Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 3950981, 9 pages
https://doi.org/10.1155/2021/3950981

Research Article

Fintech Index Prediction Based on RF-GA-DNN Algorithm

Chao Liu, Yixin Fan, and Xiangyu Zhu

School of Economics and Management, Beijing University of Technology, Beijing 100124, China

Correspondence should be addressed to Xiangyu Zhu; zhuxiangyubjut@126.com

Received 22 April 2021; Accepted 27 May 2021; Published 7 June 2021

Academic Editor: Wenqing Wu

The Fintech index has been more active in the stock market with the Fintech industry expanding. The prediction of the Fintech index is significant as it is capable of instructing investors to avoid risks and provide guidance for financial regulators. Traditional prediction methods adopt the deep neural network (DNN) or the combination of genetic algorithm (GA) and DNN mostly. However, heavy computational load is required by these algorithms. In this paper, we propose an integrated artificial intelligence-based algorithm, consisting of the random frog algorithm (RF), GA, and DNN, to predict the Fintech index. The proposed RF-GA-DNN prediction algorithm filters the key input variables and optimizes the hyperparameters of DNN. We compare the proposed RF-GA-DNN with the traditional GA-DNN in terms of convergence time and prediction accuracy. Results show that the convergence time of GA-DNN is up to 20 hours and its prediction accuracy is 97.4%. In comparison, the convergence time of our RF-GA-DNN is only about 1.5 hours and the prediction accuracy reaches 97.0%. These results demonstrate that the proposed RF-GA-DNN prediction algorithm significantly reduces the convergence time with the promise of competitive prediction accuracy. Thus, the proposed algorithm deserves to be widely recommended for predicting the Fintech index.

1. Introduction

The financial industry is reshaped by Fintech with the development of a new round of scientific and industrial revolution. Fintech provides an infinite space for innovative financial products and services [1]. At present, the market scale of the Fintech industry is in the forefront of the world [2]. The rapid expansion of the Fintech industry has drawn the attention of stock market investors to the Fintech index, which consists of 55 publicly listed Fintech companies. However, the Fintech index is subject to many factors, including financial and monetary policy and investor expectations [3, 4]. This may not only bring risk for investors but also affect financial regulation. Therefore, it is of great significance to predict the Fintech index in the stock market accurately and effectively.

Various researches have been conducted to predict indexes in the stock market. As an important stock price index in the stock market, the Fintech index is nonlinear and nonstationary [5]. The Fintech index is volatile and difficult to be predicted by the traditional time series methods. Machine learning algorithms, such as support vector regression (SVR) [6], genetic algorithm (GA) [7], and deep neural network (DNN) [8], have been widely used recently [9, 10]. SVR was introduced by Vapnik originally [11], which has a global optimum, while its hyperparameter selection needs to be determined by the experience of practitioners [5], which has strong subjectivity and may lead to poor performance in prediction [12]. GA is established on the concepts of natural selection and heredity, which can find the parameters of the algorithms that need to be optimized from a global perspective [13]. In recent years, DNN has attracted intense research interest. It has been widely adopted in many fields, such as computer vision, language processing, and speech recognition [14–17]. DNN is an algorithm of deep learning branch, which has a large number of layers and neurons. Therefore, it has a strong fitting ability and high prediction accuracy. However, overfitting is prone to occur because of the large number of layers and neurons [18].

For better prediction, some scholars have improved and integrated the algorithms for their advantages and disadvantages. Xin et al. [19] put forward the GA-SVR due to the good performance of the GA in seeking the parameters of the algorithms that need to be optimized. Similarly, the GA-DNN has recently been proposed for using in the fields of robotics

and speech separation [20, 21]. Although the GA-DNN algorithm's prediction accuracy has been greatly improved, its convergence time needs to be improved furtherly.

From the literature review mentioned above, to date, GA-DNN has not been used in the stock index forecasting research, and its convergence time is long. This study is aimed at filling the gap by proposing an integrated artificial intelligence-based algorithm, consisting of random frog algorithm (RF), GA, and DNN (RF-GA-DNN) to predict the Fintech index. In this new algorithm, RF is adopted to filter all technical variables that affect the Fintech index and some variables that have no contribution or little influence are removed firstly. Then, GA is employed to seek the parameters of DNN. Finally, DNN is used to predict the Fintech index by four times cross-validation.

he remainder of this paper is arranged as follows. Section 2 introduces the steps of RF, GA, and DNN in detail. Section 3 defines the principle of the proposed RF-GA-DNN prediction algorithm. The sample and prediction results are detailed in Section 4. The conclusions are presented in Section 5.

2. Methods

2.1. Random Frog (RF). The random frog algorithm was proposed by Li et al. [22] based on the framework of reversible jump Markov Chain Monte Carlo (MCMC). In this section, the partial least squares linear discriminant analysis (PLS-LDA) is employed to build a classifier [22]. The steps of the random frog algorithm are as follows:

Step 1. Hyperparameter initialization.

(1) N represents the number of iterations of the random frog algorithm Theoretically, the larger the N is, the model works better, whereas it takes more time to compute. Hence, N is set to 10000 according to experience

(2) Q is the number of variables in the initial variable set. Q variables make up the initial variable set V_0

(3) θ refers to a controlling factor for the variance of a normal distribution and a positive real number less than 1 in general. It is set to 0.3 by default

(4) ω is a scaling factor, which is employed to control the number of candidate variables and should be greater than 1. By default, ω is set to 3

(5) η refers to the upper bound of the probability of accepting the new variable set V^*, whose result is lower than V_0. The value ranges from 0 to 1, and the default value of η is 0.1 [22]

Step 2. The initial variable set is randomly selected as V_0, which contains Q variables. Define the set of all variables to be V.

Step 3. Q^* is the nearest rounded random number from the normal distribution with mean Q and standard deviation θQ. Then, a candidate variable subset V^* is constructed, which contains Q^* variables.

(1) If $Q^* = Q$, then $V^* = V$

(2) If $Q^* < Q$, construct a PLS-LDA method by the variable set V_0 firstly, and the regression coefficients for each variable are obtained. Then, $Q - Q^*$ variables with the smallest absolute value of the regression coefficient are removed from V_0, and V^* is composed of the remaining Q^* variables

(3) If $Q^* > Q$, $\omega(Q^* - Q)$ variables are randomly selected from $V - V_0$ to form a new variable set T. Then, T and V_0 are combined to establish the PLS-LDA, and the regression coefficients of each variable are capable of obtaining. V^* is composed of Q^* variables with the largest absolute value of the regression coefficient

Step 4. Determine whether V^* is acceptable or not. The prediction errors of the PLS-LDA established by V_0 and V^* are recorded as e_0 and e^* correspondingly.

(1) If $e^* \leq e_0$, then V^* can be accepted and $V_1 = V^*$

(2) If $e^* \geq e_0$, accept V^* with $\eta e_0/e^*$ and $V_1 = V^*$. Replace V_0 with V_1, and return to Step 2 for the next iteration until N iterations are completed

Step 5. Nvariable subsets can be obtained after N iterations. The frequency of selecting variable i is N_i; then, the selection probability of i is P_i:

$$P_i = \frac{N_i}{N}. \tag{1}$$

The variables can be selected according to their probabilities because the higher the probability of the selected variable, the more important it is.

2.2. Genetic Algorithm (GA). Holland [23] has proposed a random search intelligent algorithm, namely, genetic algorithm (GA), which simulates the evolution process of nature. It should be noted that the hyperparameters of the algorithms, which need to be optimized, can be searched by the GA [13]. The basic steps of GA are as follows:

Step 1. Initializing the population. The initial population is made up of randomly generated individuals. The application of fitness function is employed to analyze fitness factors.

Step 2. Calculating the fitness of individuals and ranking them.

Step 3. Selecting two individuals with high fitness and dealing with crossover and mutation to produce offspring individuals.

Step 4. Return to step 2 and keep optimizing the hyperparameters until the set evolutionary algebra is reached. Then, the model terminates.

2.3. Deep Neural Network (DNN). There are various structures for DNN, but the DNN structure is fixed in this paper. Based on the sample data dimension in this paper and

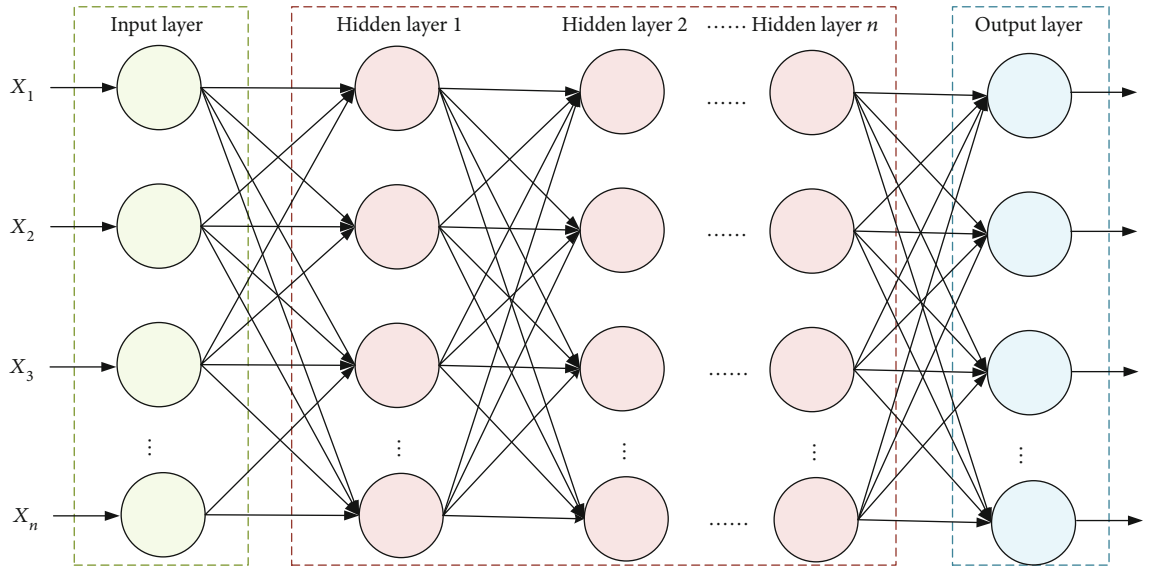

Figure 1: Deep neural network structure.

experience, the structure of DNN is determined as one input layer, three hidden layers, and one output layer. The output of the previous layer can be capable of using as the input of the next layer. The training strategy of DNN is to minimize the value of RMSE, and the samples have been divided randomly according to the rule of 80% training samples and 20% test samples. The prediction accuracy of the DNN model is also affected by some core hyperparameters, such as activation function, training times, number of monolayer neurons, learning rate, and batch size, which will be determined in Section 4.2. The input data of DNN is the 20 technical variables that affect the Fintech index, while the output data is the predicted value of the previous day's closing price of the Fintech Index. The operation steps of DNN are shown as follows.

Step 1. Supposing that $v^l = [v_i^l]$ is the hidden layer vector of layer l. The visible layer vector is $h^l = [h_i^l]$. The connection weight matrix of the hidden layer is $W^l = [w_{ij}^l]$. $a^l = [a_i^l]$ and $b^l = [b_i^l]$ are the bias vectors of the hidden layer and visible layer, respectively.

Step 2. Assigning values to W and a randomly in the range of (0,1).

Step 3. Calculating the probability that a hidden layer unit can be activated.

$$p\left(h_j^l = 1 \mid v^l\right) = \sigma\left(b_j^l + \sum_{i=1}^{I} w_{ij}^l v_i^l\right), \tag{2}$$

where $\sigma(\bullet)$ is set to ReLU function because of its better performance than other functions in this experiment. It can be described as follows:

$$\sigma(x) = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{if } x \le 0. \end{cases} \tag{3}$$

Step 4. The value of the hidden layer unit is determined by Gibbs sampling.

$$h_j^l = \begin{cases} 1, & p\left(h_j^l = 1 \mid v^l\right) > r_j, \\ 0, & p\left(h_j^l = 1 \mid v^l\right) \le r_j, \end{cases} \tag{4}$$

where r_j is the random number generated on [0,1].

Step 5. Calculating the activation probability of the reconstructed visible layer unit and reconstructing the visible layer by Gibbs sampling.

$$p\left(v^{l*} = 1 \mid h^l\right) = \sigma\left(a_i^l + \sum_{j=1}^{n} w_{ji}^l h_j^l\right), \tag{5}$$

where $v^{l*} = [v_i^*]$ is the unit vector of the reconstructed visible layer.

Step 6. Calculating the activation probability of the hidden layer element and updating the parameter values according to the visible layer element vector v^{l*}.

$$\begin{cases} w_{ij}^{l*} = w_{ij}^l + \lambda\left[v_i^l p\left(h_j^l = 1 \mid v^l\right) - v_i^{l*} p\left(h_j^{l*} = 1 \mid v^{l*}\right)\right], \\ b_j^{l*} = b_j^l + \lambda^l\left[p\left(h_j^l = 1 \mid v^l\right) - p\left(h_j^{l*} = 1 \mid v^{l*}\right)\right], \\ a_i^{l*} = a_i^l + \lambda^l\left(v_i^l - v_i^*\right), \end{cases} \tag{6}$$

where w_{ij}^{l*}, b_j^{l*}, and a_i^{l*} are the updated values of w_{ij}^l, b_j^l, and a_i^l, respectively.

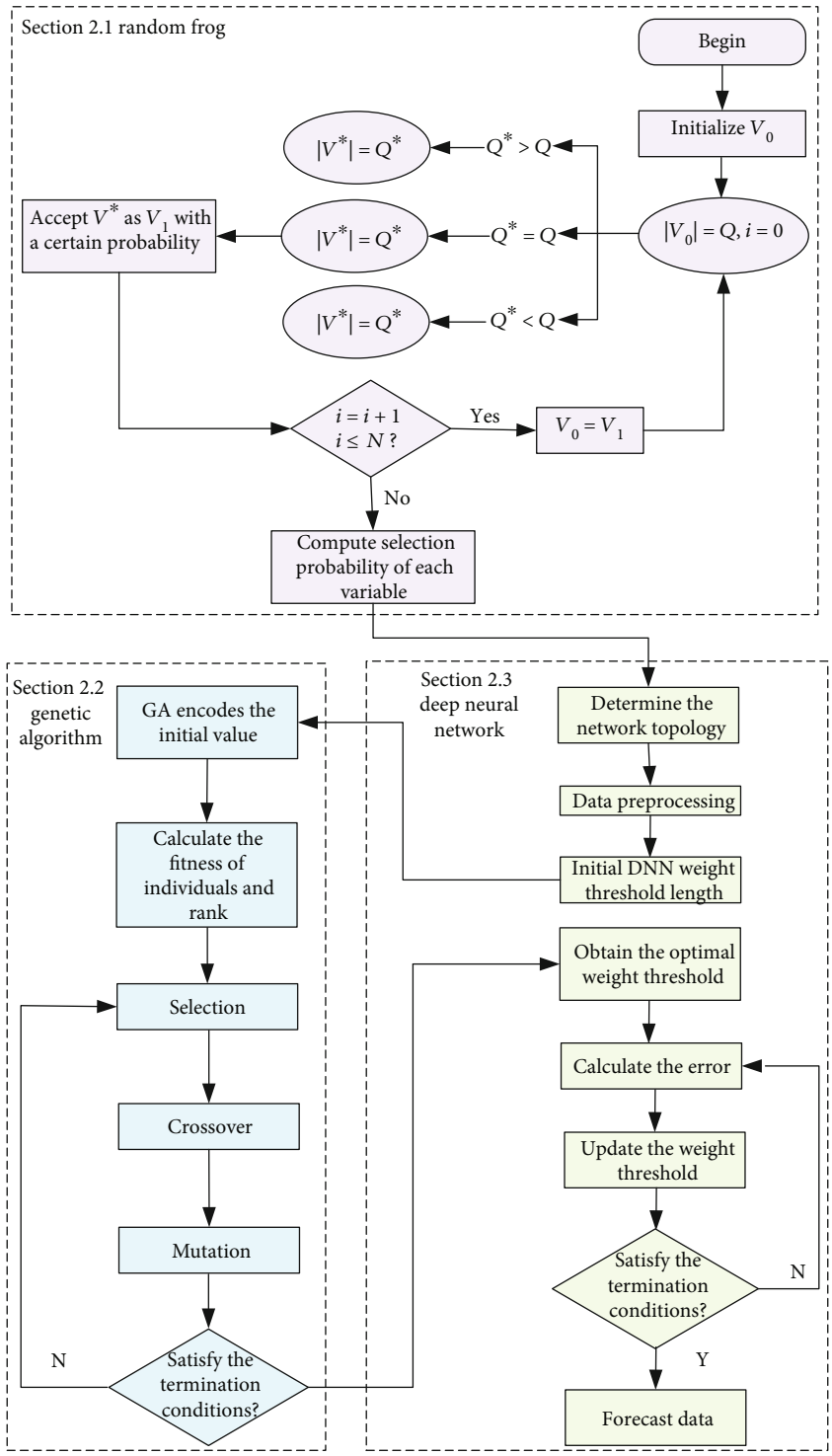

Figure 2: The proposed RF-GA-DNN algorithm workflow.

The input vector y^l of the next neural layer can be expressed as follows:

$$y^l = \sigma\left(W^l a^l + b^l\right). \tag{7}$$

Step 7. Steps 3–6 are looped until the number of iterations is finished. Then, the model terminates.

Table 1: The RMSE of different activation functions.

Activation function	RMSE
Sigmoid	1199.16
Softmax	1315.71
Tanh	1196.33
ReLU	113.74

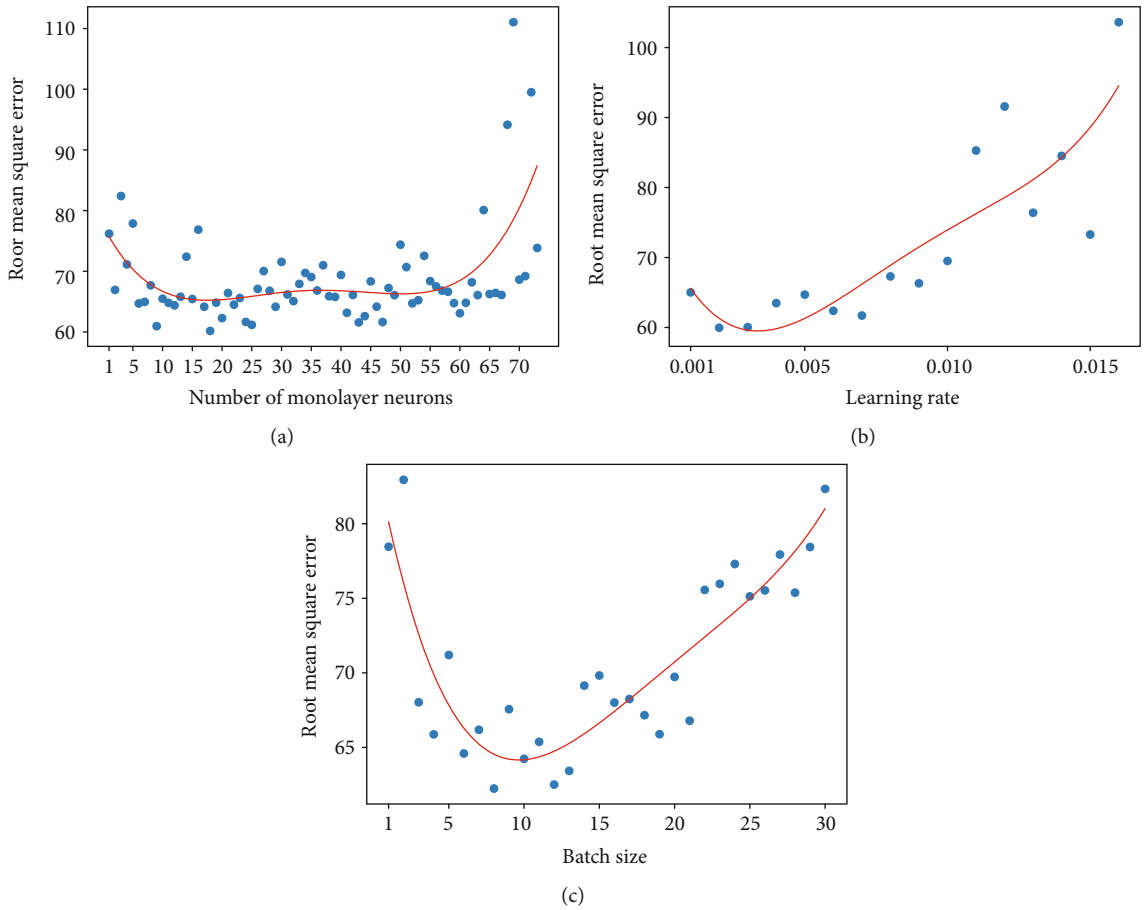

FIGURE 3: The RMSE of monolayer neurons, learning rate, and batch size (the red line represents a polynomial curve for fitting the change trend of the three hyperparameters).

Step 8. Reversing the fine-tuning of parameters by a gradient descent method, which adjusts the weights of interconnections and minimizes the output error.

$$\begin{cases} W^{l*} = W^l - \eta \dfrac{\partial E}{\partial W^l}, \\ b^{l*} = b^l - \eta \dfrac{\partial E}{\partial b^l}, \end{cases} \tag{8}$$

where η is the learning rate and E is the cost function, and it can be expressed as

$$E = \frac{1}{2N} \sum_{n=1}^{N} \| y^n - o^n \|, \tag{9}$$

where y^n and o^n are the predicted and actual values obtained from the nth training sample in one training time, respectively. N is the size of the training sample. The specific DNN network structure is shown in Figure 1.

3. The Proposed RF-GA-DNN Prediction Algorithm

In this section, an integrated artificial intelligence-based algorithm, consisting of random frog algorithm (RF), GA, and DNN, is proposed to predict the Fintech index.

The core hyperparameters of DNN have a significant influence on its prediction accuracy, but the value of the core hyperparameters cannot be determined by calculation directly at present. GA is capable of searching for the optimal hyperparameters of DNN from a global scope. The principle of GA-DNN is to find the optimal hyperparameters through the GA firstly, and then, DNN is employed to predict. However, the convergence time will be increased for too many variables. The random frog algorithm is able to filter the key variables and reduce the convergence time. Therefore, the random frog algorithm has been added to GA-DNN, and we have proposed a hybrid RF-GA-DNN prediction algorithm to reduce the convergence time.

The workflow of this new algorithm is shown in Figure 2. As can be seen from Figure 2, firstly, the random frog algorithm was adopted to screen out input variables for obtaining variables related to the Fintech index. Secondly, the screened variables were imported into DNN with different

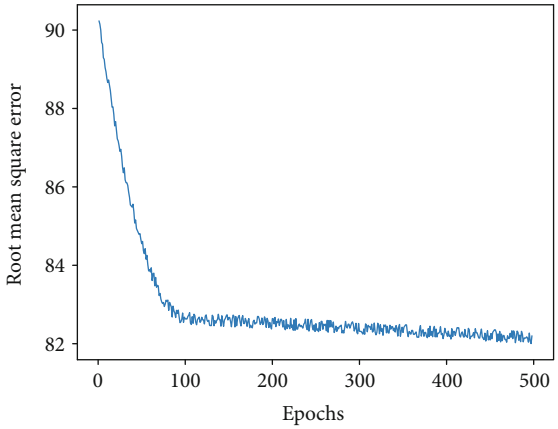

Figure 4: Effect of training times on RMSE.

hyperparameter values to get the predicted values of the test samples. Then, to determine the value range of GA searching for hyperparameters, RMSE of predicted and true values was calculated. And then, within the value range obtained by the GA, take RMSE returned by each DNN model as a fitness function to get the optimal hyperparameter combination of DNN. Finally, the selected variables were imported into the DNN model with the optimal hyperparameter setting from the GA. Then, the predicted values are obtained, and the root mean square error can be gotten by using the predicted and the true values. Specific steps of RF-GA-DNN have been listed in Section 2.1, Section 2.2, and Section 2.3.

4. Experiment and Analysis

4.1. Sample and Variable Selection. The sample interval of this paper is from March 8, 2015, to April 9, 2021, with 1488 daily data totally. The next day's closing price of the Fintech index is selected as the output variable. The input variables include 20 technical variables (MACD, BBI, DDI, DMA, MTM, TRIX, RSI, ROC, B3612, BIAS, CCI, OSC, W&R, MASS, WVAD, CR, PSY, VR, BOLL, turnover rate). 80% of the data are the training samples, and 20% of the data are selected as the test samples. All data comes from the Wind database.

4.2. Core Hyperparameter Selection and Value Range Determination. Common hyperparameters in DNN include activation function, monolayer neurons, learning rate, batch size, and training times [18]. All of them are highly related to the prediction accuracy and convergence time. Therefore, their value range is investigated.

For the activation function, Sigmoid, Softmax, Tanh, and ReLU are the common activation functions. Table 1 tabulates the root mean square error (RMSE) of these activation functions. The root mean square error of the previous day's closing price and forecast price of the Fintech Index is presented by RMSE. Through the analysis of Table 1, the RMSE of ReLU is smaller than Sigmoid, Softmax, and Tanh. Therefore, ReLU is selected as the activation function of the method.

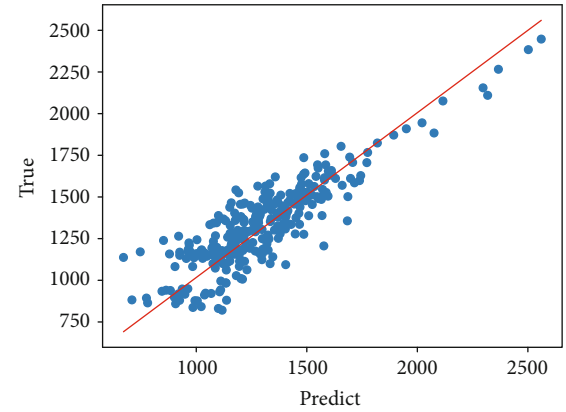

Figure 5: The degree of deviation between the actual and predicted values of DNN (the red line represents the true value equal to the predicted value).

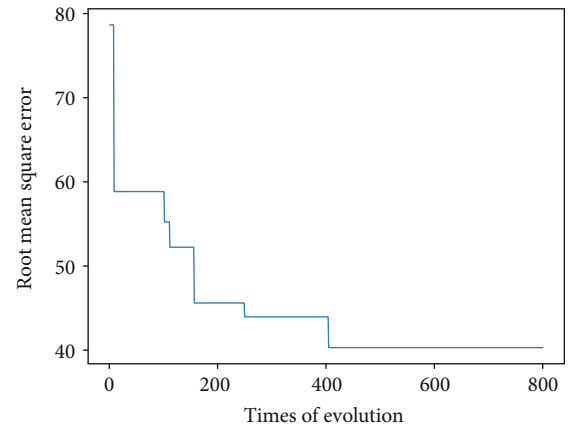

Figure 6: Effect of evolution times on RMSE of GA-DNN.

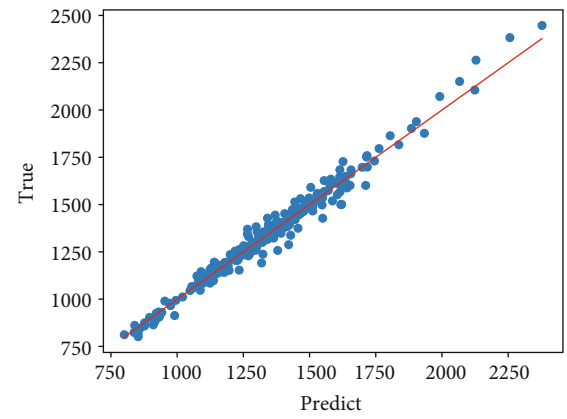

Figure 7: The degree of deviation between the actual and predicted values of GA-DNN (the red line represents the true value equal to the predicted value).

A trial parameter method is employed to obtain the optimization range. The RMSE of the number of monolayer neurons, learning rate, and batch size are shown in Figure 3. Figure 3(a) illustrates that the RMSE of the number of monolayer neurons is smaller in [10, 65]. Figure 3(b) reveals that the learning rate error can be fixed in [0.001, 0.015].

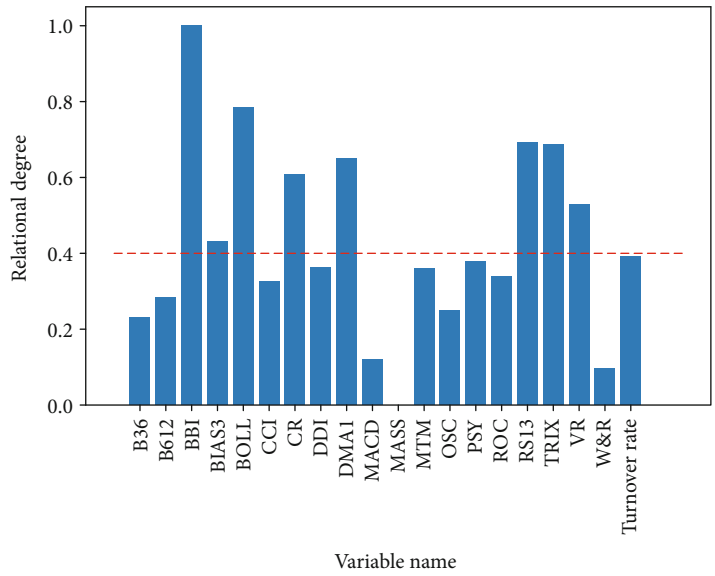

Figure 8: Correlation of different variables.

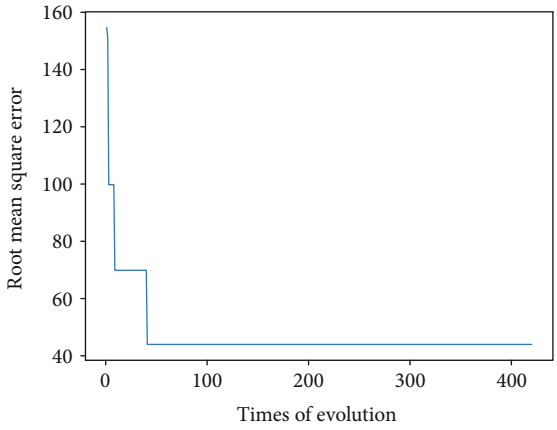

Figure 9: Effect of evolution times on RMSE of RF-GA-DNN.

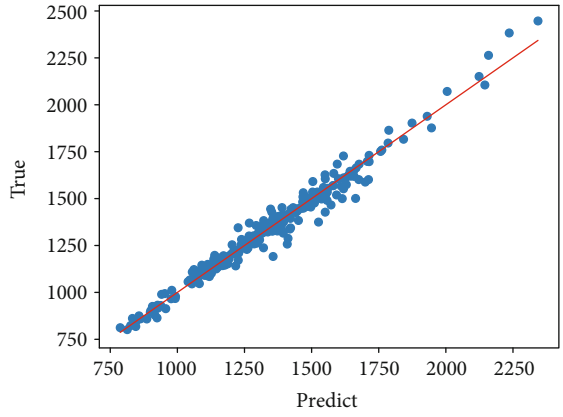

Figure 10: The degree of deviation between the actual and predicted values of RF-GA-DNN (the red line represents the true value equal to the predicted).

According to the analysis in Figure 3(c), [2, 20] is a better range for batch size due to its smallest RMSE in this interval.

To determinate the training times, based on the experience of deep learning, the number of monolayer neurons is set to 10 preliminarily, the learning rate is 0.001, and the batch size is random gradient descent by default. The RMSE has been obtained through four-time cross-validation, which is shown in Figure 4. Figure 4 reveals that the RMSE is significantly reduced between 0 and 100 training times, while there is almost no difference between 100 and 500 training times. Then, 100 is regarded as the number of training times for DNN, and the degree of deviation between the actual and predicted values of DNN is shown in Figure 5. We see that the prediction effect can be further optimized. However, it is not advisable to increase the training times. Therefore, the following models (including DNN, GA-DNN, and RF-GA-DNN) were completed under the condition of running a single DNN training of 100 times.

In summary, we select ReLU as the activation function. The range of number of monolayer neurons, learning rate, and batch size is [10, 65], [0.001, 0.015], and [2, 20], respectively. 100 is set to the training times. The values of monolayer neurons, learning rate, and batch size can only be determined in a range by a trial parameter method, and then, the optimal combination of the hyperparameters can be found by the GA. In addition, the experiment shows that the single operation time of DNN is about 10 s when a specific hyperparameter combination is given. However, an enumeration method is needed to determine the optimal hyperparameter combination. By calculation, it will take years and be difficult to achieve.

Table 2: The RMSE of different activation functions.

Algorithm	Convergence time	Prediction accuracy
GA-DNN	20 h	97.4%
RF-GA-DNN	1.5 h	97.0%

4.3. Prediction of GA-DNN Algorithm. Based on the analysis of Section 4.2, GA is introduced in DNN due to its excellent performance in searching for the optimal hyperparameters' combination. According to the sample data dimension and experience, the number of hidden layers in DNN is set to 3. The optimal hyperparameter combination searched by the GA is as follows: the number of neurons in the three hidden layers is 35, 6, and 12; the learning rate is 0.002; and the batch size is 8. Input the optimal hyperparameter combination into DNN, train 100 times, and iterate 800 generations (each generation has 20 individuals). The optimal RMSE and forecast renderings can be gained, as shown in Figures 6 and 7.

Figure 6 shows the effect of the evolutionary times on the RMSE of GA-DNN. It can be seen from Figure 6 that GA-DNN converges from generation 405, and the RMSE is 40.49 after convergence. Figure 7 shows the degree of deviation between the actual and predicted values of GA-DNN. The prediction accuracy of GA-DNN is 97.4%. However, it is worth noting that the convergence time of GA-DNN is 73694 s (about 20 hours). From this, we know that the convergence time of GA-DNN is still too long and it could be optimized furtherly.

4.4. Prediction of RF-GA-DNN Algorithm. Random frog is an efficient approach for variable selection [22], which can move between fixed-dimensional and transdimensional in different methods to realize a search. We introduced the random frog algorithm into GA-DNN to screen out key variables. Variables filtered by the random frog algorithm are shown in Figure 8. We take 0.4 as the boundary to filter variables based on the characteristics of our variables. The optimal hyperparameter combination of RF-GA-DNN is as follows: the number of neurons in the three hidden layers is 23, 17, and 13; the learning rate is 0.001; and the batch size is 12.

Figure 9 illustrates the effect of the evolutionary times on the RMSE of RF-GA-DNN. It can be concluded that RF-GA-DNN converges from generation 41, and the RMSE is 43.15 after convergence. The convergence time is 5462 s (about 1.5 hours), which is much less than GA-DNN. Figure 10 shows the degree of deviation between the actual and predicted values of RF-GA-DNN. The prediction accuracy of RF-GA-DNN is 97.0%, which is very close to GA-DNN.

In summary, the comparison results of GA-DNN and RF-GA-DNN are shown in Table 2. As seen in Table 2, it can be concluded that the convergence time of RF-GA-DNN is only 1/13 of GA-DNN with the promise of a competitive prediction accuracy. Hence, RF-GA-DNN has an excellent performance in convergence time.

5. Conclusions

Predicting the Fintech index benefits various stakeholders, e.g., assisting investors to design profitable short, medium, and long-term strategies for investing and guiding financial regulators to make precise and effective regulatory policies. However, the convergence time of current prediction algorithms is far from eligible bearing this in mind. In this paper, we have proposed a hybrid RF-GA-DNN algorithm and employed this algorithm to predict the Fintech index.

We have examined the performance of the proposed RF-GA-DNN on predicting the Fintech index. The samples come from the Wind database with a time period from March 8, 2015, to April 9, 2021. Through the analysis of samples, the proposed hybrid RF-GA-DNN has endowed with excellent superiority compared with the traditional GA-DNN. More importantly, the convergence time of RF-GA-DNN is only 1/13 of that of GA-DNN. These results demonstrate that the proposed hybrid RF-GA-DNN prediction algorithm can be employed as a more effective tool to predict the Fintech index. In practical applications, the hybrid RF-GA-DNN prediction algorithm can be expected to not only provide a reference for investors to formulate investment strategies but also support financial regulators to supervise the market.

On the whole, this paper has proposed an efficient algorithm to predict the Fintech index. However, it still remains an open issue to determine a reasonable evolutionary algebra range to guarantee the convergence of the algorithm. For future work, it is worthwhile to provide a convergence analysis for the proposed algorithm. In addition, the proposed hybrid RF-GA-DNN prediction algorithm can be further utilized as an efficient tool to deal with other indexes in the stock market.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there is no conflict of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (grant numbers: 61773029 and 62073007).

References

[1] A. Alexandrov, R. Bergmann, S. Ewen et al., "The stratosphere platform for big data analytics," *The VLDB Journal*, vol. 23, no. 6, pp. 939–964, 2014.

[2] McKinsey, "What's next for China's booming fintech sector?," 2016, https://www.mckinsey.com/industries/financial-services/our-insights/whats-next-for-chinas-booming-fintech-sector.

[3] J. S. Chou and T. K. Nguyen, "Forward forecast of stock price using sliding-window metaheuristic-optimized machine-learning regression," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3132–3142, 2018.

[4] S. Chen, Y. Sun, and Y. Liu, "Forecast of stock price fluctuation based on the perspective of volume information in stock and exchange market," *China Finance Review International*, vol. 8, no. 3, pp. 297–314, 2018.

[5] A. Kazem, E. Sharifi, F. K. Hussain, O. K. Hussain, and M. Saberi, "Support vector regression with chaos-based firefly algorithm for stock market price forecasting," *Applied Soft Computing*, vol. 13, no. 2, pp. 947–958, 2013.

[6] V. V. Gavrishchaka and S. Banerjee, "Support vector machine as an efficient framework for stock market volatility forecasting," *Computational Management Science*, vol. 3, no. 2, pp. 147–160, 2006.

[7] K. Dasgupta, B. Mandal, P. Dutta, S. Dam, and J. K. Mandal, "A genetic algorithm (GA) based load balancing strategy for cloud computing," *Procedia Technology*, vol. 10, pp. 340–347, 2013.

[8] J. Duan, "Financial system modeling using deep neural networks (DNNs) for effective risk assessment and prediction," *Journal of the Franklin Institute*, vol. 356, no. 8, pp. 4716–4731, 2019.

[9] A. Dingli and K. S. Fournier, "Financial time series forecasting – a deep learning approach," *International Journal of Machine Learning and Computing*, vol. 7, no. 5, pp. 118–122, 2017.

[10] H. Cao, T. Lin, Y. Li, and H. Zhang, "Stock price pattern prediction based on complex network and machine learning," *Complexity*, vol. 2019, Article ID 4132485, 12 pages, 2019.

[11] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, 1995.

[12] C. Y. Yeh, C. W. Huang, and S. J. Lee, "A multiple-kernel support vector regression approach for stock market price forecasting," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2177–2186, 2011.

[13] A. Lambora, K. Gupta, and K. Chopra, "Genetic algorithm-a literature review," in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pp. 380–384, Faridabad, India, 2019.

[14] W. Liu, Z. Wang, X. Liu, Y. Liu, F. E. Alsaadi, and N. Zeng, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[15] D. Yu and L. Deng, "Deep learning and its applications to signal and information processing [exploratory dsp]," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 145–154, 2011.

[16] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649, Vancouver, BC, Canada, 2013.

[17] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing [review article]," *IEEE Computational Intelligence Magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[18] S. Zhong, J. Hu, X. Fan, X. Yu, and H. Zhang, "A deep neural network combined with molecular fingerprints (DNN-MF) to develop predictive models for hydroxyl radical rate constants of water contaminants," *Journal of Hazardous Materials*, vol. 383, article 121141, 2020.

[19] N. Xin, X. Gu, H. Wu, Y. Hu, and Z. Yang, "Application of genetic algorithm-support vector regression (GA-SVR) for quantitative analysis of herbal medicines," *Journal of Chemometrics*, vol. 26, no. 7, pp. 353–360, 2012.

[20] S. Sivapatham, R. Ramadoss, A. Kar, and B. Majhi, "Monaural speech separation using GA-DNN integration scheme," *Applied Acoustics*, vol. 160, no. 3, article 107140, 2020.

[21] X. Chen, Q. Zhang, and Y. Sun, "Evolutionary robot calibration and nonlinear compensation methodology based on GA-DNN and an extra compliance error model," *Mathematical Problems in Engineering*, vol. 2020, Article ID 3981081, 15 pages, 2020.

[22] H. D. Li, Q. S. Xu, and Y. Z. Liang, "Random frog: an efficient reversible jump Markov Chain Monte Carlo-like approach for variable selection with applications to gene selection and disease classification," *Analytica Chimica Acta*, vol. 740, no. 8, pp. 20–26, 2012.

[23] J. H. Holland, *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, Ml, 1975.