energies

Article

Fuzzy First-Order Transition-Rules-Trained Hybrid Forecasting System for Short-Term Wind Speed Forecasts

Yulong Bai *, Lihong Tang, Manhong Fan, Xiaoyan Ma and Yang Yang

College of Physics and Electrical Engineering, Northwest Normal University, Lanzhou 730070, China; tlh953886@sina.com (L.T.); manhongfan@nwnu.edu.cn (M.F.); maxiaoyan_2020@sina.com (X.M.); yangyangshine668@sina.com (Y.Y.)

* Correspondence: baiyulong@nwnu.edu.cn

Received: 11 May 2020; Accepted: 29 June 2020; Published: 30 June 2020

Abstract: Due to the ever-increasing environmental pollution becoming progressively more serious, wind power has been widely used around the world in recent years. However, because of their randomness and intermittence, the accurate prediction of wind speeds is difficult. To address this problem, this article proposes a hybrid system for short-wind-speed prediction. The system combines the autoregressive differential moving average (ARIMA) model with a three-layer feedforward neural network. An ARIMA model was employed to predict linear patterns in series, while a feedforward neural network was used to predict the nonlinear patterns in series. To improve accuracy of the predictions, the neural network models were trained by using two methods: first-order transition rules and fuzzy first-order transition rules. The Levenberg–Marquardt (LM) algorithm was applied to update the weight and deviation of each layer of neural network. The dominance matrix method was employed to calculate the weight of the hybrid system, which was used to establish the linear hybrid system. To evaluate the performance, three statistical indices were used: the mean square error (MSE), the root mean square error (RMSE) and the mean absolute percentage error (MAPE). A set of Lorenz-63 simulated values and two datasets collected from different wind fields in Qilian County, Qinghai Province, China, were utilized as to perform a comparative study. The results show the following: (a) compared with the neural network trained by first-order transition rules, the prediction accuracy of the neural network trained by the fuzzy first-order transition rules was higher; (b) the proposed hybrid system attains superior performance compared with a single model; and (c) the proposed hybrid system balances the forecast accuracy and convergence speed simultaneously during forecasting. Therefore, it was feasible to apply the hybrid model to the prediction of real time-series.

Keywords: wind speed forecasting; auto regressive integrated moving averaged (ARIMA); first-order transition rules (FOTR); fuzzy first-order transition rules (FFOTR); hybrid system

1. Introduction

Wind speed forecasting is an extremely relevant metric for wind resource assessment (WRA) for wind parks in renewable sector. With environmental pollution becoming increasingly more serious, the development of green and environmentally safe renewable energy has become a research hotspot. As one form of green renewable energy, wind energy has been widely used around the world in recent years. The accurate prediction of wind speeds is an important component of effectively utilizing wind energy to generate electricity. However, due to the stochasticity and intermittence of wind speed [1–4], it is difficult to accurately predict the wind speed estimation errors and meet the actual requirements of wind power generation. Therefore, the accurate prediction of wind speed is of great significance.

Wind speed predictions can be divided into short, medium and long term predictions according to the sample time interval [5]. There are also various prediction models: physical models, statistical models and artificial intelligence models and hybrid methods [6]. Each model has its specialty: for example, physical models are suitable for long-term predictions, and statistical methods are applicable to short-term predictions [5]. With the development of artificial intelligence methods in recent years, machine learning methods have also been applied to wind speed prediction. The feedforward neural network is the most commonly used neural network, which due to its fast learning speed and good generalization performance. Various feedforward neural networks have been used in various scientific fields. Authors in [6] proposed training the support vector machine (SVR) and artificial neural network (ANN) models with different training sets for short-term wind speed prediction and compared their prediction results. Authors in [7] proposed the use of feedforward neural networks to predict stock index and sunspot movements. The results show that the method is effective at predicting stock and sunspot activity by improving the training set of the neural network. However, whether this method is suitable for short-term wind speed prediction is a topic for further research. In addition, the main problem with these models was that utilizing a single neural network to model and predict wind-speed data did not provide sufficient results [8–12].

As for hybrid methods, Authors in [13] the idea of combining predictions, which combines different prediction methods in an appropriate way. In recent decades, different models such as linear/nonlinear, supervised/unsupervised and statistical/intelligent have been combined to form various hybrid models, including the following: ARIMA–ANN [8,9] ARIMA–LSSVM [10], EMD–ANN [11], ARIAM–BP [12], ARIMA-Kalman [14], ARIMA–MLP and ARIMA–SVR [15]. Aasim [16] proposed an RWT–ARIMA model based on the continuous wavelet transform, which improved the accuracy of very-short-term wind speed predictions. Wang et.al [5] proposed a nonlinear combination model based on data feature extraction and multi-objective optimization for short-term wind speed predictions. The results show that the model has higher prediction accuracy and stability than the comparison model. The hybrid models proposed in the above literature [17–24] combine the single models through different methods, and the results verify that the combined prediction effect is better than that of the single model.

In recent years, the idea of a fuzzy set has been introduced into time-series prediction algorithms. Through the fuzzy preprocessing of original time-series, fuzzy relation are established and defuzzification is applied to improve the prediction accuracy. A large number of definitions and applications of fuzzy-time series are proposed in [25–31]. Jiang et.al [32] developed a mixed prediction system consisting of a data pretreatment module, an optimization module and a prediction module. The multi-objective differential evolutionary algorithm is used to optimize the fuzzy-time series to balance the prediction accuracy and stability. Although the fuzzy-time-series (FTS) methods has been widely used in other scientific fields, to the best of our knowledge, it is rarely used to predict wind speed. Therefore, we apply the fuzzy set idea to wind-speed data prediction, which is a great contribution to the application of fuzzy-time series to meteorology.

Therefore, in order to solve the above problems to improve the accuracy of time-series prediction, we propose a new hybrid model in this study, which is indicated to enhance the forecast accuracy. It combines the ARIMA model, the three-layer feedforward neural network and fuzzy sets. This hybrid system uses the ARIMA and feedforward neural network to predict the original series and utilizes the dominance matrix to optimize the weight of the single model to maximize the accuracy of the hybrid model prediction. Additionally, we evaluate it with a set of Lorenz-63 and two sets of actual wind-speed data and analyze the results by using three evaluation indices. All of these approaches confirm that the novel hybrid model can be used to predict time series. The innovations of our study are as following:

First, the overwhelming majority of existing studies used ARIMA methods to preprocess the original time-series, and they then used other intelligent models to model and predict the residuals of the original time-series. In this study, the ARIMA is employed to model and predict the original time-series, and then the neural network is employed to process the original time-series, forming a parallel linear-nonlinear hybrid model. Additionally, the neural network model in this study can be used as a preprocessing method for the original time-series and can also be used as a postprocessing method to deal with the errors.

Second, to validate the effectiveness of the framework of the proposed ARIMA–NN–FFOTR model, another hybrid model is proposed based on the framework: the ARIMA–NN–FOTR. Furthermore, this study reports the results of a comparative study on single models and the proposed model, including the following: the ARIMA, the NN–FOTR and the NN–FFOTR. Additionally, the three-layer feedforward neural network is employed, which has many advantages over other neural networks, the most important of which are its extremely short training time and fast convergence speed.

Third, in existing studies, most fuzzy-time series are mainly used for enrollment rate or stock index predictions. In this article, the fuzzy-time-series method is applied to wind speed predictions, which is a great contribution to the application of fuzzy-time series to meteorology. Structure of our proposed hybrid forecasting system is described in Figure 1.

The rest of this study is organized as follows: Section 2 describes the methods of the proposed hybrid system. Section 3 shows the experimental set up and the results of the proposed system. Section 4 presents relevant aspects of the proposed system. Finally, Section 5 contains the concluding remarks and the future work.

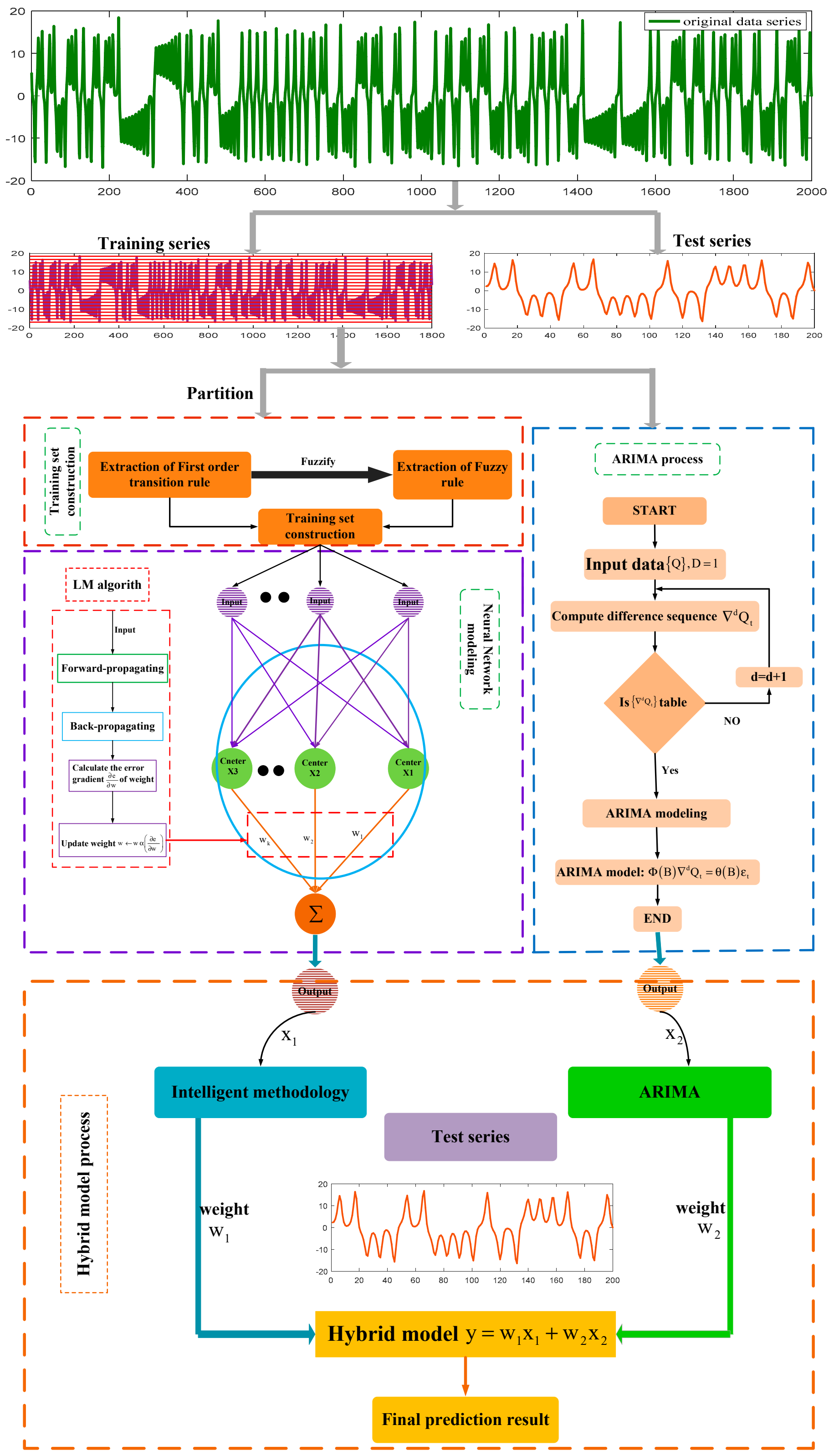


Figure 1. Structure of our proposed hybrid forecasting system.

2. Background Theories

In this section, we summarize some background theories which are relative to our hybrid method in detail.

2.1. Fuzzy Sets

A fuzzy set is different from a regular set, in that the elements of the fuzzy set belong to a certain membership value in the range of the closed interval $[0, 1]$. The definition of the fuzzy set is defined as follows Section 2.1.1.

2.1.1. Definition

Given a set U that is the "universe of discourse", a fuzzy set in the universe U is defined as a set of 2-tuples [7], as show below

$$A = \{ (x, \mu_A(x))x \in U\} \tag{1}$$

where x is an element of universe U and $\mu_A(x) \in [0, 1]$ denotes the membership value of element x in fuzzy set A. If the universe of discourse U is continuous and infinite, it is not possible to define a set of discrete 2-tuples as given in Equation (1) for fuzzy set A. In such cases, a fuzzy membership function $\mu_A : U \rightarrow [0, 1]$ is defined to map each element in the universe of discourse to its corresponding membership value in fuzzy set A.

2.2. Time-Series of Partition

2.2.1. Definition

Given a time series C, let c_{max} and c_{min} represent the maximum and minimum values of the time-series, respectively. We define partitioning as the act of dividing the range into h non-overlapping contiguous Intervals such that the following two conditions jointly hold [7]:

$$P_i \cap P_J = O, \forall i \neq j, i, j \in \{1, 2, \ldots, h\} \tag{2}$$

$$\overset{h}{\underset{i=1}{\cup}} P_i = R \qquad \text{for integer i} \tag{3}$$

Let a partition $P_i = \left[P_i^-, P_i^+\right]$ be defined as a bounded set, where P_i^- and P_i^+ represent the lower and upper bounds of the partition, respectively. Clearly, a time-series data point $c(t)$ belongs to the partition P_i if and only if $P_i^- \leq c(t) \leq P_i^+$.

2.2.2. Definition

Let $c(t)$ and $c(t+1)$ denote two consecutive data points in a time-series $\overline{C}$ and let P_i and P_j be the partitions to which these data points belong. Then, we denote $P_i \rightarrow P_j$ as a first-order [7].

2.3. Basis Function (RBF) Networks

The radial basis function (RBF) network is a three-layer (input, hidden layer and output) feedforward neural network, and its input consists of a source node that connects the network to the outside world. Its hidden layer uses a nonlinear radial basis function as the activation function. The output of the network is typically a linear combination of hidden layer functional values. Let the network output be a vector $\overrightarrow{Y} = \left(y_1, y_2, \ldots y_k\right)$. There are n hidden layer neurons in the network, and the output of the RBF network is as follows:

$$\phi\left(\overline{Y}\right) = \sum_{j=1}^{n} \alpha_j \rho\left(\|\overline{Y} - \overline{C_j}\|\right) \qquad j = 1, 2, \ldots q \tag{4}$$

$\overrightarrow{C_j}$ represents the center vector of the jth hidden layer neuron, $\|\|$ is the Euclidean norm, ϕ is the activation function, and q is the number of hidden layer unit activation functions. There are a variety of hidden layer activation functions and the most commonly used radial basis function is the Gaussian function [33]. It is defined as follows:

$$\rho\left(\|\overline{Y}-\overline{C_j}\|\right)=\exp\left(-\beta\left(\|\overline{Y}-\overline{C_j}\|\right)^2\right) \tag{5}$$

where β is a positive integer. Since the activation function depends on the distance between the input vector and the center vector of the hidden-layer neuron, the function is radially symmetric with the central vector of the neuron. RBF neurons are used as pre-selectors to determine when the neural network triggers based on the pre-trained neural network transition rules. The linear weighted sum is used to convert the weighted sum of the output layers to the output of the neural network and the activity of the ith unit in the output layer can be calculated according to the following formula:

$$Z=w_0+\sum_{p=1}^{q} w_{pr}\phi_q\left(\vec{Y}\right) \qquad r=1,2,\ldots N \tag{6}$$

where w_0 is the bias term, w_{pr} is the connecting weight between the pth hidden unit and the rth output unit, w_{pr} is calculated by using the Levenberg–Marquardt (LM) back-propagation algorithm [34], Y is the response of the qth hidden unit resulting from all input data and N is the number of output units.

2.4. First-Order Transition Rules Based on the Neural Network Model

In this section, the neural networks are trained by using first-order transition rules [7] for a given time series. S_i is a set of all first-order transition rules and $p\left(P_j/P_i\right)$ indicates the probability that the next partition is P_j when the current partition is P_i. Then the formula for $p\left(P_j/P_i\right)\mathbb{F}$ is as follows:

$$p\left(P_j/P_i\right)=\frac{\text{count}\left(P_i\rightarrow P_j\right)}{\sum\limits_{\forall k}\text{count}(P_i\rightarrow P_k)} \tag{7}$$

where count $\left(P_i\rightarrow P_j\right)$ is the sum of the number of occurrences of the transition rule $\left(P_i\rightarrow P_j\right)$. The set S_i can be expressed as follows:

$$S_i=\{P_i\rightarrow P_k\,|\,p\left(P_k/P_i\right)>0\} \tag{8}$$

Given the current partition, in order to predict the next partition, consider the weighted contributions of all transition rules with P_i as an antecedent, which can be achieved by designing a neural network model. This model allows all rules in the set S_i to be triggered at the same time. To implement the model, a set of neural networks is used that satisfy the following conditions.

Condition 1: Given the current input partition P_i, each neural network in the ensemble P_i can trigger at most one transition rule with P_i in the antecedent.

Condition 2: All transition rules with partitions P_i in the antecedent must be triggered by the ensemble. In other words, all rules contained in ensemble S_i must be triggered.

Condition 3: There are no two neural networks in ensemble that can trigger the same transition rules, such as: $P_i\rightarrow P_a$ and $P_i\rightarrow P_b$ simultaneously triggering $P_a\rightarrow P_b$.

The above conditions can be satisfied by constraining the training set of the neural network. See the relevant theorems from these conditions [7].

The steps of grouping the set transition rules into training sets are as follows:

Step 1: Partition. Divide set S into a subset or group of transition rules. All rules in a group have the same partition in the antecedent, and the antecedent contains partition P_i. The number of these subsets is equal to the total number of different partitions that occurred before the transition rules. The choice of the number of partitions will affect prediction accuracy [25], and the number of partitions of the model is set to 20 (multiple trials). The partition diagram of Lorenz-63 is shown in Figure 2.

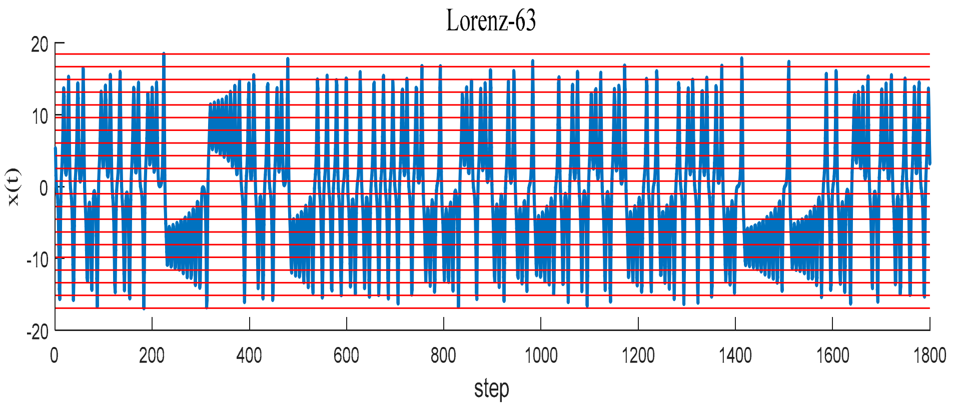


Figure 2. Partition diagram of the Lorenz-63 series.

Step 2: Construct a training set. The transition rules in each subset are ordered. By collecting the transition rules of the position in each subset, the training set of the ith neural network in the set is constructed. By repeating this process, multiple training sets are constructed. These training sets contain transition rules, and the antecedent and consequence of each rule represent a partition label. The regression neural network model is used to obtain the time-series data points that are input as the current time-series data points and output as the future time-series data points. Therefore, the label of each partition is replaced by the corresponding partition median value to get the modified training set $T = \left\{\left(\left(m_{1q}, m_{2q}, \ldots m_{iq}\right), m_{iq}\right\}\middle| q \in \{1, 2, \ldots 20\}\right.$. The Levenberg–Marquardt (LM) back propagation algorithm is used to train the neural network.

Step 3: Neural network training. The first-order transition rules training set constructed in step 2 is used to train the neural network.

Step 4: Predicting the next data point. In the prediction phase, assuming that the partition containing the current time-series data points (system inputs) is P_i, it may happen that the neural network in the ensemble has not been trained according to the rules containing P_i in the antecedent, which will lead to an approximation error. Therefore, the RBF is used as the pre-selector to trigger the appropriate neural network in the given input set. The output ρ_i of the ith hidden layer neuron is as follows:

$$\rho_i = \exp\left(-\beta\left(\left|x^2 - m_i^2\right|\right)\right) \tag{9}$$

where x is the input to the RBF neuron, m_i is the median of the ith partition and $\beta = 1$. If the current partition P_i exists in the antecedent of any training set rules of neural network NN_j, neural network NN_j is enabled. The enable signal of NN_j is obtained by the logical OR operation of the RBF neuron output.

Given the current data point $c(t)$, the final prediction $c'(t+1)$ is calculated using the weighted sum of the single output of the neural network triggered by the pre-selector RBF neurons. Suppose there are υ elected neurons whose outputs $o_1, o_2, \ldots o_\upsilon$ are located in partitions $p_{o_1}, p_{o_2}, \ldots p_{o_\upsilon}$, respectively, and the final output is as follows:

$$c'(t+1) = \sum_{i=1}^{\upsilon}\left(o_i \times \rho\left(p_{o_i}/p_c\right)\right) \tag{10}$$

2.5. Fuzzy First-Order Transition Rules Based on the Neural Network Model

In this section, the first-order transition rules are fuzzified to get the fuzzy first- order transition rules to build the training set and then train the neural network [7]. It should be noted that the best way to represent a time-series partition is by using the center of the partition, but the data points of time series cannot be completely represented by the partition. Therefore, by approximating the data points in the partition and their corresponding midpoints, some approximate errors that increase with the width of the partition itself can be deliberately studied [10]. One way to avoid this error is to treat each partition as a fuzzy set. Obviously, each partition is associated with a fuzzy membership function, and each time-series data point will have some membership values in the set $[0, 1]$. The classical Gaussian function is chosen as the membership function, and its peak corresponds to the center of the partition. Near the boundary of the partition, both sides of the peak have values that gradually decrease and approach 0. The membership value at the partition boundary is not necessarily 0. These membership values can be used to train the neural network in the model. The advantage of this approach is that it takes advantage of the inherent fuzziness involved in assigning partitions to time-series data points and uses partitions to obtain better results. The membership function corresponding to the ith partition is as calculated as follows:

$$u_i(x) = \exp\left(\frac{(x - m_i)^2}{2\sigma_i^2}\right) \tag{11}$$

Then, we train the neural network using the improved fuzzy-first-order-transition rules, as follows.

Step 1: Partition. Set the number of partitions to 40 (multiple trials) to get the best prediction results.

Step 2: Construct the first-order transition rules and training set. The training set T constructed by the fuzzified first-order transition rules is $T = \left\{\left(\mu_1\left(m_{iq}\right), \mu_2\left(m_{iq}\right), \ldots \mu_k\left(m_{iq}\right)\right), m_{jq}\middle| q \in \{1, 2, \ldots 40\}\right.$ where m_{iq} and m_{jq} are the median values corresponding to the partitions P_{iq} and P_{jq}, respectively and $\mu_i(x)$ is the membership value of x in the fuzzy set.

Step 3: Neural network training. The fuzzy-first-order-transition-rules training set constructed in step 2 is used to train the neural network.

Step 4: Predicting the next data point. This is the same as the first-order-transition-rules-based model discussed in Section 2.4.

2.6. Proposed Model

2.6.1. Determination of the Weights of the Proposed Model

The general form of a hybrid model uses the weighted sum of each single prediction model. Therefore, the key point of the hybrid model is to determine the weight coefficients. If the weight coefficients of each single prediction model are properly assigned, the prediction accuracy of the whole hybrid model will be improved accordingly. However, when using the hybrid model, how to determine the weight coefficients of the single models is a big problem. In this regard, many scholars have proposed their own methods for determining the weights. The current commonly used methods are the arithmetic average method, the optimal weight method, the variance reciprocal method, etc. In this study, we use the dominance matrix method to determine the prediction weights of two single models. Let the ARIMA model be model 1 and the neural network model be model 2. By calculating the relative errors of two single-model predictions, we determine the size of the relative error of two single models. For example, when the relative error of the predicted value at a certain point in model 1 is less than the relative error of the predicted value at the same point in model 2. It shows that the prediction effect of model 1 is better than model 2. We judge the relative error of 200 predicted values of two single models by analogy, it can be obtained that the prediction effect of model 1 is better than model 2 m_1 times, the prediction effect of model 2 is better than model 1 m_2 times, and the total number of predicted steps is n. Then, the weights are $w_1 = \frac{m_1}{n}$ and $w_2 = \frac{m_2}{n}$ respectively.

2.6.2. Proposed Model Prediction

After determining the weights, the predicted value of the hybrid model can be obtained by the formula $y = w_1x_1 + w_2x_2$. w_1 is the weight of the ARIMA model, w_2 is the weight of the neural network model, x_1 is the predicted value of the ARIMA model and x_2 is the predicted value of the neural network model. The weights of the new proposed model for the three time-series data sets are shown in Table 1 below.

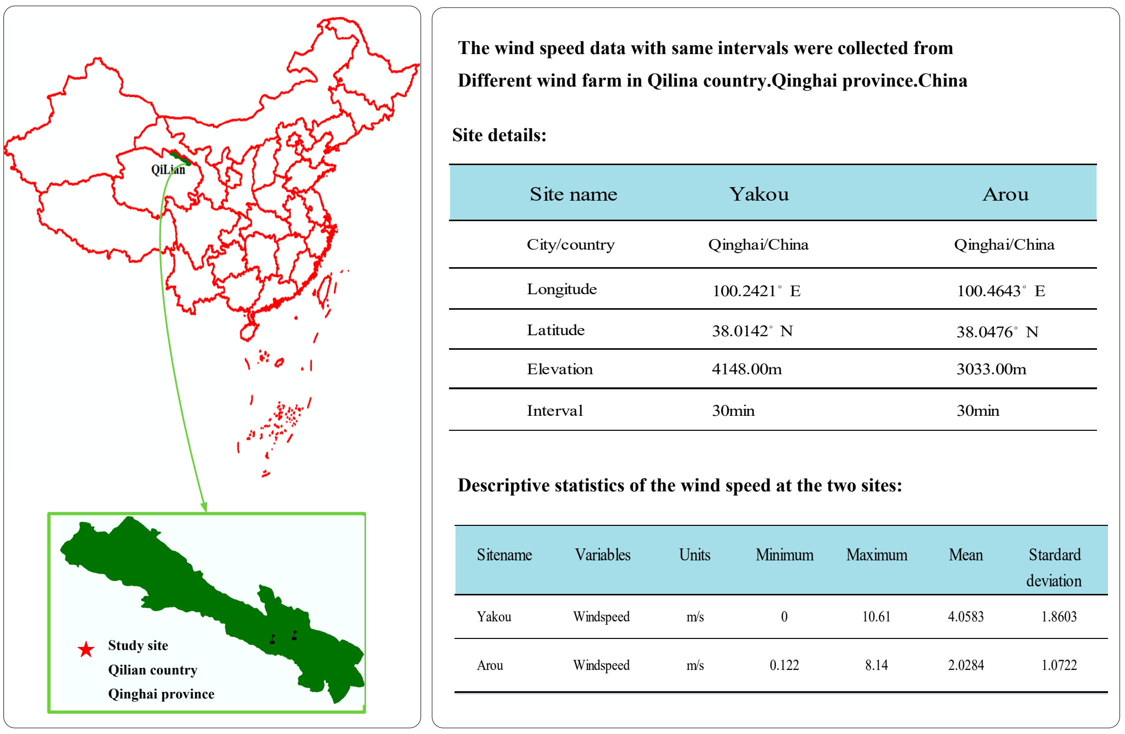
Table 1. Single-model weights.

Time Series	Lorenz-63	Yakou (Wind Speed)	Arou (Wind Speed)
$w_1(\mathrm{ARIMA})$	0.6500	0.4950	0.6900
$w_2(\mathrm{NN-FOTR})$	0.3400	0.5000	0.3500
$w_2(\mathrm{NN-FFOTR})$	0.4300	0.6000	0.6900

3. Experiments Design and Results

3.1. Experimental Data

In this study, the experimental data were the simulated values generated by the Lorenz-63 model and the wind-speed data of two sites: Yakou station and Arou station that were located in the Qilian County, Qinghai Province, The height of the wind speed measurement was 10 m and the sampling interval was 30 min. The time range of wind-speed data collection of Yakou station was: 00:00 1 October 2013 to 15:00 11 November 2013; and the time range of wind-speed data collection of Arou station was: 00:00 1 January 2013 to 15:00 11 February 2013. All three samples have 2000 data points. The first 1800 values were used as the training sample. The training time of three sets of training data under two models was as follows: NN–FOTR (9.41 s) and NN–FFOTR (7.89 s); NN–FOTR (13.14 s) and NN–FFOTR (11.37 s); NN–FOTR (18.60 s) and NN–FFOTR (14.32 s) and the remaining 200 values were used as testing sample. The three sample data sets were used to verify the validity of the new proposed model's prediction. The site details and the statistical information were shown in Figure 3.



The wind speed data with same intervals were collected from Different wind farm in Qilina country.Qinghai province.China

Site details:

Site name	Yakou	Arou
City/country	Qinghai/China	Qinghai/China
Longitude	100.2421° E	100.4643° E
Latitude	38.0142° N	38.0476° N
Elevation	4148.00m	3033.00m
Interval	30min	30min

Descriptive statistics of the wind speed at the two sites:

Sitename	Variables	Units	Minimum	Maximum	Mean	Stardard deviation
Yakou	Windspeed	m/s	0	10.61	4.0583	1.8603
Arou	Windspeed	m/s	0.122	8.14	2.0284	1.0722

Figure 3. Descriptive information about wind-speed data of sites in the Qilian region.

3.2. Evaluation Indices

To evaluate the prediction accuracy, we use three well-known error metrics, which are the mean square error (MSE), the root mean square error (RMSE) and the mean absolute percentage error (MAPE). Let be the time-series value of the test cycle at time t, and $c'(t)$ is the predicted time-series value at the same time. The calculation formulas of the error metrics mentioned above are defined as Equations (12), (13) and (14), respectively, and the results of neural network are shown in Table 2

$$\mathrm{MSE} = \frac{\sum_{t=1}^{N} (c_{test}(t) - c'(t))^2}{N} \tag{12}$$

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{t=1}^{N} (c_{test}(t) - c'(t))^2}{N}} \tag{13}$$

$$\mathrm{MAPE} = \frac{100\%}{N} \sum_{t=1}^{N} \left| \frac{c'(t) - c_{test}(t)}{c_{test}(t)} \right| \tag{14}$$

where N is the total number of test phase time-series data points and $\bar{c}(t)$ is the average of N data points.

Table 2. Performance indexes evaluation of the neural network prediction model under different training rule.

Train Sets	**Lorenz-63**			**Yakou**			**Arou**		
Indexes	RMSE (m/s)	MSE (m/s)	MAPE (%)	RMSE (m/s)	MSE (m/s)	MAPE (%)	RMSE (m/s)	MSE (m/s)	MAPE (%)
NN-FOTR	0.1847	0.0621	9.1243	0.1537	0.0161	18.7885	0.0419	0.0033	13.8783
NN-FFOTR	0.1816	0.5020	8.4147	0.1528	0.0058	17.3454	0.0404	0.0033	13.4124

3.3. Experiment I: Compare the Prediction Results of the Neural Networks Trained by Two Training Sets

The main purpose of Experiment I was to compare the prediction results of the first-order-transition-rules-trained neural network model with those of the fuzzy-first-order-transition-rules-trained neural network model. The neural network models used in this study were all three-layer-feedforward neural networks. For the neural network, the relevant parameters were set as follows Table 3 below. From Figures 4–6, it could be seen that the prediction effect of the neural network model trained by the fuzzy-first-order transition rules was better than that trained by the first-order transition rules. The first-order transition rules replaces the complete data band with the median value of the partition, which resulted in an approximation error. The first-order transition rules were fuzzified and replaced the antecedent of the partition by using the fuzzy membership value. The neural network with the large weight value was triggered according to the size of the membership value, which avoided triggering an unnecessary neural network and improved the speed and prediction accuracy of the network convergence.

Table 3. Experimental parameters setting in the neural network model.

Training Rules	**Experimental Parameters**	**Value**
FOTR	Maximum number of iterations	1000
	Number of node-points in the input layer	20
	Number of node-points in the hidden layer	30
	Number of node-points in the output layer	1
FFOTR	Maximum number of iterations	1000
	Number of node-points in the input layer	40
	Number of node-points in the hidden layer	50
	Number of node-points in the output layer	1

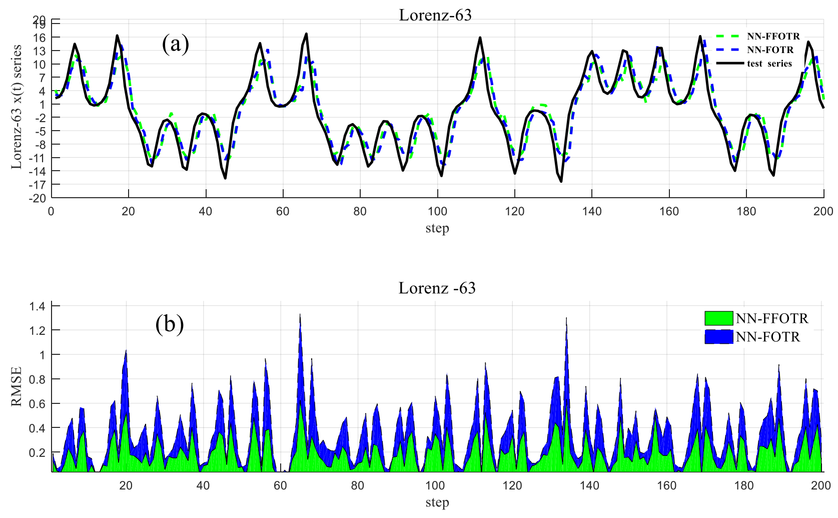


Figure 4. Results for the Lorenz-63. (**a**) Prediction, (**b**) RMSEs.

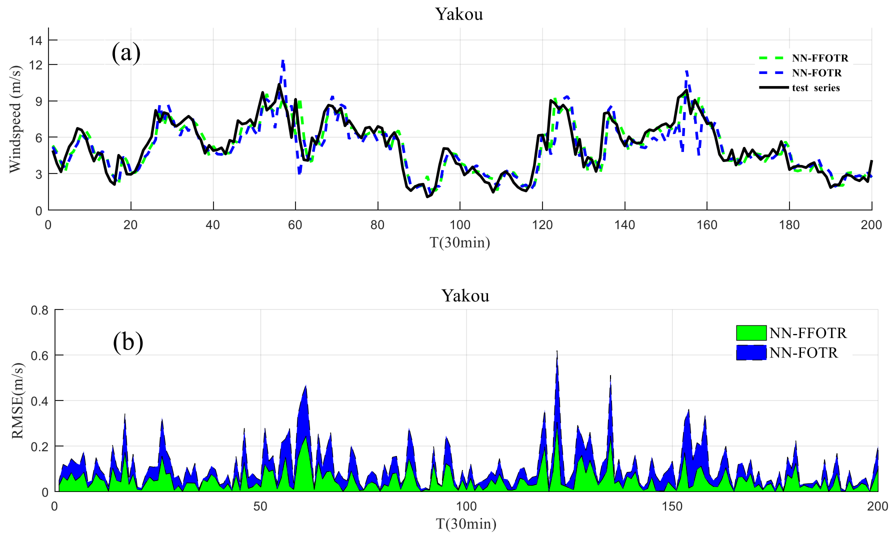


Figure 5. Results for the wind speeds at the Yakou site. (**a**) Prediction, (**b**) RMSEs.

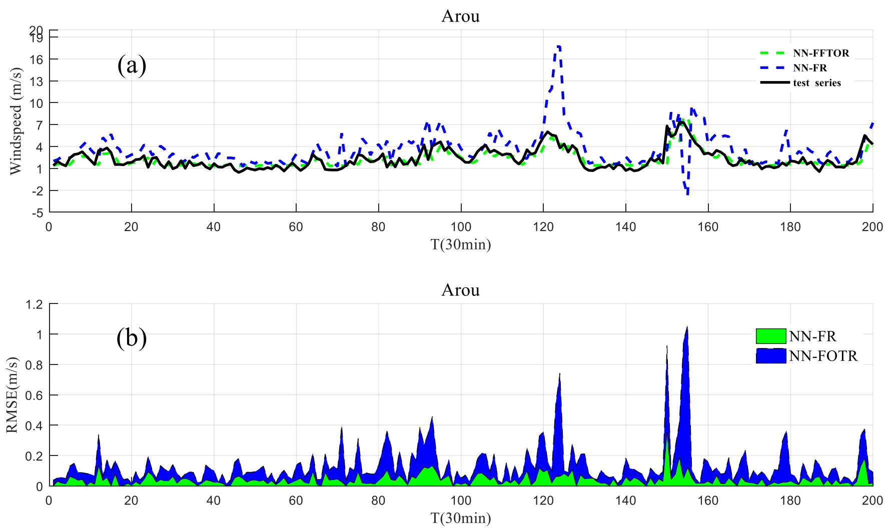


Figure 6. Results for the wind speeds at the Arou site. (**a**) Prediction, (**b**) RMSE).

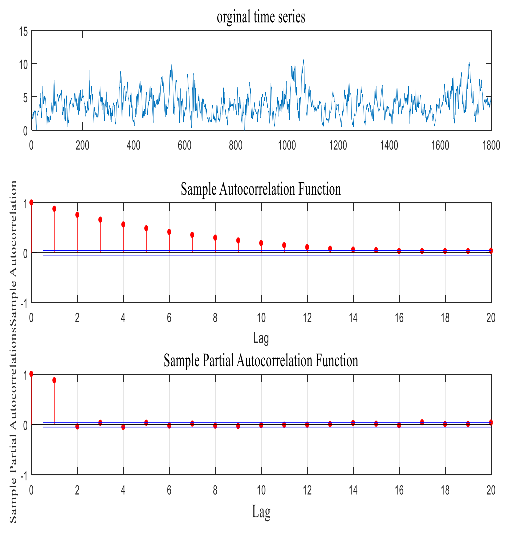
The Lorenz system is a set of three dimensional first-order ordinary differential equations set up by the American meteorologist Lorenz to simulate the atmospheric circulation model. The expression is as follows:

$$\begin{cases} \frac{dx}{dt} = a(-x + y) \\ \frac{dy}{dt} = bx - y - xz \\ \frac{dz}{dt} = xy - cz \end{cases} \tag{15}$$

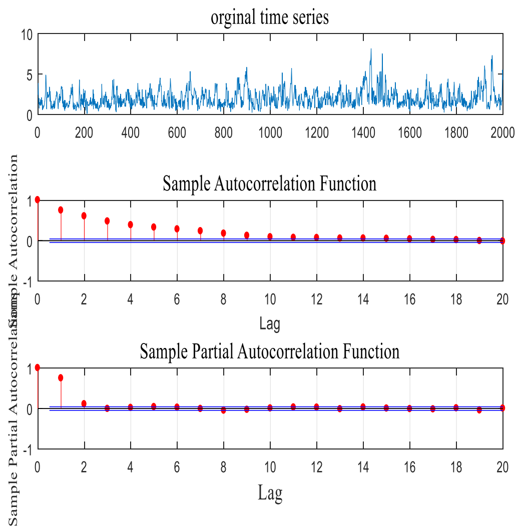
where x, y and z are state variables of Lorenz-63 system. We use a step size of 0.01 and use the Runge–Kutta method to generate 2000 sample points. Then, the neural networks trained using the first-order transition rules and the fuzzy first-order transition rules are, respectively constructed. Moreover, their prediction performances are compared.

3.4. Experiment II: Compare the Prediction Results of Hybrid Models and Single Models

The main purpose of Experiment II was to compare the prediction effects of single models and hybrid models. Three sets of time-series data were modeled by the ARIMA model. As seen from Figure 7, the first 1800 series of the two sets of wind-speed data satisfied the AR or ARMA model. Further, the ACF had a trailing character, and the PACF had a decreasing and oscillating phenomenon. The same was true for the ACF and PACF with the theoretical value series of Lorenz-63. Table 4 showed the determined model order. Figure 8 shows the ACF and PACF of the residual series after two sets of original wind-speed data after modeling. It can be seen that the ACF and PACF of the residual series were both within the 95% confidence interval. Similarly, the ACF and PACF of residual series of Lorenz-63 were the same. Thus, the residual series was white noise, and the ARIMA model was reasonable and could be predicted. According to Figure 9 when modeling and predicting theoretical time-series, the single model ARIMA (RMSE = 0.1893), NN–FOTR (RMSE = 0.1847) and NN–FFOTR (RMSE = 0.1816) had similar prediction accuracies, but the prediction curve of the NN–FFOTR was closer to the true curve than the NN–FOTR. This result showed that the ARIMA model fits the linear patterns of the series well, and the neural network model fits the nonlinear patterns of the series well. Compared with the first-order training set constructed by partition median, the neural network prediction accuracy of the fuzzy first-order training set was higher. However, there were still errors at individual points. This result showed that the fuzzy membership value could describe the weight distribution of each data point in the partition well. Compared with the single model, the prediction curve of the hybrid model was closer to the real curve. The hybrid ARIMA–NN–FFOTR model (RMSE = 0.1370) had the best prediction effect, and the ARIMA–NN–FOTR (RMSE = 0.1101) had the second best. Thus, the hybrid model could better capture the data characteristics of the time series. According to Figure 10, in the actual time-series prediction, such as in the wind-speed data prediction of the Yakou station, the prediction curves of the ARIMA (RMSE = 0.1505) and NN–FOTR (RMSE = 0.1537) fit the real curves well. The prediction curves of the NN–FFOTR (RMSE = 0.1528) were better than both of the other evaluated models, The prediction curves of the ARIMA–NN–FOTR (RMSE = 0.0515) and ARIMA–NN–FFOTR (RMSE = 0.0454) were better than those of the single models in general, and the prediction accuracy of the ARIMA–NN–FFOTR was better than that of the ARIMA–NN–FOTR. Figure 11 show that the prediction effect of the Arou site was the same. The prediction and evaluation indices of the three groups of data were shown in Table 5.



(**a**) the Yakou site

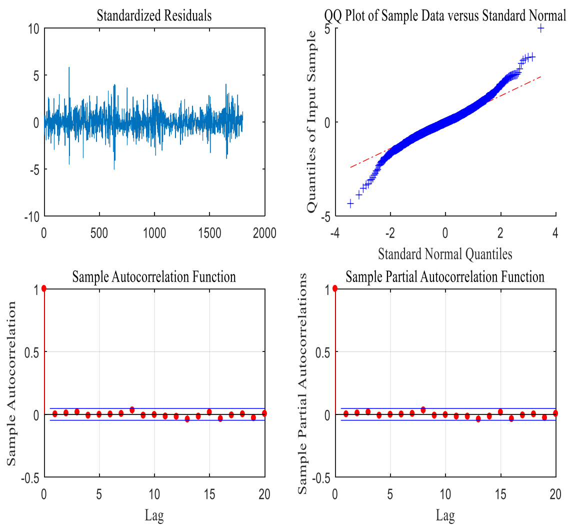


(**b**) the Arou site

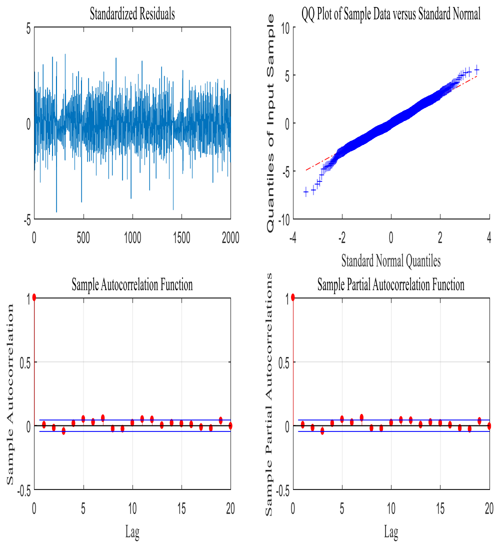
Figure 7. Origin series of wind speed and the ACF and PACF of the series.

Table 4. ARIMA model of three sets of time-series data.

Original Time-Series	ARIMA(P,D,Q)
Lorenz-63	ARIMA(2,0,2)
Yakou (wind speed)	ARIMA(1,0,0)
Arou (wind speed)	ARIMA(3,0,1)



(**a**)



(**b**)

Figure 8. The residual test of wind-speed data after modeling. (**a**) Yakou site; (**b**) Arou site.

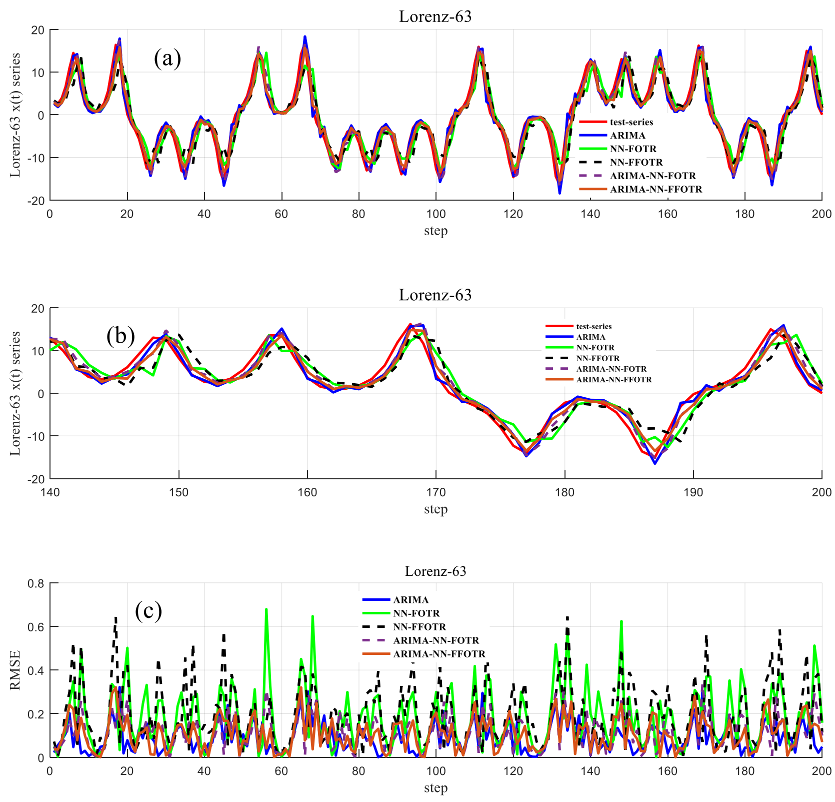


Figure 9. Results of the predictions for the Lorenz-63 series. (**a**) Overall, (**b**) local enlargement, (**c**) RMSEs for the Lorenz-63 series.

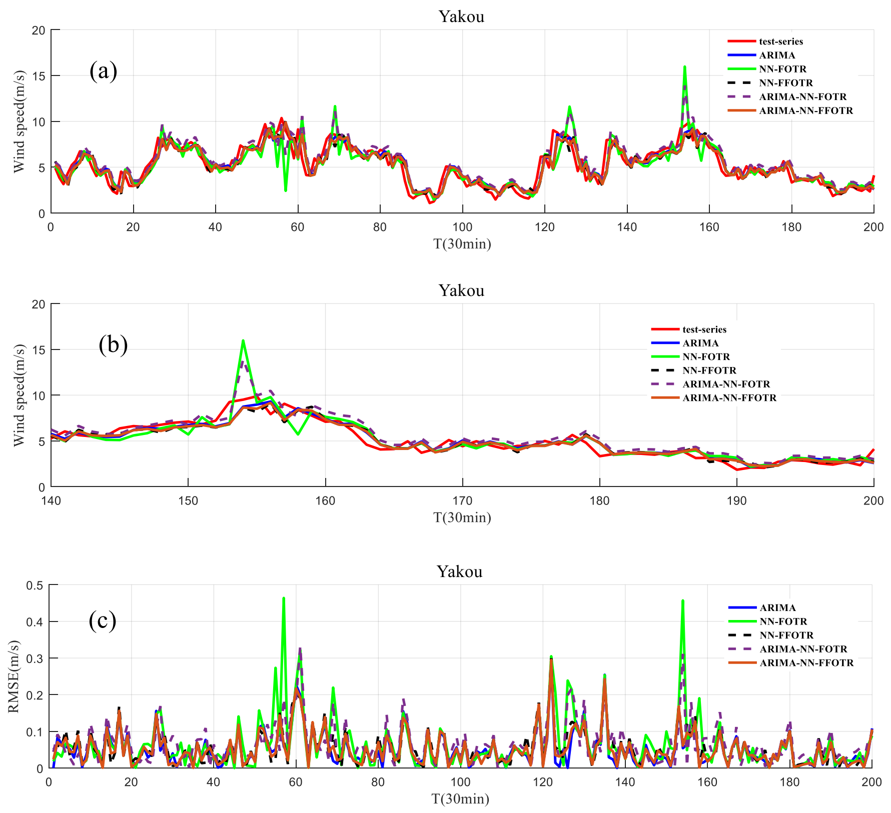


Figure 10. Results of the predictions for the wind speed at the Yakou site. (**a**) Overall, (**b**) local enlargement, (**c**) RMSEs for the wind speed at the Yakou site.

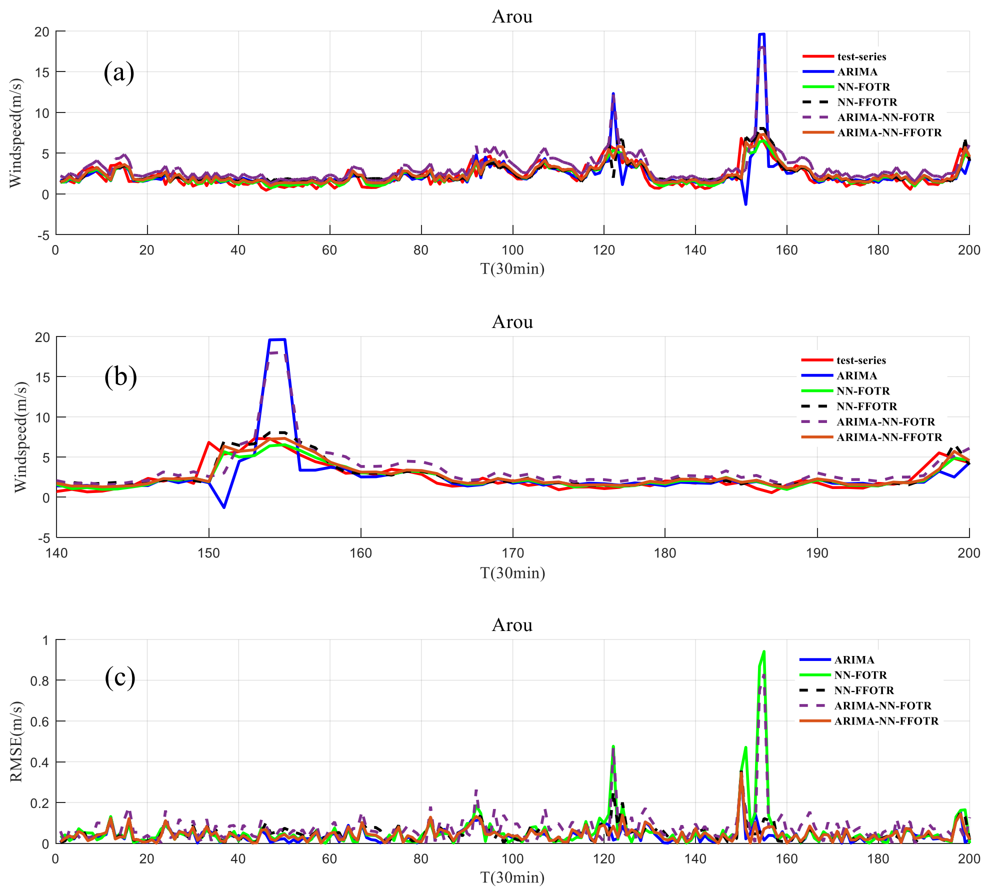


Figure 11. Results of the predictions for the wind speed at the Arou site. (**a**) Overall, (**b**) local enlargement, (**c**) RMSEs for the wind speed at the Arou site.

Table 5. Forecasting results of time-series under different models.

Title	Methods	MSE (m/s)	RMSE (m/s)	MAPE (%)
Lorenz-63	ARIMA	0.0524	0.1893	9.0435
	NN–FOTR	0.0621	0.1847	9.1243
	NN–FFOTR	0.5020	0.1816	8.4147
	ARIMA–NN–FOTR	0.0271	0.1370	8.1312
	ARIMA–NN–FFOTR	**0.0168**	**0.1101**	**7.4512**
Yakou	ARIMA	0.0147	0.1505	16.0866
	NN–FOTR	0.0161	0.1537	18.8775
	NN–FFOTR	0.0058	0.1528	17.3454
	ARIMA–NN–FOTR	0.0048	0.0515	16.0866
	ARIMA–NN–FFOTR	**0.0034**	**0.0454**	**14.0995**
Arou	ARIMA	0.0038	0.0385	13.5569
	NN–FOTR	0.0033	0.0419	13.8783
	NN–FFOTR	0.0033	0.0404	13.4124
	ARIMA–NN–FOTR	0.0032	0.0426	12.6489
	ARIMA–NN–FFOTR	**0.0018**	**0.0253**	**10.5307**

4. Discussion

Aiming at the problems and shortcomings of the existing time-series prediction models, this study proposes a comparative study of two hybrid prediction models based on the ARIMA and a neural network. Each model obtains a better prediction with respect to the Lorenz theoretical value and wind-speed data. Based on the existing work, we can conduct in-depth discussions from the following aspects.

(1) Effectiveness of the new algorithm

This study proposes a new linear-nonlinear parallel combination model. For the theoretical and actual wind-speed data, the combined ARIMA–NN–FFOTR model proposed in this study has higher prediction accuracy than any single model. This study uses a three-layer feedforward neural network with a single input and a single output. The resulting training time is short, and the convergence rate of the whole prediction system is effectively improved. The effectiveness of the new algorithm is verified by comparative experiments. The results show that the hybrid model effectively combines the advantages of a single model to make up for the shortcomings of a single model prediction, and it improves the prediction accuracy and convergence speed of the hybrid model.

(2) Robustness of the fuzzy-time series

This study applies fuzzy-time series to wind-speed data. The original wind-speed data are partitioned and then fuzzified, which will describe the distribution of the membership values of the data points in the partition, and the comparison experiment verifies the robustness of the neural network prediction by using fuzzy set training. Based on the existing research work, the robustness of the fuzzy-time series in real-time-series predictions is further proved.

(3) Applicability in real-time series

In this study, the proposed model is verified by using the Lorenz-63 theoretical value and actual wind-speed data, which shows the validity of this model. The model can be applied to any real-time-series prediction.

5. Conclusions

This work proposes a hybrid system that performs time-series forecasting in three steps: The linear component of time-series modeling using ARIMA model; the nonlinear component of time-series forecasting with feedforward neural network model; and linear combination of the forecasts of ARIMA and feedforward neural network using the dominant matrix theory. To maximize the accuracy, the feedforward neural network is trained by first-order transition rules and fuzzy first-order transition rules. Generating two hybrid systems: ARIMA–NN–FOTR and ARIMA–NN–FFOTR.

Using three evaluation metrics to evaluate experiments, the results show that the ARIMA–NN–FFOTR attains a better performance than single models and ARIMA–NN–FOTR model. The ARIMA–NN–FFOTR reaches a higher accuracy because it is able to forecast separately the linear and nonlinear patterns of the series through ARIMA and NN–FFOTR model; compare with the ARIMA–NN–FOTR model, after the input partition Gaussian of the feedforward neural network is fuzzified, the contribution of the weighted data points in the partition is effectively improved.

In future work, we aim to develop a method to calculate the optimal weight. Additionally, this study considers only short-term wind-speed predictions. Therefore, future studies should evaluate the performance of the developed method for mid-term and long-term wind speed predictions.

Author Contributions: Conceptualization, L.T. and Y.B.; formal analysis, X.M.; methodology, L.T., Y.B.; resources, M.F.; supervision, Y.B.; validation, Y.Y. and L.T.; writing—original draft preparation, L.T.; writing—review and editing, Y.B.; funding acquisition, Y.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the NSFC (National Natural Science Foundation of China) project (grant number: 41861047, 41461078), the Northwest Normal University Scientific research capability upgrading program (NWNU-LKQN-17-06).

Acknowledgments: The authors would like to thank the valuable comments and suggestions of the anonymous reviewers.

Conflicts of Interest: The authors declare no conflicts of interest.

Nomenclature

NN	neural network
RBF	radial basis functions
FOTR	first-order transition rules
FFOTR	fuzzy first-order transition rules
BP	back propagation
MLP	multi-layer perception
SVR	supporting vector machine
MAE	mean absolute error
RMSE	root mean square error
MAPE	mean absolute percentage error
PACF	partial auto correlation function
ACF	auto correlation function
RWT	repeated wavelet transform
ANN	artificial neural network
ARIMA	autoregressive integrated moving averaged
SARIMA	seasonal autoregressive integrated moving averaged
LSSVM	least squares support vector machines
EMD–ANN [11]	empirical mode decomposition-artificial neural network
ARIAM-BP [12]	autoregressive integrated moving averaged-artificial neural network
ARIMA–ANN [8,9]	autoregressive integrated moving averaged-artificial neural network
ARIMA–MLP [15]	autoregressive integrated moving averaged-multi layer perception
ARIMA–SVR [15]	autoregressive integrated moving averaged-supporting vector machine
RWT–ARIMA [16]	repeated wavelet transform-autoregressive integrated moving averaged
ICEEMDAN–ARIMA [5]	improved complementary ensemble empirical mode decomposition with adaptive noise-autoregressive integrated moving averaged
SRIMA–LSSVM [20]	seasonal autoregressive integrated moving averaged-least squares support vector machines
ICEEMDAN	improved complementary ensemble empirical mode decomposition with adaptive noise
ARIMA–NN–FOTR	autoregressive integrated moving averaged-neural network-first-order transition rules
ARIMA–NN–FFTOR	autoregressive integrated moving averaged-neural network fuzzy first-order transition rules.

References

1. Calif, R.; Schmitt, F. Multiscaling and joint multiscaling of the atmospheric wind speed and the aggregate power output from a wind farm. *Nonlinear Process. Geophys.* **2014**, *21*, 379–392. [CrossRef]
2. Calif, R.; Schmitt, F. Modeling of atmospheric wind speed sequence using a lognormal continuous stochastic equation. *J. Wind. Eng. Ind. Aerodyn.* **2012**, *109*, 1–8. [CrossRef]
3. Tiwari, H.; Tiwari, L. Wind energy-present scenario. *Electr. India Mag.* **2010**, *50*, 68–77.
4. Munteanu, I.; Bratcu, A.I.; Cutululis, N.A.; Ceanga, E. *Optimal Control of Wind Energy Systems: Towards a Global Approach*; Springer Science & Business Media: Berlin, Germany, 2008.
5. Wang, L.L.; Li, X.; Bai, Y.L. Short-term wind speed prediction using an extreme learning machine model with error correction. *Energy Convers. Manag.* **2018**, *162*, 239–250. [CrossRef]
6. Yang, J.; Zhao, X.; Wei, H.K.; Zhang, K.J. Sample Selection Based on Active Learning for Short-Term Wind Speed Prediction. *Energies* **2019**, *12*, 337. [CrossRef]
7. Konar, A.; Bhattacharya, D. *Time-Series Prediction and Application: A Machine Iintelligence Approach*; Springer: Berlin/Heidelberg, Germany, 2017.
8. Cadenas, E.; Rivera, W. Wind speed forecasting in three different regions of Mexico, using a hybrid ARIMA–ANN model. *Renew. Energy* **2010**, *35*, 2732–2738. [CrossRef]
9. Khashei, M.; Bijari, M. A novel hybridization of artificial neural networks and ARIMA models for time series forecasting. *Appl. Soft Comput. J.* **2010**, *11*, 2664–2675. [CrossRef]

10. Domingos, S.D.; Santos, J.; João, F.L.; Oliveira, D.; Paulo, S.G.; Mattos, N. An intelligent hybridization of ARIMA with machine learning models for time series forecasting. *Knowl.-Based Syst.* **2019**, *175*, 72–86. [CrossRef]
11. Liu, H.; Chen, C.; Tian, H.Q.; Li, Y.F. A hybrid model for wind speed prediction using empirical mode decomposition and artificial neural networks. *Renew. Energy* **2012**, *48*, 545–556. [CrossRef]
12. Guo, Z.H.; Wu, J.; Lu, H.Y.; Wang, J.Z. A case study on a hybrid wind speed forecasting method using BP neural network. *Knowl.-Based Syst.* **2011**, *24*, 1048–1056. [CrossRef]
13. Noel, D.U. Forecasting: A hybrid approach. *Omega* **1997**, *5*, 463472.
14. Liu, H.; Tian, H.Q.; Li, Y.F. Comparison of two new ARIMA-ANN and ARIMA-Kalman hybrid methods for wind speed prediction. *Appl. Energy* **2012**, *98*, 415–424. [CrossRef]
15. Wang, J.Z.; Wang, S.Q.; Yang, W.D. A novel non-linear combination system for short-term wind speed forecast. *Renew. Energy* **2019**, *143*, 1172–1192. [CrossRef]
16. Singh, S.N.; Mohapatra, A. Repeated wavelet transform based ARIMA model for very short-term wind speed forecasting. *Renew. Energy* **2019**, *136*, 758–768. [CrossRef]
17. Jiang, P.; Wang, Y.; Wang, J. Short-term wind speed forecasting using a hybrid model. *Energy* **2017**, *119*, 561–577. [CrossRef]
18. Mirikitani, D.T.; Nikolaev, N. Recursive Bayesian recurrent neural networks for time-series modeling. *IEEE Trans. Neural Netw.* **2010**, *21*, 262–274. [CrossRef]
19. Rahimi, Z.H.; Khashei, M. A least squares-based parallel hybridization of statistical and intelligent models for time series forecasting. *Comput. Ind. Eng.* **2018**, *118*, 44–53. [CrossRef]
20. Hamzacebi, C. Improving artificial neural networks performance in seasonal time series forecasting. *Inf. Sci.* **2008**, *178*, 4550–4559. [CrossRef]
21. Lee, C.M.; Ko, C.N. Time series prediction using RBF neural networks with an onlinear time-varying evolution PSO algorithm. *Neuro Comput.* **2009**, *73*, 449–460. [CrossRef]
22. Han, H.G.; Chen, Q.L.; Qiao, J.F. An efficient self-organizing RBF neural network for water quality prediction. *Neural Netw.* **2011**, *24*, 717–725. [CrossRef]
23. Gan, M.; Peng, H. Stability analysis of RBF network-based state-dependent autoregressive model for nonlinear time series. *Appl. Soft Comput. J.* **2011**, *12*, 174–181. [CrossRef]
24. Grigonyte, E.; Butkeviciute, E. Short-term wind speed forecasting using ARIMA model. *Energetika* **2016**, *62*, 45–55. [CrossRef]
25. Surendra, S.G.; Abhishekh; Singh, S.R. A New High-Order Approach for Forecasting Fuzzy Time Series Data. *Int. J. Comput. Intell. Appl.* **2018**, *17*, 1850019. [CrossRef]
26. Shanoli, S.P.; Samarjit, K. Fuzzy Time Series Model for Unequal Interval Length Using Genetic Algorithm. *Inf. Technol. Appl. Math.* **2018**, *699*, 205–216. [CrossRef]
27. Lee, M.H.; Sadaei, H.J.; Suhartono. Improving TAIEX forecasting using fuzzy time series with Box–Cox power transformation. *J. Appl. Stat.* **2013**, *40*, 2407–2422. [CrossRef]
28. Chen, M.Y. A high-order fuzzy time series for forecasting model for internet stock trading. *Future Feneration Comput. Syst.* **2014**, *37*, 461–467. [CrossRef]
29. Liu, Z.; Zhang, T. A second-order fuzzy time series model stock price analysis. *J. Appl. Stat.* **2019**, *46*, 2514–2526. [CrossRef]
30. Vovan, T. An improved fuzzy time series forecasting model using variations of data. *Fuzzy Optim. Decis. Mak.* **2019**, *18*, 151–173. [CrossRef]
31. Chen, S.M.; Chung, N.Y. Forecasting enrollments using high order fuzzy time series and genetic algorithms. *Int. J. Intell Syst.* **2006**, *21*, 485–501. [CrossRef]
32. Jiang, P.; Yang, H.F.; Heng, J.N. A hybrid forecasting system based on fuzzy time series and multi-objective optimization for wind speed forecasting. *Appl. Energy* **2019**, *235*, 786–801. [CrossRef]
33. Weik, M.H. Gaussian function. In *Computer Science and Communication Dictionary*; Springer: Berlin/Heidelberg, Germany, 2017.
34. Madsen, K.; Nielsen, H.B.; Tingleff, Q. Method for Non-Linear Least Squares Problems. 2004. Available online: http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3215/pdf/imm3215.pdf (accessed on 29 June 2020).