risks

MDPI

Article

An Efficient Method for Pricing Analysis Based on Neural Networks

Yaser Ahmad Arabyat [1,*], Ahmad Ali AlZubi [2], Dyala M. Aldebei [3] and Samerra'a Ziad Al-oqaily [4]

[1] Faculty of Business, Department of Economic and Finance, Al-Blaqa Applied University, Al Salt 19110, Jordan
[2] Computer Science Department, Community College, King Saud University, P.O. Box 28095, Riyadh 11437, Saudi Arabia; aalzubi@ksu.edu.sa
[3] Faculty of Business, Department of Accounting, Al-Blaqa Applied University, Al Salt 19110, Jordan; dyaladebei@bea.edu.jo
[4] Business School, Al-Blaqa Applied University, Al Salt 19110, Jordan; samoqaily@bea.edu.jo
* Correspondence: yaser_arabyat@bau.edu.jo

Abstract: The revolution in neural networks is a significant technological shift. It has an impact on not only all aspects of production and life, but also economic research. Neural networks have not only been a significant tool for economic study in recent years, but have also become an important topic of economics research, resulting in a large body of literature. The stock market is an important part of the country's economic development, as well as our daily lives. Large dimensions and multiple collinearity characterize the stock index data. To minimize the number of dimensions in the data, multiple collinearity should be removed, and the stock price can then be forecast. To begin, a deep autoencoder based on the Restricted Boltzmann machine is built to encode high-dimensional input into low-dimensional space. Then, using a BP neural network, a regression model is created between low-dimensional coding sequence and stock price. The deep autoencoder's capacity to extract this feature is superior to that of principal component analysis and factor analysis, according to the findings of the experiments. Utilizing the coded data, the proposed model can lower the computational cost and achieve higher prediction accuracy than using the original high-dimensional data.

Keywords: neural network; stock exchange; accounting systems; finance

Citation: Arabyat, Yaser Ahmad, Ahmad Ali AlZubi, Dyala M. Aldebei, and Samerra'a Ziad Al-oqaily. 2022. An Efficient Method for Pricing Analysis Based on Neural Networks. *Risks* 10: 151. https://doi.org/10.3390/risks10080151

Academic Editors: Krzysztof Jajuga and Józef Dziechciarz

Received: 22 June 2022
Accepted: 20 July 2022
Published: 28 July 2022

1. Introduction

The stock market is a market closely related to our daily lives, and it is very important to the economic development of our country and even the world. As important participants in the stock market, changes in people's sentiments will be quickly reflected in the market. In recent years, with the Internet and the rapid development of technology, the speed and channels of information dissemination are also increasing, and the ways for investors to obtain information are becoming more and more diversified (Hecht et al. 2020). Financial news, social media, etc. have gradually begun to affect investors' investment decisions. Therefore, stock prices will not only be affected by politics, the economy, and the military, but also by "emotional" factors (Lu et al. 2012; Craighead et al. 2009). With the influence of news, Weibo, blogs, post bars, forums and other social networks, financial news is more rapid and intuitive. This induction magnifies investors' attitudes towards the stock market, leading to great uncertainty and volatility in stock prices, making stock price prediction a major problem for research (Brandt and Brandt 2004).

In recent years, neural networks (Hecht et al. 2020) has become a learning boom. After decades of development, neural networks have become a popular technology (Garnet et al. 2002), and have continued to make breakthroughs in various fields, such as stock price forecasting (Brandt and Brandt 2013), time series forecasting, text analysis, computer vision (Van Velthoven et al. 2005; Zohar et al. 2002), etc.

Financial data modeling is a method of abstract representation of financial time series. The most important purpose of this task is to predict the future trend of financial series based on current data and to help decision-makers formulate strategies. It is currently widely used in accounting, business investment, stock valuation and other fields. Among them, the stock market has a higher rate of return on investment compared with other industries and has attracted the attention of a large number of investors. However, the strong volatility of stocks also causes many potential risks. Therefore, being able to accurately predict stock prices can not only obtain considerable profits, but also avoid potential risks.

The stock market has the characteristics of non-linearity, instability, complexity and so on. Regarding stock forecasting methods, scholars at home and abroad have conducted much research. At present, the methods of stock forecasting are roughly divided into two aspects: forecasting methods based on news and forecasting methods based on historical data. In the forecasting method based on news, Hai Nguyen et al. (2015) predicted the trend of stocks by mining people's sentiments about stocks from social media. Li et al. (2015) proposed a method of using news digests of stock-related reports to predict stock prices, and its effect is better than the method of using complete articles for prediction. Prachyachuwong and Vateekul (2021) designed a stock prediction model that combines word embedding and deep learning methods. This model can extract effective information from news and has higher accuracy. The research methods are based on historical data originated earlier, and are more diverse. Rundo et al. (2019) used an improved ARMA model to predict stock prices. The authors of (Chopra and Sharma 2021) used a model combining gray neural networks and Markov methods to study stock prices. Li et al. (2017) used a deep-belief-networks-based stock forecasting method, and introduced the concept of "intrinsic plasticity", which significantly improves the forecasting effect of the network.

Because the domestic stock market is not yet sound, there are many misleading factors, and retail investors participate more; thus the method is complicated. The research methods based on the existing literature are not suitable for the domestic stock market. Compared with the information based on the news, the historical trading data of stocks is more true and objective, and has reference value. Therefore, this article uses the method based on historical data to study stock prices. First of all, this paper constructs a deep autoencoder to reduce the dimensionality and feature-extraction of stock index data, then uses a BP neural network to build a regression model, predicts the stock price based on the DAE dimensionality reduction coding sequence, and finally verifies its effectiveness through experiments.

2. Deep Autoencoder

An autoencoder (AE) is a type of neural network. Unlike other neural network models, AE is an unsupervised learning algorithm that does not require additional label information during training. AE has an input layer, a hidden layer h and an output layer, and its structure is shown in Figure 1.

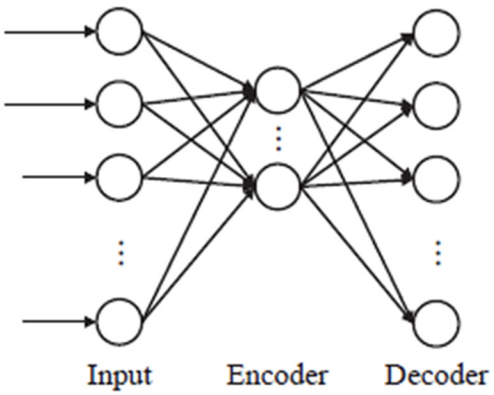


Figure 1. Autoencoder structure.

The AE uses the same structure of input and output, and through repeated training, the function $h\_(\boldsymbol{W},b)\,(x) \approx x$ is trained to make the input and output as similar as possible. The hidden layer added after training is also called an encoder, and the output of the encoder is the result of encoding the input data. The output layer is called the decoder, which can decode the encoded data, but usually does not pay attention to the output of the decoder. The working principle of an AE is expressed as follows:

Encoder:

$$h = f(\boldsymbol{W}x + b) \tag{1}$$

Decoder:

$$\hat{x} = f\left(\boldsymbol{W}'h + b'\right) \tag{2}$$

Among them, x is the input data, f is the activation function, $\boldsymbol{W}$ and b are the weights and bias values and $\hat{x}$ is the output of the decoder.

The use of autoencoders can achieve the purpose of feature extraction and feature dimensionality reduction. By setting the number of hidden layer nodes, dimensionality reduction can be achieved for any dimension. At the same time, the output of the encoder also expresses the higher-level features of the input data.

A deep autoencoder (DAE) is a deep neural network formed by stacking shallow autoencoders. It contains multiple hidden layers. Compared with shallow autoencoders, its advantage is that it can learn features in the data layer by layer. For example, we can input raw data into the DAE network, and generate a coding sequence in the first layer of the network. By analogy, the k-th in the DAE extracts features layer by layer based on the code of the $k-1$ layer output. As the number of network layers increases, the extracted features become more abstract (Wang et al. 2016). Deep autoencoders usually have better feature extraction capabilities than shallow autoencoders, and can extract non-linear features in the high levels of the data. The structure of the deep autoencoder is shown in Figure 2.

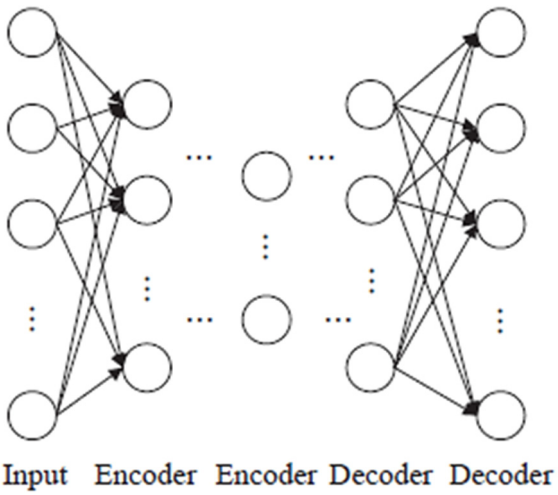


Figure 2. The structure of a DAE.

Although a DAE has the same structure as the multilayer feedforward neural network, its training strategy is different from that of the multilayer feedforward neural network. DAE uses a layer-by-layer greedy training method to pre-train the weights of each layer. Then the pre-trained weights are substituted into the corresponding layer. The global optimization algorithm fine-tunes the weight to obtain the final weight of the DAE.

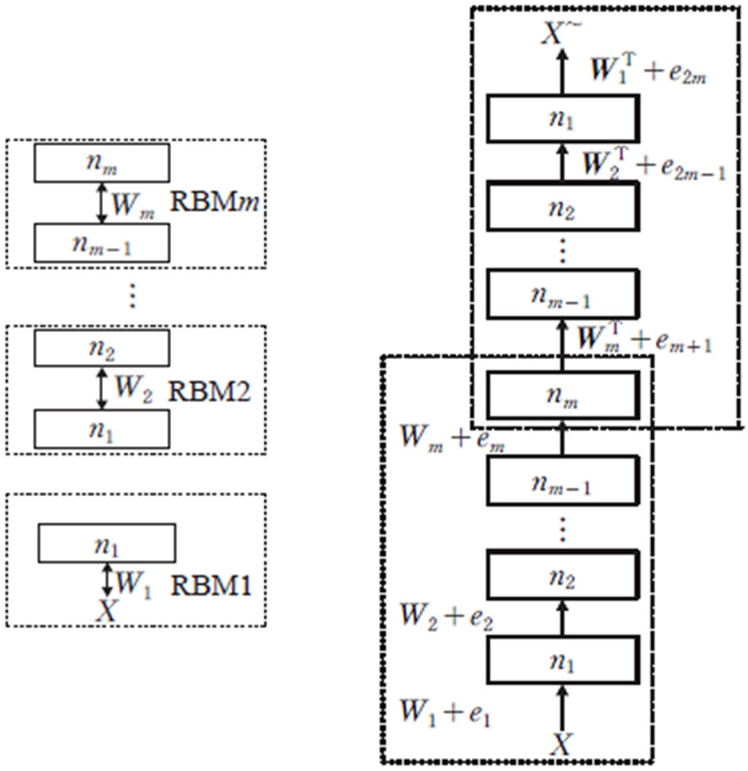
Deep autoencoders are currently widely used and have been successful in many fields; for example, in the field of image recognition, deep autoencoders are used for feature extraction (Xu et al. 2016; Gao et al. 2015) and image compression (Hinton and Salakhutdinov 2006). In the field of natural language processing, deep autoencoders are used for semantic hashing (Salakhutdinov and Hinton 2009), text generation (Babić et al.

2020) and speech processing (Araki et al. 2015). By using the deep autoencoder to reduce the original data, the algorithm becomes more efficient in the low-dimensional data space. Therefore, the deep autoencoder is an important tool and method in deep learning.

3. Proposed Model

3.1. Dimensionality Reduction of Stock Data Based on DAE

There is often multiple collinearity between stock indicators and indicators. For example, the J in KDJ (a stochastic indicator) is obtained by linear operation of K and D. Although these indicators with multi-collinearity provide convenience for technical analysis, they do not have much practical significance in the calculation of neural networks. Instead, they increase the computational cost of the model and the risk of overfitting. Therefore, it is necessary to reduce the dimensionality of the stock indicators to eliminate the multi-collinearity, reduce data redundancy, and greatly reduce the calculation and complexity of the model. This paper constructs a DAE model based on a Restricted Boltzmann Machine (RBM) and uses it for dimensionality reduction and feature extraction of stock index data. The structure of the model is shown in Figure 3.



(**a**) Trained layer by layer using RBM (**b**) After fine-tuning

Figure 3. Proposed model based on RBM.

A DAE is a deep neural network. Although the error back propagation algorithm can be used to train a DAE, the training effect is not ideal. As the number of neural network layers deepens, the gradient disappears when training the neural network using the error back propagation algorithm (Glorot and Bengio 2010). Aiming at the problem of the difficulty of deep neural network training, the literature (Le Roux and Bengio 2008) gives a method based on the RBM training method.

The training process is mainly divided into three steps:

(1) First construct multiple RBMs, as shown in Figure 3a, and pre-train the models separately using a layer-by-layer greedy training strategy. Among them, $\boldsymbol{X}$ is the

original data; each RBM uses the output of the previous RBM as input, and $\boldsymbol{W}$ is the pre-trained weight.

(2) Stack the pre-trained RBM layer by layer to build a symmetric model as shown in Figure 3b. The first to m-th layers of the model are called encoders, and each layer of the encoder uses the corresponding $\boldsymbol{W}$ as weight. The $m + 1$ layer to the 2 m layer of the model are called decoders, and each layer of the decoder uses the corresponding $\boldsymbol{W}^{\mathrm{T}}$ as the weight.

(3) Use the BP algorithm to fine-tune the model and update the weight to $\boldsymbol{W} + e$ to make the final output $\widetilde{\boldsymbol{X}}$ of the model as similar as possible to the input $\boldsymbol{X}$; the output of the m-th layer is the coding result. The following describes the RBM-based DAE training process in detail.

The RBM is an undirected neural network model based on statistical mechanics, and its structure is shown in Figure 4. The RBM has two layers that are the visible and hidden layer.

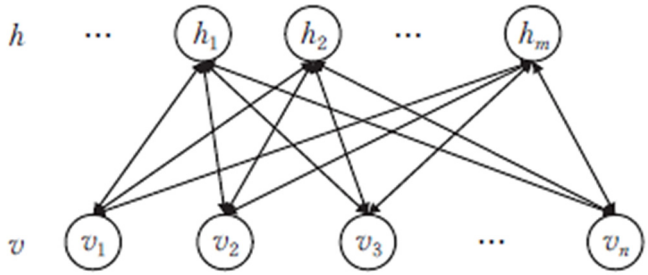


Figure 4. RBM model structure diagram.

For a set of states (v, h) given by the visible layer and the hidden layer, the energy function of the RBM in this state can be defined as:

$$E_\theta(v,h) = -\boldsymbol{a}^{\mathrm{T}}v - \boldsymbol{b}^{\mathrm{T}}h - h^{\mathrm{T}}\boldsymbol{W}v \tag{3}$$

Among them, $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{W}$ are the parameters of the model, a represents the bias value of the visible layer, $\boldsymbol{b}$ represents the bias value of the hidden layer, and $\boldsymbol{W}$ represents the weight matrix between the visible and the hidden layer.

Based on the energy function defined by Equation (3), the joint probability distribution of (h, v) can be obtained:

$$P(v,h|\theta) = \frac{e^{-E(v,h|\theta)}}{\sum_{v,h} e^{-E(v,h|\theta)}} \tag{4}$$

where

$$\theta = \{\boldsymbol{W},\ \boldsymbol{a}, \boldsymbol{b}\} \tag{5}$$

With the joint probability distribution of (h, v), the marginal distributions of h and v can be calculated separately. In practical applications, the probability distribution of the visible layer is more important. The distribution function is expressed as:

$$P(v|\theta) = \frac{\sum_h e^{-E(v,h|\theta)}}{\sum_{v,h} e^{-E(v,h|\theta)}} \tag{6}$$

The process of training the RBM is essentially to find a set of θ so that the output of the visible layer is as similar as possible to the distribution of the training sample S. In order to achieve this goal, the likelihood function is defined:

$$\ln L_{\theta,S} = \sum_{i=1}^{n_s} \ln P(v_i) \tag{7}$$

A set of optimal parameters θ can be obtained by seeking the maximum value of the likelihood function, where v_i is the i-th neuron in the visible layer, and n_s is the number of training samples. In order to calculate the maximum value of Equation (7), it is necessary to derive it, and the joint probability distribution $P(h|v)$ appears in the derivation result, because $P(h|v)$ is difficult to calculate directly. Therefore, Gibbs sampling is used to approximate it.

The RBM has good properties. For example, given the state of neurons in the visible layer, the activation conditions of neurons in the hidden layer are independent. Similarly, given the state of neurons in the hidden layer, the activation conditions of neurons in the visible layer are also independent. Therefore, as long as the RBM model is designed reasonably, any discrete probability distribution can be fitted (Hinton 2002). Based on the above properties of the RBM, when the state of the neurons in the visible layer is known, the activation probability of any j-th hidden-layer neuron can be obtained:

$$P(h_j = 1|v, \theta) = \sigma(\sum_i v_i \boldsymbol{W}_{ij} + b_j) \tag{8}$$

where σ is the sigmoid activation function. Since the RBM has only two layers, after obtaining the state of the neurons in the hidden layer, the state of the neurons in the visible layer can be calculated in the same way. The activation probability of the i-th neuron in the visible layer is:

$$P(v_i = 1|h, \theta) = \sigma(\sum_j h_j \boldsymbol{W}_{ij} + a_i) \tag{9}$$

Repeating k times using Equations (8) and (9) can express the k-th sampling result, and obtain an approximate joint probability distribution $\widetilde{P}(h|v)$, and the RBM training can be completed after m cycles.

Based on the above-mentioned RBM training method, the problem of training a DAE can be simplified to the problem of training multiple RBMs. In the actual training process, the contrast divergence (CD) algorithm is used to train an RBM, and the sampling results of a few Gibbs samples can be used to approximate the joint probability distribution of (h, v). It greatly reduces the training cost for an RBM.

The trained RBMs are stacked into a DAE network as shown in Figure 3b. At this time, the DAE network weights have not reached the optimal state, and the weights need to be fine-tuned. The specific method is that the original stock data $\boldsymbol{X}$ is input to the DAE, and the error back propagation algorithm is used to fine-tune the network to minimize the error of the output decoded vectors $\widetilde{X}$ and $\boldsymbol{X}$. After fine-tuning, the weight in the DAE network reaches the optimal value, and the output of the m-th layer encoder in the middle of the network is a share-coded sequence $\boldsymbol{X}'$ after dimensionality reduction of ticket data.

3.2. Stock Prediction Based on BP Neural Network

A BP neural network is an algorithm based on delta learning rules. Its principle is to use the method of gradient descent to transmit the error generated by each training forward, update the weight and bias value of each layer in turn, repeat iteratively, and finally achieve the iteration ends after the convergence condition. The BP neural network uses a non-linear activation function, so it has good non-linear fitting ability. The stock system is a complex non-linear dynamic system. Its price trend has strong volatility, and there are many factors that affect the price. The linear model cannot solve the stock prediction problem well. Therefore, the neural-network-based model is very suitable for the stock system.

This paper uses a BP neural network to build a regression model to predict stock prices. The model is divided into three layers: input layer, hidden layer and output layer. The neural network structure is shown in Figure 5.

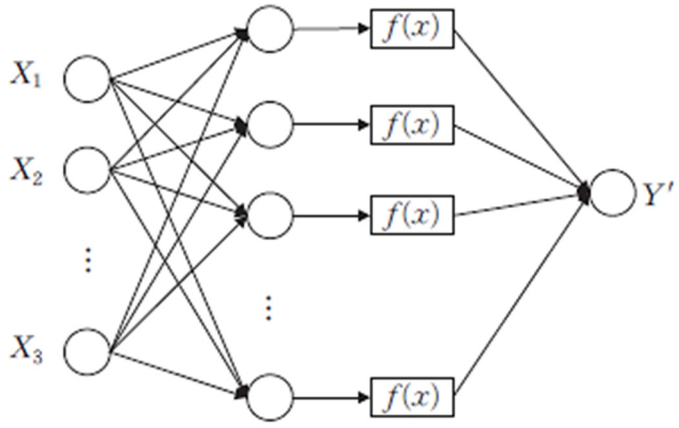


Figure 5. Illustration of BP neural network architecture.

The input of the model is the result $\boldsymbol{X}'$ of the stock data dimensionality reduction of the previous day, and the output is the predicted price $\boldsymbol{Y}'$ of the stock from the next day, where $f(x)$ is the activation function.

3.3. Algorithm Description

The proposed algorithm flow to predict the stock price is as follows in Algorithm 1.

Algorithm 1 Proposed algorithm for stock price prediction

1: For num in layer
2: Initialize RBM (num) weight $\boldsymbol{W}$ and $\boldsymbol{b}$ offset value
3: Enter stock data $\boldsymbol{X}$
4: for I in epoch1
5: Use the method in Section 3.1 to train RBM(num)
6: $\boldsymbol{W}$.append($\boldsymbol{W}$), $\boldsymbol{b}$.append($\boldsymbol{b}$)
7: Return $\boldsymbol{W}$, $\boldsymbol{b}$
8: Use the trained RBM to construct a DAE with a symmetrical structure
9: Load $\boldsymbol{W}$ and $\boldsymbol{b}$ into the corresponding DAE network
10: for i in epoch2
11: Input stock data $\boldsymbol{X}$ to DAE network to fine-tune network parameters
12: End
13: Input stock data $\boldsymbol{X}$ to the trained DAE network
14: Obtain the encoding result $\boldsymbol{X}'$ of the intermediate layer encoder
15: Divide $\boldsymbol{X}'$ into training and test set
16: Build BP neural network
17: For i in epoch3
18: Input the training set to train the BPNN
19: End
20: The BP neural network uses the test set to predict the stock price $\boldsymbol{Y}'$.

Steps (1)~(7) of the proposed algorithm are the process of training the RBM layer by layer. Steps (8)~(12) are the fine-tuning process of the DAE network, and steps (13)~(15) are the code reduction of the original stock data. In the process of dimensioning, steps (16)~(19) are the training process of BP neural network, and step (20) is the prediction process.

4. Experimental Results

4.1. Data Set

This article uses the real transaction data of Kweichow Moutai (600519) as the data set. The data set contains 48 dimensions of data, including commonly used indicators in technical analysis. They are OBV (OBV.OBV, OBV.MAOBV), HSL (HSL.HSL, HSL.MAHSL, VRSI.RSI1), VRSI (VRSI.RSI2, VRSI.RSI3), PSY (PSY.PSY, PSY.PSYMA), WAD (WAD.WAD,

WAD.MAWAD), VR (VR.VR, VR.MAVR), CYR (CYR.CYR, CYR.MACYR), CCI (CCI.CCI), RSI (RSI.RSI1, RSI.RSI2, RSI.RSI3), BIAS (BIAS.BIAS1, BIAS.BIAS2, BIAS.BIAS3), CYF (CYF.CYF), DKX (DKX.DKX, DKX.MADKX), MARSI (MARSI.RSI10, MARSI.RSI6), DMA (DMA.DIF, DMA.DIFMA), ATR (ATR.MTR, ATR.ATR), DPO (DPO.DPO, DPO.MADPO), PAVE (PAVE.CV, PAVE.MCV, PAVE.DIFF), CYE (CYE.CYEL, CYE.CYES), EMV (EMV.EMV, EMV.MAEMV), MACD (MACD.DIF, MACD.DEA, MACD.MACD), MTM (MTM.MTM, MTM.MTMMA) and ROC (ROC.ROC, ROC.MAROC). There are a total of 1156 trading days in the data set; 900 pieces of data are divided into the training set, and 256 pieces of data are used as the test set.

Due to the different dimensions of the data of different stock indicators, the data with larger dimensions has a greater impact on the final result, whereas the data with a smaller dimension has almost no effect on the original result. Data for different dimensions will cause great interference to both the dimensionality reduction stage and the prediction stage, so it is necessary to unify the data to the same order of magnitude. This article uses the linear function normalization (Min − Max scaling) method; the normalization method is:

$$x' = \frac{x - \min}{\max - \min} \tag{10}$$

Through the normalization operation, the data is compressed to between (0, 1), which eliminates the influence of a large amount of dimensional data on data dimensionality reduction, and prevents the phenomenon of gradient saturation of neurons when the logistic and tanh activation functions are used.

4.2. Comparison of DAE Dimensionality Reduction Effects of Different Depths

The autoencoder can be trained as a deep autoencoder with multiple hidden layers. Generally, the more layers of the deep autoencoder, the more abstract the extracted features, and the encoded result represents a higher layer in the data. In order to compare the impact of encoding results of different depths on the prediction results, this paper designs five DAEs, and the number of layers is gradually increased from two to six. The results of the five DAE encodings are respectively used for stock price prediction, and the prediction results are shown in Table 1.

Table 1. The prediction effect of using different depths of DAE encoding.

DAE Network Structure	Number of Network Layers	Training Error (MSE)	Test Error (MSE)
48-5	2	0.00051 ± 0.00021	0.0054 ± 0.0009
48-24-5	3	0.00041 ± 0.00016	0.0030 ± 0.0004
48-24-12-5	4	0.00036 ± 0.00015	0.0025 ± 0.0007
48-30-20-10-5	5	0.00047 ± 0.00012	0.0034 ± 0.0005
48-40-30-20-10-5	6	0.00670 ± 0.00170	0.0220 ± 0.0040

From the prediction results in Table 1, it can be seen that the results of using different depths of DAE encoding sequences to predict stock prices are different. Among them, the data after the use of four-layer-structure DAE dimensionality reduction is more effective in stock prediction, and its training error and test errors are all the smallest.

In order to further measure the encoding effect of encoders with different depths, this paper obtains the results $\widetilde{X}$ after DAE decoding of different depths, calculates the Euclidean distance between X and $\widetilde{X}$ to find the reconstruction error, and uses the reconstruction error to measure the degree of information loss in the DAE encoding–decoding process. The calculation results are shown in Table 2.

Table 2. Reconstruction error of DAE at different depths.

Network Structure	Reconstruction Error
48-5	20.60
48-24-5	17.10
48-24-12-5	15.60
48-30-20-10-5	16.70
48-40-30-20-10-5	19.79

The formula for calculating the Euclidean distance of a vector in a multidimensional space is:

$$d_{12} = \sqrt{\sum_{k=1}^{n} (x_{1k} - x_{2k})^2} \tag{11}$$

Among them, n represents the dimension of matrix $\boldsymbol{X}$.

Under normal circumstances, as the number of layers of the DAE deepens, its ability to abstract and generalize becomes stronger. A shallow DAE can complete the dimensionality reduction of the data through only one encoding, and some important information will be lost during the encoding process, so the reconstruction error is relatively large. The deep DAE extracts features layer by layer, which can more completely retain the important features in the data, with less information loss, so the reconstruction error is small. However, the depth of the DAE is not at its maximum. As the number of encodings increases, the extracted features will become more and more abstract, and the detailed information in the data will be lost, so the reconstruction error will be larger.

4.3. Comparison with Other Dimensionality Reduction Methods

Principal component analysis (PCA) and factor analysis (FA) are two commonly used methods in multivariate analysis, which are usually used for dimensionality reduction analysis of observation samples.

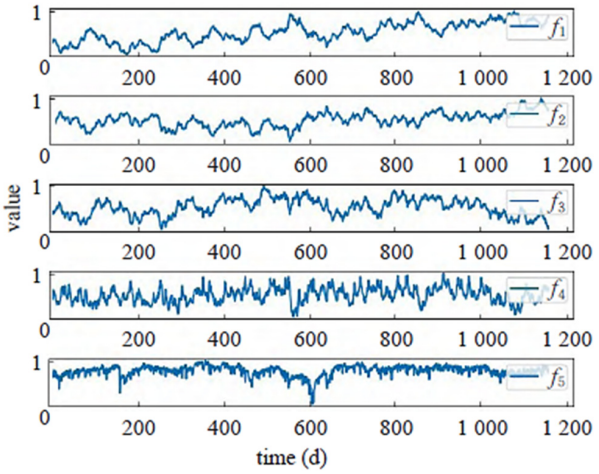
PCA is a linear dimensionality reduction method. The principle is to remove the data in the original high-dimensional space from the data with smaller variance dimensions and simplify it into low-dimensional data, and then rotate the low-dimensional data into the low-dimensional coordinate system. In the coordinate system, the data of each dimension is a linear combination of the original data (Bakshi 2010), the purpose of which is to make the reconstructed data closer to the original data. Therefore, PCA is also called linear autoencoding.

FA is also a method of linear dimensionality reduction. Its principle is to synthesize variables into a few common factors according to their correlation, and replace a group with a strong correlation with one factor. The correlation of variables between different groups is weak, and the correlation of variables in the same group is strong. Similar to the idea of clustering, the original variables can be expressed as a linear combination of common factors (Kaiser 2016).

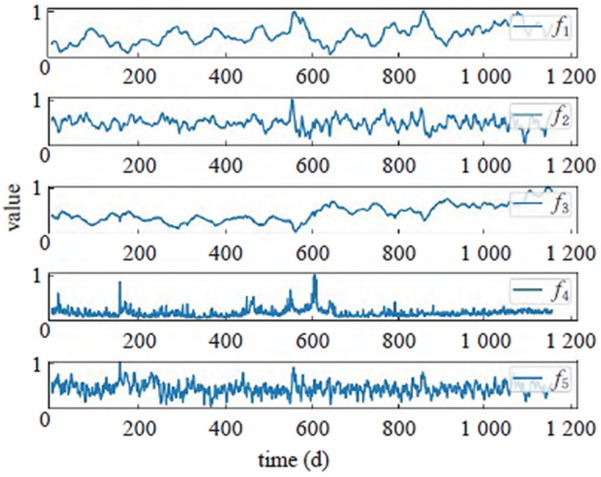
Different from the above two methods, the DAE used in this article is a non-linear dimensionality reduction method. The reason it is called non-linear is that a non-linear activation function can be used between the layers of the DAE, such as sigmoid, tanh, relu, etc., to achieve non-linear mapping of input data and output data. As the number of DAE layers increases, the more times it performs non-linear mapping, and the more abstract the extracted features.

Using the PCA method to reduce dimensionality, when the number of principal components is five, the cumulative variance contribution is 88.27%, so the five-dimensional dimension is selected as the final dimension after dimensionality reduction. FA and DAE are based on the dimension after PCA dimensionality reduction. It is also reduced to five dimensions, which is convenient for comparing the dimensionality reduction effects of different methods. Among them, DAE uses the best four-layer structure model in Section 4.2.

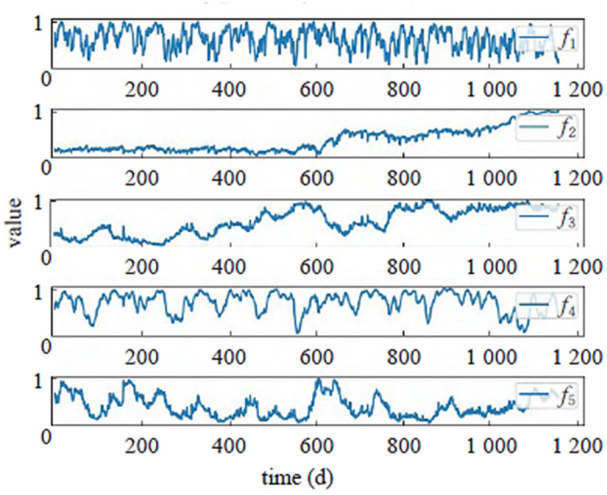
Using the above-mentioned dimensionality reduction methods to reduce the stock data to five dimensions, the coding sequence of each dimension is shown in Figure 6.



(**a**) The result after using PCA for dimensionality reduction



(**b**) The result after dimensionality reduction using FA



(**c**) The result after dimensionality reduction using DAE

Figure 6. Coding sequence of stock data after dimensionality reduction.

Intuitively analyzing the sequence encoded by the three methods in Figure 6, it can be found that the characteristics of the encoded sequence after using PCA dimensionality reduction are not obvious, and it does not reflect the basic situation of the stock trend. The coding sequence after dimensionality reduction using FA and DAE contains the characteristics of stock price trends, among which DAE's stock price trend characteristics are the most obvious (as shown by f_2 in Figure 6c), which can be inferred because the effect of dimensionality reduction using FA and DAE data forecast is better.

In order to verify the above statement, the sequences obtained using the above three methods are used as the input of the BP neural network, and the prediction results are generated using the BP neural network calculation. The prediction results are shown in Figure 7.

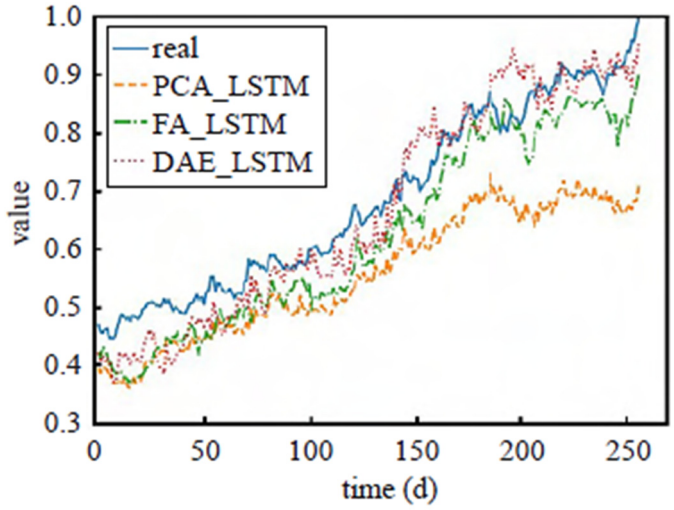


Figure 7. The results of predicting stock prices with different coding sequences.

In order to measure the error between the predicted result and the real result, this article uses three error evaluation indicators, namely MSE (Mean Square Error), MRE (Mean Relative Error) and MAE (Mean Absolute Error). The calculation results of the three errors are shown in Table 3. Analyzing the data in Table 3, we can see that the proposed algorithm has a better prediction effect, whereas PCA-BP and FA-BP have poor prediction results. The single-day rise and fall of the Chinese stock market is only 10%, and the relative error of using the PCA dimensionality reduction code to predict the stock price has reached 17.9%, far exceeding the upper limit of the stock's rise and fall. Therefore, the PCA dimensionality reduction code is used. Predicting stock prices has no practical significance. The data after dimensionality reduction using FA barely meets the practical requirements, but still has a large relative error. The proposed model is used to reduce the dimensionality of the stock data. The reduced data has a higher prediction accuracy and meets practical requirements.

Table 3. Errors in predicting stock prices with different coding sequences.

Algorithm	Error		
	MAE	**MSE**	**MRE**
FA_BP	0.062	0.0045	9.72%
PCA_BP	0.127	0.02	17.9%
Proposed DAE_BP	0.043	0.0025	6.94%

In order to explore the reasons why different encoding methods have different effects on the prediction results, this article further analyzes the encoded sequence. Stock sequence is a time series with strong volatility. From a theoretical point of view, the variables used to predict stock prices also need to have similar strong volatility in order to predict

stock prices well. Variance is used in statistics to measure the degree of dispersion of a sequence. The larger the variance, the greater the volatility of the sequence. Similarly, the smaller the variance, the smaller the volatility of the sequence. Therefore, this paper calculates the variance of the coded sequence after dimensionality reduction by various dimensionality reduction methods, which is used to measure the degree of fluctuation of the coded sequence after dimensionality reduction. Among them, *Di* represents the *i*-th dimension of the coding sequence.

Through calculation, in Table 4, the variance of the stock price sequence is 0.061. Analysis of the data in Table 4 shows that the variance of the sequence after DAE dimensionality reduction is the closest to the variance of the stock price, so the following conclusions can be drawn. The DAE should be used to compare stock data to perform dimensionality reduction, and the FA and PCA methods should be used for comparison. When the data is reduced to the same dimension, the DAE can retain more useful information in the original data, and the extracted features are more in line with the main features in the original data.

Table 4. The variance of the coding sequence of stock data after dimensionality reduction.

Algorithm	Variance				
	***D*1**	***D*2**	***D*3**	***D*4**	***D*5**
FA	0.049	0.027	0.039	0.030	0.015
PCA	0.039	0.015	0.038	0.009	0.018
DAE	0.053	0.073	0.086	0.051	0.056

4.4. Comparison with Different Forecasting Methods

This article compares the prediction results of the DAE-BP model with the three commonly used models of Multilayer Perceptron (MLP), Support Vector Regression (SVR) and Multiple Linear Regression (MLR). The MLP, SVR and MLR use stocks without dimensionality reduction. The index data predicts the price, and the prediction results are shown in Table 5. From the data in Table 5, it can be seen that although the MLR has the shortest running time, its prediction effect is the worst. Compared with other prediction models, the proposed algorithm has better performance in terms of prediction accuracy and running time.

Table 5. Comparison of prediction results of different methods.

Algorithm	Running Time	Error		
		MAE	**MSE**	**MRE**
SVR	9.16 ms	0.056	0.0039	7.71%
MLR	0.13 ms	0.160	0.029	22.9%
MLP	0.44 ms	0.063	0.0047	9.98%
Proposed	0.29 ms	0.043	0.0025	6.94%

5. Conclusions

This paper designs a stock prediction model based on the DAE-BP neural network. Since the original indicator data of stocks has a high dimensionality and there is multiple collinearity between the indicators, directly using the original indicators to predict stock prices will greatly increase the complexity of the model and computational overhead. Thus, this article uses a deep autoencoder to reduce the dimensionality of stock data. The effect of DAE dimensionality reduction is related to the number of layers. If the number of layers is too deep or too shallow, the reconstruction error will increase. The DAE with a four-layer structure has the best dimensionality reduction effect. This article also compares the dimensionality reduction effects of the three dimensionality reduction methods of DAE, PCA and FA. When the data is reduced to the same dimension, the data after DAE dimensionality reduction can retain the more essential characteristics of the data and use

it for stocks. It has high accuracy when forecasting. In comparison with other prediction models, DAE-BP has more advantages in terms of running time and prediction accuracy.

Using the model designed in this paper to predict stock prices still produces a large error; the relative error reaches 6.94%, and although this is less than the single-day limit of stocks, it is still misleading in practical applications. Therefore, the next step is to adjust and optimize the prediction part of the model to further reduce the prediction error and make the model more practical.

Author Contributions: Formal analysis, Y.A.A.; Investigation, D.M.A.; Methodology, S.Z.A.-o.; Project administration, Y.A.A.; Resources, D.M.A.; Software, A.A.A.; Validation, A.A.A.; Visualization, A.A.A.; Writing—review & editing, S.Z.A.-o. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Researchers Supporting Project (No. RSP-2021/395), King Saud University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

Araki, Shoko, Tomoki Hayashi, Marc Delcroix, Masakiyo Fujimoto, Kazuya Takeda, and Tomohiro Nakatani. 2015. Exploring multichannel features for denoising-autoencoder-based speech enhancement. Paper presented at IEEE International Conference on Acoustics, Speech and Signal Processing, South Brisbane, QLD, Australia, April 19–24, pp. 116–20.

Babić, Karlo, Sanda Martinčić-Ipšić, and Ana Meštrović. 2020. Survey of neural text representation models. *Information* 11: 511. [CrossRef]

Bakshi, Bhavik. 2010. Multiscale PCA with application to multivariate statistical process monitoring. *Aiche Journal* 44: 1596–610. [CrossRef]

Brandt, Andreas, and Manfred Brandt. 2004. On the two-class M/M/1 system under preemptive resume and impatience of the prioritized customers. *Queuing Systems* 47: 147–68. [CrossRef]

Brandt, Andreas, and Manfred Brandt. 2013. Workload and busy period for M/GI/1 with a general impatience mechanism. *Queuing Systems* 75: 189–209. [CrossRef]

Chopra, Ritika, and Gagan Deep Sharma. 2021. Application of artificial intelligence in stock market forecasting: A critique, review, and research agenda. *Journal of Risk and Financial Management* 14: 526. [CrossRef]

Craighead, Christopher W., Kirk R. Karwan, and Janis L. Miller. 2009. The effects of severity of failure and customer loyalty on service recovery strategies. *Production and Operations Management* 13: 307–21. [CrossRef]

Gao, Shenghua, Yuting Zhang, Kui Jia, Jiwen Lu, and Yingying Zhang. 2015. Single sample face recognition via learning deep supervised autoencoders. *IEEE Transactions on Information Forensics and Security* 10: 2108–18. [CrossRef]

Garnet, Ofer, Avishai Mandelbaum, and Martin. I. Reiman. 2002. Designing a call center with impatient customers. *Manufacturing & Service Operations Management* 4: 208–27.

Glorot, Xavier, and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research* 9: 249–56.

Hai Nguyen, Thien, Kiyoaki Shirai, and Julien Velcin. 2015. Sentiment analysis on social media for stock movement prediction. *Expert Systems with Applications* 42: 9603–11. [CrossRef]

Hecht, Amelie A., Crystal L. Perez, Michele Polascek, Anne N. Thorndike, Rebecca L. Franckle, and Alyssa J. Moran. 2020. Influence of food and beverage companies on retailer marketing strategies and consumer behavior. *International Journal of Environmental Research and Public Health* 17: 7381. [CrossRef]

Hinton, Geoffrey. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation* 14: 1771–800. [CrossRef] [PubMed]

Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science Journal* 313: 504–7. [CrossRef]

Kaiser, Henry F. 2016. The application of electronic computers to factor analysis. *Educational & Psychological Measurement* 20: 141–51.

Le Roux, Nicolas, and Yoshua Bengio. 2008. Representational power of restricted Boltzmann machines and deep belief networks. *Neural Computation* 20: 1631–42. [CrossRef] [PubMed]

Li, Xiaodong, Haoran Xie, Yangqiu Song, Shanfeng Zhu, Qing Li, and Fu Lee Wang. 2015. Does summarization help stock prediction? News impact analysis via summarization. *IEEE Intelligent Systems* 30: 26–34. [CrossRef]

Li, Xiumin, Lin Yang, Fangzheng Xue, and Hongjun Zhou. 2017. Time series prediction of stock price using deep belief networks with intrinsic plasticity. Paper presented at Control and Decision Conference, Chongqing, China, May 28–29, pp. 1237–42.

Lu, Yina, Andrés Musalem, Marcelo Olivares, and Ariel Schilkrut. 2012. Measuring the effect of queues on customer purchases. *Measurement Science* 59: 1743–63. [CrossRef]

Prachyachuwong, Kittisak, and Peerapon Vateekul. 2021. Stock trend prediction using deep learning approach on technical indicator and industrial specific information. *Information* 12: 250. [CrossRef]

Rundo, Francesco, Francesca Trenta, Agatino Luigi di Stallo, and Sebastiano Battiato. 2019. Machine learning for quantitative finance applications: A survey. *Applied Sciences* 9: 5574. [CrossRef]

Salakhutdinov, Ruslan, and Geoffrey Hinton. 2009. Semantic Hashing. *International Journal of Approximate Reasoning* 50: 969–78. [CrossRef]

Van Velthoven, J., Benny Van Houdt, and Chris Blondia. 2005. Response time distribution in a D-MAP/PH/1 queue with general customer impatience. *Stochastic Models* 21: 745–65. [CrossRef]

Wang, Yao, Wan-Dong Cai, and Peng-Cheng Wei. 2016. A deep learning approach for detecting malicious javascript code. *Security & Communication Networks* 9: 1520–34.

Xu, Jun, Lei Xiang, Qingshan Liu, Hannah Gilmore, Jianzhong Wu, Jinghai Tang, and Anant Madabhushi. 2016. Stacked sparse autoencoder for nuclei detection of breast cancer histopathology images. *IEEE Transactions on Medical Imaging* 35: 1193–204. [CrossRef] [PubMed]

Zohar, Ety, Avishai Mandelbaum, and Nahum Shimkin. 2002. Adaptive behavior of impatient customers in tele-queues: Theory and empirical support. *Management Science* 48: 566–83. [CrossRef]