Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 9546792, 17 pages
https://doi.org/10.1155/2020/9546792

Research Article

A Novel Fuzzy Time Series Forecasting Model Based on Multiple Linear Regression and Time Series Clustering

Yanpeng Zhang,[1] **Hua Qu**,[1,2] **Weipeng Wang**,[2] **and Jihong Zhao**[3]

[1]*School of Software Engineering, Xi'an Jiao Tong University, Yan Xiang Road, Xi'an, China*
[2]*School of Electronics and Information Engineering, Xi'an Jiao Tong University, Yan Xiang Road, Xi'an, China*
[3]*School of Communication and Information Engineering, Xi'an University of Posts & Telecommunications, Chang An Road, Xi'an, China*

Correspondence should be addressed to Yanpeng Zhang; hnypzhang@gmail.com

Received 2 July 2019; Revised 19 October 2019; Accepted 4 December 2019; Published 13 January 2020

Academic Editor: Bogdan Smolka

Time series forecasting models based on a linear relationship model show great performance. However, these models cannot handle the the data that are incomplete, imprecise, and ambiguous as the interval-based fuzzy time series models since the process of fuzzification is abandoned. This article proposes a novel fuzzy time series forecasting model based on multiple linear regression and time series clustering for forecasting market prices. The proposed model employs a preprocessing to transform the set of fuzzy high-order time series into a set of high-order time series, with synthetic minority oversampling technique. After that, a high-order time series clustering algorithm based on the multiple linear regression model is proposed to cluster dataset of fuzzy time series and to build the linear regression model for each cluster. Then, we make forecasting by calculating the weighted sum of linear regression models' results. Also, a learning algorithm is proposed to train the whole model, which applies artificial neural network to learn the weights of linear models. The interval-based fuzzification ensures the capability to deal with the uncertainties, and linear model and artificial neural network enable the proposed model to learn both of linear and nonlinear characteristics. The experiment results show that the proposed model improves the average forecasting accuracy rate and is more suitable for dealing with these uncertainties.

1. Introduction

Prediction for future data based on analyzing temporal data is an important way to explore the value of data since a precise prediction is conductive to make policy analysis and decision in many fields, such as government [1], economics, and management [2]. However, considering the instability of the data sources and the unreliability of data collecting process, most of the collecting data contain incomplete, imprecise, and ambiguous records, which makes preprocessing an indispensable procedure for machine learning. Thus, forecasting methods based on fuzzy time series have been proposed to cope with uncertainties caused by vagueness, ambiguity, and other nonprobabilistic reasons and widely applied in finance domain, such as Taiwan Stock Exchange Capitalization Weighted Stock Index (TAIEX) forecasting [3–11], the NTD/USD exchange rates forecasting [6, 8], and the market price of shares of the State Bank of India (SBI) at the Bombay Stock Exchange (BSE) forecasting [12–19].

In 1965, Zadeh [20] developed fuzzy set theory. By replacing the values of time series with fuzzy sets, a fuzzy time series (FTS) model was proposed by Song and Chissom [21–23] in 1993. In past decades, many forecasting methods based on this framework were proposed. Most of these researches used an interval-based FTS model to handle the fuzzification of the time series and applied fuzzy logic relationship, which can be executed on the FTS dataset, for making forecast. For example, Chen [24] developed a method for forecasting enrollments based on high-order

fuzzy time series. Chen and Chang [3] published a method for multivariable fuzzy forecasting based on fuzzy clustering and fuzzy rule interpolation techniques. Huarng [15] determined interval lengths by using distribution-based and average-based lengths to improve the forecasting accuracy. Chen and Chen [25] constructed a hybrid fuzzy time series model based on granular computing. Cai et al. [9, 10] selected the partition points of the universe of discourse by adopting the metaheuristic optimization algorithm. Qu et al. [26] applied multipartition to improve the process of fuzzification and adopted a linear model to deal with samples that are not suitable for fuzzy logic relationship. Recently intuitionistic fuzzy set (IFS) [27] and hesitant fuzzy set (HFS) [28] have gained attention of researchers [16, 29] and proposed a fuzzy time series forecasting model based on intuitionistic fuzzy logical relations, respectively. Bisht et al. [17, 30] explained fuzzy time series forecasting models built by hesitant fuzzy logical relations to address issue of aforesaid nonstochastic hesitation. Gupta and Kumar [19] proposed an aggregation operator to aggregate hesitant probabilistic fuzzy elements to fuzzy elements. These models are focused on modeling with fuzzy logic relationship. However, after high-order fuzzy time series was proposed, many works began to employ the linear model to make forecast. Cai et al. [9] adopted Levenberg–Marquardt to build the forecasting model of high-order time series. Askari et al. [31] depicted a forecasting model based on fuzzy clustering and linear combinations of the input variables (CFTS) instead of fuzzy logic. These models make great performances on forecasting accuracy.

1.1. Motivation. In 2015, Askari et al. [31] depicted CFTS which applied fuzzy C-means clustering to replace interval-based fuzzification. CFTS [31] made a great improvement of predict accuracy, which suggests that a cluster-based linear model is much more suitable for this problem than other existing models applied. Clustering is widely adopted by the fuzzy time series model, and it is mainly used to determine the fuzzy sets to make a more suitable partition of the universe of discourse [32]. In fact, as for time series, clustering is usually executed on the set of subtime series [3, 33] and integrate all forecasting results given by models constructed on each clusters. The time series within a cluster is highly similar, which could be modeled with some simple models, such as the linear model. Chen and Chang [3] applied fuzzy C-means clustering algorithm to construct fuzzy rules, with which to make forecast. Cheng et al. [11] defined a similarity with fuzzy logic relationship and employed K-means to improve the forecasting accuracy. Aladag et al. [34] executed fuzzification with Gustafson–Kessel fuzzy clustering to eliminate the influence of the number of intervals. All of them still adopt fuzzy logic to construct the relationships and make a little improvement.

Recently, most of FTS research studies focus on applying the linear model to construct the relationship between observed data and the data to predict. Talarposhti et al. [35] employed an exponential model to construct the forecasting model based on fuzzy time series. These models show a little improvement in accuracy compared with the model based on fuzzy logic relationship. Cai et al. [9, 10] applied Levenberg–Marquardt algorithm to build the model on the relationship based on interval-based fuzzification. Also, CFTS employed a linear model to build the relationship instead of fuzzy logic. These models do improve the forecasting accuracy of FTS.

However, most of the economic research studies suggest that the linear model is not always suitable for market price [36] since the markets are commonly imperfect as, for example, in the case of a higher interest rate for borrowing or different risk-premia for the seller and the buyer. Artificial neural network is capable of modeling these nonlinear data characteristics and has attracted a lot of interest as a popular tool of artificial intelligence in the late years. More and more research studies try to develop a forecasting model based on ANN. Huarng and Yu [37] adopted artificial neural network (ANN) for building the nonlinear model of fuzzy time series. Yolca and Alpaslan [38] integrated all steps into one ANN to reduce the training error propagation between steps. Selvin et al. [39] employed three deep learning approaches, including convolutional neural network (CNN), recursive neural networks (RNN), and long short term memory (LSTM), for making forecast of National Stock Exchange. ANN shows a great ability to model the time series of market prices than the linear model.

Askari's CFTS model [31] suggests that the cluster-based linear model could make an improvement on the forecasting accuracy, which reduces the forecasting error to 50% of other existing FTS models. However, unlike interval-based fuzzification, CFTS replaces the process of fuzzification with fuzzy C-mean clustering [40], and it works with the time series instead of fuzzy time series. Besides, the fuzzy C-mean clustering just gives the grade of membership by exact distance, and it could not handle the incomplete, imprecise, and ambiguous data. Essentially, CFTS is not a real fuzzy time series forecasting model.

For reasons above, we proposed a new fuzzy time series forecasting model based on multiple linear regression and time series clustering. This work makes three main contributions:

(1) First, we applied preprocessing to transform fuzzy time series set into a weighted time series set. Then, we applied the synthetic minority oversampling technique (SMOTE) to deal with the unbalanced samples and finally transformed the weighted time series set into a time series set, on which the multiple linear regression model (MLRM) and ANN could be worked.

(2) Second, we developed a novel high-order time series clustering algorithm based on the multiple linear regression model, which clusters data by similarity of linear relationship instead of shape, to extract suitable linear models for dataset.

(3) Finally, we designed a new forecasting model for FTS based on the multiple linear regression model and ANN, which employs the ANN to give the weights of each multiple linear regression. We devised a specific

learning algorithm for our forecasting model to train these ANNs together. Unlike ANN-based fuzzy models, we adopted ANN to give the weight of linear models instead of forecasting results, which could model the nonlinear characteristics of the market prices. Besides, for each clusters, the forecasting model builds a linear model on it instead of fuzzy logic relationship, which other cluster-based FTS models used.

The experimental results show that the proposed model has a higher average forecasting accuracy rate than other existing FTS-based models.

The rest of this paper is organized as follows. Section 2 briefly reviews some basic concepts and algorithms. Section 3 explains a new forecasting model based on fuzzy time series and describes the key steps in detail, and it shows how the proposed model works with a case of TAIEX. Section 4 shows the comparison of forecasting results of the proposed model and other existing models. The conclusion is provided in Section 5.

2. Preliminaries

This section will introduce terminologies and briefly reviews basic concepts and algorithms.

2.1. Fuzzy Time Series. To deal with the fuzzy, incomplete sequences containing noise, Song and Chissom [21–23] introduced the concept of fuzzy mathematics into time series and propose the concept of fuzzy time series.

In fuzzy time series, values are represented by fuzzy sets defined on the universe of discourse U, where $U = \{u_1, u_2, \ldots, u_n\}$. A fuzzy set A can be represented by

$$A = \frac{f_A(u_1)}{u_1} + \frac{f_A(u_2)}{u_2} + \cdots + \frac{f_A(u_n)}{u_n}, \quad (1)$$

where f_A denotes the membership function of fuzzy set A, $f_A: U \longrightarrow [0, 1]$ and $f_A(u_i)$ denotes the grade of membership of u_i belonging to the fuzzy set A and $i = 1, 2, \ldots, n$.

Definition 1. Let $Y(t)(t = 0, 1, \ldots)$ be the universe of discourse consisting of real numbers, on which fuzzy sets $f_i(t)(i = 1, 2, \ldots)$ defined. Let $F(t)$ be a collection of $f_i(t)(i = 1, 2, \ldots)$. Then, $F(t)$ is called a fuzzy time series defined on $Y(t)(t = 0, 1, \ldots)$.

Definition 2. Let $F(t)(t = 1, 2, \ldots)$ be a fuzzy time series. Assume that there is a relationship $R(t-1, t)$ between $F(t)$ and $F(t-1)$ that satisfies $F(t) = F(t-1) \circ R(t-1, t)$, where $F(t)$ and $F(t-1)$ are fuzzy sets and $\circ$ is max-min composition operator; then, the $R(t-1, t)$ is a fuzzy logic relationship denoted by $F(t-1) \longrightarrow F(t)$.

Definition 3. Suppose $F(t)$ is caused by $F(t-1), F(t-2), \ldots, F(t-k)$, it can be presented by a fuzzy logical relationship: $F(t-k+1), \ldots, F(t-1), F(t-1) \longrightarrow F(t)$.

The fuzzy time series model uses a four-step framework to make forecast: (1) define the universe of discourse and partition it into intervals; (2) determine the fuzzy sets on the universe of discourse and fuzzify the time series; (3) build the model of the existing fuzzy logic relationships in the fuzzified time series; and (4) make forecast and defuzzify the forecast values.

3. A New Forecasting Model Based on Fuzzy Time Series and High-Order MLRM-Based Time Series Clustering

In this section, we proposed a new forecasting method based on fuzzy time series and the clustering algorithm.

The proposed model is different from traditional FTS-based models that adopt fuzzy logic relationship. It builds the model which makes forecasting by combining the results of several linear models as CFTS [31]. This model consists of a set of linear models, and each model represents an individual rule that time series changes. For each sample, we could calculate the similarities related to every linear models and make forecast by weighted summation of the outputs of linear models according to the similarities. On the one hand, it is important to construct a suitable set of linear models; on the other hand, a corresponding precise estimation of similarity between each sample and the linear model is required. Thus, we proposed a high-order MLRM-based time series clustering algorithm to give a suitable set of linear models. After that, for each linear model, we adopted an ANN to obtain its weight for each time series sample. With the output of linear models and their weights, we could make forecast by weighted summation.

This model has four main steps during training process as shown in Figure 1:

Step 1: generate the high-order fuzzy time series with the origin data set. The high-order time series is generated and fuzzified with method defined in [24], which generates fuzzy sets by partitioning the discourse of universe equally.

Step 2: transform the set of fuzzy time series into a set of time series by the preprocessing defined in Section 3.1.

Step 3: build a set of multiple linear regression models with a clustering algorithm described in Section 3.2.

Step 4: train an ANN to calculate the grades of membership between high-order time series and the multiple linear regression model constructed, which is explained in Section 3.3.

After training, we get a model consisting of a set of linear models and a set of ANNs. As displayed in Figure 1, for an observed time series, the details of making forecasting are as follows.

Step 5: generate a high-order fuzzy time series corresponding to the observed time series data as Step 1.

Step 6: transform the high-order fuzzy time series into a weighted high-order time series set that contains several samples as the preprocessing defined in Section 3.1.

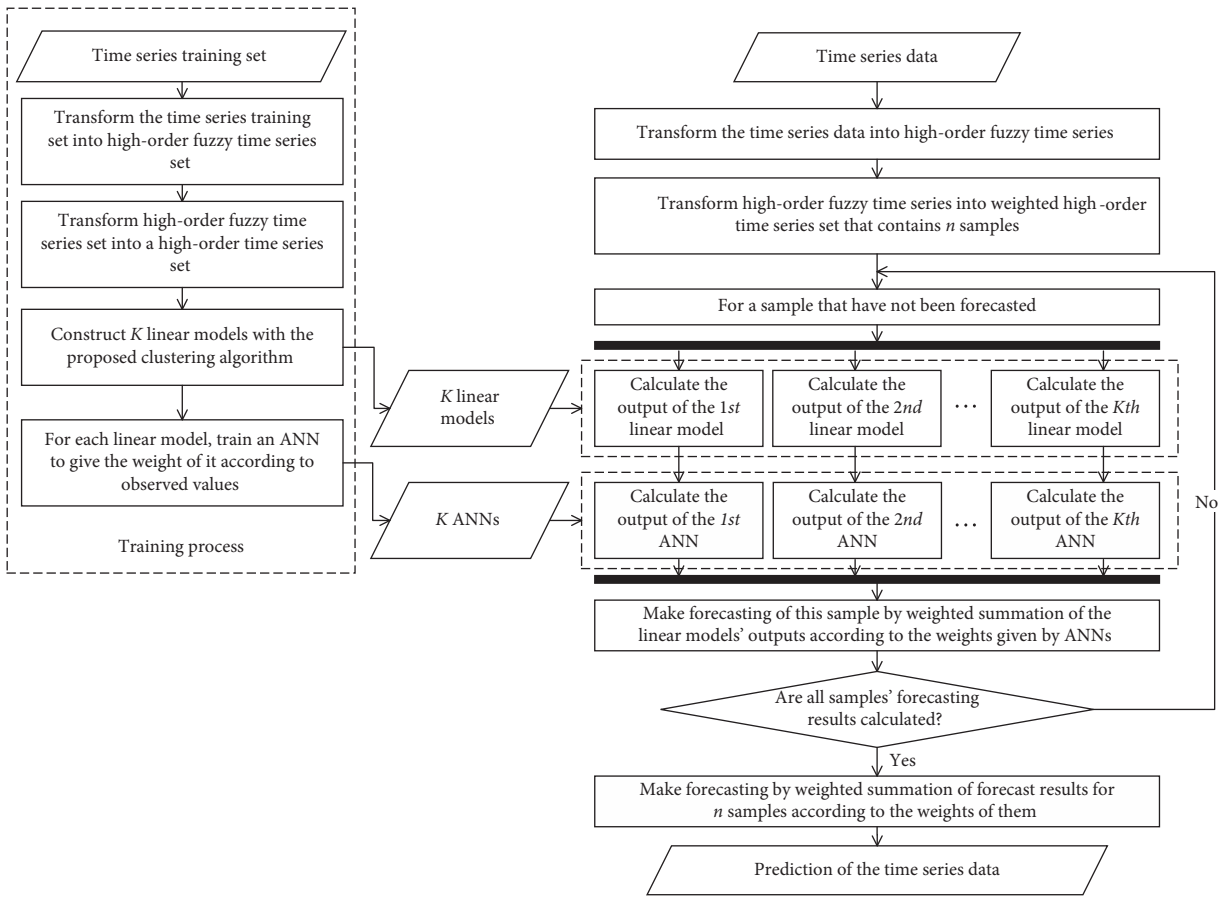

Figure 1: The procedures of the proposed forecasting model.

Step 7: for each weighted sample, without considering the weight, calculated the outputs of linear models and ANNs are obtained during training process. Then, calculate the weighted summation of the outputs of linear models according to the weights given by ANNs, which is described in Section 3.3. The results of the weighted summation are the forecasting results of this sample.

Step 8: after all weighted samples have been forecasted without considering the weight, give the final output by making a weighted summation of these forecasting results according to these samples' weights. The final output is the forecasting result of the observed time series.

Besides, we take the TAIEX data of 1999 as a case to show how our proposed model works in Section 3.4.

3.1. Preprocess of the High-Order Fuzzy Time Series. For high-order fuzzy time series, it easily to build a linear model with their corresponding actual high-order time series, if values of these actual high-order time series are not incomplete, ambiguous, which could be represented by a numeric value. However, it is hard to employ the linear model to build relationships with fuzzy sets, neither with the actual time series, when the actual time series contains incomplete, ambiguous value. Suppose $\overrightarrow{x}_t^* = [A_{t-L+1}, A_{t-L+2}, \ldots, A_t]$ is a high-order fuzzy time series that contains incomplete, ambiguous data. The detailed preprocessing of $\overrightarrow{x}_t^*$ is as follows:

Step 1: replace each dimension with the corresponding numeric value if could. Then, we obtain a hybrid vector $\overrightarrow{B}_0 = [B_{0,1}, B_{0,2}, \ldots, B_{0,L}]$, where B_i could represent either a numeric value x_{t-L+i} or a fuzzy set A_{t-L+i}. And, we construct two empty sets of tuple V and W. If $\overrightarrow{B}_0$ contains fuzzy set, add $\{\overrightarrow{B}_0, 1\}$ into V; otherwise, add it into W and set the number of tuples $N_B = 1$.

Step 2: if V is empty, then go to Step 5; otherwise, choose a tuple $\{\overrightarrow{B}_j, w_j\}$ from V and remove it from V.

Step 3: for $\overrightarrow{B}_j$, suppose that $B_{j,i}$ is described by fuzzy set instead of numeric value, and $B_{j,i}$ could be represented as a vector of membership to all fuzzy sets, $B_{j,i} = [\text{mem}_1, \text{mem}_2, \ldots, \text{mem}_p]$, where p denotes the number of intervals that determined the fuzzy sets. We construct p new tuples by replace $B_{j,i}$ with numeric values related to intervals defined before as follows:

$$\begin{aligned}
&\left\{\overrightarrow{B}_{N_B} = \left[B_{j,1}, \ldots, B_{j,i-1}, x_{u_1}, B_{j,i+1}, \ldots, B_{j,L}\right], w_{N_B}\right\},\\
&\left\{\overrightarrow{B}_{N_B+1} = \left[B_{j,1}, \ldots, B_{j,i-1}, x_{u_2}, B_{j,i+1}, \ldots, B_{j,L}\right], w_{N_B+1}\right\},\\
&\ldots\\
&\left\{\overrightarrow{B}_{N_B+p-1} = \left[B_{j,1}, \ldots, B_{j,i-1}, x_{u_p}, B_{j,i+1}, \ldots, B_{j,L}\right], w_{N_B+p-1}\right\},
\end{aligned} \tag{2}$$

where x_{u_k} denotes the average value of the kth interval, and the weight of $\overrightarrow{B}_{N_B+k}(k = 1, 2, \ldots, p)$ could be calculated as follows:

$$w_{N_B+k} = w_j \cdot \frac{\text{mem}_k}{\sum_{m=1}^{p} \text{mem}_m}. \tag{3}$$

Step 4: for each constructed hybrid vector of tuple $\overrightarrow{B}_j (j = N_B, N_B + 1, \ldots, N_B + p - 1)$, if its related weight $w_j \neq 0$ and $\overrightarrow{B}_j$ still contains fuzzy sets, add $\left\{\overrightarrow{B}_j, w_j\right\}$ into W; otherwise, if its related weight $w_j \neq 0$ and $\overrightarrow{B}_j$ only contains numeric values, add $\left\{\overrightarrow{B}_j, w_j\right\}$ into V. Then, set the number of tuples $N_B = N_B + p$ and go to Step 2.

Step 5: for each tuple $\left\{\overrightarrow{B}_j, w_j\right\}$ in W, $\overrightarrow{B}_j$ is a high-order time series sample, which only contains numeric values, and w_j denotes the weight of it.

After the preprocessing, the high-order fuzzy time series are transformed into a new weighted high-order time series set, which consists of only numeric values. And, we applied this processing on each sample of the high-order fuzzy time series set and combined all of the weighted high-order time series sets obtained into one set $\overrightarrow{X}w$. However, it is hard to deal with the weighted dataset, which is actually a kind of unbalanced sampling dataset. The commonly solution is to applied SMOTE [41] to deal with the undersampling. The SMOTE could regenerated a set of high-order time series $\overrightarrow{X}'$ with the weighted high-order time series set $\overrightarrow{X}w$. For a particular sample $\left\{\overrightarrow{B}_j, w_j\right\}$ in $\overrightarrow{X}w$, it would have several corresponding samples in $\overrightarrow{X}'$, and

$$\frac{w_j}{\sum_{\left\{\overrightarrow{B}_i, w_i\right\} \in \overrightarrow{X}w} w_i} = \frac{N_{\left\{\overrightarrow{B}_j, w_j\right\}}}{N_{\overrightarrow{X}'}}, \tag{4}$$

where $N_{\left\{\overrightarrow{B}_j, w_j\right\}}$ denotes the number of corresponding samples in $\overrightarrow{X}'$ of $\left\{\overrightarrow{B}_j, w_j\right\}$ and $N_{\overrightarrow{X}'}$ denotes the number of samples in $\overrightarrow{X}'$.

The regenerated high-order time series set $\overrightarrow{X}'$ has the same distribution of sample as the weighted high-order time series set and could be easily processed by the multiple linear regression model and ANN.

3.2. A High-Order MLRM-Based Time Series Clustering Algorithm. The proposed model makes forecast by weighted summation of several linear models; this section describes the method to construct these linear models.

Forecasting of high-order time series is the matter of forecasting future values of a process with a certain number of observed time points. Supposed that we have observed $\overrightarrow{x}_t = [x_{t-L+1}, x_{t-L+2}, \ldots, x_t]$ with given integer t, and L is the number of observed time points of a high-order time series and wish to forecast x_{t+h}. The forecasting model of time series could be described as follows:

$$x_{t+h} = f\left(\overrightarrow{x}_t\right), \tag{5}$$

where $f(*)$ is a map function. The multiple linear regression model is one of the most popular map functions.

As for forecasting models that adopt multiple the linear regression model, some of them classify time series into several groups by performing clustering algorithm [42], which usually applies similarity defined by shape-based distance (e.g., Euclidean distance). Since those time series in the same group are similar in shape with each other, the multiple linear regression model should also be similar. However, the similarity of the multiple linear relationship is not completely equivalent to the similarity of shape, which means that similar time series may cause different future values and vice versa. Since the reason mentioned above, we proposed a high-order MLRM-based time series clustering algorithm.

With the preprocessing described in Section 3.1, we transformed the set high-order of fuzzy time series into a set of high-order time series. Let $\overrightarrow{X}' = [\overrightarrow{x}_1, \overrightarrow{x}_2, \ldots, \overrightarrow{x}_{N_{\overrightarrow{X}'}}]$ be the set of high-order time series obtained, where $\overrightarrow{x}_j = [x_{j,1}, x_{j,2}, \ldots, x_{j,L}, x_{j,L+h}] (j = 1, 2, \ldots, N_{\overrightarrow{X}'})$. The given dataset should be classified into K clusters.

Suppose $\mathbf{R} = [\overrightarrow{r}_1, \overrightarrow{r}_2, \ldots, \overrightarrow{r}_K]$, where $\overrightarrow{r}_i = [r_{i,0}, r_{i,1}, r_{i,2}, \ldots, r_{i,L}] (i = 1, 2, \ldots, K)$, be a set of multiple linear regression models, where multiple linear regression model $\overrightarrow{r}_i$ corresponding to cluster $\mathbf{C}_i$. The distance $D_{i,j}$ between cluster $\mathbf{C}_i$ and high-order time series $\mathbf{x}_j$ can be calculated as follows:

$$D_{i,j} = \sum_{l=1}^{L} x_{j,l} \cdot r_{i,l} + r_{i,0} - x_{j,L+h}. \tag{6}$$

The distance suggests the fitness of multiple linear regression model to the high-order time series. In other words, small value of $D_{i,j}$ means that the multiple linear regression model of time series $\overrightarrow{x}_j$ is similar with the multiple linear regression model $\overrightarrow{r}_i$.

The clustering algorithm could group the time series into clusters according to the distance defined in equation (6) so that the multiple linear regression models of time series within a cluster are highly similar. Thus, the proposed clustering algorithm aims at minimizing objective function J, given by

$$J(\mathbf{R}) = \sum_{i=1}^{K} \sum_{\overrightarrow{x}_j \in \mathbf{C}_i} D_{i,j}^2, \tag{7}$$

where $\mathbf{C}_i$ denotes the set of time series classified into the ith cluster and $\overrightarrow{x}_j \in \mathbf{C}_i$ indicates that the time series $\overrightarrow{x}_j$ is classified into cluster C_i.

Figure 2 shows the process of high-order MLRM-based time series clustering algorithm, and the details are described as follows.

3.2.1. Initialization. The above discussion shows that there are $N_{\overrightarrow{\mathbf{X}}'}$ high-order time series in dataset $\overrightarrow{X}'$ and K clusters wished to obtain. Since it is not easy to determine whether a couple of time series are similar in linear relationship or not, it is hard to give a better initialization of the cluster. Besides, for each clusters, in order to initialize its related linear model with multiple linear regression algorithm, there should be enough time series belonged to it. Thus, we divide $\overrightarrow{X}'$ into K clusters equally and randomly. Therefore, each cluster has around $N_{\overrightarrow{\mathbf{X}}'}/K$ time series belonged to it.

Since each cluster has a corresponding multiple linear regression model, K multiple linear regression models need to be constructed. As for cluster $\mathbf{C}_i$, which consists of a set of time series, each series has an individual weight. In order to adopt multiple linear regression, we applied a weighted multiple linear regression model which is described in [43]. $\mathbf{C}_i$ could be partitioned into a matrix of explanatory variables $\mathbf{C}_i^E$ and a vector of response variables $\mathbf{C}_i^R$. Besides, $\mathbf{C}_i^E$ and $\mathbf{C}_i^R$ could be obtained as follows:

$$\mathbf{C}_i^E = \begin{bmatrix} x_{1,1}^{\mathbf{C}_i} & x_{1,2}^{\mathbf{C}_i} & \cdots & x_{1,L}^{\mathbf{C}_i} \\ x_{2,1}^{\mathbf{C}_i} & x_{2,2}^{\mathbf{C}_i} & \cdots & x_{2,L}^{\mathbf{C}_i} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N_i,1}^{\mathbf{C}_i} & x_{N_i,2}^{\mathbf{C}_i} & \cdots & x_{N_i,L}^{\mathbf{C}_i} \end{bmatrix},$$
$$\mathbf{C}_i^R = \begin{bmatrix} x_{1,L+h}^{\mathbf{C}_i} \\ x_{2,L+h}^{\mathbf{C}_i} \\ \vdots \\ x_{N_i,L+h}^{\mathbf{C}_i} \end{bmatrix}, \tag{8}$$

where $\overrightarrow{x_j^{\mathbf{C}_i}} = [x_{j,1}^{\mathbf{C}_i}, x_{j,2}^{\mathbf{C}_i}, \ldots, x_{j,L}^{\mathbf{C}_i}, x_{j,L+h}^{\mathbf{C}_i}]$ is the jth time series which belonged to cluster $\mathbf{C}_i$ and N_i is the number of time series belonged to cluster $\mathbf{C}_i$.

According to the given $\mathbf{C}_i^E$ and $\mathbf{C}_i^R$, the multiple linear regression models $\overrightarrow{r}_i$ of cluster $\mathbf{C}_i$ is initialized as follows:

$$\overrightarrow{r}_i^0 = \mathrm{MLR}\left(\mathbf{C}_i^E, \mathbf{C}_i^R\right), \tag{9}$$

where MLR(*) is the regular process of multiple linear regression.

Hence, the set of multiple linear regression models $\mathbf{R}$ can be initialized as $\mathbf{R}^0 = [\overrightarrow{r}_1^0, \overrightarrow{r}_2^0, \ldots, \overrightarrow{r}_K^0]$. And, each multiple linear regression model relates to a cluster constructed.

Initialize a matrix of distance D^0 that has K rows and $N_{\overrightarrow{\mathbf{X}}'}$ columns. $D_{i,j}^0$ is the initial distance that between cluster C_j and sample x_i and can be initialized as

$$D_{i,j}^0 = \sum_{l=1}^{L} x_{j,l} \cdot r_{i,l}^0 + r_{i,0}^0 - x_{j,L+h}. \tag{10}$$

where $\mathbf{r}_i^0 = [r_{i,0}^0 r_{i,1}^0, \ldots, r_{i,L}^0]$ represents the initial multiple linear model corresponding to cluster C_i.

3.2.2. Update of the Distance Matrix and Classification of Time Series. During the pth round of iteration, assume that $\mathbf{R}^{p-1}$ is the set of multiple linear regression models obtained in the previous round. The distance matrix $\mathbf{D}^p$ is updated according to $\mathbf{R}^{p-1}$. As the distance defined in equation (6), $D_{i,j}^p$ can be updated by:

$$D_{i,j}^p = \sum_{l=1}^{L} x_{j,l} \cdot r_{i,l}^{p-1} + r_{i,0}^{p-1} - x_{j,L+h}, \tag{11}$$

where $\mathbf{r}_i^{p-1} = [r_{i,0}^{p-1} r_{i,1}^{p-1}, \ldots, r_{i,L}^{p-1}]$ represents the multiple linear model corresponding to cluster C_i in pth iteration.

Each cluster's time series set would be set as empty, and each time series would be reclassified into a cluster, constructed in the previous round, which leads to the minimum distance. For example, given time series $\mathbf{x}_j$, and $D_{i,j}^p$ represents the distance between $\mathbf{x}_j$ and cluster $\mathbf{C}_i^p$, which could be obtained by distance matrix defined by equation (11). By comparing $D_{i,j}^p (i = 1, 2, \ldots, K)$, $\mathbf{x}_j$ will be added to the time series set of the cluster that with the least distance. In other words, $\mathbf{x}_j$ would be reclassified into $\mathbf{C}_i^p$ only when

$$D_{i,j}^p = \mathrm{Min}\left(D_{i,1}^p, D_{i,2}^p, \ldots, D_{i,K}^p\right), \tag{12}$$

where Min(*) represents the minimal function.

3.2.3. Update of the Set of Multiple Linear Regression Models. After all of the time series have been reclassified, the multiple linear models of clusters should also be updated. For cluster $\mathbf{C}_i$ in the pth round, if there is no time series belonged to it, continue with the next cluster; otherwise, the corresponding multiple linear regression model will be calculated according to all of time series that are reclassified into $\mathbf{C}_i$ by the method described in the step of initialization. Yet, the multiple linear regression model $\mathbf{r}_i^p$ could be built by partitioning $\mathbf{C}_i$ into a matrix of explanatory variables and a vector of response variables and applying the multiple linear regression.

After all of the multiple linear regression models have been updated, if there are empty clusters that are ignored, new models should be generated to replace them. Suppose that there are n_e empty clusters. First, an estimation of each nonempty cluster should be done. The estimation result $\mathbf{E}_i$ of cluster $\mathbf{C}_i$ is calculated as follows:

$$\mathbf{E}_i = \sum_{\mathbf{x}_j \in \mathbf{C}_i} w_j \cdot D_{i,j}^2. \tag{13}$$

The small $\mathbf{E}_i$ suggests that time series are highly similar with each other within $\mathbf{C}_i$.

In order to generate n_e new models, we choose n_e worst-performing clusters, whose estimations $\mathbf{E}_i$ are larger than others. For each cluster, half of time series belonged to it is randomly selected, and a new multiple linear regression model is generated based on a cluster formed by them. Then, n_e new models are obtained to replace old models of empty clusters.

If the number of clusters obtained $K - n_e$ is less than n_e, which means we could not find n_e worst-performing clusters, let $n_e' = K - n_e$, and generate n_e' new models by the process described above.

3.2.4. Stopping Criterion. If the multiple linear regression $\mathbf{R}^p$ shows no differences with $\mathbf{R}^{p-1}$, or this process reaches the given maximum number of iterations, the iteration process

ends. $\mathbf{R}^p$ is the final output of the proposed clustering algorithm.

Finally, we get a set of multiple linear regression models $\mathbf{R}$, and each of them relates to a cluster.

3.3. A New Forecasting Model Based on Fuzzy Time Series and High-Order MLRM-Based Time Series Clustering. After the clustering defined former, we obtained a set of linear models. In order to make forecasting, we need to calculate the weighted summation of these linear models' outputs. This section describes a forecasting model which adopts ANNs to give the weights of linear models.

The fuzzy time series model usually exacts fuzzy logic relationships to build the relationships between current state and next state, and the state transition matrix is the core of the model. In the MLRM-based forecasting algorithm, time series are clustered as other fuzzy time series models, but model is built without using state transition matrix. Instead, models of multiple linear regression are applied for mapping variables of current state into space of next state.

The proposed fuzzy time series model works on the high-order time series built as described in Section 3.1, and it contains four steps as other existing fuzzy time series models explained in Section 2. During making forecast, existing fuzzy time series models based on linear model [31] employed grade of membership, obtained by fuzzy C-means clustering. However, the distance defined by equation (6) depends on the numeric value to forecast. Since we could not obtain the numeric value before forecasting, this distance cannot be calculated. Thus, we build a model for each multiple linear regression model to calculate the grade of membership with only observed data.

Suppose the set of multiple linear regression models R, calculated as described in Section 3.2, containing K models. $\overrightarrow{r}_i = [r_{i,0}, r_{i,1}, r_{i,2}, \ldots, r_{i,L}]\,(i = 1, 2, \ldots, K)$ is the model corresponding to the ith cluster. Let the membership function of ith cluster be $\mathrm{Mem}_i(*)$. Then, for a high-order time series $\overrightarrow{x}_j = [x_{j,1}, x_{j,2}, \ldots, x_{j,L}]$ obtained by the methods described in Section 3.1, the forecast result $x_{j+h}^{\mathrm{predict}}$ of x_{j+h} can be calculated by weighted summing results of all multiple linear regression models according to the grades of membership:

$$x_{j+h}^{\mathrm{predict}} = \sum_{i=1}^{K} \mathrm{Mem}_i\left(\overrightarrow{x}_j\right) \cdot \left(\overrightarrow{x}_j \cdot \overrightarrow{r}_i\right), \tag{14}$$

where $\overrightarrow{x}_j \cdot \overrightarrow{r}_i$ is an algebraic dot production and $\mathrm{Mem}_i(\overrightarrow{x}_j)$ is the grade of membership between $\overrightarrow{x}_j$ and the ith cluster.

With given time series, the forecast model aims at finding a set of membership functions $\mathbf{M}$ so that the root mean squared error (RMSE) of prediction results is minimized as follows:

$$\begin{aligned}\mathbf{M} &= \left\{\widehat{\mathrm{Mem}}_i(*) \mid i = 1, 2, \ldots, K\right\} \\ &= \mathrm{argmin}_{\mathbf{M}} \sqrt{\frac{\sum_{j=1}^{N_{\overrightarrow{\mathbf{x}}'}} \left(x_{j+h} - x_{j+h}^{\mathrm{predict}}\right)^2}{N_{\overrightarrow{\mathbf{x}}'}}}.\end{aligned} \tag{15}$$

To build the membership function models, we construct an ANN for each cluster. Then, the forecasting model can be described as in Figure 3.

3.3.1. Learning Algorithm of Forecasting Model Based on High-Order MLRM-Based Time Series Clustering. Traditional ANN usually adopts supervised learning, which uses a set of example pairs $(x, y), x \in X$ and $y \in Y$, and aims at finding a function $f: X \longrightarrow Y$ in the allowed class of functions that matches the examples. Generally, the cost function is related to the mismatch between the mapping f and the data. Therefore, the expected output Y of the ANN is necessary.

In the forecasting model based on high-order MLRM-based time series clustering, each ANN corresponds to a membership function. Since we cannot give expected grades of membership between samples and multiple linear regression model, the expected outputs of ANNs is missing. In other words, we cannot train each ANNs individually. Thus, we designed a learning algorithm of the whole forecasting model.

The learning algorithm can be divided into two phases: propagation and backpropagation.

During the propagation, for each sample $\overrightarrow{x}_j$, each ANN takes anterior L-dimension of sample $\overrightarrow{x}_j$ as inputs and generates its own output value $\mathrm{Mem}_i(\overrightarrow{x}_j)$ as a grade of membership. After all grades of membership obtained, the forecasting model could calculate the prediction result by equation (14). The cost of the forecasting model is defined by squared error function as follows:

$$E = \frac{1}{2}\left\|e^j\right\|^2, \tag{16}$$

where $e^j = x_{j+h}^{\mathrm{prediction}} - x_{j+h}$ denotes the forecasting error of the jth sample.

After the process of propagation, the forecasting error propagates backward throughout the model to adjust the parameters. The derivative of the forecasting error with respect to a weight $\mathrm{Mem}_i(\overrightarrow{x}_j)$ can be expressed as follows:

$$\frac{\partial E}{\partial \mathrm{Mem}_i\left(\overrightarrow{x}_j\right)} = \frac{1}{2} \cdot \frac{\partial \left\|e^j\right\|^2}{\partial \mathrm{Mem}_i\left(\overrightarrow{x}_j\right)}. \tag{17}$$

Given j, the derivative part of equation (17) can be simplified to

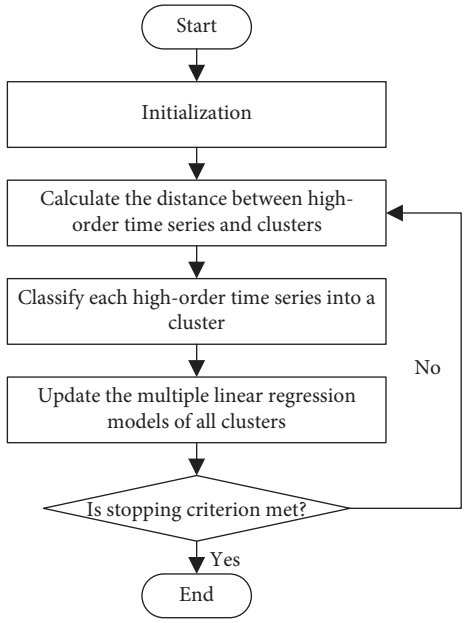

Figure 2: Flowchart of proposed high-order MLRM-based time series clustering algorithm.

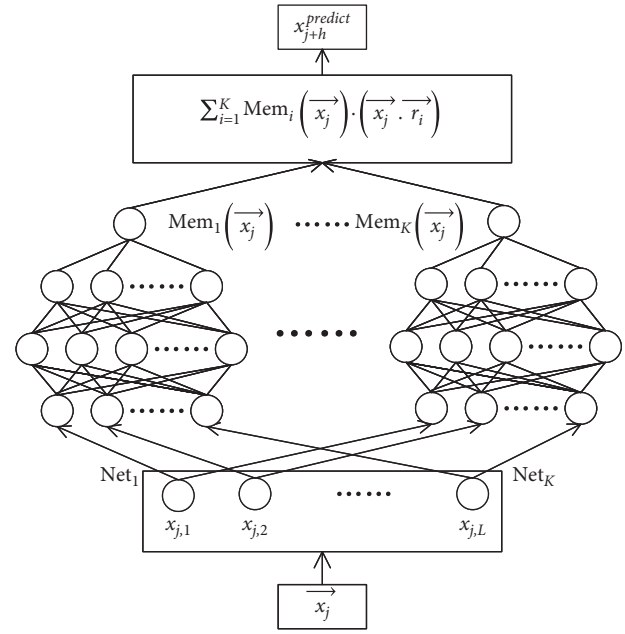

Figure 3: Forecasting model based on high-order MLRM-based time series clustering.

$$\begin{aligned}\frac{\partial\left\|e^j\right\|^2}{\partial \mathrm{Mem}_i\left(\overrightarrow{x}_j\right)} &= \frac{\partial\left\|e^j\right\|^2}{\partial x_{j+h}^{\mathrm{prediction}}}\frac{\partial x_{j+h}^{\mathrm{prediction}}}{\partial \mathrm{Mem}_i\left(\overrightarrow{x}_j\right)}\\ &= \frac{\partial\left\|x_{j+h}^{\mathrm{prediction}}-x_{j+h}\right\|^2}{\partial x_{j+h}^{\mathrm{prediction}}}\frac{\partial\left(\sum_{k=1}^{K}\mathrm{Mem}_k\left(\overrightarrow{x}_j\right)\cdot\left(\overrightarrow{x}_j\cdot\overrightarrow{r}_k\right)\right)}{\partial \mathrm{Mem}_i\left(\overrightarrow{x}_j\right)}\\ &= 2e^j\left(\overrightarrow{x}_j\cdot\overrightarrow{r}_i\right).\end{aligned} \tag{18}$$

Then, equation (17) could be expressed as follows:

$$\frac{\partial E}{\partial \mathrm{Mem}_i\left(\overrightarrow{x}_j\right)} = e^j\left(\overrightarrow{x}_j\cdot\overrightarrow{r}_i\right). \tag{19}$$

Treating all ANNs as equal, we supposed a constant η^0 as learning ratio for the weight sum layer of our model. Let

$$\sum_{i=1}^{K}\frac{\partial E}{\partial \mathrm{Mem}_i\left(\overrightarrow{x}_j\right)}\eta^0 = E, \tag{20}$$

then, η^0 can be obtained as follows:

$$\eta^0 = \frac{\sum_{j=1}^{N_{\overrightarrow{x}'}}\left\|e^j\right\|^2}{\sum_{i=1}^{K}e^j\left(\overrightarrow{x}_j\cdot\overrightarrow{r}_i\right)}. \tag{21}$$

Thus, for the ANN corresponding to the ith multiple linear regression model, by using the result of equations (19) and (21), the cost E_i is defined as follows:

$$E_i = \frac{\partial E}{\partial \mathrm{Mem}_i\left(\overrightarrow{x}_j\right)}\eta^0 = e^j\left(\overrightarrow{x}_j\cdot\overrightarrow{r}_i\right)\frac{\sum_{j=1}^{N_{\overrightarrow{x}'}}\left\|e^j\right\|^2}{\sum_{i=1}^{K}e^j\left(\overrightarrow{x}_j\cdot\overrightarrow{r}_i\right)}, \tag{22}$$

which suggests the error of the ith ANN.

Since we got error of each ANN, the traditional backpropagation [44] algorithm could be executed. During the backpropagation of ANN, all neurons update their weights and offsets according to gradients. After that, another sample will be employed to train the model unless the stopping criterion is satisfied.

The pseudocode for learning algorithm of the proposed forecasting model is presented as follows (Algorithm 1):

3.4. A Case of Forecasting the TAIEX. TAIEX data are frequently used for evaluation of the FTS algorithms. This section presents a detailed example on the TAIEX data of 1999 to explain our model. There are 266 samples in this dataset. 221 samples are used for training and others for testing. The proposed forecasting model is presented as follows:

Step 1. In order to explain the details of the proposed model, we initialize number of clusters $K = 5$, number of observed time points of a high-order time series $L = 3$, and time to predict $h = 1$.

Input:
All multiple linear regression models $\{\overrightarrow{r_i} \mid i = 1, \ldots, K\}$;
Training sets $\overrightarrow{X}'$;
Output:
Forecasting model with ANNs $\{\text{Net}_i \mid i = 1, \ldots, K\}$;
(1) $\{\text{Net}_i \mid i = 1, \ldots, K\} \longleftarrow$ initialize all networks
(2) $\overrightarrow{X}' \longleftarrow$ normalize $\overrightarrow{X}'$
(3) **Repeat**
(4) **For all** $\overrightarrow{x_j} \in \overrightarrow{X}'$ do
(5) **For all** $\text{Net}_i \in \{\text{Net}_i \mid i = 1, \ldots, K\}$ do
(6) Calculate the output value $\text{Mem}_i(\overrightarrow{x}_j)$ of Net_i
(7) **End for**
(8) $x_{j+h}^{\text{predict}} \longleftarrow \sum_{i=1}^{K} \text{Mem}_i(\overrightarrow{x}_j) \cdot (\overrightarrow{x}_j \cdot \overrightarrow{r}_i)$,
(9) $E \longleftarrow$ cost defined in equation (16)
(10) **For all** $\text{Net}_i \in \{\text{Net}_i \mid i = 1, \ldots, K\}$ do
(11) Calculate the partial derivative defined in equation (19)
(12) **End for**
(13) $\eta^0 \longleftarrow (\sum_{j=1}^{N_{\overrightarrow{x}'}} \|e^j\|^2 / \sum_{i=1}^{K} e^j(\overrightarrow{x}_j \overrightarrow{r}_i))$
(14) **For all** $\text{Net}_i \in \{\text{Net}_i \mid i = 1, \ldots, K\}$ do
(15) $E_i \longleftarrow (\partial E / \partial \text{Mem}_i(\overrightarrow{x}_j))\eta^0$
(16) Compute Δw and Δb for all weights and offsets
(17) Update network weights and offsets
(18) **End for**
(19) **End for**
(20) Until RMSE is small enough or fallen into a local minimum

Algorithm 1: Learning algorithm of the proposed model.

Step 2. Normalize the data by calculating the daily percentage change of the closing price as the universe of the discourse. Let the $D_{\max}$ and $D_{\min}$ be the maximum and the minimum of daily change. Then, the scope of the universe can be represented as $U = [D_{\min} - \delta(D_{\max} - D_{\min}), D_{\max} + \delta(D_{\max} - D_{\min})]$ $(\delta \geq 0)$. The percentage change is calculated by using

$$x_j = \frac{\text{closed}_j - \text{closed}_{j-1}}{\text{closed}_{j-1}}. \tag{23}$$

Step 3. Transform the time series of percentage changes, in both training set and testing set, into vectors of time series as follows:

$$\overrightarrow{x_j} = \left(x_{j-L+1}, x_{j-L+2}, \ldots, x_j, x_{j+h}\right). \tag{24}$$

Step 4. For the scope of the universe U, apply an equivalent-interval partition described in [21]. Let the number of intervals $p = 10$. With obtained intervals $u_1, u_2, \ldots, u_p$, fuzzy sets $A_1, A_2, \ldots, A_p$ are defined as follows:

$$\begin{aligned}
A_1 &= \frac{1}{u_1} + \frac{0.5}{u_2} + \cdots + \frac{0}{u_{p-1}} + \frac{0}{u_p},\\
A_2 &= \frac{0.5}{u_1} + \frac{1}{u_2} + \cdots + \frac{0}{u_{p-1}} + \frac{0}{u_p},\\
&\cdots\\
A_{p-1} &= \frac{0}{u_1} + \frac{0}{u_2} + \cdots + \frac{1}{u_{p-1}} + \frac{0.5}{u_p},\\
A_p &= \frac{0}{u_1} + \frac{0}{u_2} + \cdots + \frac{0.5}{u_{p-1}} + \frac{1}{u_p}.
\end{aligned} \tag{25}$$

Furthermore, if there are missing values existing, these values could be represented by a special fuzzy set, $A_0 = [\{u_1, 1/p\}, \{u_2, 1/p\}], \ldots, \{u_p, 1/p\}]$, which employs equal grade of membership for each interval.

Then, the high-order time series $\overrightarrow{x_j}$ can be fuzzified as follows:

$$\overrightarrow{x_j}^* = \left(A_{x_{j-L+1}}, A_{x_{j-L+2}}, \ldots, A_{x_j}, x_{j+h}\right), \tag{26}$$

where Ax_i denotes the fuzzy set corresponding to x_i.

Step 5. In order to employ the linear model, we transmit $\overrightarrow{x_j}^*$ into numeric vectors as the methods explained in Section 3.1. Since, each fuzzy set could be replaced by p intervals and the grades of membership, defined in equation (25), $\overrightarrow{x_j}^*$ could generate p^L time series vectors. And, for each vector, the grade of membership related to it can be calculated by accumulating grades of membership related to intervals in each dimension. For example, $(A_1, A_2, A_{10}, x_j + h)$ could be transformed into

$$\begin{Bmatrix} \{[u_1,u_1,u_1],0\}, & \cdots & \{[u_1,u_1,u_10],0.5\}, \\ \{[u_1,u_2,u_1],0\}, & \cdots & \{[u_1,u_2,u_10],1\}, \\ \cdots & \cdots & \cdots \\ \{[u_10,u_10,u_1],0\}, & \cdots & \{[u_10,u_10,u_10],0\} \end{Bmatrix}$$
$$\Rightarrow \begin{Bmatrix} \{[u_1,u_1,u_9],0.25\}, & \{[u_1,u_1,u_{10}],0.5\}, \\ \{[u_1,u_2,u_9],0.5\}, & \{[u_1,u_2,u_{10}],1\}, \\ \{[u_1,u_3,u_9],0.25\}, & \{[u_1,u_3,u_{10}],0.5\}, \\ \{[u_2,u_1,u_9],0.125\}, & \{[u_2,u_1,u_{10}],0.25\}, \\ \{[u_2,u_2,u_9],0.25\}, & \{[u_2,u_2,u_{10}],0.5\}, \\ \{[u_2,u_3,u_9],0.125\}, & \{[u_2,u_3,u_{10}],0.25\} \end{Bmatrix}. \tag{27}$$

After the transformation, we get a set of tuples. For each tuple, it consists of a vector of intervals and a weight calculated by accumulating grades of membership related to intervals in each dimension. So, we just collect the tuples with nonzero weight and formed a weight high-order time series set $\overrightarrow{X}w$.

Step 6. After that, we get weighted numeric vectors set $\overrightarrow{X}w$. Besides, grade of membership s_j represents the weight of the vectors in training set, and the value of it can only vary within $\{1, 0.5, 0.125\}$ in this case since this dataset does not contain missing value. Then, to form a new training set with high-order time series $\overrightarrow{X}'$ with $\overrightarrow{X}w$, we applied the SMOTE defined in [41]. Also, we partition the $\overrightarrow{X}'$ into two parts, including training set $\mathbf{X}_{\text{train}}$ and testing set $\mathbf{X}_{\text{test}}$.

Step 7. Based on the clustering algorithm described in Section 3.2, cluster these high-order time series vectors in $\mathbf{X}_{\text{train}}$ and calculate the centers of these clusters. The final output $\mathbf{R}^p$ of the clustering and linear models corresponding is

$$\mathbf{R}^p = \begin{bmatrix} -1.258 & -0.018 & 1.198 & 0.212 \\ 1.449 & -0.143 & -0.236 & 0.670 \\ -0.384 & -0.591 & -0.173 & 0.015 \\ -0.339 & 0.379 & 0.418 & 0.372 \\ 1.358 & 0.035 & -0.047 & -0.450 \end{bmatrix}$$
$$\Rightarrow \begin{cases} y_1 = -1.258 - 0.018x_{j-2} + 1.198x_{j-1} + 0.212x_j, \\ y_2 = 1.449 - 0.143x_{j-2} - 0.236x_{j-1} + 0.670x_j, \\ y_3 = -0.384 - 0.591x_{j-2} - 0.173x_{j-1} + 0.015x_j, \\ y_4 = -0.339 + 0.379x_{j-2} + 0.418x_{j-1} + 0.372x_j, \\ y_5 = 1.358 + 0.035x_{j-2} - 0.047x_{j-1} - 0.450x_j. \end{cases} \tag{28}$$

Step 8. With training set $\mathbf{X}_{\text{train}}$ and linear models $\mathbf{R}^p$, a forecasting model is trained by methods described in Section 2. K ANNs are built during training process, and each takes anterior L-dimension of sample $\overrightarrow{x_j}$ as input and gives the grade of membership $\text{Mem}_i(\overrightarrow{x_j})(i = 1, 2, \ldots, K)$ between sample and multiple linear regression model as output.

Step 9. Make a forecast of each sample in the testing set $\mathbf{X}_{\text{test}}$. The forecast value x_{j+1}^{predict} of sample $\overrightarrow{x_j}$ can be obtained by

$$\begin{aligned} x_{j+1}^{\text{predict}} &= \sum_{i=1}^{K} \text{Mem}_i\left(\overrightarrow{x_j}\right) \cdot y_i \\ &= \text{Mem}_1\left(\overrightarrow{x_j}\right) \cdot y_1 + \text{Mem}_2\left(\overrightarrow{x_j}\right) \cdot y_2 + \text{Mem}_3\left(\overrightarrow{x_j}\right) \cdot y_3 \\ &\quad + \text{Mem}_4\left(\overrightarrow{x_j}\right) \cdot y_4 + \text{Mem}_5\left(\overrightarrow{x_j}\right) \cdot y_5. \end{aligned} \tag{29}$$

Note that if the sample in the testing set contains uncertainness, the sample should be transformed into a weighted high-order time series set by the method explained in Section 3.1. Then, make forecasting for each weighted high-order time series as formula (29), and the final forecasting result could be obtained by weighted summation of these forecast results according to the weights of these high-order time series.

Step 10. The forecasting results are formulated by anti-normalizing the result of Step 9.

The number of clusters K is not optional and should be computed. Optimal number of clusters is simply found by running the algorithm for $K = 1, 2, 3, \ldots$ and choosing K corresponding to the minimum testing error. During the process of finding optimal K, the iteration should be stopped if there are redundant clusters during Step 7.

4. Experimental Results and Analysis

To demonstrate the performance of the proposed model, we made experiments on the dataset of historical closing prices of the TAIEX [45]. For each year, we used historical data from January to October to construct the training set and the data from November to December as the testing set. Besides, the structure of ANN, including the number of layers, the number of neurons ,and the type of transfer function were decided by experimenting on the training set to give the best performance. Some main parameters are listed in Table 1.

First of all, we tested our high-order MLRM-based time series clustering algorithm with other clustering algorithms, such as K-means clustering, hierarchical clustering and density-based spatial clustering of applications with noise (DBSCAN). For the results of clustering, we applied multiple linear regression to construct a linear model for each cluster. So, the cost of the clustering results can be calculated by summing all linear models' cost, which can be expressed as

$$E = \sum_{i=1}^{K} L_i, \tag{30}$$

Table 1: Main simulation parameters.

Parameters	Value
Time to predict h	1
Number of the intervals p	10
Stopping criterion threshold of the proposed clustering algorithm	0.01
Number of cluster K	5
Learning rate of ANN	0.05
Learning goal of ANN	10^{-5}

where K is number of clusters and the L_i denotes the cost, defined by common linear regression model, for ith cluster. Given $K = 4$ (for proposed algorithm, K-means clustering, and hierarchical clustering) and $L = 2$, we made an experiment by using the TAIEX data from 1999 to 2004, and the experiment results is given by Figure 4. Since the smaller cost suggests the better performance, our high-order MLRM-based time series clustering algorithm is better than other three algorithms. Especially, the small cost means that the cluster results are suitable for the linear model, so our clustering algorithm is more suitable for the proposed forecasting model.

For a given order $L = 2$, we made an experiment to choose the suitable number of ANNs K by using the TAIEX data of year 1999. During the experiment, the RMSE of the forecasting results is calculated for each K. Since the model built by ANN is random, for each K, the forecasting should be repeated 10 times to calculate the average RMSE. Figure 5 shows the relationship between the number of ANNs K and the average RMSE of the forecasting results. The results suggest that the RMSE decreases when the number of ANNs K increases. However, the RMSE becomes stabilized when $K \geq 4$, since there are redundant clusters.

In the similar way, we made an experiment to decide the order of high-order time series with given $K = 4$ by using the TAIEX data of year 1999. The relationship between the order L and the average RMSE is depicted by Figure 6. The RMSE of the forecasting results varies little when the order of high-order time series $L \geq 2$.

Also, we compared running time of the proposed model with different L and K. For each K and L, the proposed model has been run for 10 times, and the average running time is calculated. Results are given in Figure 7. The running time has a linear relationship with the number of ANNs K. Also, it has an exponential relationship with the order of high-order time series L. Thus, the following experiments applied parameters that $K = 4$ and $L = 2$, which ensures a well forecasting accuracy as well as an acceptable running time.

Besides, we made forecast of TAIEX, with the data from 1990 to 2004, by the proposed model and other existing FTS-based forecast models [4–8, 10–12]. All of the RMSEs in Tables 2 and 3 are given by [11]. Table 2 shows the comparison of RMSE and average RMSE between the proposed model and other existing FTS methods by using the TAIEX historical data from 1990 to 1999, and Table 3 shows the comparison by using the TAIEX historical data from 1999 to 2004. The proposed method gets a much more accurate forecast result than other methods, and the average RMSE of the proposed method, which equals 42.91 with the data of the TAIEX from 1990 to 1999, and 36.91 with the data of from 1999 to 2004. Since the forecasting results depend on the results of clustering and ANN and both of these two procedures' results are dynamic, the forecasting results obtained varied in each run. Thus, we repeated our model for 100 times with TAIEX data from 1990 to 2004 and calculated the average $\overline{\text{RMSE}}$, variance Var, and standard deviation Std of the forecasting RMSEs. The results are displayed in Table 4.

Moreover, since most of the IFS-based [16, 18] and HFS-based [17, 19] models are experimented on the dataset of State Bank of India (SBI) at BSE, we compared our model with other models by using the market prices of SBI share at BSE. As the experiment in [19], we took one sample per month as testing sample, and others as training samples. Figure 8 shows the forecasting results of testing set. Our proposed model suggests a better forecasting accuracy than other FTS models. To make more clear comparison, we repeat the experiment 10 times and calculated the RMSE, AFE, R, and R^2 as defined in [19]. The forecasting results given in Table 5 suggests that our model gives the best RMSE and AFE, which indicates the highest forecasting accuracy.

In addition, we compared our model with some nonfuzzy methods, including Askari's CFTS model [31], SVM-based model [46], and LSTM-based model [47] by forecasting the TAIEX data from 1999 to 2004. The results are presented in Table 6. Our model and CFTS model is much better than others, which indicates that linear model is more suitable for this problem. Furthermore, our model shows the best performance with the RMSE equaled 36.40, which is a little better than CFTS model's 40.72.

At last, we tested the robustness by using data from year 1999 to 2004. We randomly generated deviations within ρ% for each value of time point in the training set. Then, we compared the change of RMSE with other nonfuzzy methods [31, 46, 47]. The change of RMSE can be evaluated as

$$r_{\text{change}} = \frac{\text{RMSE}' - \text{RMSE}}{\text{RMSE}}, \tag{31}$$

where RMSE denotes the forecasting RMSE of origin dataset and RMSE$'$ denotes the forecasting RMSE of modified dataset. Figure 9 shows that the proposed model is less insensitive to fluctuation of the training set than other models, which suggests that the proposed model has better robustness (Figures 9 and 10).

Furthermore, we randomly removed values of σ% time points of the training set. For nonfuzzy models, we replaced the value which is removed with the average value of whole time points in training set. For fuzzy time series model, these values are defined by equivalent grades of membership related to intervals of universal. This experiment is also executed on the data of year 2002. Figure 10 shows that the missing values have little influence on forecasting results of the proposed model. Also, the proposed model shows a better robustness to the missing values than other two models.

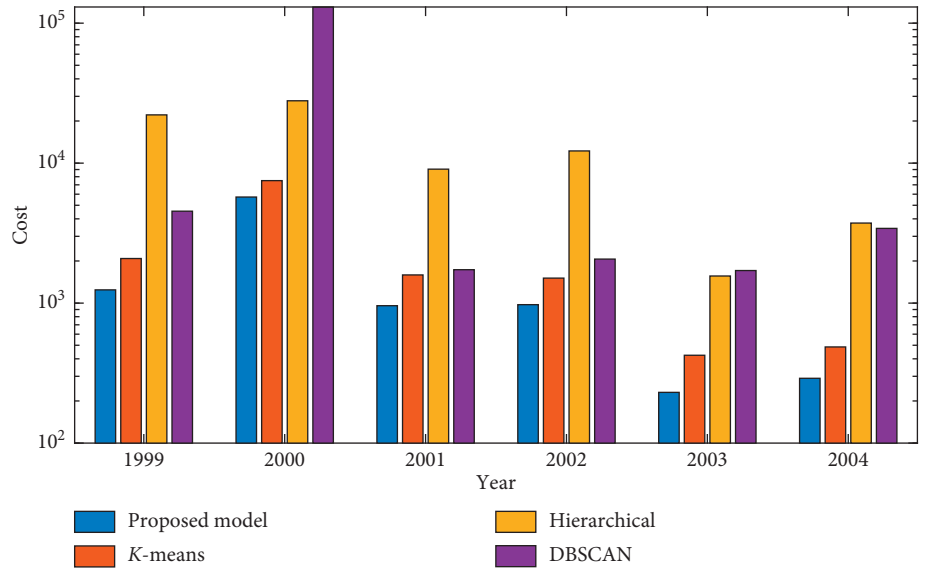

Figure 4: Comparison of performance between high-order MLRM-based time series clustering algorithm with other clustering algorithms.

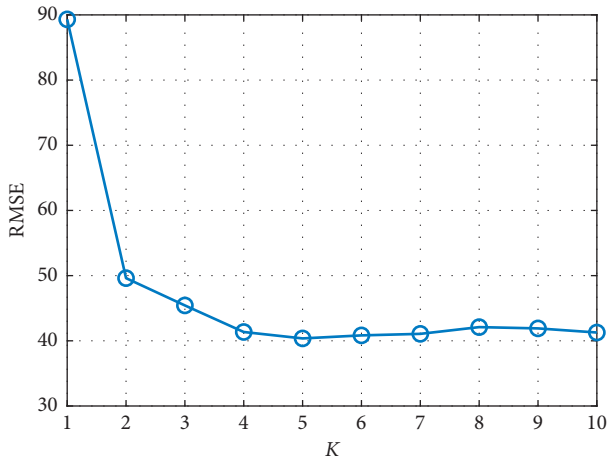

Figure 5: Relationship between the number of ANNs K and the average RMSE of the forecasting results.

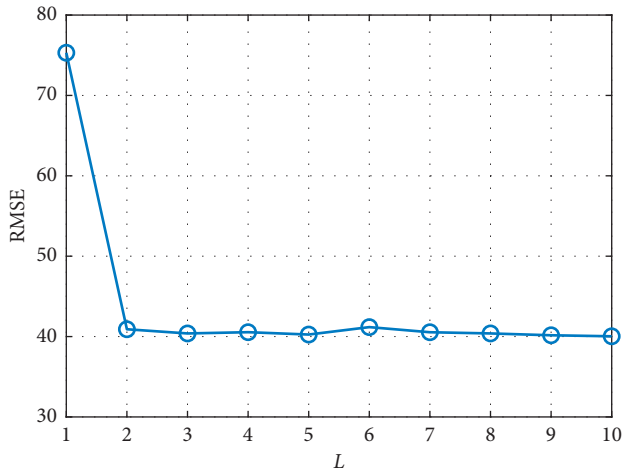

Figure 6: Relationship between the order of high-order time series L and the average RMSE of the forecasting results.

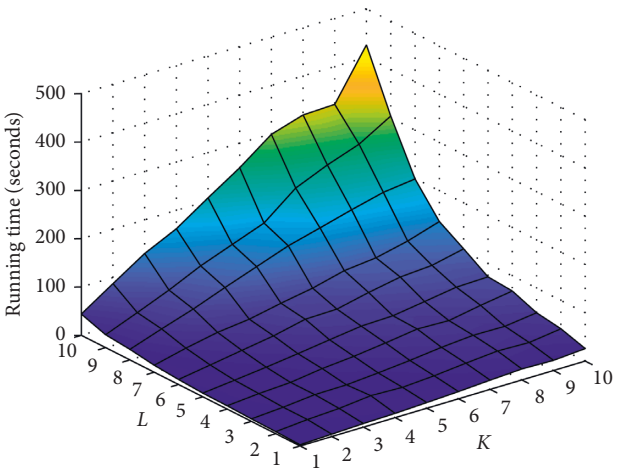

Figure 7: Running time of the proposed model with different L and K.

Table 2: Comparison of RMSEs and average RMSEs for different FTS methods for forecasting the TAIEX from 1990 to 1999.

Models	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	Average
Conventional model [4]											
Average-based lengths	220	80	60	110	112	79	54	148	167	149	117.9
Distribution-based lengths	270	79	60	105	132	79	52	149	159	159	124.4
Weighted model [4]											
Average-based lengths	227	61	67	105	135	70	54	133	151	142	114.5
Distribution-based lengths	266	67	56	105	114	70	52	152	154	145	118.1
Chen and Chen's model [5]											
Use Dow Jones	172.89	72.87	43.44	103.21	78.63	66.66	59.75	139.68	124.44	114.47	97.70
Use NASDAQ	169.93	66.12	49.61	104.75	75.66	67.01	60.90	140.86	144.13	119.32	99.83
Use Dow Jones and NASDAQ	172.99	74.85	43.78	101.38	78.13	68.14	61.26	139.29	132.94	116.64	98.94
Chen's model [7]											
Use Dow Jones	175.67	43.31	42.82	104.34	94.72	54.00	50.47	138.31	120.00	99.87	92.31
Use NASDAQ	176.60	43.73	42.42	104.18	95.04	54.49	49.84	139.85	119.22	102.60	92.80
Use Dow Jones and NASDAQ	174.62	43.22	42.66	104.17	94.60	54.24	50.50	138.51	117.87	101.33	92.17
Chen's model [8]											
Use Dow Jones	174.35	43.78	43.12	108.02	88.32	53.69	51.02	139.86	113.58	102.34	91.81
Use NASDAQ	176.17	43.16	43.34	106.66	87.95	53.30	51.10	138.41	113.88	102.11	91.61
Chen's model [6]											
Use Dow Jones	180.36	43.80	43.06	104.89	75.35	55.06	50.06	133.82	112.11	103.90	90.24
Use NASDAQ	174.15	45.04	42.10	104.94	76.40	54.96	50.17	133.45	113.37	104.99	89.96
Cai's model [10]	187.10	39.58	39.37	101.80	76.32	56.05	49.45	123.86	118.41	102.34	89.44
Cheng's model [12]	168.77	46.63	44.53	105.30	77.17	50.34	53.19	131.45	115.89	100.74	89.40
The proposed model	71.73	29.09	20.18	45.55	51.99	29.60	30.15	56.88	53.04	40.91	42.91

Table 3: Comparison of RMSEs and average RMSEs for different FTS methods for forecasting the TAIEX from 1999 to 2004.

Models	1999	2000	2001	2002	2003	2004	Average
Chen's model [4]							
Use NASDAQ	123.64	131.10	115.08	73.06	66.36	60.48	94.95
Use Dow Jones	101.97	148.85	113.70	79.81	64.08	82.32	98.46
Use M1B	156.92	142.70	132.76	96.06	90.27	100.10	119.80
Use Dow Jones and NASDAQ	106.34	130.13	113.33	72.33	60.29	68.07	91.75
Use NASDAQ and M1B	116.22	134.63	116.59	76.48	53.51	69.29	94.45
Use NASDAQ and Dow Jones and M1B	111.7	129.42	113.67	66.82	56.1	64.76	90.41
Chen's model [6]							
Use Dow Jones	115.47	127.51	121.98	74.65	66.02	58.89	94.09
Use NASDAQ	119.32	129.87	123.12	71.01	65.14	61.94	95.07
Use M1B	120.01	129.87	117.61	85.85	63.10	67.29	97.29
Use Dow Jones and NASDAQ	116.64	123.62	123.85	71.98	58.06	57.73	91.98
Use Dow Jones and M1B	116.59	127.71	115.33	77.96	60.32	65.86	93.96
Use NASDAQ and M1B	114.87	128.37	123.15	74.05	67.83	65.09	95.58
Use NASDAQ and Dow Jones and M1B	112.47	131.04	117.86	77.38	60.65	65.09	94.08
Chen's model [3]							
Use NASDAQ	123.64	131.10	115.08	73.06	66.36	60.48	94.95
Use Dow Jones	101.97	148.85	113.70	79.81	64.08	82.32	98.46
Use M1B	156.92	142.70	132.76	96.06	90.27	100.10	119.80
Use Dow Jones and NASDAQ	106.34	130.13	113.33	72.33	60.29	68.07	91.75
Use NASDAQ and M1B	116.22	134.63	116.59	76.48	53.51	69.29	94.45
Use NASDAQ and Dow Jones and M1B	111.70	129.42	113.67	66.82	56.10	64.76	90.41
Chen's model [10]							
Use Dow Jones	102.34	131.25	113.62	65.77	52.23	56.16	86.89
Use NASDAQ	102.11	131.30	113.83	66.45	52.83	54.17	86.78
Use M1B	103.52	131.36	112.55	66.23	53.20	55.36	87.04
Chen's model [7]							
Use Dow Jones	103.90	127.32	115.37	64.71	52.84	53.36	86.25
Use NASDAQ	104.99	124.52	114.66	64.79	53.63	52.96	85.93
Use M1B	105.61	127.37	115.46	66.07	53.67	53.30	86.91
Chen's model [8]							
Use TAIFEX	100.83	122.20	120.30	72.23	56.89	55.40	87.98
Use Dow Jones	99.87	122.75	117.18	68.45	53.96	52.55	85.79

Table 3: Continued.

Models	1999	2000	2001	2002	2003	2004	Average
Use NASDAQ	102.60	119.98	114.81	69.07	53.16	53.57	85.53
Use M1B	101.22	123.99	117.75	70.63	54.92	55.29	87.30
Use Dow Jones and NASDAQ	101.33	121.27	114.48	67.18	52.72	52.27	84.88
Use Dow Jones and M1B	100.59	124.10	116.28	68.11	53.50	53.33	85.99
Use NASDAQ and M1B	102.25	122.47	115.02	68.51	52.82	53.99	85.84
Use NASDAQ and Dow Jones and M1B	101.47	122.88	114.47	67.17	52.49	52.84	85.22
Cai's model [11]	102.22	131.53	112.59	60.33	51.54	50.33	84.75
Cheng's model [12]	100.74	125.62	113.04	62.94	51.46	54.25	84.68
Yolca's model [38]		113.38	110.89	58.68	50.07	51.60	76.92
Vovan's model [32]	73.07	85.28	78.74	52.21	77.05	39.78	67.69
The proposed model	40.91	46.68	41.16	30.21	28.13	31.34	36.40

Table 4: The average, variance, and standard deviation of the forecasting RMSEs.

Year	$\overline{\text{RMSE}}$	Var	Std	Year	$\overline{\text{RMSE}}$	Var	Std
1990	76.621	33.297	5.756	1998	49.769	16.049	3.996
1991	31.158	6.799	2.601	1999	43.909	19.021	4.350
1992	22.446	3.670	1.911	2000	51.563	21.070	4.579
1993	47.161	13.080	3.608	2001	46.767	18.514	4.292
1994	43.312	12.490	3.525	2002	28.016	5.713	2.384
1995	27.661	3.266	1.803	2003	35.399	7.632	2.756
1996	27.977	4.420	2.097	2004	31.985	6.578	2.558
1997	59.726	21.111	4.583				

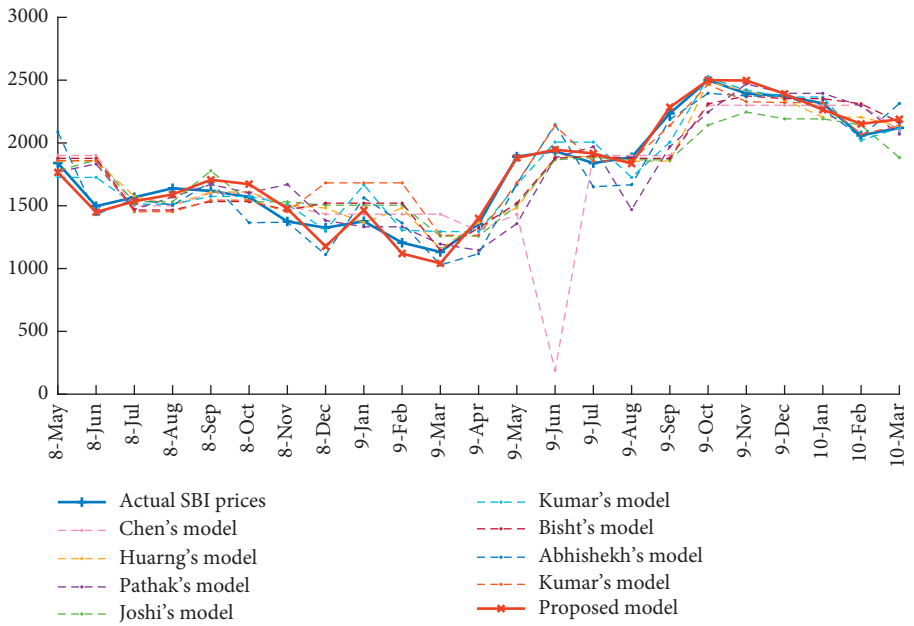

Figure 8: Forecasting results of SBI prices with different models.

Table 5: Comparison of RMSE, AFE, R, and R^2 of different models.

A	RMSE	AFE	R	R^2
Chen's model [13]	187.26	8.26	0.8839	0.7813
Huarng's model [15]	164.04	6.29	0.9110	0.8314
Pathak's model [14]	205.96	8.95	0.8685	0.7544
Kumar's model [16]	134.28	6.30	0.9446	0.8924
Bisht's model [17]	179.03	7.86	0.9001	0.8101
Abhishekh's model [18]	153.77	8.53	0.8442	0.7319
Kumar's model [19]	192.98	7.86	0.9001	0.8101
Proposed model	105.62	5.78	0.8651	0.7542

Table 6: Comparison of RMSEs and average RMSEs with nonfuzzy methods for forecasting the TAIEX from 1999 to 2004.

Models	1999	2000	2001	2002	2003	2004	Average
CFTS model [31]	39.90	60.78	54.71	36.77	24.57	27.61	40.72
SVM-based model [46]	129.89	111.71	69.046	55.255	53.77	53.198	78.81
LSTM-based model [47]	158.67	136.35	101.43	89.379	91.492	69.963	107.88
The proposed method	40.91	46.68	41.16	30.21	28.13	31.34	36.40

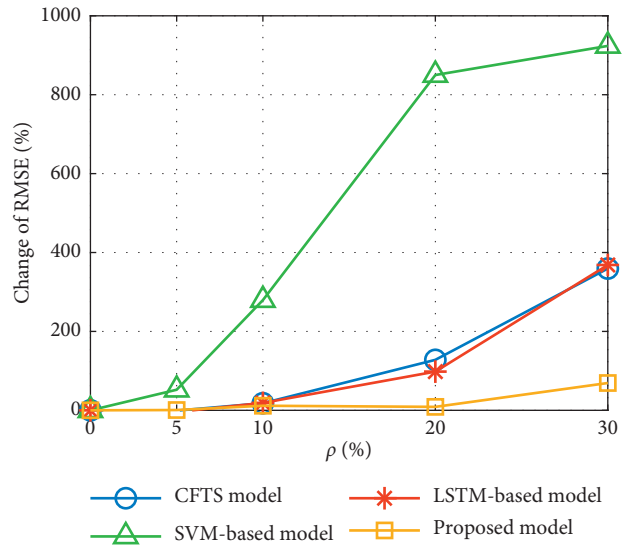

Figure 9: Change of RMSE related to fluctuation ratio ρ of the training set.

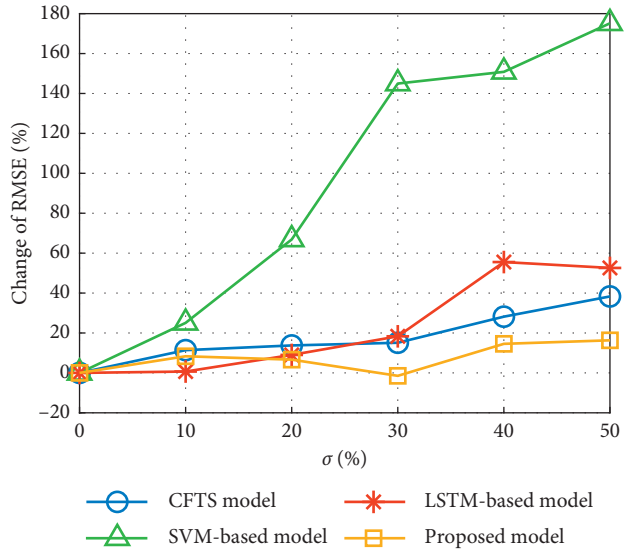

Figure 10: Change of RMSE related to missing ratio σ of the training set.

5. Conclusion

This paper proposes a robust fuzzy time series forecasting model based on multiple linear regression and time series clustering. And, it makes three main contributions.

First, we apply a preprocessing to transform high-order fuzzy time series set into a weighted high-order time series set, which consists of numeric values. Futhermore, we employ SMOTE to generate a high-order time series with the weighted set. After preprocessing, the generated dataset could be processed by multiple linear regression model and ANN.

Second, to build the multiple linear model of the dataset, a novel high-order time series clustering algorithm is employed to cluster data by their linear relationships instead of shapes, which constructs more suitable linear model than other cluster-based models. The proposed clustering algorithm is compared with other three clustering algorithms and built the best suitable linear model which gives the least cost.

At last, we design a forecasting model based on ANN, which calculates the weight of related multiple linear model, as well as its learning algorithm.

The proposed forecasting model is compared with other FTS-based model on the TAIEX, BSE, and it gave a better forecasting accuracy than others. Also, by comparing with forecasting model that are FTS-based, the results suggest that the proposed forecasting model could handle the incomplete, imprecise data, and a better robustness than others.

Appendix

Proof of the convergence of high-order MLRM-based time series clustering algorithm

Proof. Clustering high-order time series is equivalent to minimize the cost function given by formula (4). Since each time series should only belong to one cluster, the cost function could also be presented as follows:

$$J(\mathbf{R}) = \sum_{j=1}^{N} A_j^2, \tag{A.1}$$

where A_j denotes the cost of time series x_j, which equals the distance between x_j and the cluster that x_j belongs to.

For the pth round of iteration, assume $J(\mathbf{R})^{p-1}$ is the cost after the $(p-1)$th iteration, and A_j^{p-1} is the corresponding cost for x_j.

During the process of updating the distance matrix and classification of time series, each time series would be classified into the cluster which leads to the minimum distance defined, which means

$$A_j^{p,*} = \mathrm{Min}\left(D_{i,1}^{p-1}, D_{i,2}^{p-1}, \ldots, D_{i,K}^{p-1}\right) \le A_j^{p-1}, \tag{A.2}$$

where $D_{i,j}^{p-1}$ is the distance between time series x_j and cluster C_i in the $(p-1)$th iteration and $A_j^{p,*}$ is the new cost for x_j after update. Thus,

$$J(\mathbf{R})^{p,*} = \sum_{j=1}^{N} A_j^{p,*2} \leq \sum_{j=1}^{N} A_j^{p-12} = J(\mathbf{R})^{p-1}, \tag{A.3}$$

where $J(\mathbf{R})^{p,*}$ denotes the new cost after update. Hence, the cost decreases after the process of updating the distance matrix and classification of time series.

During the process of update of the set of multiple linear regression models, for each cluster, the linear regression model is aimed to minimizing the cost function:

$$L(\mathbf{C}_i) = \sum_{\overrightarrow{x}_j \in \mathbf{C}_i} D_{i,j}^2. \tag{A.4}$$

Let $D_{i,j}^{p,*}$ be the distance matrix after former process and $D_{i,j}^{p}$ be the distance matrix after linear regression, the cost after the pth iteration could be expressed as follows:

$$J(\mathbf{R})^{p} = \sum_{i=1}^{K} \sum_{\overrightarrow{x}_j \in \mathbf{C}_i} \left(D_{i,j}^{p}\right)^2 \leq \sum_{\overrightarrow{x}_j \in \mathbf{C}_i}^{K} \sum_{i=1} \left(D_{i,j}^{p,*}\right)^2 = J(\mathbf{R})^{p,*}. \tag{A.5}$$

□

Finally, we could conclude that the high-order MLRM-based time series clustering algorithm is convergent since $J(\mathbf{R})^{p} \leq J(\mathbf{R})^{p-1}$ according to formula (A.3) and (A.5).

Data Availability

All data created during this research is openly available from https://finance.yahoo.com/quote/LV30.TW and https://finance.yahoo.com/quote/SBIN.NS.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded by the National Natural Science Foundation of China (grant number 61531013) and the National Science and Technology Major Project of China (grant number 2018ZX03001016).

References

[1] C. S. Chen, Y. D. Jhong, W. Z. Wu, and S. T. Chen, "Fuzzy time series for real-time flood forecasting," *Stochastic Environmental Research and Risk Assessment*, vol. 33, no. 3, pp. 645–656, 2019.

[2] H. Qu, Y. Zhang, J. Zhao, G. Ren, and W. Wang, "A hybrid handover forecasting mechanism based on fuzzy forecasting model in cellular networks," *China Commnunications*, vol. 15, no. 6, pp. 84–97, 2018.

[3] S.-M. Chen and Y.-C. Chang, "Multi-variable fuzzy forecasting based on fuzzy clustering and fuzzy rule interpolation techniques," *Information Sciences*, vol. 180, no. 24, pp. 4772–4783, 2010.

[4] H.-K. Yu, "Weighted fuzzy time series models for taiex forecasting," *Physica A: Statistical Mechanics and Its Applications*, vol. 349, no. 3-4, pp. 609–624, 2005.

[5] S.-M. Chen and C.-D. Chen, "Taiex forecasting based on fuzzy time series and fuzzy variation groups," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 1, pp. 1–12, 2011.

[6] S. M. Chen and S. W. Chen, "Fuzzy forecasting based on two-factors second-order fuzzy-trend logical relationship groups and the probabilities of trends of fuzzy logical relationships," *IEEE Transactions on Cybernetics*, vol. 45, no. 3, pp. 391–403, 2017.

[7] S. M. Chen, H. P. Chu, and T. W. Sheu, "TAIEX forecasting using fuzzy time series and automatically generated weights of multiple factors," *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, vol. 42, no. 6, pp. 1485–1495, 2012.

[8] S. M. Chen, G. M. Manalu, J. S. Pan, and H. C. Liu, "Fuzzy forecasting based on two-factors second-order fuzzy-trend logical relationship groups and particle swarm optimization techniques," *IEEE Transactions on Cybernetics*, vol. 43, no. 43, pp. 1102–1117, 2013.

[9] Q. Cai, D. Zhang, B. Wu, and S. Leung, "A novel stock forecasting model based on fuzzy time series and genetic algorithm," *Procedia Computer Science*, vol. 18, pp. 1155–1162, 2013.

[10] Q. Cai, D. Zhang, W. Zheng, and S. C. H. Leung, "A new fuzzy time series forecasting model combined with ant colony optimization and auto-regression," *Knowledge-Based Systems*, vol. 74, no. 1, pp. 61–68, 2015.

[11] S.-H. Cheng, S.-M. Chen, and W.-S. Jian, "Fuzzy time series forecasting based on fuzzy logical relationships and similarity measures," *Information Sciences*, vol. 327, pp. 272–287, 2016.

[12] S. S. Gangwar and S. Kumar, "Partitions based computational method for high-order fuzzy time series forecasting," *Expert Systems with Applications*, vol. 39, no. 15, pp. 12158–12164, 2012.

[13] S.-M. Chen, "Forecasting enrollments based on fuzzy time series," *Fuzzy Sets and Systems*, vol. 81, no. 3, pp. 311–319, 1996.

[14] H. K. Pathak and P. Singh, "A new bandwidth interval based forecasting method for enrollments using fuzzy time series," *Applied Mathematics*, vol. 2, no. 4, pp. 504–507, 2011.

[15] K. Huarng, "Effective lengths of intervals to improve forecasting in fuzzy time series," *Fuzzy Sets and Systems*, vol. 123, no. 3, pp. 387–394, 2001.

[16] S. Kumar and S. Gangwar, "Intuitionistic fuzzy time series: an approach for handling nondeterminism in time series forecasting," *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 6, pp. 1270–1281, 2015.

[17] K. Bisht and S. Kumar, "Fuzzy time series forecasting method based on hesitant fuzzy sets," *Expert Systems with Applications*, vol. 64, pp. 557–568, 2016.

[18] Abhishekh, S. S. Gautam, and S. R. Singh, "A refined method of forecasting based on high-order intuitionistic fuzzy time series data," *Progress in Artificial Intelligence*, vol. 7, no. 4, pp. 339–350, 2018.

[19] K. K. Gupta and S. Kumar, "Hesitant probabilistic fuzzy set based time series forecasting method," *Granular Computing*, vol. 4, no. 4, pp. 739–758, 2019.

[20] L. A. Zadeh, "Fuzzy sets and systems," in *Advances in Fuzzy Systems—Applications and Theory, Fuzzy Sets, Fuzzy Logic, & Fuzzy Systems*, pp. 35–43, World Scientific Publishing Co. Pte. Ltd., Singapore, 1996.

[21] S. Qiang and B. S. Chissom, "Fuzzy time series and its models," *Fuzzy Sets and Systems*, vol. 54, no. 3, pp. 269–277, 1993.
[22] Q. Song and B. S. Chissom, "Forecasting enrollments with fuzzy time series—part I," *Fuzzy Sets and Systems*, vol. 54, no. 1, pp. 1–9, 1993.
[23] S. Qiang and B. S. Chissom, "Forecasting enrollments with fuzzy time series—part II," *Fuzzy Sets & Systems*, vol. 62, no. 1, pp. 1–8, 1991.
[24] S.-M. Chen, "Forecasting enrollments based on high-order fuzzy time series," *Cybernetics and Systems*, vol. 33, no. 1, pp. 1–16, 2002.
[25] M.-Y. Chen and B.-T. Chen, "A hybrid fuzzy time series model based on granular computing for stock price forecasting," *Information Sciences*, vol. 294, pp. 227–241, 2015.
[26] H. Qu, Y. Zhang, W. Liu, and J. Zhao, "A robust fuzzy time series forecasting method based on multi-partition and outlier detection," *Chinese Journal of Electronics*, vol. 28, no. 5, pp. 899–905, 2019.
[27] K. T. Atanassov, "Intuitionistic fuzzy sets," *Fuzzy Sets and Systems*, vol. 20, no. 1, pp. 87–96, 1986.
[28] V. Torra and Y. Narukawa, "On hesitant fuzzy sets and decision," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, Jeju Island, South Korea, August 2009.
[29] S. S. Gangwar and S. Kumar, "Probabilistic and intuitionistic fuzzy sets-based method for fuzzy time series forecasting," *Cybernetics and Systems*, vol. 45, no. 4, pp. 349–361, 2014.
[30] K. Bisht and S. Kumar, "Hesitant fuzzy set based computational method for financial time series forecasting," *Granular Computing*, vol. 1–15, 2018.
[31] S. Askari, N. Montazerin, and M. H. F. Zarandi, "A clustering based forecasting algorithm for multivariable fuzzy time series using linear combinations of independent variables," *Applied Soft Computing*, vol. 35, pp. 151–160, 2015.
[32] T. Vovan and N. Ledai, "A new fuzzy time series model based on cluster analysis problem," *International Journal of Fuzzy Systems*, vol. 21, no. 3, pp. 852–864, 2019.
[33] X. L. Shao, D. Ma, Y. Liu, and Y. Quan, "Short-term forecast of stock price of multi-branch lstm based on *k*-means," in *Proceedings of the 4th International Conference on Systems and Informatics (ICSAI)*, Hangzhou, China, November 2018.
[34] C. H. Aladag, U. Yolcu, V. R. Uslu, and N. A. Erilli, "Fuzzy time series forecasting method based on Gustafson–kessel fuzzy clustering," *Expert Systems with Applications An International Journal*, vol. 38, no. 8, pp. 10355–10357, 2011.
[35] F. M. Talarposhti, H. J. Sadaei, R. Enayatifar, F. G. Guimarães, M. Mahmud, and T. Eslami, "Stock market forecasting by using a hybrid model of exponential fuzzy time series," *International Journal of Approximate Reasoning*, vol. 70, pp. 79–98, 2016.
[36] N. E. Karoui and M. C. Quenez, "Non-linear pricing theory and backward stochastic differential equations," *Lecture Notesin Mathematics*, vol. 1656, Springer, Berlin, Germany, 1997.
[37] K. Huarng and T. H.-K. Yu, "The application of neural networks to forecast fuzzy time series," *Physica A: Statistical Mechanics and Its Applications*, vol. 363, no. 2, pp. 481–491, 2006.
[38] O. C. Yolca and F. Alpaslan, "Prediction of taiex based on hybrid fuzzy time series model with single optimization process," *Applied Soft Computing*, vol. 66, pp. 18–33, 2018.
[39] S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, V. K. Menon, and K. P. Soman, "Stock price prediction using lstm, rnn and cnn-sliding window model," in *Proceedings of the International Conference on Advances in Computing*, Udupi, India, September 2017.
[40] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: the fuzzy *c*-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2-3, pp. 191–203, 1984.
[41] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321–357, 2002.
[42] T. Warren Liao, "Clustering of time series data-a survey," *Pattern Recognition*, vol. 38, no. 11, pp. 1857–1874, 2005.
[43] C. C. Chuang, C. M. Wang, and C. W. Li, "Weighted linear regression for symbolic interval-values data with outliers," in *Proceedings of the Industrial Electronics & Applications*, Taichung, Taiwan, June 2010.
[44] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.
[45] "TAIEX dataset," 2019, https://finance.yahoo.com/quote/LV30.
[46] J. Wang, R. Hou, C. Wang, and L. Shen, "Improved v -support vector regression model based on variable selection and brain storm optimization for stock price forecasting," *Applied Soft Computing*, vol. 49, pp. 164–178, 2016.
[47] C. Kai, Z. Yi, and F. Dai, "A lstm-based method for stock returns prediction: a case study of China stock market," in *Proceedings of the IEEE International Conference on Big Data*, Santa Clara, CA, USA, October 2015.