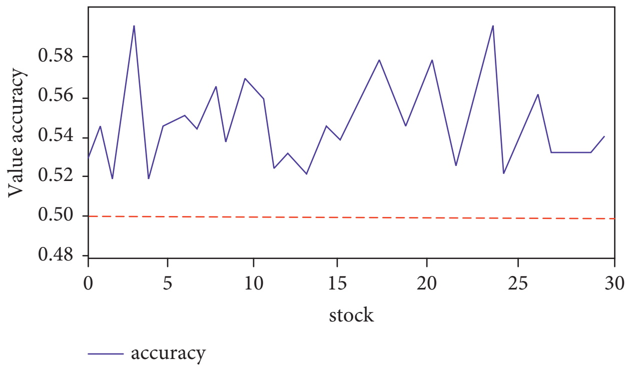

Figure 9: The forecast result of the closing price after BP is changed to LSTM-GAN.

First, use the data of the Shanghai Composite Index to train the model. The parameter tuning of the model is all carried out around the Shanghai Composite Index. When a relatively satisfactory result is achieved, the model will be used to verify other large-cap indexes, such as Shenzhen Component Index, Shanghai and Shenzhen 300, and Small and Medium-Sized Index. Because these indexes are not easy to be manipulated by "bookmakers," and their stock prices and transaction data can more reflect investors' judgments on the market, these large-cap indexes are more suitable for evaluating forecasting models. Specific experimental results are shown in the following.

5.2.1. Shanghai Composite Index. The data set uses the historical daily data of the Shenzhen Component Index from April 1, 2009, to February 27, 2020, 1894 strip. Considering that historical stock market data before 2009 did not have much reference value, especially the economic crisis in 2008, data collection began in 2009. 75% of the data is used as training data, and the remaining 25% is used for verification testing.

Daily line data include a total of 1921 data: training set, 1447; test set, 474.

After 20 iterations, the predicted result is shown in Figure 10:

5.3. Experimental Results. The experimental results prove that the designed LSTM-GAN regression prediction model is effective. And the accuracy of the model after tuning for a

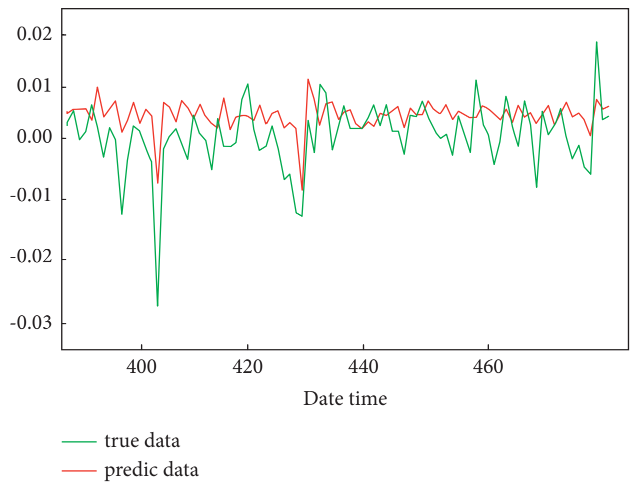

Figure 10: Details of the forecast results of the closing price increase of the Shenzhen Component Index. The accuracy rate of forecasting changes: 61.603%; increase forecast error value: 0.0120.

single stock market reaches 61.603%. Moreover, without modifying the model parameters, it is applied to other large-cap indexes, and its average accuracy rate exceeds 60%, which fully verifies the effectiveness and universality of the designed model for the A-share large-cap index. The potential of LSTM in stock market forecasting is worthy of in-depth study and exploration. At the same time, another innovative point of the experiment is that the regression prediction value is the predicted value of the closing price increase, and the average predicted increase error value is also obtained during the process of training the model, which can be provided to investors for reference.

Through experiments, it can also be shown that although China's stock market is a policy market, its rise and fall are more affected by policies, and stock price data will have a certain "singularity," but the market index is relatively stable and quite measurable.

6. Conclusion

This paper systematically expounds on the theoretical basis of Generative Adversarial Networks in deep learning and applies these two models to the stock market to predict the ups and downs of A-share stocks in the next day. The main research results of this paper are as follows.

First, from the perspective of deep learning, while introducing neural networks, this paper also clarifies how LSTM-GAN neural networks solve the shortcomings of traditional BP neural networks one by one.

Second, apply deep learning technology to the stock price prediction. Take historical transaction data as input data and use the LSTM-GAN model to predict the rise and fall (up, adjustment, and fall) of stocks. The prediction accuracy of the model reached 60%.

Compared with the traditional neural network, the average error of this new network method is smaller, and in most cases, the relative error is better than the traditional neural network. However, there are still deficiencies in the experiment process that need to be improved. Although the prediction model based on the generative confrontation network introduces a preattention mechanism, it solves the problem of difficulty in convergence and difficulty in training and avoids the problem of nonconvergence and collapse of the algorithm in the budget process. However, this model is compared with other generative models. The GAN model does not need to be modeled in advance. In the case of more pixels, the model is too free to cause uncontrollable problems.

Data Availability

The dataset can be accessed upon request to the author.

Conflicts of Interest

The author declares no conflicts of interest.

References

[1] I. Goodfellow, J. Pouget-Abadie, and M. Mirza, "Generative adversarial nets," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2672–2680, Cambridge MIT Press, Cambridge, MA, USA, June 2014.

[2] S. Talwar, M. Viberg, and A. Paulraj, "Blind separation of syn-chronous co-channel digital signals using an antenna array (part I. algorithms)," *IEEE Transactions on Signal Processing*, vol. 44, no. 5, pp. 1184–1197, 1995.

[3] Y. P. Wei, L. K. Liu, and H. Guo, "Approach to blind estimation of pseudo random code sequence for DSSS signaling multipath," *Computer Engineering and Applications*, vol. 49, no. 12, pp. 195–199, 2013.

[4] S. Reed, Z. Akata, and X. Yan, "Generative adversarial text to image synthesis," in *Proceedings of the International Conference on Machine Learning*, pp. 1060–1069, ACM, New York, NY, USA, May 2016.

[5] S. E. Reed, Z. Akata, and S. Mohan, "Learning what and where to draw," *Advances in Neural Information Processing Systems*, vol. 12, pp. 217–225, 2016.

[6] X. Zhang, J. Zhao, and Y. Lecun, "Character-level convolutional networks for text classification," *Advances in Neural Information Processing Systems*, vol. 1, pp. 649–657, 2015.

[7] H. Zhang, T. Xu, and H. Li, "StackGAN++:Realistic image synthesis with stacked generative adversarial networks," in *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1, IEEE, Venice, Italy, October 2017.

[8] Z. Xu, J. Du, and J. J. Wang, "Satellite image prediction relying on GAN and LSTM neural networks," in *Proceedings of the IEEE International Conference on Communications*, Shanghai, China, May 2019.

[9] A. Joulin, E. Grave, and P. Bojanowski, "Bag of tricks for efficient text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, ACL, Valencia, Spain, September 2016.

[10] S. Reed, Z. Akata, and H. Lee, "Learning deep representations of finegrained visual descript-tions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 49–58, IEEE, Las Vegas, NV, USA, June 2016.

[11] E. Denton, S. Chintala, and A. Szlam, "Deep generative image models using a Laplacian pyramid of adversarial networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS'15)*, pp. 1486–1494, ACMDL, Montréal, Canada, December 2015.

[12] T. Karras, T. Aila, and S. Laine, "Progressive growing of GANs for improved quality, stability, and variation," in *Proceedings of the International Conference on Learning Representations*, ICLR, Vancouver, Canada, May 2018.

[13] C. Ledig, L. Theis, and F. Huszár, "Photorealistic single image superresolution using a generative adversarial network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4681–4690, IEEE, Honolulu, HI, USA, July 2017.

[14] K. He, X. Zhang, and S. Ren, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, IEEE, Las Vegas, NV, USA, June 2016.

[15] A. Santoro, D. Raposo, and D. G. Barrett, "A simple neural network module for relational reasoning," in *Proceedings of the Conference and Workshop on Neural Information Processing Systems*, NIPS, Denver, CO, USA, June 2017.

[16] J. Song, L. Gao, Z. Guo, W. Liu, D. Zhang, and H. T. Shen, "Hierarchical LSTM with adjusted temporal attention for video captioning," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, August 2017.

[17] X. Mao, Q. Li, and H. Xie, "Least squares generative adversarial networks," in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2813–2821, IEEE, Venice, Italy, October 2017.

[18] B. Dai, D. Lin, and R. Urtasun, "Towards diverse and natural image descriptions via a conditional GAN," in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Venice, Italy, October 2017.

[19] L. Yu, W. Zhang, and J. Wang, "SeqGAN: sequence generative adversarial nets withPolicy gradient," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI 2017)*, Cornell University, Ithaca, NY, USA, August 2017.

[20] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press Cambridge, London, England, 1998.

[21] S. J. Rennie, E. Marcheret, and Y. Mroueh, "Self-critical sequence training for image captioning," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Venice, Italy, July 2017.