Knowledge and Information Systems (2021) 63:2693–2718
https://doi.org/10.1007/s10115-021-01602-3

REGULAR PAPER

A turning point prediction method of stock price based on RVFL-GMDH and chaotic time series analysis

Junde Chen[1] · Shuangyuan Yang[1] · Defu Zhang[1] · Y. A. Nanehkaran[1]

Received: 11 May 2020 / Revised: 18 July 2021 / Accepted: 24 July 2021 / Published online: 26 August 2021

Abstract
Stock market prediction is extremely important for investors because knowing the future trend of stock prices will reduce the risk of investing capital for profit. Therefore, seeking an accurate, fast, and effective approach to identify the stock market movement is of great practical significance. This study proposes a novel turning point prediction method for the time series analysis of stock price. Through the chaos theory analysis and application, we put forward a new modeling approach for the nonlinear dynamic system. The turning indicator of time series is computed firstly; then, by applying the RVFL-GMDH model, we perform the turning point prediction of the stock price, which is based on the fractal characteristic of a strange attractor with an infinite self-similar structure. The experimental findings confirm the efficacy of the proposed procedure and have become successful for the intelligent decision support of the stock trading strategy.

Keywords Turning point prediction · Chaotic time series · Phase space reconstruction · RVFL-GMDH · Stock market

1 Introduction

The stock market is a complicated and varied system due to the chaotic, blaring, and non-stationary data. The fast and accurate prediction of the stock market becomes challenging among the investors to invest the capital for making profits. Therefore, plenty of previous research has been done in this field, and they can be generally summarized in two categories: linear models based on statistical theories; and nonlinear models based on machine learning

✉ Defu Zhang
dfzhang@xmu.edu.cn

Junde Chen
23020180155666@stu.xmu.edu.cn

Shuangyuan Yang
yangshuangyuan@xmu.edu.cn

Y. A. Nanehkaran
artavil20@gmail.com

[1] School of Informatics, Xiamen University, Xiamen 361005, China

[1, 2]. In the existing methods of statistical models, time series [3], gray [4], vector autoregression (VAR) [5], and others are usually used. These prediction methods such as linear regression, moving average, and GARCH are much favorable for financial time series forecasting because of their interpretability [6]. They perform well in linear and stationary time series while do not accomplish ideally on the nonlinear and non-stationary data.

Apart from the traditional statistical methods, data mining and machine learning have carved their own niches in time series analysis [7, 8]. Regarding data mining studies utilized in daily stock data, there are prediction works based on the support vector machine (SVM) [9, 10]. To determine whether the new pattern data belong to a certain category, artificial neural networks (ANN) [11, 12] has achieved successful predictions even with complicated relationships between variables. The fuzzy time series (FTS) [13, 14] method is also employed to predict and identify the time series variation. Including particle swarm optimization (PSO), genetic algorithm (GA), and rough sets theory (RS), the well-known optimization algorithms have been used in this field as well. Besides, some other prediction research works are performed using the method of word analysis for several news articles [15, 16], etc. In particular, since White first employed the ANN to predict the daily return rate of IBM ordinary stock [17], the usage of the ANN in stock price prediction has become a hot research topic [18–20]. More recently, the deep learning method, and especially convolutional neural networks (CNN), has shown impressive performance in pattern recognition and classification; it is also employed in forecasting stock price from historical exchange data [21]. For example, Tsantekidis et al. [22] proposed a deep learning methodology, based on Convolutional Neural Networks, to predict the stock prices, and they gained the highest 67.38% precision in 20 prediction horizons. Deng et al. [23] trained a recurrent deep neural network (NN) for real-time financial signal representation and trading; the experimental results on both the stock and the commodity future markets demonstrated the effectiveness of their proposed approach. Zarkias et al. [24] reported a novel price trailing approach based on deep reinforcement learning (RL) that went beyond traditional price forecasting, and the experiment conducted on a real dataset demonstrated their proposed method outperforming a competitive deep RL method. Besides, some designed recurrent neural networks (RNNs) have been applied to forecasting the stock data as well [25, 26]. These studies make full use of the advantages of neural networks such as self-adapting, self-learning, distributed processing, etc., and overcome the shortcomings of conventional prediction methods. Nevertheless, there are still problems for these neural network methods, including low precision, slow convergence, and easy inclination to a local minimum. In addition, a very large number of parameters that need to be trained by a vast amount of labeled data is also a problem for deep neural networks. On the other hand, such the price trailing forecast, which is a variant of the price prediction method, does not really overcome the limitations of traditional price forecasting influenced by chaos, volatility, and noise interference. Despite these constraints, the previous studies successfully indicated the potential of neural network methods in stock forecasting.

As mentioned previously, most of the existing research considered the prediction of price changes with the aim of creating accurate models that predict the exact value of a stock price instead of the trading strategy itself, such as determining the buy or sell points regarding the knowledge of intelligent stock trading. However, in reality, investors are more concerned with making trading decisions than forecasting daily prices. So far, there are only a few examples of research that involve the turning points of a stock, and notably this problem is still a large one as regards academic researchers and industrial practitioners [1, 11]. Reviewing the latest literature, we found the following research correlated with the turning point prediction. Tang et al. [27] integrated piecewise linear representation (PLR) and weighted support vector machine (WSVM) for forecasting stock turning points. Intachai et al. [28] reported a Local

Average Model (LAM) to predict the turning points of the stock price. Chang et al. [29] applied an ANN model to learn the connection weights from historical turning points, and afterward, an exponential smoothing based dynamic threshold model was used to forecast the future trading signals, etc. The commonly used algorithms including SVM, ANN, and others have shown the effectiveness in stock turning point prediction. However, as stated earlier, the conventional algorithms such as ANNs exist some disadvantages in avoiding over-fitting and trapping at a local minimum. Thus, referring to the above literature, this paper proposes a RVFL-GMDH based turning point prediction method for the time series analysis of stock price. Through chaotic time series analysis, the turning signal $\Phi(t)$ of stock price can be got in advance. Then, the RVFL-GMDH networks, which randomly initializes the weight and bias parameters between the input layer and the hidden layer is employed for the turning point prediction of the stock price. A series of experiments are performed and the empirical analysis results demonstrate the validity of the proposed procedure.

The rest of this writing is structured in the following hierarchy. Section 2 introduces the chaotic time series analysis and describes the theoretical background. Section 3 discusses the modeling methodology to accomplish the task of turning point prediction along with related concepts and the proposed procedure. Section 4 dedicates to the algorithm experiments; multiple experiments are conducted as well as empirical analysis implemented in practical application scenarios. This paper is ultimately summarized in Sect. 5.

2 Chaotic time series analysis

2.1 Determination of chaotic attractor

Generally, it is easy to get a set of time series values in the economic system, i.e., the one-dimensional information of the system. Although the one-dimensional information contains the characteristics of the system, the dynamic or multi-dimensional features of the system cannot be reflected by this one-dimensional representation fully, and thus some multi-dimensional features of the system may be lost. Therefore, Packard proposed to reconstruct the phase space with the delay coordinates of a variable in the original system, and Takens proved that a suitable embedding dimension could be found [30]. It means that if the dimension of the delay coordinate m is greater than or equal to $2d + 1$ ($m \geq 2d + 1$, d is the dimension of the dynamic system), the regular trajectory (attractor) can be restored in the embedding dimension space. Thus, on the orbit of the reconstructed R^m space, the original dynamic system maintains differential homeomorphism, which lays a solid theoretical foundation for the prediction of chaotic time series.

Let the chaotic time series of a single variable be $\{x(i), i = 1, 2, ..., N\}$: Then, if the embedding dimension m and time delay τ are selected appropriately, the phase space of the time series can be expressed in Eq. (1).

$$\begin{aligned} (t_i) &= [x(t_i), x(t_i + \tau), x(t_i + 2\tau), \ldots, x(t_i + (m-1)\tau)] \\ i &= 1, 2, \ldots, m \end{aligned} \tag{1}$$

where τ represents the delay time, m indicates the embedding dimension. Referring to Takens' theorem, the dynamic characteristics of the attractor can be recovered in the sense of topological equivalence.

A small embedding dimension m_0 is first given to form a reconstructed phase space, and the distance between vectors in phase space can be defined in Eq. (2).

$$\left|y_i - y_j\right| = \max_{1 \le k \le m_0} \left|y_{ik} - y_{jk}\right| \tag{2}$$

The correlation integral of embedded time series is computed by

$$C_{m_0}(r) = \frac{1}{N^2} \sum_{i,j=1}^{N} \theta(r - \left|y_i - y_j\right|) \tag{3}$$

where $\theta(\cdot)$ is the Heaviside function, r denotes a certain probability value, and $C(r)$ is a cumulative distribution function, representing the probability that the distance in phase space is less than r. The function $\theta(\cdot)$ is expressed using Eq. (4).

$$\theta(x) = \begin{cases} 0, & x \le 0 \\ 1, & x > 0 \end{cases} \tag{4}$$

For an appropriate range of r, the dimension d of attractors and the cumulative distribution function $C(r)$ should satisfy the logarithmic linear relationship, as written in Eq. (5).

$$d(m_0) = \ln C_{m_0}(r) / \ln r \tag{5}$$

Thus, $d(m_0)$ is the calculated value of the correlation dimension corresponding to m_0. Increase the embedding dimension ($m_1 > m_0$), and repeat the above calculation process until the corresponding dimension value $d(m)$ no longer changes with a certain error range as m increases. The d obtained at this time is the correlation dimension of the attractor. If d increases with m and does not converge to a stable value, it indicates that the time series is a random time series and does not have the characteristics of chaos dynamics, which provides a basis for the identification of chaotic time series.

2.2 Chaotic analysis prediction

According to Packard's theorem of reconstructive phase space and Takens' proof [30] of maintaining differential homeomorphism between reconstructed R^m space and original dynamical system, we can obtain a homeomorphic dynamical system in m dimensional phase space R^m. Thus, there is a smooth map f: $R^m \to R^m$, and it gives the nonlinear dynamic relationship in phase space, as expressed in Eq. (6).

$$X(t+1) = f(X(t)) \tag{6}$$

For the time series $x_1, x_2, \ldots, x_{n\text{-}1}, x_n$, the phase space can be reconstructed on the basis of the calculated optimal embedding dimension m and time delay τ. It is defined as:

$$X(t_i) = [x(t_i - \tau), x(t_i - 2\tau), \ldots, x(t_i - m\tau)], \quad i = 1, 2, \ldots \tag{7}$$

From Takens' theorem, the dynamic characteristics of the attractor can be recovered in the sense of topological equivalence. Therefore, a dynamic system F: $R^m \to R$ can be obtained on the space R^m, which satisfies Eq. (8).

$$x(t_i - \tau + 1) = F[x(t_i - \tau), x(t_i - 2\tau), \ldots, x(t_i - m\tau)], \quad i = 1, 2, \ldots \tag{8}$$

where $F(\cdot)$ is a mapping from m-dimensional states to a one-dimensional real number. If the resolution or dynamic equation for the mapping $F(\cdot)$ can be obtained, it is possible to predict the time series according to the inherent regularity of the chaotic time series.

For most of the time series is the complex and nonlinear system, there are some difficulties in obtaining the functional equation directly, and the strong nonlinear mapping ability of neural networks just provides a good way to deal with such issues. Moreover, the Kolmogorov continuity theorem [31] tells us: for any continuous function $\psi: E^m \to R^n$, $Y = \psi(x)$, the ψ can be precisely implemented by a three-layer neural network, where E^m is the m dimensional unit cube $[0,1]^m$. Thus, this theorem mathematically guarantees the feasibility of neural networks for chaotic time series prediction.

3 Nonlinear modeling based on RVFL-GMDH

3.1 Related theories

3.1.1 RVFL neural networks

Random vector functional link (RVFL) networks proposed first by Pao et al. [32] is a randomized version of feedforward neural networks and has received much attention all along due to its universal approximation capability and outstanding generalization performance [33–37]. Different from the conventional neural networks, the RVFL networks randomly initializes all the weight and bias parameters ($\{w_j, b_j\}, j = 1,2,\ldots,m$) between the input layer and hidden layer; it should be noted that these parameters are fixed and do not need to be tuned during the whole training stage. In addition, compared with the influential extreme learning machines (ELM) [38] and random kitchen sinks (RKS) [39] neural networks, there are extra direct connections between the input layer and the output layer for the RVFL, although they are all random networks. Also, the RVFL as well as ELM randomly fix the weight and bias in the hidden layer and RKS adopts a bank of arbitrary randomized nonlinearities. Figure 1 depicts a schematic diagram of the RVFL networks and a brief description of each layer is presented below.

Input layer

The main function of the input layer is to enter a training set $\{(x_i, y_i)\}$ with n samples, where $i = 1, 2,\ldots, n$, $x \in R^n$, $y \in R$.

Hidden layer

The hidden layer obtains the value of the activation function ($h(\cdot)$) for each hidden layer node, and the *sigmoid* function is usually employed to compute the value of $h(\cdot)$, as defined in Eq. (9).

$$h(x, w, b) = \frac{1}{1 + \exp\{-w^T x + b\}} \tag{9}$$

where w and b represent the weights and biases from the input layer to the hidden layer, respectively. Then, the kernel mapping matrix H of the hidden layer can be formed to calculate the output, as written using Eq. (10).

$$H = \begin{bmatrix} h_1(x_1) & \cdots & h_k(x_1) \\ \vdots & \ddots & \vdots \\ h_1(x_n) & \cdots & h_k(x_n) \end{bmatrix} \tag{10}$$

where k is the number of hidden layer nodes.

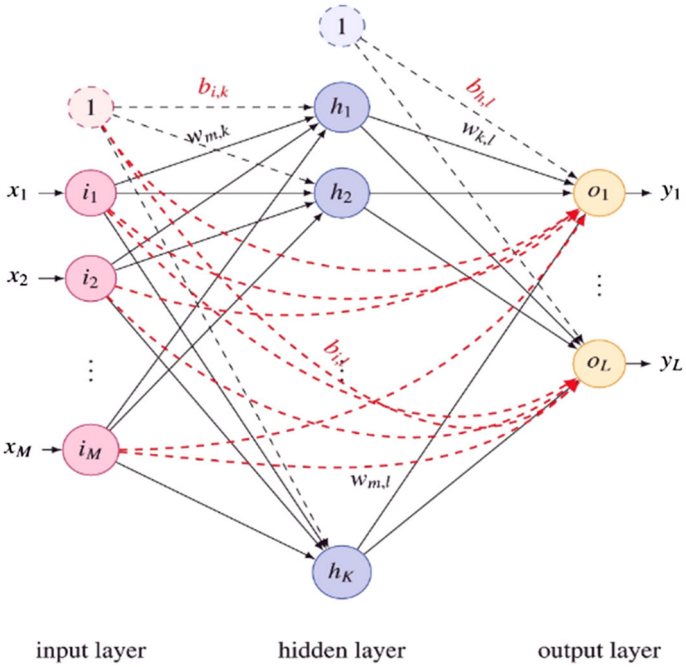


Fig. 1 The schematic diagram of RVFL networks [33]

Output layer

The key task of mode training for the RVFL networks is to learn the optimal weights W_o from the hidden layer to the output layer, which can be calculated using the least-square method and eventually solved by

$$W_o = (H^T H)^{-1} H^T Y \tag{11}$$

where Y is the training target.

3.1.2 GMDH method

Group method of data handling (GMDH) is the core algorithm of self-organizing data mining, and it can determine the variables to enter the model, confirm the structure and parameters of the model in a self-organizing manner [40–42]. GMDH is suitable for the modeling of nonlinear complex systems. Figure 2 depicts a typical GMDH network architecture.

As seen in Fig. 2, the initial input variables (models) are combined with each other to generate the intermediate candidate models, and the optimal intermediate models are selected (e.g., black-filled nodes) by the operations such as heredity, mutation; Then, after further iterating, the processes of heredity, mutation, selection, and evolution are repeated, and the complexity of the intermediate models are continuously increased until the optimal complexity model is obtained.

Different from the classical ANN family, the GMDH adopts the form of mathematical description which is termed the referential function to establish a general mapping relationship between the input and output variables for modeling. The discrete form of the Volterra

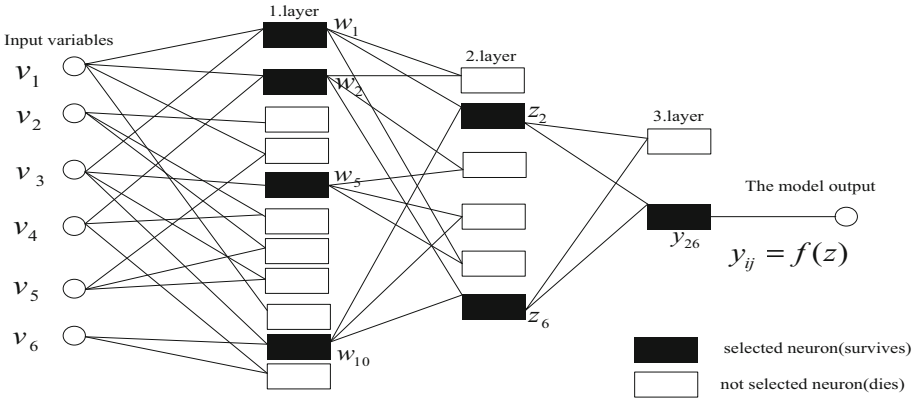


Fig. 2 A typical GMDH network

functional series or Kolmogorov–Gabor (KG) polynomial is usually taken into account for the referential function. Most frequently, K-G polynomial is utilized as the initial input model of this algorithm, and the K-G polynomial composed of variables $(x_1, x_2,\ldots, x_m)$ is established by Eq. (12)

$$y = f(x_1, x_2, ..., x_m) = \sum_{i=1}^{m} a_i x_i + \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij} x_i x_j + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m} a_{ijk} x_i x_j x_k + \cdots \quad (12)$$

where $X = (x_1, x_2,\ldots, x_m)$ represents the vector of input variables, $a = (a_1, a_2,\ldots, a_m)$ is the vector of coefficients or weights, and y denotes the output variable. Ideally, as the increase in independent variables and polynomial degree (aliased as complexity) [40], the polynomial sequence can fit numeric data with any required precision. Therefore, the GMDH method is usually employed to address the prediction problem in practical application scenarios of various domains.

3.2 Proposed approach

3.2.1 RVFL-GMDH modeling

As mentioned earlier, RVFL has the nonlinear modeling ability and the advantages of simplicity, optimal approximation, and fast solution by using the randomization method, while GMDH can resist noise interference, effectively avoid the over-fitting problem, and has interpretability. There are just some complementarities between the two algorithms. Therefore, by taking the merits of both, the modified RVFL and GMDH networks were fused to generate a new model called RVFL-GMDH, which was used for the turning point prediction of the stock price. Figure 3 depicts the schematic of the model, and the specific descriptions of these processes are presented below.

1. For a given dataset D, it is divided into the training set A and test set B, thus $D = A + B$. Suppose the training set $A = \{x_1,\ldots, x_n\}$ with n samples input to the model, where $i = 1,2,\ldots,n$, $x \in R^n$.

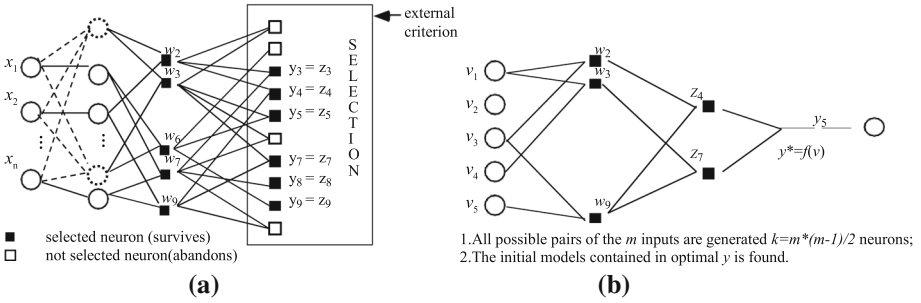


Fig. 3 **a** Network structure of RVFL-GMDH, and **b** Output predicted value

2. Using the input variables, the value of the nodes in the first hidden layer is calculated by the activation function $h(\cdot)$, where the *sigmoid* function is employed, as expressed in Eqs. (13,14).

$$f(x_i)=\sum_{i=1}^{n}(w_i x_i + b_i) \tag{13}$$

$$h_i = 1/(1+e^{-f(x_i)}) \tag{14}$$

where w_i and b_i represent the random weight and bias of RVFL, x_i is the input variable, and h_i denotes the output of the activation function.

3. After conducting Step 2, the K-G polynomial is employed as the reference function for the proposed method, as seen in Eq. (12). The form of the first order K-G polynomial including n neurons (variables) is displayed as follows:

$$f(x_1, x_2, ..., x_n) = a_0 + a_1x_1 + a_2x_2 + \cdots + a_nx_n \tag{15}$$

4. Generate the candidate models: By pairwise coupling, the nodes in the former layer are combined to generate the intermediate candidate models, which are regarded as the new input of the next layer. Specifically, considering all the sub-items of Eq. (15), there are $n+1$ initial input models: $v_1 = a_0$, $v_2 = a_1x_1$, …, $v_{n+1} = a_nx_n$, and every two nodes are composed as one unit with the reference function $y = f(v_i, v_j) = a_1 + a_2v_i + a_3v_j$. Therefore, there are $n_1 = C_{n_0}^2$($n_0 = n+1$) candidate models in the hidden layer:

$$y_k^1 = a_1^k + a_2^k v_i + a_3^k v_j, \quad i,\ j = 1, 2, ..., n_0, i \neq j, \quad k = 1, 2, ..., n_1 \tag{16}$$

where $y_k{}^1$ represents the estimation output; $a_1{}^k$, $a_2{}^k$, $a_3{}^k$ (k = 1,2,…,n_1) are the coefficients obtained in the training set by the least-squares (LS) method.

5. Model selection: Based on the threshold measurement (external criterion), F_1 ($\leq n_1$) candidate models are selected as the input of the next layer. Similarly, there are also $n_2 = C_{F_1}^2$ intermediate candidate models obtained in the subsequent layer:

$$y_k^2 = b_1^k + b_2^k y_i^1 + b_3^k y_j^1, \quad i,\ j = 1, 2, ..., F_1, i \neq j, k = 1, 2, ..., n_2 \tag{17}$$

6. Repeat Steps 4–5 until obtaining the optimal complexity model by termination principle (see Fig. 3a). A brief description of the above processes is displayed in Algorithm 1.

Algorithm 1: RVFL-GMDH modelling.

Input:

The sample training set $A=\{x_1,\ldots, x_n\}$, where $i=1,2,\ldots,n$, $x\in R^n$.

Begin

1: Define the input layer and confirm the initial input variables of the model

2: Borrowing the idea of RVFL, randomize the weight and bias parameters from the input layer to the hidden layer and calculate the new nodes except original input using sigmoid function $h(\cdot)$

$$h(x,w,b)=\frac{1}{1+\exp\{-w^T x+b\}}$$

3: Based on the output of the hidden layer, K-G polynomial is employed to generate middle candidate models, as calculated by

$$y=f(h_1,h_2,\ldots,h_m)=\sum_{i=1}^{m}a_i h_i+\sum_{i=1}^{m}\sum_{j=1}^{m}a_{ij}h_i h_j+\sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}a_{ijk}h_i h_j h_k+\ldots$$

4: Select the middle candidate models by external criterion.

$$y_k^2=b_1^k+b_2^k y_i^1+b_3^k y_j^1, i,j=1,2,\ldots,F_1, i\neq j, k=1,2,\ldots,n_2$$

5: Repeat Steps 2-3 until finding the optimal complexity model.

Output:

Get the final prediction results $\{y_1, y_2,\ldots, y_k\}$.

End.

3.2.2 Generate turning point sequence

Stock price forecasting represents a specific type of economic prediction and has its own unique characteristics. As shown in the following Fig. 4, it is the daily *K* curve of a stock. For making a successful stock market operation, it is necessary to trade according to the vertical arrow, buying at the lowest point and selling at the highest point. A good trading system should only give reminders when the real turning point appears, and the false positives need to be avoided. For example, as indicated by the horizontal arrow, the market only drops by a few percentage points and then continues to rise. So, it is not a turning point at this time, and should not be reminded.

In practice, we need to know the direction of the observed market, and thus determine the trading operations such as long-term, short-term, buying, selling, and so on. What we want is to use historical data to predict future price trends. Therefore, the model does not need to predict the exact closing price of the next trading day but discover our trading strategy from the predicted trend. In other words, we need to predict the turning points of the stock price rather than the exact value.

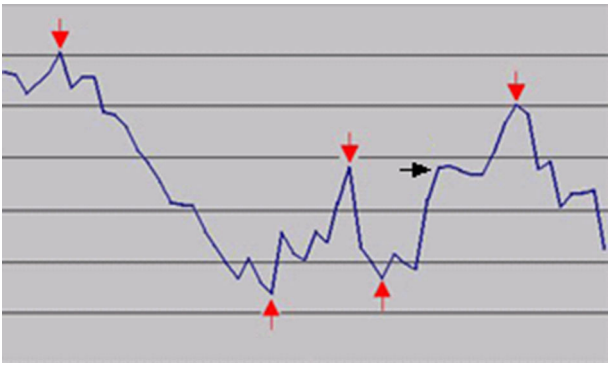
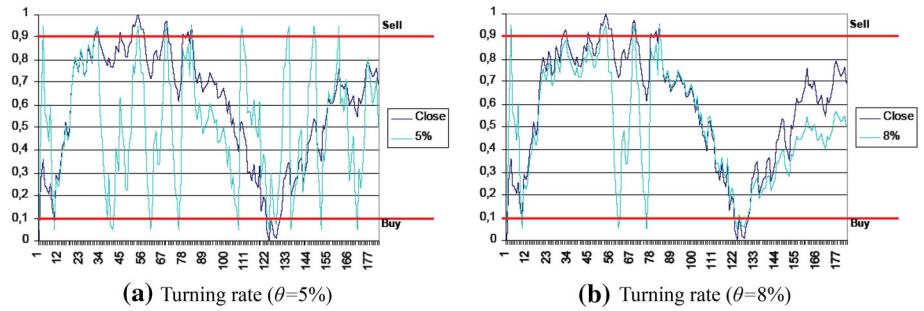


Fig. 4 The daily *K* curve of a stock



(a) Turning rate (θ=5%) **(b)** Turning rate (θ=8%)

Fig. 5 The relation diagram between turning indicator $\Phi(t)$ and turning rate θ

As for the neural networks, the supervisory signal must be obtained to learn the model in the training phase. The model we designed uses the time series turning indicator $\Phi(t)$ as the supervisory signal. Let θ be the expected turning rate, and the calculation method of $\Phi(t)$ can be described as follows.

1. Starting from the first data in the time series, the increment between adjacent data $y(t)$ and $y(t\text{-}1)$ is calculated in turn, and it is accumulated.
2. When the market price rise rate above the expected turning rate θ, the previous low point is marked as the "Buy" signal, and the corresponding output value of $\Phi(t)$ is set to 0.05.
3. When the market price fall rate below the expected turning rate θ, the previous high point is marked as a "Sell" signal, and the corresponding output value of $\Phi(t)$ is set to 0.95.
4. The other output values of $\Phi(t)$ in this time window are normalized to the interval [0.05, 0.95] using the interpolation method.

Thus, the values of turning indicator $\Phi(t)$ can be well calculated on the basis of the above processes. It is worth mentioning that the expected turning rate θ cannot be set too small, otherwise, too many signals will be generated, and many of which may be useless signals. For a given time series data of stock price, Fig. 5 depicts the different results of the calculated turning indicator $\Phi(t)$ when the expected turning rate θ is set to 5% and 8%, respectively.

As seen in Fig. 5a, it generates 9 "Buy" signal points and 8 "'Sell'" signal points when the turning rate θ is set to 5%. By contrast, in the same periods, there are 4 "Buy" signal points and 3 "Sell" signal points when the turning rate θ is set to 8%, as presented in Fig. 5b. Naturally, the greater the number of trained turning points is, the more turning signals will be generated for the predicting of the proposed method, or the less otherwise. That is to say, the turning rate θ is a hyper-parameter threshold and we recommend it to be set 5% or 8% here. After getting the value of turning signal $\Phi(t)$, the trajectory prediction in the reconstructed phase space can be carried out correspondingly. According to Takens' theorem, the reconstructed R^m space and original dynamical system maintain differential homeomorphism, and thus the turning point prediction for the stock price can be performed on the basis of the turning signal. In particular, Algorithm 2 depicts the specific calculation processes of the stock turning point sequence.

Algorithm 2: Calculation of stock turning point sequence.

Input:

Time series dataset $D=\{y_1,\ldots, y_n\}$, where $t=1,2,\ldots,n$, $y\in R^n$.

Begin

1: Calculate the difference d of stock rice, as written in

$$d_t(y_t, y_{t-1}) = y_t - y_{t-1}$$

where t and t-1 are the adjacent two points; y_t and y_{t-1} represent the corresponding stock price value.

2: Compute the change rate of normalized stock price Y_t and compare it with expected turning rate θ

$$ChangeRate = (Y_t - Y_{t-1}) / Y_{t-1}$$

3: If *ChangeRate* is greater than θ, the former point is signed as a 'Buy' signal, otherwise, the *Change-Rate* is less than $-\theta$, the former point is signed as a 'Sell' signal.

4: Using the turning point signals to update the normalized difference value $\Phi(t)$. 'Buy' signal update the $\Phi(t)$ with 0.05 while 'Sell' signal revise it with 0.95. The difference value is normalized in interval [0.05,0.95], as calculated by

$$\phi(t) = a + \frac{b-a}{max(d)-min(d)}(d_t - min(d))$$

where b is 0.95 and a is 0.05.

Output:

Get the final turning signal time series $\{\ \Phi(1),\ \Phi(2),\ldots,\ \Phi(n)\}$.

End.

4 Experimental results and analysis

We have conducted a series of experiments to validate the efficacy of the proposed procedure. Except that some graphical representations were conducted using MATLAB or R tools, the main machine learning algorithms including SVM, ANN, RF, and the proposed RVFL-GMDH were implemented using Anaconda3 (Python 3.6), scikit-learn library, and PyMC3 library, etc. [43–45]. The chaos identification and correlation dimension determination components were accomplished with statistics and machine learning toolbox in MATLAB. The essential hardware environment for the experiments contains Intel® Core™ i7-8750 CPU (2.20 GHz), 8 GB DDR4 RAM, and GeForce GTX 1060 graphics card, which was used for program operation.

4.1 Experiments on public datasets

UCI Repository [46] is an international general database comprised of widespread collected datasets, which are primarily for the prediction or classification algorithm test of machine learning. To verify the efficiency of the proposed method, five datasets including Boston, Abalone, Airfoil, Communities, and Combined Cycle Power Plant (CCPP) are downloaded from the UCI database and utilized in the experiments. They are frequently used as benchmark datasets to probe the performance of different methods.

Boston dataset that is composed of 506 instances focus on the forecasting of a median house price, and it includes 14 variables starting from a set of characteristics of the house and its correlated attributes. Through the physical measurement of Abalone molluscs, the Abalone dataset aims to predict the ages of Abalone, and this dataset contains 4177 samples determined by 8 features. Airfoil dataset is used to predict the scaled sound pressure level of aircraft, and it comprises of 1,503 samples with 6 features. Communities dataset contains 1994 instances which are determined by 128 features; for this dataset, the goal is to learn to predict the violent crime rate. CCPP dataset is used for the prediction of the hourly energy

output of the electrical net, and this dataset contains 9,568 instances determined by 4 features. Approximately half of the data in each dataset is selected as the training samples while the remainders as the test samples.

Moreover, we especially study the forecasting effect of the model on economic data, and one open Stock dataset is directly downloaded from KEEL-dataset repository [47] (http://150.214.190.154/keel/dataset.php?cod=1298), which aims at providing a set of benchmarks to analyze the behavior of the learning methods for data mining and knowledge discovery tasks. The Stock dataset provides the daily stock prices of 10 aerospace companies from January 1988 to October 1991, and there are 10 features for this dataset comprised of 950 instances. The goal of this dataset is to estimate the stock price of the 10th airline company given the stock prices of the other 9 companies, and the specific description of this Stock dataset is presented in Table 1. On this basis, we perform the model training and test for the proposed approach on this dataset, and Fig. 6 depicts the mean square error (MSE, see Eq. (19)) iteration curve of the training process. The horizontal axis of this figure is the number of iterations and the vertical axis is the mean square error. In general, it can be seen from the figure that the final mean square error converges to a very small value in the training process. Table 2 displays detailed information of these tested open datasets.

Taking into account the statistics of accuracy, we may verify the efficiency of the models with metrics like the coefficient of determination R^2, the mean squared error MSE, and the explained variance E_{VAR}, which are, respectively calculated in Eqs. (18–20).

$$R^2 = 1 - \sum_{i=0}^{n} (y_i - \hat{y}_i)^2 / \sum_{i=1}^{n} (y_i - \overline{y})^2 \tag{18}$$

Table 1 The description of the Stock dataset

Feature	Company1	Company2	Company3	Company4	Company5
Domain	[17.219,61.5]	[19.25,60.25]	[12.75,25.125]	[34.375,60.125]	[27.75,94.125]
Feature	Company6	Company7	Company8	Company9	Company10
Domain	[14.125,35.25]	[58,87.25]	[16.375,29.25]	[31.5,53]	[34.62]

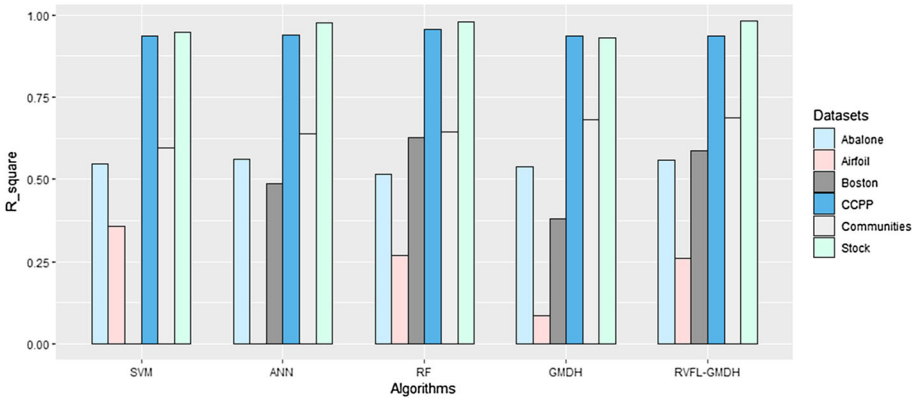


Fig. 6 The testing R^2 of different algorithms on diverse datasets

Table 2 The detailed information of the tested open datasets

Datasets	No. of instances	No. of attributes	No. of trained instances	No. of tested instances	Desired outputs
Boston	506	13	253	253	Median house value
Abalone	4177	8	2080	2097	Number of rings
Airfoil	1503	6	753	750	Sound pressure level
Communities	1994	128	997	997	Violent crimes per capita
CCPP	9568	4	4784	4784	Net hourly electrical energy
Sock	950	9	500	450	Company10

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{19}$$

$$E_{VAR} = 1 - var(y_i - \hat{y}_i)/var(y_i) \tag{20}$$

where y_i denotes the observed value, $\hat{y}_i$ is the predicted value, $\overline{y}$ is the average of the observed value, and $var(\bullet)$ is the variance function. The indicator R^2 measures the proportion of the dependent variable change that can be interpreted by the independent variable; the MSE depicts the deviation with the mean value of the squared error between the observed value and the predicted value; the E_{VAR} represents the explanatory power of the model. For both R^2 and E_{VAR}, the best possible value is equal to 1, while greater values are worse for MSE. Furthermore, to compare the proposed procedure, we took into account four influential algorithms including SVM, ANN, RF (random forest), and GMDH to perform the prediction for the contrast analysis. The training and test accuracies of different methods are, respectively obtained in Tables 3, 4. Particularly, the R^2 of diverse algorithms on test datasets are depicted in Fig. 7.

From Table 4, it can be clearly observed that the average R^2 value of the RVFL-GMDH is 0.668 in the testing set, which is the highest of all the algorithms. In particular, for the forecasting effect of the models on Stock dataset, the final mean square error MSE of the proposed method in the training set and the test set is relatively small, while the coefficient of determination R^2 and the explained variance E_{VAR} are both large, indicating that the model fits the economic data well. Additionally, for the different datasets, the RVFL-GMDH presents a relatively high R^2 value except for the Airfoil dataset. The whole bars are higher than that of other algorithms for all the regions, which depicts the stability and effectiveness of the proposed method, as observed in Fig. 7. For the average MSE and E_{VAR} values, the RVFL-GMDH also achieves the optimal results except for the RF algorithm, which is an ensemble learning method composed of multiple decision trees (Here is assigned as 20). Therefore, based on the experimental results, it can be assumed that the proposed method is superior to other algorithms and has comparable performance as the RF algorithm on the experimental

Table 3 The training accuracies of predicting the datasets

Datasets	SVM			ANN			RF			GMDH			RVFL-GMDH		
	R	*M*	*E*	*R*	*M*	*E*	*R*	*M*	*E*	*R*	*M*	*E*	*R*	*M*	*E*
Boston	0.909	0.004	0.914	0.939	0.003	0.939	0.982	0.001	0.982	0.891	0.005	0.891	0.891	0.005	0.891
Abalone	0.562	0.006	0.578	0.566	0.006	0.566	0.926	0.001	0.926	0.533	0.006	0.533	0.578	0.006	0.562
Airfoil	0.841	0.005	0.842	0.848	0.005	0.848	0.993	0.000	0.993	0.842	0.005	0.842	0.842	0.005	0.842
Comm	0.803	0.011	0.807	0.768	0.013	0.768	0.937	0.004	0.937	0.665	0.019	0.665	0.667	0.019	0.667
CCPP Seg	0.937	0.003	0.937	0.938	0.003	0.938	0.993	0.001	0.993	0.938	0.003	0.938	0.939	0.003	0.939
Stock	0.950	0.003	0.950	0.986	0.000	0.986	0.996	0.000	0.996	0.942	0.003	0.942	0.990	0.001	0.990
Average	0.834	0.005	0.838	0.841	0.005	0.841	0.971	0.001	0.971	0.802	0.007	0.802	0.818	0.007	0.815

(R: R^2, M: MSE, E: E_{VAR})

Table 4 The testing accuracies of predicting the datasets

Datasets	SVM			ANN			RF			GMDH			RVFL-GMDH		
	R	*M*	*E*	*R*	*M*	*E*	*R*	*M*	*E*	*R*	*M*	*E*	*R*	*M*	*E*
Boston	0.000	0.107	0.113	0.487	0.033	0.496	0.628	0.024	0.657	0.381	0.039	0.461	0.586	0.026	0.607
Abalone	0.547	0.006	0.562	0.562	0.006	0.562	0.515	0.006	0.515	0.537	0.006	0.538	0.559	0.006	0.563
Airfoil	0.357	0.022	0.481	0.000	0.085	0.000	0.269	0.025	0.298	0.084	0.037	0.079	0.260	0.043	0.237
Comm. ities s	0.594	0.021	0.624	0.639	0.018	0.639	0.645	0.018	0.649	0.681	0.016	0.685	0.686	0.016	0.687
CCPP Seg	0.936	0.003	0.936	0.938	0.003	0.938	0.955	0.002	0.955	0.936	0.003	0.936	0.936	0.003	0.936
Stock	0.947	0.003	0.948	0.976	0.002	0.976	0.980	0.001	0.980	0.931	0.004	0.931	0.982	0.001	0.982
Average	0.564	0.027	0.611	0.600	0.025	0.602	0.665	0.013	0.676	0.592	0.018	0.605	0.668	0.016	0.669

(R: R^2, M: MSE, E: E_{VAR})

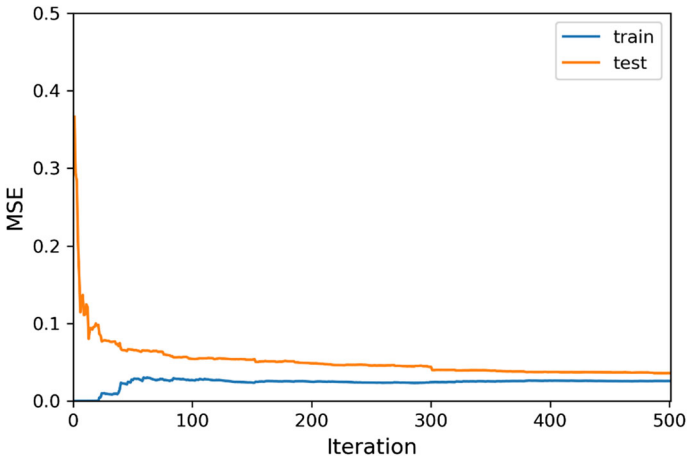


Fig. 7 The MSE iteration curve of the proposed approach on Stock dataset

datasets. The proposed procedure basically outperforms the other well-known prediction algorithms on the experimental dataset, even though the ensemble learning algorithm is adopted.

4.2 Empirical analysis experiment

By obtaining the historical data of stock trading from Yahoo Finance [48], we can randomly choose the stock price as the analysis object. Here, the Pci-Suntek (stock code: 600,728) and one FTSE-100 index constituent stock, Tesco (stock code: TSCO.L) are selected in our empirical analysis. The close price of the Pci-Suntek stock from January 2017 to September 2018 is obtained and the data normalization is processed as the following Eq. (21).

$$x(i) = (y(i) - \frac{1}{N}\sum_{i=1}^{N} y(i)) \Big/ \left\{ \frac{1}{N}\sum_{j=1}^{N} (y(j) - \frac{1}{N}\sum_{i=1}^{N} y(i))^2 \right\}^{1/2} \tag{21}$$

The chaos of time series is first judged using the method of chaotic attractor determination reported in Sect. 2. Based on the normalized time series data, the m-dimensional phase space with the time delay of τ can be constructed, and the point in the phase space is $y(t) = x(t)$,$x(t + \tau)$,...,$x(t + (m-1)\tau)$. Considering that there are five trading days a week in the stock market, we can set the τ value as $\tau = 5$, and the embedding dimension m is taken a positive integer such as $m = 3, 4,.., 10$, etc. Furthermore, for the recent data that has a greater impact on the prediction, the recent data from January 1, 2018 to July 1, 2018 is selected as the verification data and the correlation integral function $C(r)$ is computed in Eq. (3).

The value of the correlation integral is related to the given distance r. Theoretically, there is a range of r value for the attractor dimension d and the integral function $C(r)$ satisfies the log-linear relationship. That is, the $d(m_0)$ is equal to the $\ln C_{m0}(r)/\ln r$ (see Eq. (5)). Under different embedding dimensions, a series of correlation integral values $C_m(r)$ can be calculated for a series of r values. Taking $\ln(r)$ as the horizontal axis and $\ln(C_{m0}(r))$ as the vertical axis, the graph can be drawn as Fig. 8.

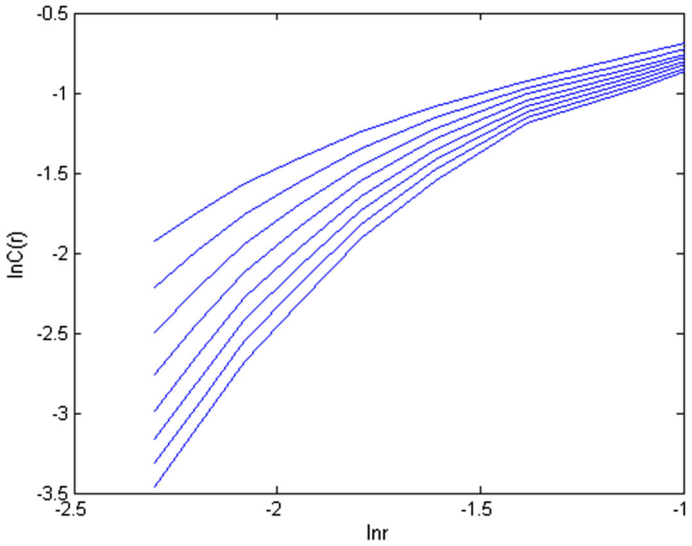


Fig. 8 The Ln$C(r)$-ln(r) diagram for Pci-Suntek

As seen in Fig. 8, the curve from top to bottom indicates that the embedding dimension m = 3, 4, 5,…,10 increases gradually. With the increase of m, the curve becomes steeper and the slope values are more stable. The obtained stable value is just the estimated value of the correlation dimension, and the details are listed in the following Table 5.

It can be visualized from Table 5 that when the embedding dimension m is increased to 8, the correlation dimension $d(m)$ tends to be stable. The value is around 3.0, and the best embedding dimension is 7. This implies that the time series is not a random time series and has chaotic dynamics characteristics. The correlation value of the chaotic attractor is about 3.0. Moreover, the embedding dimension 7 is greater than the 2*3.0287, which satisfies the condition of restoring the differential homeomorphic dynamical system in Takens' theorem [31]. Therefore, we can reconstruct the phase space based on the embedded dimension 7.

We take the historical data of Pci-Suntek stock from January 1, 2018 to September 1, 2018, and divide it into two sections on July 1, 2018. The former section is used as the training sample while the latter is as the prediction sample. Based on the method mentioned in Sect. 3, we can compute the turning indicator value, and which is loaded into the RVFL-GMDH network for the model training. Subsequently, the data from July 1, 2018 to September 1, 2018 is used as the test data and the turning points of stock prices in this period are predicted. Table 6 displays the partial sample data. In particular, it should be noted that the commonly used approaches such as ANN and time series methods are chosen to predict the turning points for comparative analysis. The results of turning point predictions are depicted in Fig. 9. Among the figures, Fig. 9a is the trend of the actual stock price, Fig. 9b displays the prediction results of the time series method of Auto-Regressive and Moving Average Model (ARMA), Fig. 9c presents the prediction results of ANN method, and Fig. 9d depicts the prediction results of the proposed RVFL-GMDH method.

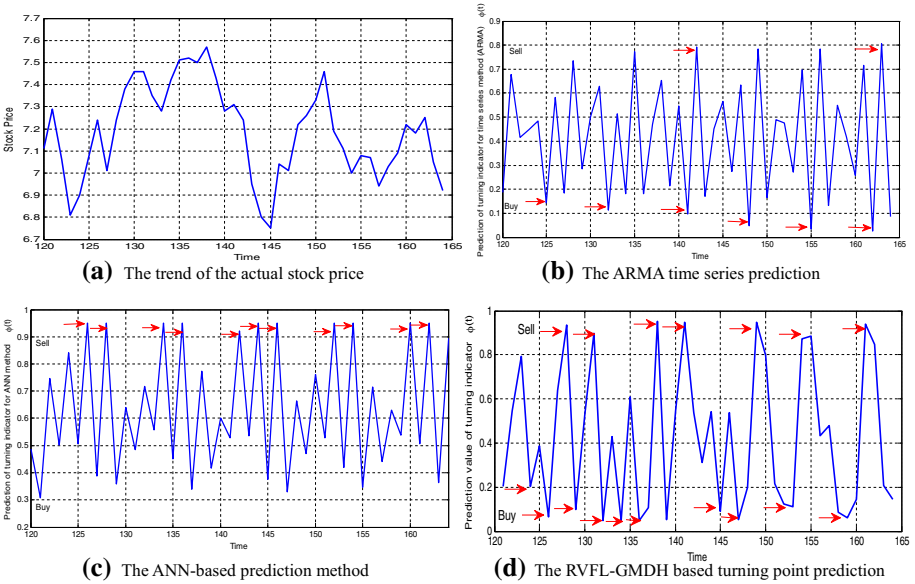
It can be observed from Fig. 9 that the RVFL-GMDH based turning point prediction method has predicted most of the turning points for the samples generally, and the position of the turning point is basically consistent with the actual trend of the stock price curve.

Table 5 The estimates of correlation dimension

m	1	2	3	4	5	6	7	8	9	10
$d(m)$	0.9342	1.6962	2.6015	2.5350	2.9764	2.7095	3.0287	3.5455	2.7484	2.9281

Table 6 The partial sample data of turning signal generation ($\theta = 5\%$)

Date	Close price	Difference	Change rate	Turning signal	Normalized diff	Adjusted nmdiff
2018–01-02	8.25	0.00	0.00000	–	0.05000	0.05
2018–01-03	8.31	0.06	0.02952	–	0.47332	0.47332
2018–01-04	8.27	− 0.04	− 0.01911	Buy	0.437747	0.05
2018–01-05	8.58	0.31	0.15101	–	0.562253	0.56225
2018–01-08	8.57	− 0.01	− 0.00423	–	0.448419	0.44842
2018–01-09	8.46	− 0.11	− 0.04675	–	0.412846	0.41285
2018–01-10	8.47	0.01	0.00446	Buy	0.455534	0.05
2018–01-11	8.62	0.15	0.06658	Sell	0.505336	0.95
2018–01-12	8.40	− 0.22	− 0.09156	Sell	0.373715	0.95
2018–01-15	8.08	− 0.32	− 0.14660	–	0.338142	0.338142



(a) The trend of the actual stock price

(b) The ARMA time series prediction

(c) The ANN-based prediction method

(d) The RVFL-GMDH based turning point prediction

Fig. 9 The actual stock price curve and the predictions of turning indicator $\Phi(t)$

Moreover, as shown in Fig. 9d, the most of predicted turning points (e.g., the marked red arrow in Fig. 9d) are just located in the position of the cusp in the actual stock price curve (see Fig. 9a), which proves the effectiveness of the proposed procedure. By contrast, the other method such as time series of ARMA mainly obtains the "Buy" turning points except for 2 "Sell" turning point signals because the prediction value is relatively small, and the correctly predicted turning points are not too much as well. Moreover, the results of ANN prediction method are similar and the primary turning point signals predicted by this method are not consistent with the cusp of the stock price curve. Most of the "Buy" turning point signals are gained because the predicted value of the ANN method is relatively big, as shown in Fig. 9c.

Table 7 The partial sample data of TSCO.L

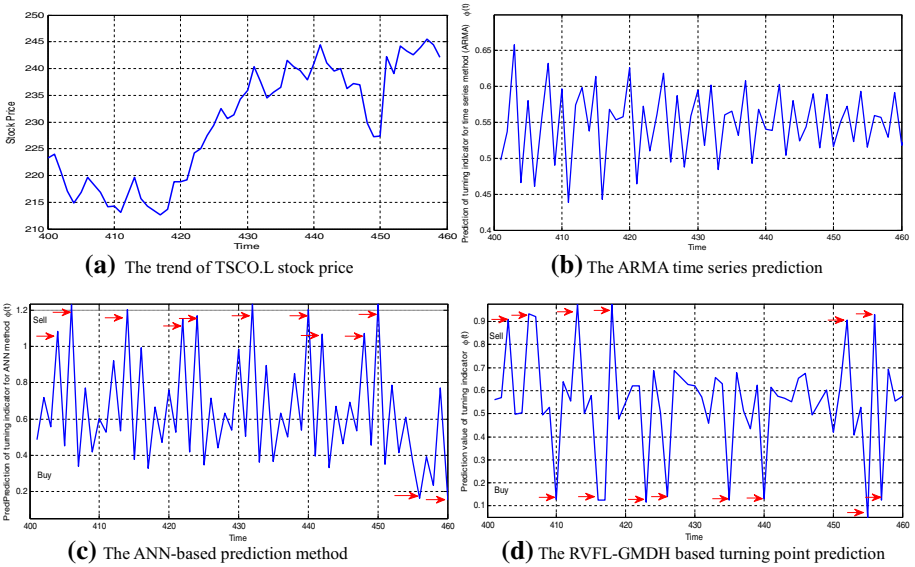
Date	Close price	Difference	Change rate	Turning signal	Normalized diff	Adjusted nmdiff
2019–01-02	191.55	1.45	0.30	Buy	0.601983	0.05
2019–01-03	199.35	7.8	1.25	Sell	0.763881	0.95
2019–01-04	197.4	− 1.95	− 0.14	Buy	0.515297	0.05
2019–01-07	202.5	5.1	0.42	Buy	0.695043	0.05
2019–01-08	208.1	5.6	0.33	Buy	0.707791	0.05
2019–01-09	211.8	3.7	0.16	Buy	0.659348	0.05
2019–01-10	216.4	4.6	0.17	–	0.682295	0.682295
2019–01-11	218	1.6	0.05	–	0.605808	0.605808
2019–01-14	218	0	0.00	–	0.565014	0.565014
2019–01-15	218.1	0.1	0.00	–	0.567564	0.567564

Therefore, referring to the cusp position of the RVFL-GMDH prediction curve, we can use it as a selling or buying signal to assist our trading operations. Likewise, the experiment is further conducted on TSCO.L, which is one of the FTSE-100 index constituent stocks in UK stock market. We have extracted the historical data of TSCO.L stock price from January 1, 2015 to October 30, 2020, a total of 1,476 instances, to perform the empirical analysis of turning point prediction of stock price. In particular, it is worth mentioning that considering the emergency impact of the new coronavirus (COVID-19) on the economy, the data of 2020 is not used to validate the effect of the model. That is to say, we have used the historical data of TSCO.L stock price from January 1, 2015 to July 31, 2019 to train the model, while the data from August 1, 2019 through October 31, 2019 (3 months, time window) to verify the effectiveness of the proposed procedure. Table 7 displays the partial sample data and the estimated values of the correlation dimension are computed in Table 8. Also, from Table 8, it can be seen that the correlation dimension tends to be stable when the embedded dimension is increased to 6, which indicates that the time series has chaotic dynamics characteristics and the phase space can be reconstructed.

As seen in Table 7, after gaining the turning signal according to the method introduced in Sect. 3.2.2, we use it to update the normalized difference value, which is normalized to the interval of [0.05, 0.95] and forms the sample data of modeling (last column of Table 7). Therefore, based on the constructed sample data, we build the model using the RVFL-GMDH method along with other comparative methods to predict the turning point data for the next 3 months. Figure 10a presents the specific trend of the actual TSCO.L stock price and the predicted results of different methods are depicted in Fig. 10b–d. As seen in Fig. 10b, the predicted data are all limited in the interval of [0.2, 0.8], and no obvious turning points are predicted by the time series method of ARMA. The situation of the neural network method is similar and most of the predicted value is relatively big, which caused the "Sell" turning point signals are primarily obtained but few "Buy" turning point signals, as shown in Fig. 10c. Particularly, the correctly predicted turning points are not too much. On the other hand, looking at Fig. 10d, it is apparent that the proposed approach has successfully predicted most of the effective turning points for the stock price. As indicated by the arrow in this figure, the turning points with the predicted index value greater than 0.8 or less than 0.2, which signals the selling or buying strategy for the stock exchange, are almost effective

Table 8 The correlation dimension estimation of TSCO.L

m	1	2	3	4	5	6	7	8	9	10
$d(m)$	0.9707	1.8226	2.7227	3.6645	4.2767	3.8125	3.5297	3.6707	3.7770	3.8182



(a) The trend of TSCO.L stock price **(b)** The ARMA time series prediction

(c) The ANN-based prediction method **(d)** The RVFL-GMDH based turning point prediction

Fig. 10 The actual stock price curve and predicted turning indicator $\Phi(t)$ of TSCO.L

turning points verified by the actual stock price. Moreover, the predicted trend of the turning indicator is consistent with the trend of the actual stock price as well. For example, in the period of the continuous increase of actual stock price, the predicted trend of turning point indicator ("Adjusted nmdiff" variable) is flat too, as seen in the middle part of Fig. 10d. Also, we can conduct the simulation studies to evaluate the performance of turning point prediction using the financial metrics like Profit and Loss (PnL), which measures an absolute gain of the price index for total transactions, as calculated by [2]

$$PnL = \sum_{k=1}^{N} (Closing\_index_s - Closing\_index_b) \tag{22}$$

where N denotes the number of holding stock, *closing_index* represents the closing price of the stock. Suppose that an investor buy 100 shares at the time point of "410" (August 15, 2019, stock price:213.1) according to the turning point signals predicted by the proposed procedure, and he sells them at the next long selling point of "452" (October 15, 2019, stock price:244.2). By doing this, the investor will obtain a profit of 3,110 since the Profit and Loss is computed as 100*(244.2–213.1). Furthermore, suppose that the investor carries out the short-term trading strategy of each signal point, and he buys or sells at least 100 shares each time. Thus, the investor buys 100 shares at the time point of "410" (August 15, 2019, stock price:213.1) and sells them out at the time point of "413" (August 20, 2019, stock price:215.7). In the same way, the investor buys 100 shares at the time point of "416" (August 23, 2019, stock price:212.3) and sells them out at the time points of "418" (August 28, 2019, stock price:218.8). As such, the investor buys 100 shares at the four time points of "423" (September 04, 2019, stock price:227.5), "426" (September 09, 2019, stock price:230.6), "435" (September 20, 2019, stock price:241.5), "440" (September 27, 2019, stock price:244.5), respectively. And he sells them out at the time point of "452" (October 15, 2019, stock price: 244.2). Similarly, at the time point of "455" (October 18,

2019, stock price: 244), the investor buys 100 shares and sells them out at the time point of "456" (October 21, 2019, stock price: 245.5). In this manner, the PnL of the trading strategy based on turning point signals in this time window can be calculated as 100*(215.7–213.1) + 100*(218.8–212.3) + 100*(4*244.2–227.5–230.6–241.5–244.5) + 100*(245.5–244) = 4,330. In summary, the results of the empirical analysis reveal the feasibility and effectiveness of the proposed procedure, which is successful in predicting the turning points of stock time series and can also be employed in more real-world application scenarios.

5 Conclusions

This work proposes a novel turning point prediction approach for the time series analysis of stock price. The modeling of the nonlinear dynamic system based on the RVFL-GMDH network is introduced in the paper, and the chaos theory is utilized to analyze the time series. Using the fractal characteristic of strange attractor with infinite self-similar structure, the trajectory trend prediction in the reconstructed phase space is accomplished. This method not only exploits the prediction ability brought by the periodic denseness in chaotic dynamic systems but also avoids the issue that the single predicted value is not accurate due to the sensitivity of the initial value. A series of experiments are performed for the proposed approach on both the open dataset and practical stock price data. The experimental findings demonstrate the effectiveness of the model compared to the other state-of-art methods and present the proposed procedure with the substantial performance for forecasting the turning points of stock time series in real-life application scenarios. Based on the experimental analysis, it can be concluded that the proposed procedure has a significant capability for the turning point prediction of stock price and can also be extended to other fields such as electricity load forecasting, material demand planning, and international oil price forecast. In future development, we plan to deploy it on embedded systems to form the production for automatically predicting and recognizing the turning points of stock prices. Meanwhile, we intend to apply it to more real-world applications.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10115-021-01602-3.

Acknowledgements The authors express their acknowledges to the Data Mining and Computing Intelligence Research Group at Xiamen University for funding this work through grant No. of 20720181004 and 61672439 separately. The authors also thank the expert, Mr. Zhang Liang-jun from Tiptech Ltd., for the beneficial discussion and assistance. The authors are thankful to the editors and all the anonymous reviewers for their critically reviewing and constructive suggestions.

References

1. Shen W, Guo X, Wu C, Wu D (2011) Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm. Knowl-Based Syst 24(3):378–385
2. Chen TL, Chen FY (2016) An intelligent pattern recognition model for supporting investment decisions in stock market. Inf Sci 346:261–274
3. Babu CN, Reddy BE (2015) Prediction of selected Indian stock using a partitioning–interpolation based ARIMA–GARCH model. Appl Comput Informat 11(2):130–143
4. Hamzaçebi C, Pekkaya M (2011) Determining of stock investments with grey relational analysis. Expert Syst Appl 38(8):9186–9195
5. Wen D, Wang GJ, Ma C, Wang Y (2019) Risk spillovers between oil and stock markets: a VAR for VaR analysis. Energy Econom 80:524–535

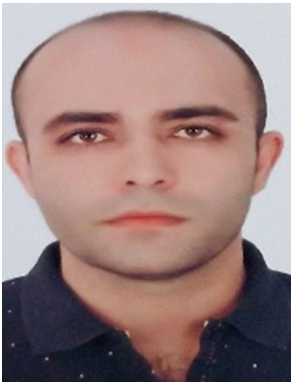
6. Long W, Lu Z, Cui L (2019) Deep learning-based feature engineering for stock price movement prediction. Knowl-Based Syst 164:163–173
7. Dash R, Dash PK (2016) A hybrid stock trading framework integrating technical analysis with machine learning techniques. The Journal of Finance and Data Science 2(1):42–57
8. Fujimaki R, Nakata T, Tsukahara H, Sato A, Yamanishi K (2009) Mining abnormal patterns from heterogeneous time-series with irrelevant features for fault event detection. Statist Analy Data Mining: The ASA Data Sci J 2(1):1–17
9. Nahil A, Lyhyaoui A (2018) Short-term stock price forecasting using kernel principal component analysis and support vector machines: the case of Casablanca stock exchange. Procedia Comput Sci 127:161–169
10. Lahmiri S (2018) Minute-ahead stock price forecasting based on singular spectrum analysis and support vector regression. Appl Math Comput 320:444–451
11. Chang, P. C., Fan, C. Y., & Liu, C. H. (2008). Integrating a piecewise linear representation method and a neural network model for stock trading points prediction. *IEEE Trans Syst, Man, Cybernet, Part C (Applications and Reviews)*, *39*(1), 80–92
12. Göçken M, Özçalıcı M, Boru A, Dosdoğru AT (2016) Integrating metaheuristics and artificial neural networks for improved stock price prediction. Expert Syst Appl 44:320–331
13. Efendi R, Arbaiy N, Deris MM (2018) A new procedure in stock market forecasting based on fuzzy random auto-regression time series model. Inf Sci 441:113–132
14. Cheng SH, Chen SM, Jian WS (2016) Fuzzy time series forecasting based on fuzzy logical relationships and similarity measures. Inf Sci 327:272–287
15. Kim Y, Jeong SR, Ghani I (2014) Text opinion mining to analyze news for stock market prediction. Int J Adv Soft Comput Appl 6(1):2074–8523
16. Nikfarjam, A., Emadzadeh, E., & Muthaiyah, S. (2010, February). Text mining approaches for stock market prediction. In *2010 The 2nd international conference on computer and automation engineering (ICCAE)* (Vol. 4, pp. 256–260). IEEE
17. White, H. (1988, July). Economic prediction using neural networks: The case of IBM daily stock returns. In *ICNN* (Vol. 2, pp. 451–458)
18. Baba, N., & Kozaki, M. (1992, June). An intelligent forecasting system of stock price using neural networks. In *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks* (Vol. 1, pp. 371–377). IEEE
19. de Oliveira FA, Nobre CN, Zárate LE (2013) Applying Artificial Neural Networks to prediction of stock price and improvement of the directional prediction index–Case study of PETR4, Petrobras. Brazil Expert syst appl 40(18):7596–7606
20. Laboissiere LA, Fernandes RA, Lage GG (2015) Maximum and minimum stock price forecasting of Brazilian power distribution companies based on artificial neural networks. Appl Soft Comput 35:66–74
21. Sayavong, L., Wu, Z., & Chalita, S. (2019, September). Research on stock price prediction method based on convolutional neural network. In *2019 International Conference on Virtual Reality and Intelligent Systems (ICVRIS)* (pp. 173–176). IEEE
22. Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2017, July). Forecasting stock prices from the limit order book using convolutional neural networks. In *2017 IEEE 19th Conference on Business Informatics (CBI)* (Vol. 1, pp. 7–12). IEEE
23. Deng Y, Bao F, Kong Y, Ren Z, Dai Q (2016) Deep direct reinforcement learning for financial signal representation and trading. IEEE Trans Neural Netw Learn Syst 28(3):653–664
24. Zarkias, K. S., Passalis, N., Tsantekidis, A., & Tefas, A. (2019, May). Deep reinforcement learning for financial trading using price trailing. In *ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 3067–3071). IEEE
25. Rather AM, Agarwal A, Sastry VN (2015) Recurrent neural network and a hybrid model for prediction of stock returns. Expert Syst Appl 42(6):3234–3241
26. Dixon M (2018) Sequence classification of the limit order book using recurrent neural networks. J Comput Sci 24:277–286
27. Tang H, Dong P, Shi Y (2019) A new approach of integrating piecewise linear representation and weighted support vector machine for forecasting stock turning points. Appl Soft Comput 78:685–696
28. Intachai, P., & Yuvapoositanon, P. (2017, March). The variable forgetting factor-based local average model algorithm for prediction of financial time series. In *2017 International Electrical Engineering Congress (iEECON)* (pp. 1–4). IEEE
29. Chang PC, Liao TW, Lin JJ, Fan CY (2011) A dynamic threshold decision system for stock trading signal detection. Appl Soft Comput 11(5):3998–4010
30. Takens, F. (1981). Detecting strange attractors in turbulence. In *Dynamical systems and turbulence, Warwick 1980* (pp. 366–381). Springer, Berlin, Heidelberg.

31. Kolmogorov AN (1963) On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. Trans Am Math Soc 2(28):55–59
32. Pao YH, Park GH, Sobajic DJ (1994) Learning and generalization characteristics of the random vector functional-link net. Neurocomputing 6(2):163–180
33. Ren Y, Suganthan PN, Srikanth N, Amaratunga G (2016) Random vector functional link network for short-term electricity load demand forecasting. Inf Sci 367:1078–1093
34. Gorban AN, Tyukin IY, Prokhorov DV, Sofeikov KI (2016) Approximation with random bases: pro et contra. Inf Sci 364:129–145
35. Scardapane S, Wang D, Uncini A (2017) Bayesian random vector functional-link networks for robust data modeling. IEEE Trans Cybernet 48(7):2049–2059
36. Cui W, Zhang L, Li B, Guo J, Meng W, Wang H, Xie L (2017) Received signal strength based indoor positioning using a random vector functional link network. IEEE Trans Industr Inf 14(5):1846–1855
37. Zhang PB, Yang ZX (2020) A new learning paradigm for random vector functional-link network: RVFL+. Neural Netw 122:94–105
38. Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1–3):489–501
39. Rahimi, A., & Recht, B. (2008, December). Weighted sums of random kitchen sinks: replacing minimization with randomization in learning. In *Nips* (pp. 1313–1320)
40. Mueller, J. A., & Lemke, F. (2000). Self-organising data mining: an intelligent approach to extract knowledge from data. *Hamburg: Libri*.
41. He CZ, Wu J, Müller JA (2008) Optimal cooperation between external criterion and data division in GMDH. Int J Syst Sci 39(6):601–606
42. Teng GE, He CZ, Xiao J, Jiang XY (2013) Customer credit scoring based on HMM/GMDH hybrid model. Knowl Inf Syst 36(3):731–747
43. Anaconda. Available online: https://www.anaconda.com/ (accessed on 17 Nov., 2019)
44. scikit-learn. Available online: https://scikit-learn.org/stable/ (accessed on 17 Nov., 2019)
45. PyMC3. Available online: https://docs.pymc.io/ (accessed on 17 Nov., 2019)
46. Blake, C. (1998). UCI repository of machine learning databases. http://www.ics.uci.edu/~mlearn/MLRepository.html
47. Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. (2011). Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, *17*
48. https://fiance.yahoo.com/quote/600728.SS/history?p=600728.SS

Junde Chen receives his bachelor degree from Xiangtan University , and the master degree from Sichuan Uni versity separately . His research interests include the aspects of Data Mining, Image Processing, Machine Learning and Decision Support System etc. Currently, he i s doing PHD in School of Informatics , Xiamen University , Xiamen city, in China Shuangyuan

Shuangyuan Yang received the PHD degree in School of computer science, Huazhong University of science and technology, and she is currently working at the School of Informat ics, Xiamen University. His research interests include Machine Learning, Computer Vision, Information Extraction & Retrieval, Natural Language Processing, and Data Mining Technology etc.

Defu Zhang works in School of Informatics at Xiamen University currently. His research interests include all aspects of computational intelligence, image analysis and data mining, etc. He has published papers in the following Journals: INFORMS Journal on Computing, Computers Operations Research, European Journal of Operational Research, Expert System with Applications, etc.



Y. A. Nanehkaran received the B.E. degree from IAU of Ardabil Branch, Ardabil, Iran, in Power Electrical Engineering and M.Sc. degree in IT from Cankaya University, Ankara, Turkey. He is currently pursuing the Ph.D. degree in the Department of Computer Science at Xiamen University, Xiamen city, in China. His research area mainly includes data mining, big data and deep learning techniques.