SSI-DT79185035

Doctoral Thesis

Wind speed and wind power forecasting system based on data decomposition and deep learning neural network

データ分解と深層学習ニューラルネットワークに基づく風速・風力発電出力予測システム

Zhuoyi Liu

March, 2022

Division of Systems Science and Informatics
Graduate School of Information Science and Technology
Hokkaido University

Doctoral Thesis
submitted to Graduate School of Information Science and Technology
Hokkaido University
in partial fulfillment of the requirements for the degree of
Doctor of Engineering

Zhuoyi Liu

Dissertation Committee: Associate Professor Ryoichi Hara
Professor Hiroyuki Kita
Professor Hajime Igarashi
Professor Satoshi Ogasawa

Wind speed and wind power forecasting system based on data decomposition and deep learning neural network

Zhuoyi Liu

Abstract

The overexploitation of fossil fuels, such as coal, oil, and natural gas in recent years has made the development of renewable energy forms an inevitable trend in energy development. Wind energy, as an environment-friendly new energy source, has gradually become the most promising, fastest-growing, and relatively mature renewable energy generation method because of its easy access and low cost. However, because of the volatility of wind speed, wind power generation is always accompanied by uncertainty. Effective wind speed and wind power forecasting are essential for grid dispatch, controllability, and stability, and accuracy is crucial for the effective utilization of wind energy resources.

This study proposes an accurate and efficient wind power forecasting system based on a wind speed forecasting model. First, in the construction of the wind speed forecasting model, a novel wind speed forecasting system is developed based on data decomposition and deep learning neural networks for ultra-short-term wind speed (wind speed at the next moment of 10, 30, and 60 minutes interval) and short-term wind speed (24 h-ahead hourly average wind speed) forecasting. The system consists of three modules: an extraction module, a data pre-processing module, and a forecasting module. In the data extraction module, a considerable amount of valid historical data are extracted, filtered, classified, and used as training data. In the data preprocessing module, the complementary ensemble empirical mode decomposition (CEEMD) is used to decompose the wind speed data. In the forecasting module, an optimized long short-term memory network (LSTM) is employed to forecast the decomposed wind speed data and integrate them into the final forecast results. The results of numerical simulations for 10 locations in Hokkaido indicate that the proposed forecasting system has higher forecasting accuracy and better stability performance than other forecasting models for wind speed at different locations in different periods. Secondly, three forecasting models are proposed to forecast 1-hour ahead wind power based on the structure of wind speed forecasting model, and a hybrid forecasting model based on the three forecasting models with different forecasting accuracy under different conditions is proposed and numerically simulated

with the power generation data provided by the sotavento wind farm in Spain. The numerical simulation results indicate that the forecasting system can obtain good forecasting results in different time periods and that the accuracy is less affected by the environmental conditions, thereby confirming the high generalizability of the forecasting system. The comparison with other forecasting models shows that the proposed system has relatively high accuracy.

Overall, the proposed wind speed and wind power forecasting system exhibits good generality, good stability, and high accuracy and is expected to be used in practical wind power forecasting.

Key words: Wind speed forecasting, Wind power forecasting, Data area division, Complementary ensemble empirical mode decomposition, Long short-term memory, Hybrid forecasting model

Table of Contents

List of Figures

List of Tables

Abbreviations

Abbreviation	English name
AR	Auto regressive
MA	Moving average
ARMA	Auto regressive moving average
ARIMA	Autoregressive integrated moving average model
WD	Wavelet decomposition
WPD	Wavelet packet decomposition
EMD	Empirical mode decomposition
EEMD	Ensemble empirical mode decomposition
CEEMD	Complementary ensemble empirical mode decomposition
CEEMDAN	Complete ensemble empirical mode decomposition with adaptive noise
ANN	Artificial neural network
BPNN	Back propagation neural network
RNN	Recurrent neural network
RBFNN	Radial basis function neural network
ENN	Elman neural network
LSTM	Long short-term memory
VNN	Volterra neural network
ELM	Extreme learning machine
KELM	Kernel extreme learning machine
DAD	Data area division
GA	Genetic algorithm
ACO	ant colony optimization

PE	Permutation entropy
EWT	Empirical wavelet transform
MAE	Mean square error
MAPE	Mean absolute percentage error
RMSE	Root mean square error

Chapter 1. Introduction

1.1 Background and significance of the study

The price of fossil energy continues to rise with the massive consumption of fossil energy and the intensification of the worldwide energy crisis, leading to the increased prominence of environmental problems [1], [2]. The development and utilization of new energy sources have become the key to solving this problem. As a result, countries around the world have invested significant human and material resources in the research of new energy generation. New energy generation refers to the development and utilization of non-conventional energy sources based on new technologies, including wind, solar, tidal, ocean, and biomass, as well as other renewable or clean energy sources for power generation to meet the growing electricity demand. Wind energy has been receiving considerable attention from increasingly more countries in the new energy industry because of its huge commercial value and environmental benefits, making it the fastest growing and most mature form of power generation among new energy sources. Wind energy is generated by the uneven heating of the earth's surface due to the sun's illumination and the temperature difference it produces causes atmospheric convection. It is estimated that the exploitable storage capacity of wind energy on the earth can reach 2×10^7MW, and one percent of the wind energy storage capacity can be fully utilized to meet the current energy demand [1]–[3].

Wind turbines use power generation devices to convert the natural wind into electricity and transmit it to the grid for users. Because wind is formed by the asymmetric flow of air, its inherent randomness and volatility lead to high volatility in the output power of wind farms when grid-connected [4]. When the number of wind turbines running on the grid is small, the volatility of their output wind power has a lesser effect on the safe and stable operation of the power grid, whereas when the scale of the wind turbines running on the grid is large, their output wind power has greater volatility and intermittency, which will bring greater risk in terms of safety, reliability, and economy of the power system operation [2], [4]. Electrical energy cannot be stored in large quantities because of the simultaneous nature of its production, transmission, and use. Grid dispatchers need to ensure the balance of supply and demand of electric energy in real-time to ensure the reliability and safety of the power system operation. When wind turbines are not integrated into the power system, the grid regulator can control the power generation through load forecasting methods to develop power plant generation plans to achieve a real-time balance of power supply and demand. When a large number of wind turbines are integrated into the

power system, controlling the power generated can be difficult because of the volatility of wind power. However, if the wind speed or power can be forecasted accurately, it can reduce the effects of wind power volatility on the power system, making it conducive for the grid scheduler to wind farms to remove the adverse effect of timely adjustment, achieve a real-time balance of power system, and avoid large-scale power outages [3]. Therefore, technologies that can accurately forecast wind speed and wind power are essential for power systems.

1.2 Current status of research

1.2.1 Development status of wind power generation

Awareness of the need for environmental protection has been increasing gradually with the intensification of environmental pollution and the massive consumption of fossil energy, and the urgent need to find a new clean energy source that can replace traditional fossil energy should be addressed [3],[4]. Wind and solar power are green energy sources that are more mature and widely used and have a greater value for commercial development. Wind power generation has the advantages of small unit capacity, low investment cost, and wide distribution, and its development scale is expanding worldwide. Meanwhile, thermal power generation and hydroelectric power generation are affected by topographic conditions and the environment, and the investment cost is relatively high, which limits their large-scale development and application. According to a report, by the end of 2020, the total global wind power generation accounted for 6.38% of the total electricity demand, and the highest percentage of wind power generation is in Denmark, which exceeds 48% of the total power generation capacity [5].(see Table 1.1 and Fig. 1.1)

Table 1.1 Share of wind power in total electricity demand in selected countries and regions.

Country / Region	Share of wind power in total electricity demand(%)	Cumulative installed capacity of wind power(GW)
Global	6.38	743
Europe(EU 27+UK)	16.00	220
Denmark	48.00	6.2
Germany	27.00	63
United Kingdom	24.20	24
Spain	21.90	27
Sweden	16.00	10
Netherlands	11.70	6.6
Australia	9.90	7.3

Brazil	9.80	18
United States	8.40	122
China	6.20	281

Fig.1.1 Share of wind power in total electricity demand in selected countries and regions.

Compared with other energy sources, wind energy has outstanding advantages and limitations. The advantages of using wind power include large reserves, no pollution, renewable, and wide distribution. The limitations of wind energy are the uneven distribution of wind energy and the uncontrollability of wind power generation because of the intermittent and unstable nature of wind energy. The rational use of wind energy can not only reduce environmental pollution but also alleviate the increasing pressure of energy shortage that exists globally [6].

According to statistics [5], in 2020 the globally installed wind power capacity was 93 GW, representing an increase of 14.3% compared to 2019, at which time the total global installed capacity was up to 743 GW. Fig. 1.2 and Table 1.2 shows the top 10 countries in terms of global cumulative installed wind power capacity in 2020, from which it can be seen that China, the United States, and Germany ranked in the top 3 and have the highest share of wind power at 37.8%, 16.4%, and 8.5%, respectively. Between 2011 and 2020, the global installed capacity of wind turbines continued to expand significantly [7] (see Fig. 1.3). Particularly, the cumulative installed capacity of offshore wind power increased as an exponential function [8] (see Fig. 1.4). The region with the fastest development of wind power generation is the Asia Pacific, with 50.7% of the new installed capacity worldwide in 2020, followed by Europe (25.5%), and North America (16.1%). China, the U.S., and the U.K. are the top three regions with 18.8%, 13.6%, and 10.5% of global new additions, respectively. At present, one-third of the world has started to use wind power for electricity

generation, and in the process of using wind power for electricity generation, countries that developed wind power earlier, such as the United States, Germany, Denmark, and the Netherlands, have invested considerable human and material resources to develop a series of more complete development and application standards [3]. Subsequently, as the scale of wind power development increased, more and more countries used wind power as an important means to replace the traditional mode of power generation.

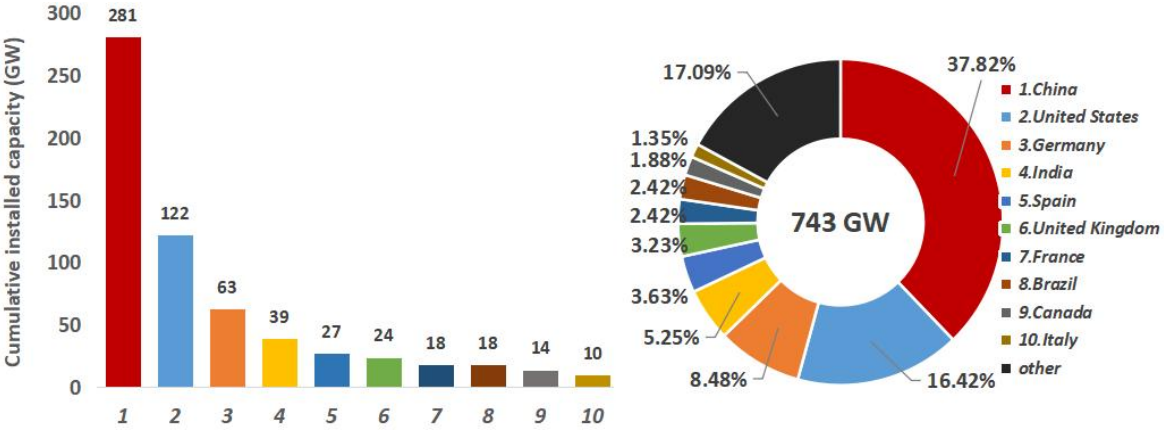


Fig. 1.2 Top 10 countries in the world in terms of cumulative installed wind power capacity in 2020.

Table 1.2 Top 10 countries in the world in terms of cumulative installed wind power capacity in 2020.

Ranking	Country / Region	Cumulative installed capacity of wind power(GW)	Global share of installed wind power capacity(%)
1	China	281	37.82
2	United States	122	16.42
3	Germany	63	8.48
4	India	39	5.25
5	Spain	27	3.63
6	United Kingdom	24	3.23
7	France	18	2.42
8	Brazil	18	2.42
9	Canada	14	1.88
10	Italy	10	1.35

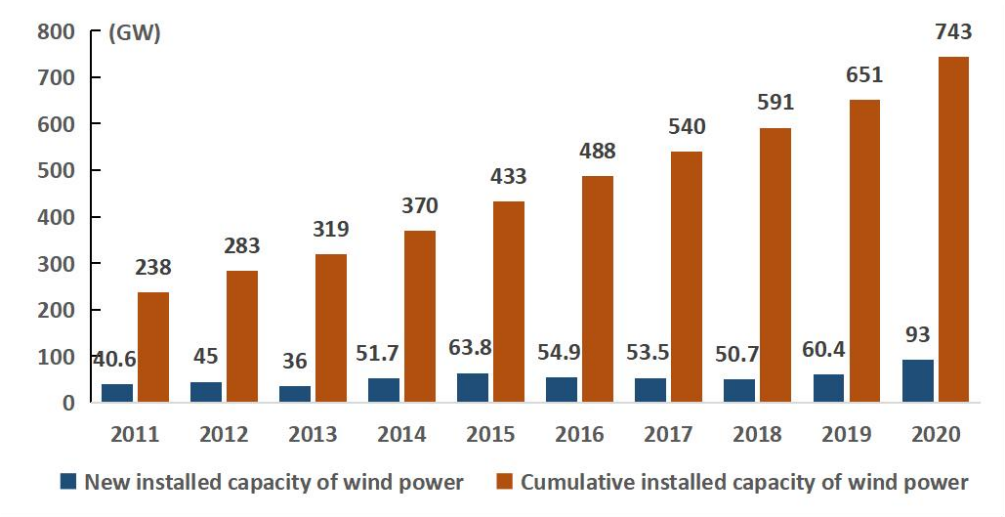


Fig. 1.3 Change in global cumulative installed wind power capacity from 2011 to 2020.

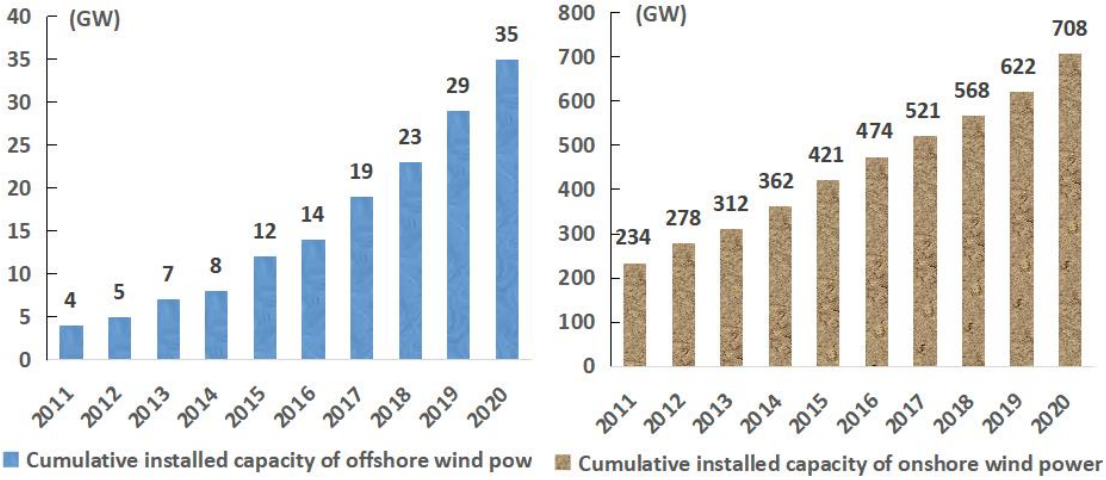


Fig. 1.4 Change in global cumulative installed capacity of onshore and offshore wind power from 2011 to 2020.

However, with the rapid development of wind power generation, the negative effects of large-scale wind power grid connection on the stable operations of the power system have also come to the fore [9]. Due to the weak grid structure of the power grid compared with the rapidly developing wind power scale, a large number of wind turbines connected to the grid are prone to voltage fluctuations and flicker resulting in wind curtailment, which limits the further development of wind power. Therefore, an effective solution to the phenomenon of wind curtailment is a prerequisite for the vigorous development of wind power generation, and accurate wind speed and wind power forecasting provide an effective way to solve the above problems.

1.2.2 Current status of research on wind speed and wind power forecasting

1.2.2.1 Classification according to forecasting methods

At present, scholars from various countries have conducted considerable research on wind speed and wind power forecasting of wind farms. Wind power forecasting methods can be divided into two categories from the general direction: the first method is to build a wind power forecasting model based on the measured wind power data. The advantage of this method is that only the wind farm power needs to be analyzed, and no other data need to be considered. The disadvantage of this method is that it requires a high level of wind power data consistency and a certain standard to exclude too large or too small values and ensure that the data changes within a reasonable range. The second method is to forecast the wind speed of the wind farm, construct a wind speed-power statistical model, and determine the wind power based on the wind speed forecasting results. The accuracy of the second wind power forecasting method depends on wind speed forecasting, which can be broadly classified into physical, statistical, and artificial intelligence methods [10]–[12] (see Fig. 1.5).

The physical model uses meteorological factors, such as the location information of the wind farm, temperature, wind direction, humidity, and barometric pressure for wind speed forecasting. The advantage of this method is that it starts from the meteorological factors and the physical characteristics of wind speed, and requires only a small amount of data to limit the accuracy of wind speed forecasting to a small area, making it more suitable for complex terrains. The disadvantage of the physical model is that it requires access to comprehensive meteorological information of the wind farm, which is relatively difficult to obtain, and its complex calculation makes it difficult to promote [13]–[16].

Statistical methods based on historical data using mathematical models for wind speed and wind power forecasting include the time series method, Kalman filter method, wavelet analysis method, artificial neural network method, etc. The basic principle of the time series method is to establish a time-series model of wind speed based on the functional relationship between measured data, random errors, and forecast data [11]. Currently, the main time series methods are the auto regressive (AR), moving average (MA), and auto regressive moving average (ARMA) models. The advantages of the time series method include the simple structure of the model, the ability to correct local trends in real-time according to data fluctuations, and the ability to calculate confidence intervals for the values to be forecast. The basic principle of the wavelet analysis method is to first decompose the wind speed time series into several smooth time series for forecasting to

ensure accuracy. Each series can use different forecasting methods, and the forecast value of the wind speed can be obtained according to the superposition principle. This method requires less data, has a better forecasting effect, can reduce the non-smoothness of the wind speed series, and is more suitable for time series with drastic wind speed changes [10], [22] . The commonly used wavelet analysis methods include wavelet decomposition (WD), wavelet packet decomposition (WPD), empirical mode decomposition (EMD) [17], etc.

In recent years, artificial neural networks (ANNs) have been used extensively as a forecasting method in the field of wind speed or wind power forecasting. ANNs have a good ability to process information in parallel, strong nonlinear fitting ability, and the ability to learn and improve autonomously that other forecasting methods do not have. The most commonly used among these models are feedforward neural networks [18], recurrent neural networks (RNNs) [15], [16], and radial basis function neural networks (RBFNNs). Seventeen ANNs show good forecasting power when the time series has nonlinear characteristics [20]. However, ANNs also have some problems that cannot be ignored, such as the tendency to fall into local optima and produce overfitting [19]. In response, many researchers have introduced hybrid forecasting models to solve this problem. For example, Wang Deyun et al. proposed a forecasting model based on WNN optimized by genetic algorithm (GA) for ultra-ultra-short-term wind speed forecasting. Numerical simulations have also indicated that the model has good forecasting accuracy [21]. Shukur et al. proposed a combination of Kalman and BPNN forecasting models according to Kalman's principle and achieved good results in very ultra-ultra-short-term wind speed forecasting. Another class of hybrid forecasting models combines data preprocessing methods and neural networks, such as using decomposition methods, including wavelet decomposition, wavelet packet decomposition [22], and empirical mode decomposition (EMD) to decompose the original wind speed or wind power time series into several subseries with relatively simple variations. forecasting is conducted separately to reduce the complexity of individual neural networks, avoid falling into local optima and generate overfitting phenomenon, thus having the effect of improving forecasting accuracy [23]–[25]. In addition, literature [26] proposed a wind power forecasting model based on the Genetic algorithm-Volterra neural network (GA-VNN) that uses multi-step forecasting for ultra-ultra-ultra-short-term wind power forecasting of a wind farm. The simulation results verified the accuracy of the model. In [27], the kernel function was introduced into the extreme learning machine (ELM), and the kernel extreme learning machine (KELM) algorithm, which optimized the parameters of the ELM network and improved the accuracy of the forecasting model, was proposed. In [28] a single wind power forecasting model was weighted according to the maximum entropy principle and a comprehensive wind power model that could improve the forecasting accuracy of wind power compared with the traditional single forecasting model was built.

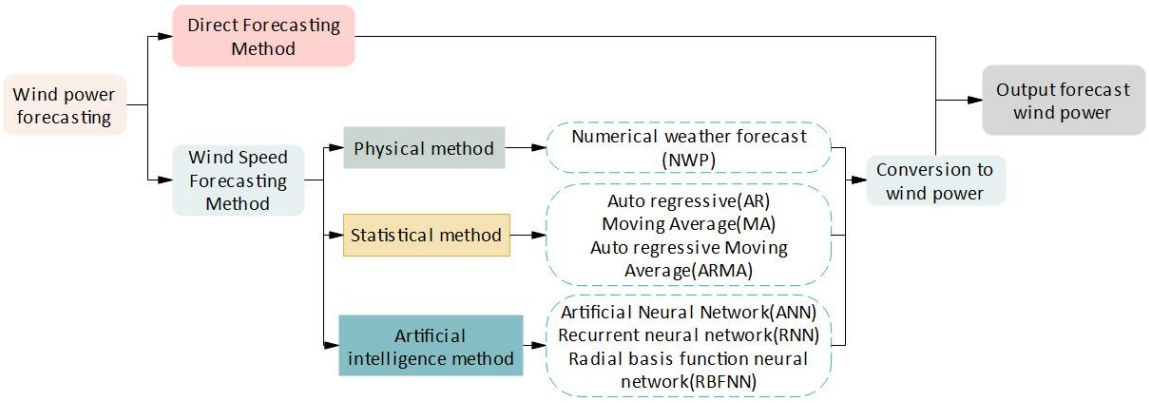


Fig. 1.5 Classification of various methods of wind power forecasting.

1.2.2.2 Classification according to the time scale of the forecast

From the above introduction, it can be seen that wind speed or wind power forecasting can be divided into ultra-short-term forecasting, short-term forecasting, mid-term forecasting and long-term forecasting according to the time scale of forecasting, in addition to the forecasting method [10]–[12]. Ultra-short-term forecasting refers to forecasts from a few minutes to about 1 hour in advance; short-term forecasting refers to forecasts from 1 hour to about 72 hours in advance; medium-term forecasting refers to forecasts from a few days or weeks or months in advance; and long-term forecasting takes years as a forecasting unit.

Medium- and long-term forecasts are used for wind resource assessment for wind farm planning and construction , or provide data support for the rational arrangement of wind turbine overhaul, commissioning, and maintenance [12]. Ultra-short-term forecasts are used for controlling wind turbines, assessing power quality, and adjusting daytime power generation plans [12]. Short-term forecasts are used for rational dispatching of power grids, maintaining, maintaining power quality, and power market trading [10].

Medium- and long-term wind speed and wind power forecasts are always difficult. Thus far, no efficient and highly accurate forecasting methods have been proposed due to the large period and too many unknown variables to be considered. The forecasting techniques for ultra-short-term and short-term wind speed and wind power forecasting have been relatively mature. A large number of researchers have proposed various forecasting methods based on physical forecasting, statistical forecasting, and forecasting models based on artificial intelligence neural networks, but there is still a large upside in terms of forecasting accuracy and stability of forecasting models [29], [30].

1.3 Research purpose and content

1.3.1 Research purpose

The ultimate goal of this study is to propose a wind power forecasting system with high accuracy and good forecasting stability, which can realize the wind speed or wind power data as input data only and output the forecast wind power data in a short period in the future.

From the above introduction, it can be seen that the wind power forecasting methods can be divided into two major categories: direct forecasting method and wind speed forecasting-based forecasting method. The advantages and disadvantages of both categories are obvious, that is, the direct forecasting method only needs to consider the wind power data, eliminating the intermediate conversion process, which can avoid the errors brought by the conversion calculation to a certain extent. However, the requirements for the neatness of the wind power data are rather demanding, making it difficult to apply this method to the special situation of large changes in wind power. In contrast, the forecasting method based on wind speed forecasting is relatively flexible and can cope with a variety of situations. The wind speed data, which are easier to collect than wind power data, can be used in the forecasting process, and the amount of training data can be expanded by collecting wind speed data from surrounding locations to improve forecasting accuracy. However, this method relies too much on the accuracy of wind speed forecasting, and when the wind speed forecasting error is large, the final error will be greatly expanded by the conversion of wind speed and wind power [30].

Based on the above analysis, this study is divided into two steps. Step 1: Develop a wind speed forecasting model to achieve high accuracy and high efficiency in forecasting ultra-short-term, short-term wind speeds. Step 2: Based on the wind speed forecasting model in the first step, a wind power forecasting model based on wind speed forecasting is developed. At the same time, a direct forecasting model for wind power is developed based on the forecasting tools used in the first step. Then, compare the forecasting accuracy of the two types of forecasting models in different situations, integrate the two types of forecasting models, and propose a hybrid forecasting model to achieve the effect of choosing the best forecasting means corresponding to different situations (see Fig. 1.6).

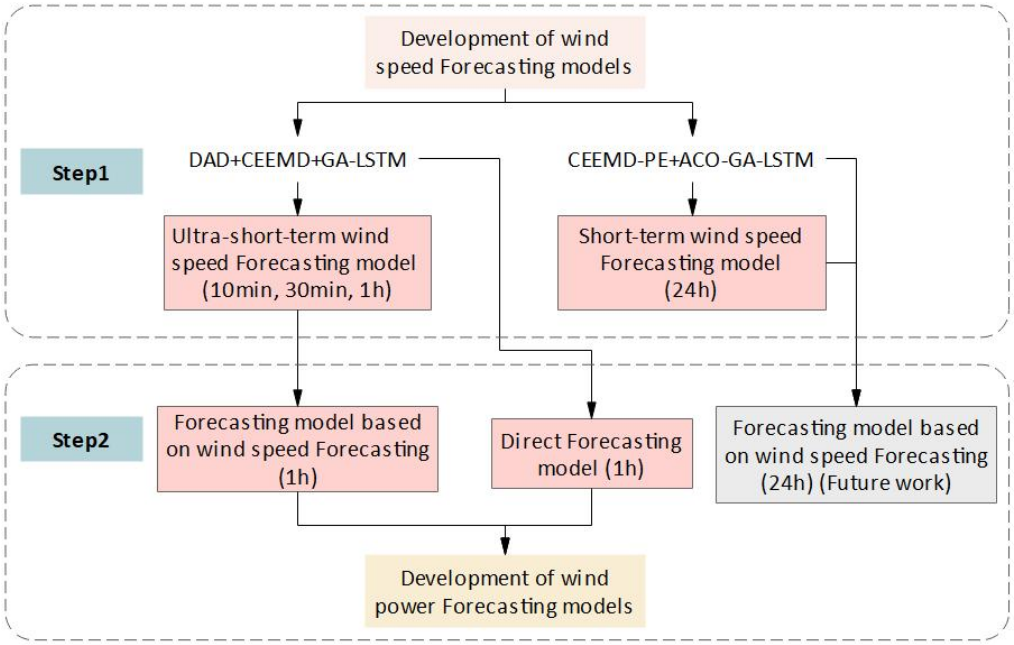


Fig. 1.6 The implementation steps of the forecasting method proposed in this study.

1.3.2 Contributions of this research

The main contributions of this paper are summarized as follows:

(1) The data area division (DAD) method is proposed for ultra-short-term wind speed forecasting. This method can regionally divide wind speed data, and a large amount of valid wind speed history data can be extracted from the area where the forecast target location is located during forecasting, thereby providing a reliable database for wind speed forecasting.

(2) The data pre-processing strategy of decomposing the training data using CEEMD is proposed. The decomposed subseries of a wind speed or wind power are selectively filtered and reconstructed using the permutation entropy (PE) method by judging the forecast time scale and the complexity of the neural network.

(3) For ultra-short-term wind speed forecasting, GA is proposed to optimize the initial parameters of the LSTM network, and for short-term wind speed forecasting, a hybrid method of ant colony optimization (ACO) and GA, ACO-GA, is proposed to improve the optimization efficiency of the initial parameters of the LSTM network.

(4) Based on the proposed wind speed forecasting model, a wind power forecasting model based on wind speed, a direct wind power forecasting model based on data division of LSTM, and a direct wind power forecasting model based on LSTM and CEEMD are proposed.

(5) For wind speed forecasting, ultra-short-term wind speeds (wind speed at the next moment of 10, 30, and 60 minutes interval) at 10 locations in Hokkaido and short-term wind speeds (24 h-ahead hourly average wind speed) at three locations in Hokkaido were used as forecasting targets, respectively. The results verify the effectiveness of the proposed wind speed forecasting model for ultra-short-term and short-term wind speed forecasting.

(6) For wind power forecasting, the ultra-short-term wind power (wind power at the next moment of 60 minutes interval) of an experimental wind farm named Sotavento in Spain is used as the forecasting target. The results validate that the proposed hybrid wind power forecasting model has better forecasting accuracy and stability compared with the single forecasting model.

1.3.3 Organization of chapters

This dissertation consists of Six chapters, and the contents are summarized as follows:

Chapter 1 introduces the background of the study and then briefly describes the current status of wind power generation today and the current status of research on wind speed and wind power forecasting.

Chapter 2 details the methods, models, neural networks, and optimization algorithms used in the various stages of forecasting model building, which form the basis of the forecasting model proposed in this thesis.

Chapters 3 and 4 presents the method for constructing the wind speed forecasting model. The wind speed forecasting models are classified into ultra-short-term forecasting models and short-term forecasting models according to the forecasting time scales. The problems faced by the two forecasting models in the three processes of data selection, data decomposition, and optimization of neural networks are discussed and dealt with by corresponding means to propose a adaptable wind speed forecasting model.

Chapter 5 presents the construction method of the wind power forecasting model. Three wind power forecasting models are proposed based on the proposed wind speed forecasting model and the method used to construct this model. The numerical calculation results show that the three forecasting models have their advantages and disadvantages in different situations. To effectively utilize the points of the three forecasting models, a hybrid forecasting model based on the three forecasting models is proposed, and the effectiveness of the method is verified by numerical simulation.

Chapter 6 is the conclusion of this thesis.

Chapter 2. Construction of forecasting model

This chapter presents the methods, models, neural networks, and optimization algorithms used at each stage of the forecasting model building. Section 2.1 presents the data preprocessing methods and K-means clustering algorithm used in the wind speed data extraction stage. Section 2.2 presents the CEEMD method used in the data decomposition stage and the PE algorithm used to further optimize the decomposition results. Section 2.3 presents the LSTM network, the core forecasting tool used in the proposed forecasting model. Section 2.4 introduces the GA and ACO algorithms used in the initial parameter optimization of the LSTM network. Section 2.5 presents the evaluation criteria for the forecasting performance used in this paper.

2.1 Wind speed data extraction

2.1.1 Data pre-processing

Large deviations in the measured and actual data can be observed in extreme weather conditions or equipment failure, and it easily results in large deviations between the measured and actual data of wind speed. Hence, if the actual data of wind farms are used directly for analysis, it will cause large errors in the simulation results and affect the accuracy of the forecasting model results. Hence, to avoid the effects of inaccurate data, reference [31] used the average interpolation method to supplement the missing data.

Different attribute characteristics have different correlations with wind speed, and the degree of correlation may vary significantly. If wind speed or wind power data are used directly with wind speed data for wind speed or wind power forecasting, the different orders of magnitude of the data may cause a phenomenon that the data with higher correlation with the quantity to be forecast contribute less to the forecasting results. Thus, to prevent this situation, the normalization method is used to process the wind speed and wind power data [32], [33], as shown in equation (2-1).

$$x_i = \frac{x_i^{'} - min\{x_j^{'}\}}{max\{x_j^{'}\} - min\{x_j^{'}\}}, j = 1,2,\ldots,n \qquad (2-1)$$

where $x_i^{'}$ denotes the initial wind speed attribute data of the wind farm, and x_i denotes the wind speed feature data after the normalization process.

The wind speed feature data are mapped to the interval [0,1] according to Equation (2-1), which constitutes the initial feature set. Similarly, after obtaining the forecast wind speed values, the forecast wind speed values need to be inversely normalized according to Equation (2–2).

$$y_i^{'} = y_i \times \left(max\{x_j\} - min\{x_j\}\right) + min\{x_j\}, j = 1,2,\ldots,N \qquad (2-2)$$

where y_i is the forecast data at the ith moment, $y_i^{'}$ is the forecast data after inverse normalization at the ith moment, and N is the number of forecast samples.

2.1.2 K-means clustering algorithm

The purpose of the clustering algorithm is to divide the huge data samples into several clusters according to certain criteria, and the data samples vary widely between different clusters. The samples within the same cluster have a high degree of similarity. Due to a large number of wind speed attribute samples and similar climate conditions wind speed has certain similarities and regularity because of the large number of wind speed attribute samples and climate conditions. The wind speed forecasting accuracy can be improved by clustering the training samples after data preprocessing and then constructing wind speed forecasting models for different clusters separately.

The K-means clustering algorithm was first proposed by MacQueenJB in 1967. The basic idea of this algorithm is to divide samples into different groups according to certain criteria based on the inherent regularity and characteristics of data samples. Since the K-means clustering algorithm has the advantages of simple principle, easy programming, and fast computation, it is widely used as a data analysis tool in scientific research and industrial applications [34].

The basic idea of the K-means clustering algorithm is as follows:

Step1: The number of clusters k is determined, and the Euclidean distance of each sample to each initial cluster center is calculated;

Step2: The samples are grouped based on the calculated Euclidean distance, and the standard measure function value and average value of each group are calculated based on the result of the grouping;

Step3: the standard measure function is judged to be If it converges, the run ends. If it does not converge, the mean value of each sample group is recalculated, and the initial clustering center is updated.

Based on the above steps, the cycle is continued until the standard measure function converges (see Fig. 2.1).

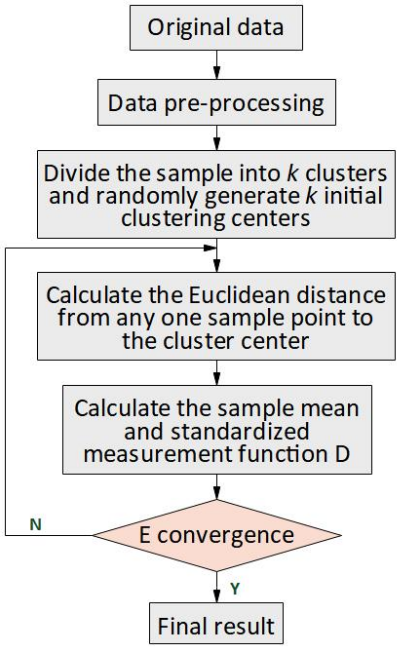


Fig. 2.1 Flowchart of K-means.

The specific steps are shown below.

(1) Data preprocessing.

(2) Divide the samples into k groups; the initial clustering center and each group of samples take the form shown in Equation (2-3).

$$s_o = (s_{o,1}, \cdots, s_{o,i}, \cdots, s_{o,l}) \qquad (2-3)$$

where o=1,2,...,k; l is the time series.

(3) Calculate the Euclidean distance from each sample point to each initial clustering center.

$$d_{(s_i,s_o)} = \sqrt{\sum_{j=1}^{n} (s_{(i,j)} - s_{(o,j)})^2} \quad (2-4)$$

Following the calculation results, the sample points are divided into groups according to the principle of minimum Euclidean distance, thus completing the initial division of the sample points.

(4) Based on the division results, the mean value of each sample group is calculated, and the sample convergence criterion (standard measure function) is determined as follows.

$$\overline{s_o} = \left(\frac{1}{N_o} \sum_{s_i \in G_o} s_{(i,1)}, \cdots, \frac{1}{N_o} \sum_{s_i \in G_o} s_{(i,j)}, \cdots, \frac{1}{N_o} \sum_{s_i \in G_o} s_{(i,l)} \right) \quad (2-5)$$

$$D = \sum_{o=1}^{k} \sum_{s_i \in G_o} |s_i - s_o|^2 \quad (2-6)$$

where s_o is the sample mean, N_o is the total number of samples in group o; G_o is the set of samples in group o; D is the standard measure function characterizing the degree of convergence of the samples.

(5) If the standard measure function does not converge, the calculated sample mean needs to be used as the new initial clustering center, and steps (3) to (5) are calculated cyclically until the algorithm converges. The specific process is shown in Fig 2.1.

2.2 Data decomposition

In the training process of the forecasting model, providing sufficient training data can ensure that the training is carried out properly and the generality of the forecasting model is improved. However, because of the noise and random fluctuations in the original wind speed series, the directly used raw data will lead to poor training results. [35] In recent years, many scholars have used signal decomposition algorithms for ultra-ultra-short-term wind speed forecasting in wind farms, proposing a class of forecasting methods based on decomposition combinations [36]. The main reason for doing so is that wind speed signals

have complex nonlinear, non-smooth, and other characteristics, and decomposing them into relatively simple subseries of frequency by signal decomposition algorithms is beneficial to the construction of forecasting models, thereby improving the accuracy of point forecasting. The commonly used decomposition combination forecasting method consists of three steps: first, the original wind speed time series is decomposed into multiple subseries by some signal decomposition algorithm, then the model is constructed for each subseries separately and the forecast values of the subseries are generated. Finally, the forecast values of each subseries are summed to obtain the forecast values of the original series (see Fig. 2.2).

The selection of the decomposition algorithm is crucial in this process. It depends on whether the wind speed features can be extracted effectively and directly affects the accuracy of wind speed forecasting. In recent years, the decomposition methods often used for wind speed forecasting include WD [37], EMD [23]–[25], etc.

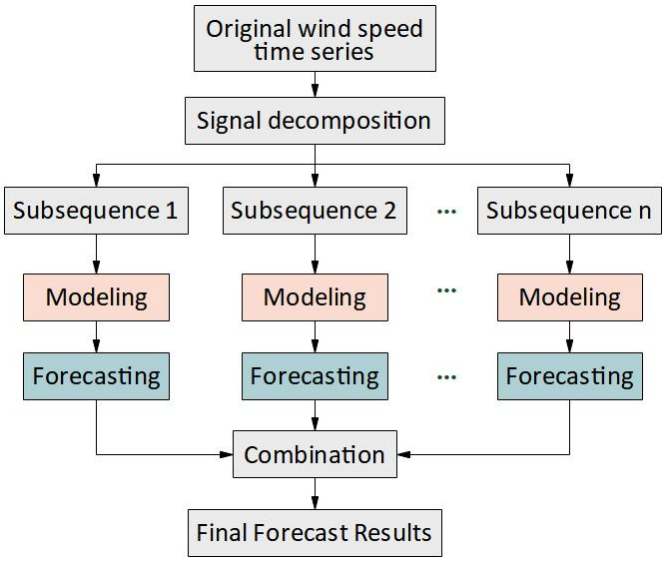


Fig. 2.2 Flowchart of decomposition forecasting method.

2.2.1 Empirical mode decomposition (EMD)

EMD [23]–[25] refers to the non-linear and non-smooth signal analysis method developed by Huang et al. in 1998. Unlike wavelet decomposition, EMD does not require a fixed basis function and has no special requirements on the type of signal. The basic idea of EMD is to decompose a complex signal into a finite and small set of sub-signals with a limited number of oscillatory modes according to the local properties of the signal's time scale. Each oscillatory mode is represented by an intrinsic mode function (IMF). This decomposition idea is based on the following assumptions.

(1) The data have at least two extremes, a maximal and a minimal value.

(2) The local time-domain properties of the data are uniquely determined by the time scale between the extreme value points.

(3) If the data has no extreme value points but inflection points, the decomposition can be obtained by differentiating the data one or more times to obtain the extreme values and then by integration.

As the core of the EMD algorithm, the IMF needs to satisfy two conditions [25]:

- The number of extremal points and the number of over-zero points must either be equal or differ by at most 1 over the entire data set.
- The mean value of the upper and lower envelopes (defined by local maxima and minima) must be zero at any point.

The flowchart of the EMD algorithm is given in Fig 2.4. Let $y(t)$ denote a given signal and the specific calculation steps are as follows

(1) Identify all local extreme points (maxima and minima) of $y(t)$, and connect all maxima and minima by interpolation to obtain the upper envelope $y_{up}(t)$ and $y_{low}(t)$, respectively.

(2) Calculate the average envelope $m(t)$

$$m(t) = \frac{y_{up}(t) + y_{low}(t)}{2} \tag{2-7}$$

Subtracting the average envelope $m(t)$ from the original sequence $y(t)$ yields the detailed components of the sequence.

$$d(t) = y(t) - m(t) \tag{2-8}$$

(3) Check whether $d(t)$ satisfies the condition of IMF. If $d(t)$ is an IMF, set $c(t) = d(t)$, and use the residual $r(t)$ instead of $y(t)$.

$$r(t) = y(t) - c(t) \tag{2-9}$$

Otherwise, use $d(t)$ instead of $y(t)$ and repeat the above two steps until the following termination condition is satisfied.

$$\sum_{t=1}^{l} \frac{\left[d_{j-1}(t) - d_j(t)\right]^2}{\left[d_{j-1}(t)\right]^2} \leq \delta \; j = 1,2,\ldots; \; t = 1,2,\ldots,l \quad (2-10)$$

where l is the length of the signal, j denotes the number of iterations computed, and δ is the stopping threshold, which usually takes a value between 0.2 and 0.3. This process is also known as screening.

(4) The three steps are repeated until all IMFs and residuals are obtained. Eventually, after decomposition, the original sequence $y(t)$ can be expressed as

$$y(t) = \sum_{i=1}^{n} c_i(t) + r_n(t) \quad (2-11)$$

where $c_i(t)(i=1, 2, ...,n)$ denotes the different IMFs and $r_n(t)$ denotes the final residuals.

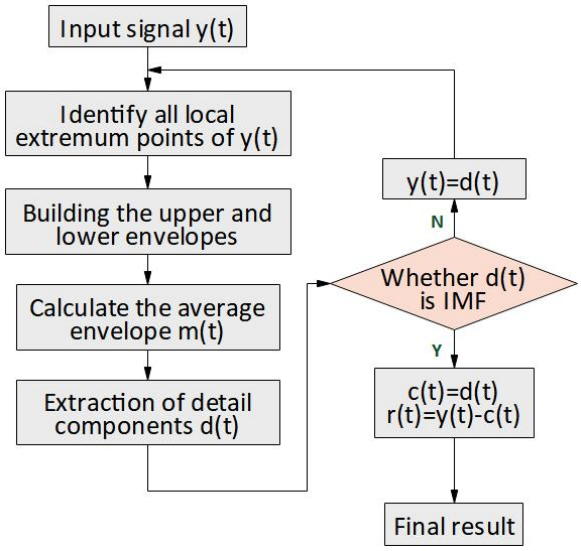


Fig. 2.3 Flowchart of EMD method.

However, EMD has an obvious drawback, namely mode mixing [35], which tends to produce a single IMF consisting of signals of significantly different scales, or signals of the same scale appearing in different IMF components. To solve this problem, in 2008, Wu and Huang proposed ensemble empirical mode decomposition (EEMD) [38], which effectively avoided mode mixing by adding white noise to the original signal. However, but it brings

the problems of inefficient decomposition and noise residual at the same time.

2.2.2 Complementary ensemble empirical mode decomposition (CEEMD)

An improved version of EEMD, namely CEEMD, was used to process the wind speed data to decompose the raw wind speed data more efficiently and accurately [36].

The CEEMD algorithm is described in Fig. 2.4, and *NE* and *Nstd* will be discussed in detail in Sections 3.2 and 4.2.

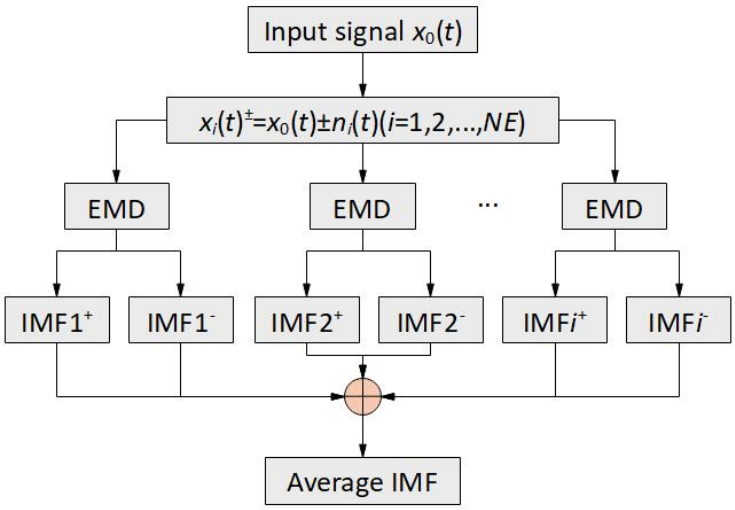


Fig. 2.4 CEEMD algorithm flowchart.

(1) In the original wind speed signal $x_0(t)$, two groups of white noise sequence $\pm n(t)$ with opposite signs are added to obtain $x_i(t)\pm$ (i is the sequence number of the added white noise).

(2) The signal $x_i(t)\pm$ with the added white noise is decomposed into IMFi by using EMD.

(3) Steps 1 and 2 are repeated using different white noises, and the corresponding IMF$i\pm$ components are obtained. The number of repeated procedures is called the ensemble number (*NE*).

(4) The average value of IMFi± components is taken as the final result or IMF.

(5) In EMD, the combination of all IMFs corresponds to the original data. However, in CEEMD, the combined result is no longer the original data because white noise has been added.

With the addition of white noise, the original wind speed signal becomes more stable,

and continuous signals can be decomposed easily, thereby avoiding the problem of mode mixing. The reason is that each time a pair of white noise ±n(t) with opposite sign is added, the effect of some white noise on the original signal can be eliminated when the IMFi± is superposed, thus restoring the characteristics of the original wind speed signal.

2.2.3 Permutation entropy (PE)

Permutation entropy (PE) is a new method proposed by Bandt et al. [39] for detecting sudden changes in the signal dynamics and randomness of time series. This algorithm has several advantages, including its simple calculation, good anti-noise ability, strong time sensitivity, and suitability for non-linear systems [40] ,[41]. The basic principle of this algorithm is discussed as follows:

The length is N time series $\{X_i, i = 1,2,\dots,N\}$, and a phase space reconstruction is carried out to obtain the following:

$$\begin{cases} X_1 = \{x_1, x_{1+t}, \cdots, x_{1+(m-1)t}\} \\ \vdots \\ X_k = \{x_i, x_{i+t}, \cdots, x_{i+(m-1)t}\} \\ \vdots \\ X_{N-(m-1)} = \{x_{N-(m-1)t}, x_{N-(m-2)t}, \cdots, x_N\} \end{cases} \quad (2-12)$$

where m is the embedding dimension, and t is the time delay.

The m vectors $X_k = \{x_i, x_{i+t}, \cdots, x_{i+(m-1)t}\}$ of the sequence are arranged in an ascending order as follows:

$$X_i = \{x_{i+(j_1-1)t} \le x_{i+(j_2-1)t} \le \cdots \le x_{i+(j_m-1)t}\} \quad (2-13)$$

If $x_{i+(j_{i1}-1)t} = x_{i+(j_{i2}-1)t}$, then the vectors are sorted according to the value of j, that is, when $j_{k1} < j_{k2}$, we have $x_{i+(j_{i1}-1)t} \le x_{i+(j_{i2}-1)t}$. For any vector of a time series, the following symbol sequences can be obtained:

$$S(I) = (j_1, j_2, \cdots, j_m) \quad (2-14)$$

According to the form of entropy, the permutation entropy of k sequences of time

series $X(i)$ can be defined as

$$H_p(m) = -\sum_{i=1}^{k} P_i \ln P_i \qquad (2-15)$$

The permutation entropy $H_p(m)$ is normalized, and the value is $0 \leq H_p \leq 1$. A larger H_p corresponds to a signal with higher complexity, whereas a lower H_p corresponds to a more stable signal.

2.3 Forecasting model

In the forecasting module, an improved version of the deep learning neural network, RNN, LSTM network, is used. Because LSTM has memory capability, it has better performance in dealing with the forecasting problem of time series compared with other deep learning neural networks. To improve the training efficiency of the network and enhance the forecasting accuracy, we also used GA and ACO to optimize the initial parameters of the LSTM network.

2.3.1 Recurrent neural network (RNN)

A recurrent neural network (RNN) is a new type of deep learning neural network, which can solve some time series-related problems that cannot be handled by traditional neural networks. In the traditional feedforward neural network, the signal flows in only one direction, first from the input layer into the network model, then from the input layer to the hidden layer, and finally from the output layer. During the signal flow, all layers are fully connected, but the nodes in the same layer have no connection. RNNs are different from feedforward neural networks in that they make connections between the neurons in the recurrent layers, giving the network the ability to remember previous information. RNNs can effectively analyze time series and use the memory function inside the network to process data input at any moment [42], [43]. A typical recurrent neural network structure is shown in Fig. 2.5.

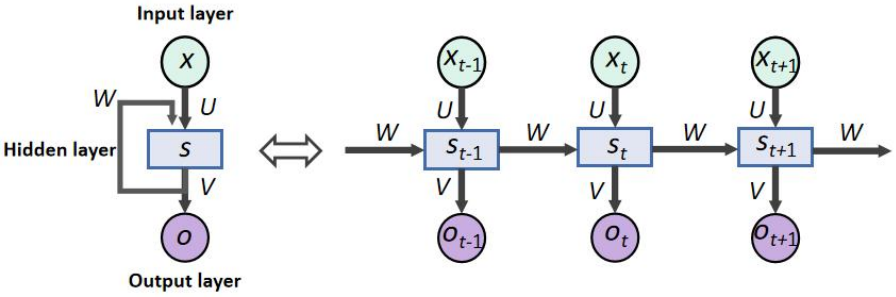


Fig. 2.5 Structure of recurrent neural network.

In the figure, x_t denotes the input at moment t, o_t denotes the output at moment t, and s_t denotes the memory at moment t, i.e., the implicit state. The unfolding diagram shows that the information flow of RNN is unfolded according to the time series, and RNN can remember and learn the features in the time series before the current moment, and then transfer the information in the time series before the current moment to the neural unit at the current moment to achieve the memory capability of time series data. the mathematical representation of RNN is formulated as follows.

The output vector o_t of the output layer is represented as follows:

$$o_t = g(V_{st}) \tag{2-16}$$

The output vector s_t of the loop layer is represented as follows:

$$s_t = f(Ux_t + Ws_{t-1}) \tag{2-17}$$

o_t is a function of st, and thus, o_t can be written as follows:

$$\begin{aligned} o_t = \; & g(Vs_t) \\ & g(Vf(Ux_t + Ws_{t-1})) \\ & g(Vf(Ux_t + Wf(Ux_{t-1} + Ws_{t-2}))) \\ & g(Vf(Ux_t + Wf(Ux_{t-1} + Wf(Ux_{t-2} + \ldots)))) \end{aligned} \tag{2-18}$$

where U represents the weight from the input layer to the hidden layer, V represents the weight from the hidden layer to the output layer, W represents the weight from the state of the hidden layer at the previous moment to the state of the hidden layer at the current moment, and f and g represent the activation function.

The output vector o_t receives the output of the network at the current moment is not

only related to the input at the current moment but also to the state of the hidden layer at the previous moment. It reflects that RNN has memory function and is easy to handle time series related problems.

2.3.2 Long short-term memory (LSTM)

Long short-term memory (LSTM) network is a special kind of RNN. The original RNN model uses backpropagation arithmetic to train the network, and when the sequence is long, the size of the returned values decreases exponentially, making the network weights update slowly and the problem of gradient disappear easily. The training process tends to update the weights according to the effects of the last input signal on the output, and the latter the input has a greater effect on the change of the weights, while the earlier signal has an increasingly lower effect, which has little influence on the final training result. This phenomenon that training results are largely determined by new input information suggests that RNNs do not have a long memory function. Hence, to solve the gradient disappearance problem of RNN during training and to make the network have a longer memory function, Hochreiter and Schmidhuber proposed the long short-term memory network (LSTM) in 1997 [44], whose basic structure is shown in Fig. 2.6.

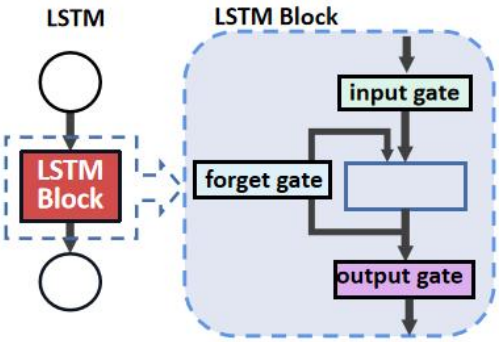


Fig. 2.6 Basic structure diagram of LSTM.

LSTM is a special variant of the RNN, which uses memory cells instead of neurons in the implicit layer of the RNN to realize the memory of past information. This mechanism controls the input, forgetting, and output information of RNNs, which can selectively remember and forget the past information, greatly expanding the learning ability of recurrent neural networks in time series and their application scope.

The original RNN network has only one state h in the hidden layer, and a state c is added in LSTM, called unit state, which can preserve long-term states (as shown in Fig. 2.7). Expanding it according to the time dimension, at moment t, the LSTM has three inputs: the input value x_t of the network at the current moment, the output value h_{t-1} of the LSTM at the previous moment, and the cell state c_{t-1} at the previous moment. The output of

the LSTM has two: the output value h_t of the LSTM at the current moment and the cell state c_t at the current moment.

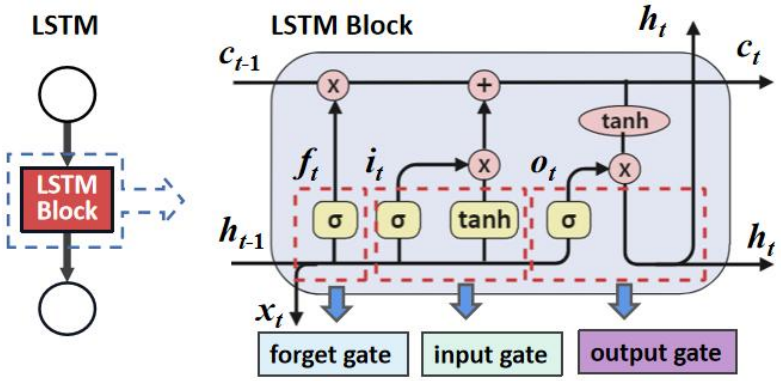


Fig. 2.7 Structure of LSTM expanded by the time dimension.

The key to the LSTM is to implement control of the long-term state c. The LSTM uses three control switches: the first switch, which is responsible for controlling the continued preservation of the long-term state c; the second switch, which is responsible for controlling the input of the immediate state to the long-term state c; and the third switch, which is responsible for controlling whether to use the long-term state c as the output of the current LSTM. The roles of the three switches are shown in Fig. 2.8.

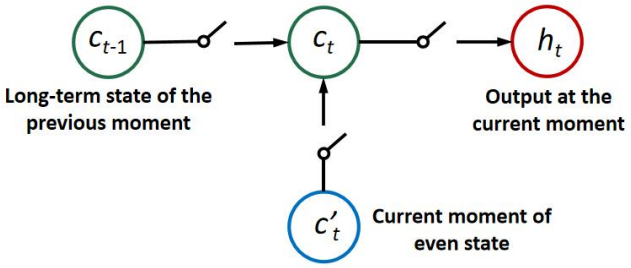


Fig. 2.8 Control chart for long-term state c.

Usually, the forward transfer process of LSTM in a single time step can be expressed by the input, forgetting, and output processes.

The input gate i_t determines the unit that needs to be updated

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \qquad (2-19)$$

where W and b are the weight matrix and the offset, respectively. The forget gate ft determines the historical information that needs to be retained

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \quad (2-20)$$

Subsequently, the state of the model is updated as follows:

$$c_t = f_t c_{t-1} + i_t tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (2-21)$$

The output gate o_t outputs the calculation result in the LSTM unit as follows:

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \quad (2-22)$$

where σ is the activation function. The hidden state information is obtained as follows:

$$h_t = o_t tanh(c_t) \quad (2-23)$$

where *tanh* is the activation function.

The complexity of the network structure is kept in a reasonable range in each step of the training process because of the inclusion of dropout layer in the LSTM network structure; thus, the overfitting problem can be alleviated to some extent [45]. The conceptual diagram of the dropout method is shown in Fig. 2.9. During the training process, for a certain layer of the neural network in one iteration, some neurons are first randomly hidden temporarily before this training and optimization. In the next iteration, some neurons continue to be hidden randomly and so on until the end of training. As a result of the random hiding, each iteration is training a different network with a relatively simple structure, and the forecasting result is the average of the output of each neural network.

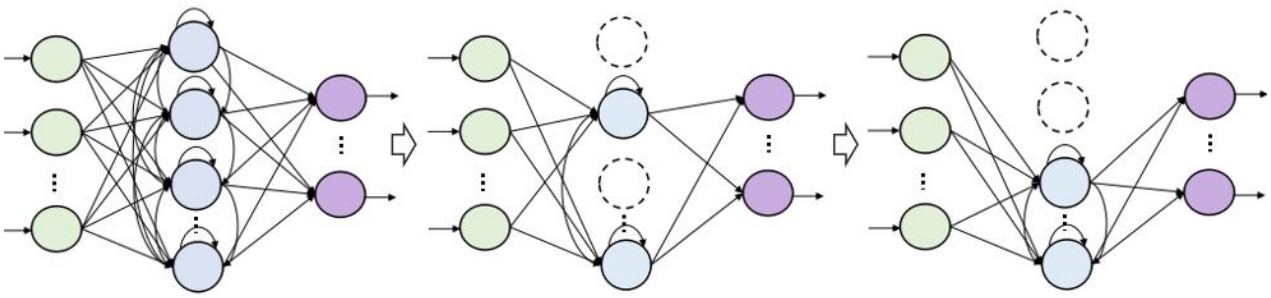


Fig. 2.9. Conceptual diagram of dropout algorithm.

LSTM is an excellent variant model of RNN, inheriting most of the characteristics of the RNN model, which adds three control units of input gate, output gate, and forgetting gate. As the information enters, the control unit in LSTM will judge the information; the information that meets the rules will be left and the information that does not meet will be forgotten. With this principle, the long-time dependence problem in an RNN can be solved, and thus, this paper will use LSTM as the basic model of wind speed forecasting..

2.4 Initial parameter initialization of the neural network

The initialization method of neural network parameters has an important effect on the optimization results of neural networks. Proper initialization of network parameters makes the model training twice as successful. On the contrary, a bad initialization scheme not only affects the network convergence but even leads to gradient disappearance or explosion. In this paper, two methods, genetic algorithm (GA) and ant colony algorithm (ACO) are used individually and in combination to determine the initial parameters of the neural network.

2.4.1 Genetic algorithm (GA)

GA was introduced by Prof. J. Holland in 1975[46]. This algorithm treats individuals belonging to a population as a solution set, introduces genetic and mutation ideas in biology into the solution set, and selects, crosses, and mutates the individuals in the solution set during the solution process. Based on the fitness value of each individual, the next generation of individuals is selected to create a new generation of population. After several iterations, the solution set of the population evolves into a population set containing the optimal solution [37].

The general steps of the genetic algorithm are outlined in Fig. 2.10 as follows:

(1) Analysis of the problem and coding of the variables individually.

(2) Construction of the fitness function.

(3) Initialization of the population.

(4) Selection, crossover, and mutation of the population to form a new generation population.

(5) Calculation of the value of the population fitness function.

(6) Judgement whether the population satisfies the termination criterion, and output the result if it does, otherwise turn to the step (4).

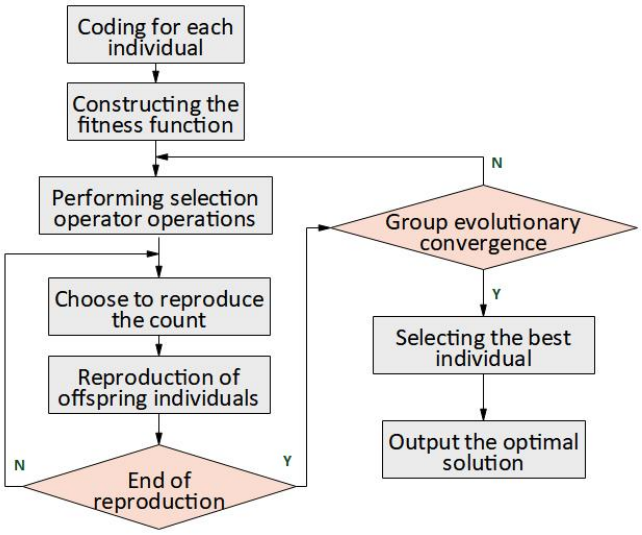


Fig. 2.10 Flowchart of genetic algorithm.

While GA has strong adaptability, versatility, and global optimization performance, it also has certain shortcomings. For instance, GA uses a random probability mechanism for iteration. When the solution reaches a certain range, GA often shows redundancy, and the speed of convergence to the optimal solution is reduced greatly, thereby reducing the efficiency in finding the optimal solution. GA also has weak local search capabilities and is prone to premature convergence when solving optimization problems.

2.4.2 Ant colony optimization (ACO)

The ACO algorithm is an evolutionary calculation method proposed by M. Dorigo et al. in the 1990s based on the foraging mechanism of real ant colonies. This algorithm solves similar optimization problems by simulating the biological process of an ant colony searching for food. ACO is known for its strong robustness, excellent distributed computer system, and easy combination with other algorithms. Despite its lack of a strict theoretical basis, ACO has been widely applied in several areas, such as intelligent optimization [47].

Take the traveling salesman problem as an example. Suppose that there are n nodes. The distance between every two nodes is determined, and the shortest loop that passes each node once and returns to the starting point is solved. When using the ACO algorithm, suppose $b_i(t)$ is the number of ants at the position of node i at time t, $\tau_i(t)$ is the amount of information on the path between nodes i and j at time t, and m is the total number of ants. The artificial ant k (k=1,2,...,m) selects the next node to go to based on the amount of information on the path between each node. A Tabu list tabuk(k=1,2,...,m) is used to store the node that is currently passed by the ant labeled k. tabuk is then dynamically adjusted along with the execution of the algorithm. The transition probability of the k-th ant from nodes i to j at time t is:

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{\left[\tau_{ij}(t)\right]^{\alpha}\cdot[\eta_{ik}(t)]^{\beta}}{\sum_{s\in allowed_k}[\tau_{is}(t)]^{\alpha}\cdot[\eta_{is}(t)]^{\beta}}, & if\ j\in allowed_k\\ 0, & otherwise\end{cases} \quad (2-24)$$

where $allowed_k$ represents the set of nodes that the k-th ant can choose in the next step, and α is the information heuristic factor that represents the relative importance of each path. A larger a corresponds to a higher tendency for the ant to choose the previous path, thereby reducing the randomness of the search path. β is the expected heuristic factor that indicates the relative importance of visibility. A larger β corresponds to a greater importance of the heuristic factor when the ant chooses a path during its movement. $\eta_{ij}(t)$ is a heuristic function, which indicates the expected degree for ants to choose the path (i, j) from nodes i to j. This function can be expressed as follows:

$$\eta_{ij}(t)=\frac{1}{d_{ij}} \quad (2-25)$$

The value of $\eta_{ij}(t)$ is determined by the distance d_{ij} between two nodes, that is, a smaller d_{ij} corresponds to a larger $\eta_{ij}(t)$ and a greater transition probability $p_{ij}^{k}(t)$.

To prevent the heuristic information between nodes from being weakened by the excessive pheromone on the corresponding path, the amount of information carried on the path must be updated each time an ant passes through a node or passes through all nodes. Generally, the amount of information on path (i, j) at time $t+n$ is updated as

$$\tau_{ij}(t+n)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}(t) \quad (2-26)$$

$$\Delta\tau_{ij}(t)=\sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t) \quad (2-27)$$

where $\rho\in[0,1)$ is the volatilization coefficient of the pheromone, and 1-ρ is the residual factor of the pheromone. This coefficient can prevent an unlimited increase of the pheromone on the path. $\Delta\tau_{ij}(t)$ is the information increment on the path (i, j) traversed by the ant during its movement at time t, and its initial value is set to $\Delta\tau_{ij}(0)=0$. Meanwhile, represents the increment of information generated by the k-th ant on path (i, j) at time t.

In this way, m ants randomly select each node and walk for a week. After each ant passes all nodes, the pheromone concentration on the path is updated to calculate the probability for the next ant to choose this path. When the number of cycles reaches a preset value or when the selection probability of each path no longer changes significantly, the cycle is stopped to determine the best route[48], [49].

2.5. Evaluation criteria of forecast performance

To evaluate the forecasting results comprehensively from the perspective of accuracy and stability, several evaluation indicators are selected to compare the performance of competitive models. MAE, MAPE, and RMSE are used to evaluate the forecasting accuracy. MAE and RMSE can display the average deviation between the forecasted value and the actual value, and the use of MAE and RMSE can solve the problem of offsetting positive and negative deviations. MAPE is a common indicator and can be used to test the overall forecasting of this combined model effect. Equations (2-28) – (2-30) indicates the specific definition of the evaluation criteria. y_t and $\hat{y}_t$ represent the actual and forecasted value at time t, respectively. N represents the total number of samples.

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2} \quad (2-28)$$

$$MAE = \frac{1}{N}\sum_{t=1}^{N}|y_t - \hat{y}_t| \quad (2-29)$$

$$MAPE = \frac{1}{N}\sum_{t=1}^{N}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \times 100\% \quad (2-30)$$

To facilitate the accuracy comparison among different models, this paper uses three improvement percentage indicators, namely, P_{RMSE}, P_{MAE}, and P_{MAPE}. The calculation method is shown in Equations (2-31) – (2-33).

$$P_{RMSE} = \left|\frac{RMSE_1 - RMSE_2}{RMSE_2}\right| \times 100\% \quad (2-31)$$

$$P_{MAE} = \left|\frac{MAE_1 - MAE_2}{MAE_2}\right| \times 100\% \quad (2-32)$$

$$P_{MAPE} = \left|\frac{MAPE_1 - MAPE_2}{MAPE_2}\right| \times 100\% \quad (2-33)$$

Chapter 3. Ultra-short-term wind speed forecasting

For ultra-short-term wind speed forecasting, most previous studies focused mainly on optimization algorithms and data processing to improve the accuracy of wind speed forecasting while ignoring the selection and extraction of wind speed data [50]. In addition, the established wind speed forecasting models concentrate on a single location only. Local historical data are used as training data for ultra-short-term forecasting of future wind speed. In this case, historical data that are too old are not informative. Moreover, providing accurate wind speed forecasting is difficult in areas where historical data are lacking or where the wind speed change cycle is short and the change range is large [51].

In the previous forecasting process, the forecasting accuracy of different areas varies greatly, and forecasting is even impossible in some areas, often because of drastic changes in wind speed and the lack of historical data of wind speed [75]. Therefore, on the basis of improving the precision, this research also focuses on improving the stability and versatility of the forecasting model.

For ultra-short-term wind speed forecasting, this chapter proposes a forecasting system that contains data extraction module, data preprocessing module, and forecasting module.

3.1 Construction of the forecasting model

3.1.1 Data extraction module

This module proposes a data area division (DAD) method to extract historical wind speed data.

In the process of wind speed forecasting, we found that wind speed data from other areas that are highly correlated with the wind speed time series of the location to be forecasted can be used as training data after processing. Considering that the wind speed time series at these locations have similar changes, the wind speeds are also very close. The extraction of wind speed data from multiple locations within a short period of time as training data can ensure a higher correlation between the selected historical data and the wind speed to be forecasted while avoiding the problem of insufficient historical data to

train the forecasting model. Thus, a DAD method is proposed based on the above point of view. The flowchart is shown in Fig. 3.1.

Step 1. Divide according to the wind speed：On the basis of the average and standard deviation of the historical wind speed time series at each location, the area to be forecasted was divided into several groups using the k-means method. With 141 locations in Hokkaido taken as an example, the specific steps are:

(1) k locations are selected randomly as initial clustering centers, $\alpha = \alpha_1, \alpha_2, \cdots, \alpha_k$.

(2) For each location x_i in the dataset, its distance to the K cluster centers is calculated and assigned it to the group corresponding to the cluster center with the smallest distance.

(3) For each group, recalculate its cluster center $\alpha_j = \frac{1}{|c_i|}\sum_{x \in c_i} x$ is recalculated. (the center of mass of all samples belonging to the group)

(4) Steps (2) and (3) are repeated until the clustering centers no longer change significantly.

The only parameter that needs to be set for the k-means method is the number of clusters K. Here, K is set to 10 (the trial-and-error method suggests that this scale is reasonable). As shown in Fig. 3.2, all locations in Hokkaido can be divided into 10 groups. Groups A, B, and C are taken as examples; group A includes 10 locations, group B includes 6 locations, and group C includes only 1 location. The wind speed data used for the grouping in this example are the historical wind speed data for various locations in January 2019; in fact, the wind speed data used for the grouping when forecasting using the proposed forecasting system is the historical wind speed data for 30 days prior to the forecast time point.

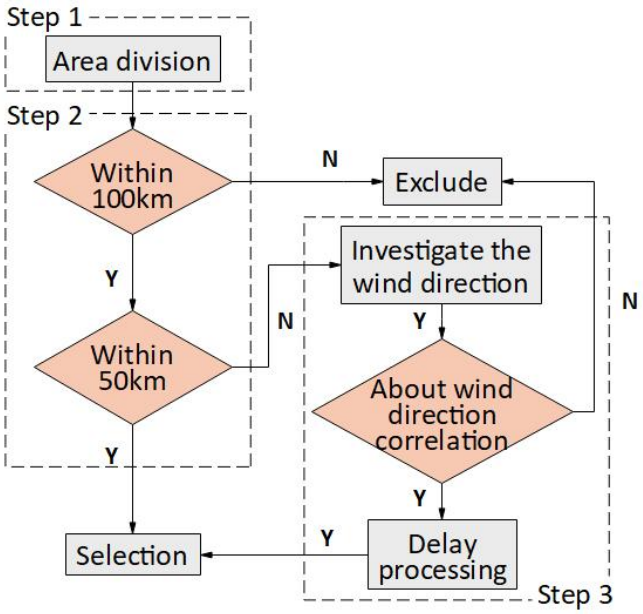


Fig. 3.1 Flowchart of the DAD method

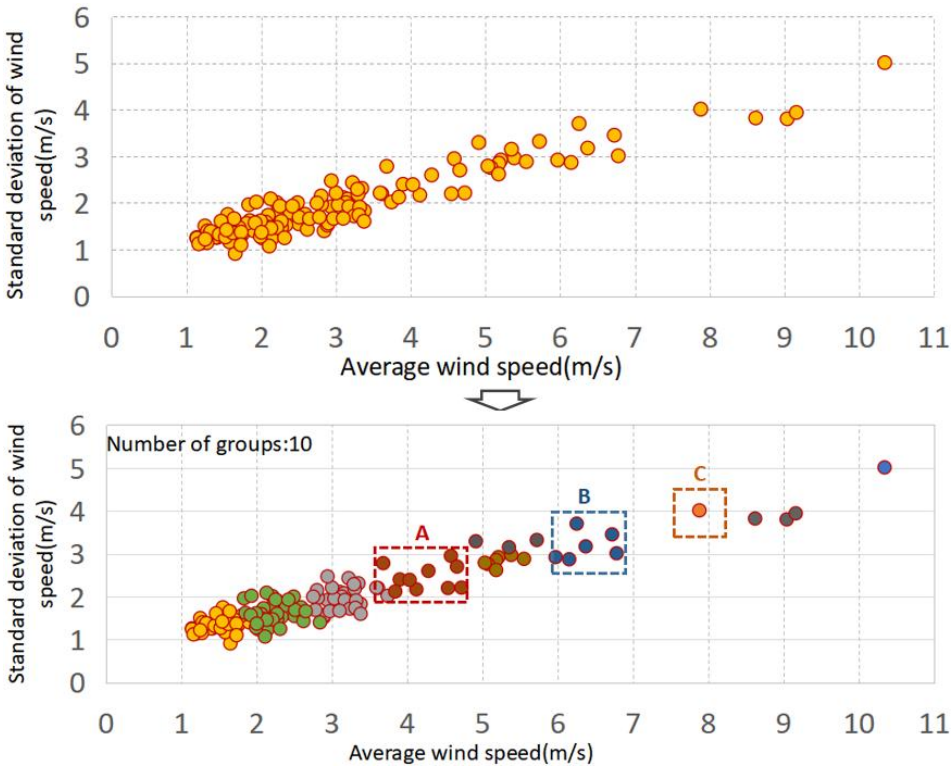


Fig. 3.2 Grouping results for the Hokkaido area

Step 2. Divide by geographic distance: On the basis of Step 1, the available wind

speed history data are selected according to the geographic distance. For example, Muroran is taken as the center, and a search for areas within radii of 0, 50, 100, and 150 km is performed. The results are shown in Table 3.1. The Pearson correlation coefficient between each location and Muroran’s wind speed time series is calculated by using Equation (3-1). The results are shown in Table 3.2. The wind speed time series of locations within 50 km have an obvious correlation with Muroran, and the wind speed time series of locations within 100 km and Muroran exhibit a certain relevance. Therefore, on the basis of the division result of Step 1, the wind speed data of the locations within 50 km of the center to be forecasted directly are used as the training data, and the wind speed data of locations within 50–100 km need to be investigated further.

$$r(V_1, V_2) = \frac{Cov(V_1, V_2)}{\sqrt{Var[V_1]Var[V_2]}} \qquad (3-1)$$

where V_i is the wind speed time series at location i, Cov(V_1 ,V_2) is the covariance of V_1 and V_2, and Var[V_i] is the variance of V_i .

Table 3.1 Locations within each radius around Muroran.

Observation radii (km)	Locations	Number of locations
0	Muroran	1
50	Date, Noboribetsu	3
100	Tomakomai, Rankoshi, Hokuto, Suttsu	7
150	Okushiri, Iwamizawa	9

Table 3.2 Correlation coefficient of wind speed time series between locations.

Correlation coefficient	Muroran
Date	0.6021
Noboribetsu	0.5766
Tomakomai	0.4123
Rankoshi	0.4526
Hokuto	0.3314
Suttsu	0.4021

Okushiri	0.2421
Iwamizawa	0.1719

Step 3. Divide by wind direction: Our investigation shows that within a certain period of time, the wind speed series of adjacent locations with roughly the same wind direction have a strong correlation. Muroran is taken as an example. Fig. 3.3 shows the wind direction distribution of Muroran in January, April, July, and October 2019. The change in wind direction shows evident seasonality. With January taken as an example, the wind direction in Muroran is mainly northwest wind. At this time, Suttsu, which is in the 50–100 km range around Muroran, is compared with Muroran. The comparison of the wind speed time series in January is shown in Fig. 3.4 (a). The wind speed time series at the two locations show a certain correlation. After several hours of delay processing, Suttsu's wind speed time series has a stronger correlation with that of Muroran. The correlation coefficients between the wind speed time series of Suttsu after each hourly time delay and the wind speed time series of Muroran, and the autocorrelation coefficients between the wind speed time series of Suttsu after each hourly time delay and the original wind speed time series are shown in Table 4, where the autocorrelation coefficients are calculated as shown in Equation (3-2).

$$r_a(h) = \frac{\sum_{i=1}^{n-h}(y_i - \hat{u})(y_{i+h} - \hat{u})}{\sum_{i=1}^{n}(y_i - \hat{u})} \qquad (3-2)$$

where h is the number of hours of delay, n is the total length of the wind speed time series, and $\hat{u}$ is the average wind speed of the wind speed series.

Table 3.3 shows that the autocorrelation coefficients of the wind speed time series of Suttsu and the original wind speed time series gradually become smaller as the number of delay hours increases. From this finding, the wind speed time series after the delay treatment can be judged to no longer have an autocorrelation with the original wind speed time series after the delay hours reach a certain range. Meanwhile, the correlation coefficients of the wind speed time series of Suttsu and Muroran after the delay treatment show that the wind speed time series of the two places have the strongest correlation after the five-hour delay treatment. The autocorrelation coefficient of the wind speed time series of Suttsu is 0.5176 at this time; thus, the wind speed time series at this time still maintains the characteristics of the original wind speed time series to some extent. The comparison chart of the wind speed time series of the two places in this case is shown in Fig. 3.4 (b). On the basis of the above analysis, Suttsu's wind speed time series after a five-hour delay

is selected when forecasting Muroran's wind speed.

Through the screening of the three aforementioned steps, when Muroran is the forecasted target location and January is the forecasted time period, the wind speed data of Date, Noboribetsu, Suttsu, and Rankoshi can be used as training data.

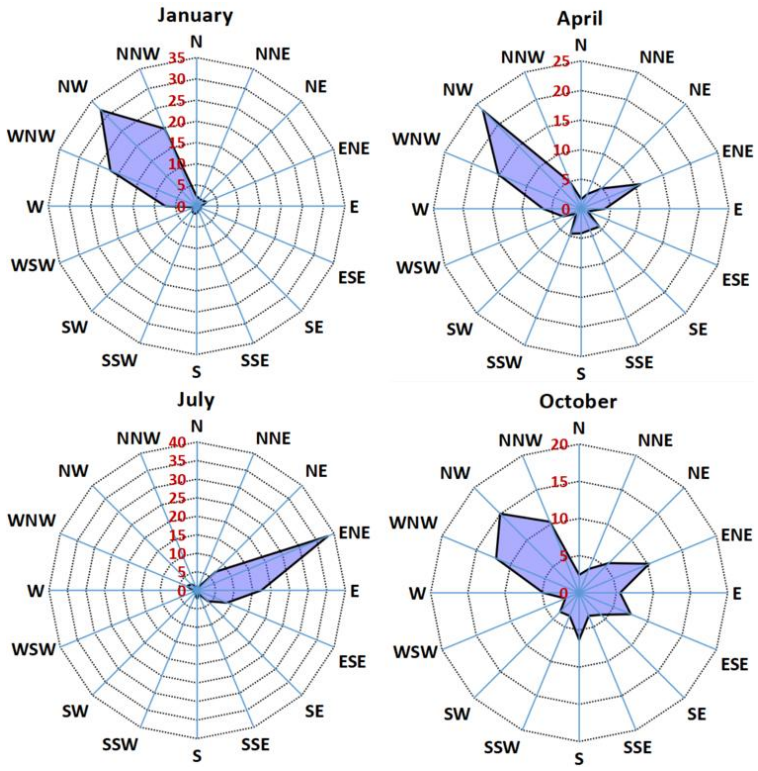
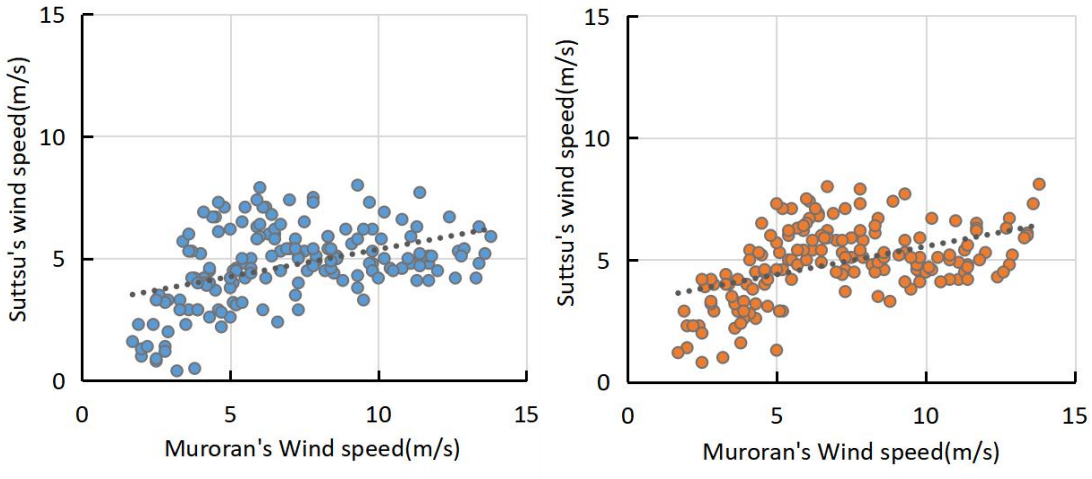


Fig. 3.3 Wind direction distribution of Muroran in January, April, July, and October 2019.



(a) (b)

Fig. 3.4 Comparison of Muroran's and Suttsu's wind speed time series.

Table 3.3 Changes in correlation and autocorrelation coefficients with time-delayed processing

Time delay	Correlation coefficient (Muroran and Suttsu)	Autocorrelation coefficient (Suttsu)
1h	0.4429	0.6986
2h	0.4365	0.6463
3h	0.4863	0.5774
4h	0.5118	0.5203
5h	0.5711	0.5176
6h	0.5321	0.4592

3.1.2 Data preprocessing module

Although the aforementioned data collection method provides sufficient training data for the forecasting model, thereby improving the versatility of the forecasting model greatly, the original data that are directly used will lead to poor training effects because of the noise and random fluctuations in the original wind speed series. To reduce the random volatility of wind speed time series, many researchers have proposed data preprocessing strategies using EMD [23] or EEMD [38].

EMD [25] refers to the nonlinear and non-stationary signal analysis method developed by Huang et al. in 1998. This method extracts a series of intrinsic mode functions (IMF) from the original signal through stepwise filtering and processes them separately. However, EMD has an evident shortcoming called mode mixing [38], in which a single IMF composed of signals with significantly different scales is easily generated or signals of the same scale appear in different IMF components. To solve this problem, Wu and Wang proposed EEMD in 2008 [38], which avoids mode mixing effectively by adding white noise to the original signal. However, it also has low decomposition efficiency and residual noise.

To decompose the original wind speed data more efficiently and accurately, an improved version of EEMD called CEEMD is used to process wind speed data [53].

The CEEMD algorithm is described as shown in subsection 2.2.2, and *NE* and *Nstd* are discussed in detail in subsections 3.2.2 and 4.2.2.

3.1.3 Forecasting module

In the forecasting module, the LSTM neural network is used, which has an improved effect on time series forecasting in the deep learning neural network. To improve forecasting efficiency, GA is used to optimize the internal parameters of LSTM.The specific descriptions of LSTM and GA are shown in subsections 2.3.2 and 2.4.1.

GA optimizes the internal parameters of the LSTM (Fig. 3.5).

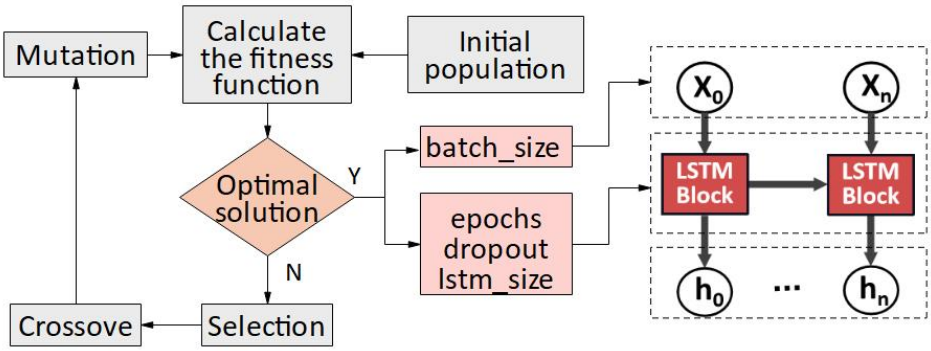


Fig. 3.5 Flowchart of GA optimization of LSTM.

GA is used to optimize the data time window step size (batch_size), the number of hidden units in the LSTM Network (lstm_size), the number of training times (epochs), and the dropout ratio (dropout). Four parameters are optimized in the search space to determine the best parameter combination.

The main process is presented as follows:

(1) The population is initialized and decoded.

(2) The mean square error of the LSTM neural network is taken as the fitness function.

(3) The selected cross mutation operation is performed on the solved individual.

(4) If the target value of the fitness function reaches the optimal value, then the next step is performed; otherwise, Step 3 is repeated.

(5) The process is followed step by step to obtain the fitness target value and optimal parameters.

(6) The forecasting mean square error is calculated based on the best parameters.

(7) The termination conditions are judged as follows: If the number of population iterations is satisfied, then the calculation is stopped. At this time, the LSTM network determines the global optimal parameter combination [batch_size, lstm_size, epochs, dropout]; otherwise, Step (6) is repeated.

3.2 Forecast preparation

In this section, the specific implementation conditions of this research are described. These conditions mainly include the selection of forecasting locations and forecasting time periods, the introduction of using historical wind speed data, the preparation of forecasting models and the setting of related parameters, and the introduction of forecasting accuracy evaluation methods.

3.2.1 Data description

Hokkaido is located in the northernmost part of Japan. It is rich in solar, wind and biomass resources, and has the greatest potential for new energy utilization in Japan. The introduction rate of wind power generation in Hokkaido ranks first in Japan (approximately 55% and 29% for onshore and offshore wind power respectively), and in 2018, the total power generation of wind power plants was 441,185 kW, with 54 power plants and 329 wind turbines. Most of the power plants are concentrated in coastal areas with abundant wind resources, and their main distribution is shown in Fig. 3.6.

On the basis of the above analysis, 10 locations in Hokkaido (i.e., Wakkanai, Embetsu, Ishikari, Otaru, Esashi, Matsumae, Muroran, Hiroo, Atsutoko, and Nemuro) (Fig. 3.6) are selected as the target locations for forecasting. The wind speed data from a total of 141 locations in the Hokkaido area, including the target location and locations near the target location where wind speed data can be collected, published on the official website of the Japan Meteorological Agency are used as training and validation data to train the proposed forecasting system. The forecast target time period selected for this study is from January to December 2019. Thus, the actual wind speed data used in this study, including the training data, are wind speed data from 141 locations in Hokkaido from November 2018 to December 2019, and the forecast system is evaluated in terms of forecast accuracy at three time intervals of 10, 30, and 60 minutes. The description of the forecast accuracy evaluation criteria will be given in subsection 3.2.3.

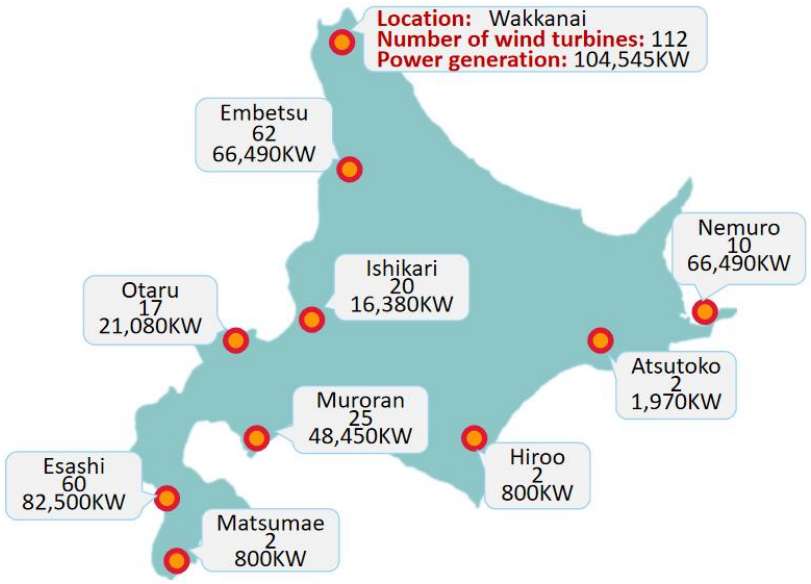


Fig. 3.6 Distribution of wind power plants in Hokkaido.

The example of Muroran is illustrated for the specific data selection method. As mentioned in subsection 3.1.1, when Muroran is used as the target location for forecasting, the historical wind speed data for five locations—Muroran, Date, Noboribetsu, Suttsu, and Rankoshi—can be used as training data. The data selection principles for the forecasting of wind speeds from January 1 to 31, 2019 are shown in Table 3.4.

Table 3.4 Principles of data selection

Time interval	Training data				Validation data			
	Number of locations	Time Period	Number of days	Number of data	Number of locations	Time Period	Number of days	Number of data
10-min	5	2018.12.2~ 2018.12.31	30	21600	1	2019.1.1~ 2019.1.31	31	4464
30-min	5	2018.12.2~ 2018.12.31	30	7200	1	2019.1.1~ 2019.1.31	31	1488
60-min	5	2018.11.2~ 2018.12.31	60	7200	1	2019.1.1~ 2019.1.31	31	744

For forecasting the wind speed at each time interval, the training data and validation data are selected for the same time interval, and the number of days was set to 30 and 60 days because, as mentioned in our published paper [75], the amount of data should be as sufficient as possible considering the timeliness and training efficiency of the data; this number was finally determined by trial and error.

In accordance with the above-mentioned data selection principle, this paper selects the training and verification data of 10 locations to be forecasted, and the wind speed in 2019 is forecasted.

3.2.2 Parameter setting of the model

CEEMD: When CEEMD is used, two parameters need to be set: the ensemble number (*NE*) and the amplitude of the white noise (*Nstd*) [53]. In this paper, *NE* is initially set to 300 and *Nstd* to 0.3 based on experience. The value was adjusted after passing the forecast result. With Muroran taken as an example (Table 3.5), the mean absolute percentage error (MAPE) of wind speed forecasts at 10-minute intervals in January 2019 varies with the changes in the values of *NE* and *Nstd*.

Table 3.5 Forecasting accuracy when setting different *NE* and *Nstd.*

MAPE (%)	*Nstd*				
NE	0.1	0.2	0.3	0.4	0.5
100	8.06	7.87	7.27	7.15	7.71
200	8.21	7.62	7.02	7.23	7.34
300	7.92	7.37	7.01	7.02	7.31
400	7.12	6.91	6.11	6.18	6.75
500	6.94	6.92	6.07	6.14	6.68
600	7.23	7.26	6.68	6.58	7.53

NE is in the range of 400–500, and the best forecasting effect can be achieved when *Nstd* is 0.3 or 0.4. Therefore, the actual decomposition effect is combined, *NE* is set to 500, and *Nstd* is set to 0.3.

LSTM: The conceptual diagram of the LSTM structure is shown in Fig. 3.5, where input_size is the number of input layer units, lstm_size is the number of units in each hidden layer, and output_size is the number of output layer units. The input_size and output_size must first be determined. The principle of setting input_size was explained in detail in a paper previously published by our team [75]. On the basis of considering the continuity characteristics of the time series, the selection of data with higher correlation performance with the time series of wind speed to be forecasted was prioritized. On the basis of previous experiences, the wind speed that has a high correlation with the wind speed to be forecasted is usually the wind speed at 3 to 5 time points before the forecast time point. With Muroran's 10-minute interval wind speed data taken as an example, let V_t be the wind speed time series to be forecasted and V_{t-i} be the wind speed time series before i time at the time point to be forecasted. The correlation between V_t and each time series is shown in Table 3.6. For the wind speed data at 10-minute intervals, the wind speed of the four time points before the forecast time point and the wind speed at the forecast time point have a relatively high correlation. Therefore, when the wind speed is forecast at 10-minute intervals, the input_size is set to 4. Similarly, the input_size for the 30-minute interval wind speed forecasting is set to 4, and the input_size for the 60-minute interval wind speed forecasting is set to 3. However, the proposed forecasting system does not forecast the original wind speed data directly and forecasts the decomposed components instead. Yet,

the decomposed components still inherit the correlation between the original wind speed time series in each time interval. Thus, the previous method is still used to set it. Output_size is set to 1 because the purpose of this research is to forecast the wind speed at the next moment.

Table 3.6 Correlation coefficient between wind speed time series V_t.

Correlation coefficient	V_{t-1}	V_{t-2}	V_{t-3}	V_{t-4}	V_{t-5}	V_{t-6}
V_t	0.7154	0.6963	0.6253	0.5232	0.3123	0.2104

In the process of numerical simulation using LSTM, batch_size (dividing the data into several groups for training with the amount of data in each group), epochs (rounds of training the model with all data), dropout (inactivation probability of dropout layer), and lstm_size (the total number of units in the hidden layer) also need to be set. The setting of lstm_size will affect the accuracy of the forecasting model directly. The batch_size and epoch settings will affect the training efficiency and convergence of the model greatly because of the large number of training data samples. The function of dropout is to inactivate the input and recursive connections of LSTM units with a certain probability in the process of forward transfer and weight update. Setting the dropout probability value correctly can avoid overfitting and improve model performance effectively. Thus, GA is used to optimize the batch_size, epochs, dropout, and lstm_size of the forecasting model. The optimization process is described in the summary of subsection 3.1.3, and the initial values are set as shown in Table 3.7.

Table 3.7 Initial parameter setting of LSTM.

Parameter	Initial Value
batch_size	100
epochs	50
dropout	0.2
lstm_size	50

3.3 Validation of the proposed method by numerical calculations

For the analysis of the versatility and practicality of the proposed hybrid forecasting system, experiments are conducted to investigate the performance of the proposed forecasting system in different situations. For the comparison of the performance of the proposed forecasting system and other forecasting models, different types of forecasting models are selected and the same historical wind speed data are used as verification data. The experimental results and related analysis will be introduced in this section.

3.3.1 Experiment I: Forecast of annual wind speed in different locations

As mentioned in the summary of section 3.1, the 10 locations in Hokkaido, namely, Wakkanai, Embetsu, Ishikari, Otaru, Esashi, Matsumae, Muroran, Hiroo, Atsutoko, and Nemuro, are used as the forecast target locations. The wind speed at 10-minute intervals is used as the forecast target to forecast the wind speed for the whole year of 2019. MAE, RMSE, and MAPE are calculated on the basis of the forecast results. The monthly average is computed as shown in Table A.1. The comparisons of MAE, RMSE, and MAPE are shown in Figs. 3.6–3.8, respectively. Figs. 3.6 and 3.7 show that the RMSE is basically maintained in the range of 0.2 to 0.6 in the annual wind speed forecast for the whole site. The maximum value of RMSE is approximately 0.6, which appears in the location forecasting of Locations 1 and 5 in January. From November to February, the RMSE value is relatively high, mainly concentrated in the range of 0.40 to 0.55, while the range of MAE is between 0.35 and 0.50, and the overall deviation is not obvious. The minimum RMSE value appeared in the forecasting for Location 6 in July. In other locations, except for Location 3, the RMSEs in July and August are also small, mainly concentrated in the range of 0.15 to 0.25. The range of MAE is between 0.10 and 0.30. Thus, the preliminary conclusion is that the stability of the forecasting model will be affected by the season and the geographic location of the forecasted location, but the overall impact is insignificant. The forecasting results obtained from the 10 locations are relatively similar. MAE shows a certain periodic change, but not much difference is observed because of the change in location.

Fig. 3.8 shows that MAPE has obvious seasonal changes. Although certain differences are observed in the monthly averages of various locations, the overall performance suggests that the forecast results from July to September have the highest accuracy, and MAPE is in the range of 3.0% to 4.5%, whereas the forecast accuracy from April to June is unstable. MAPE is more common in the range of 3.0%–5.0%. The forecast results from November to March are relatively poor. MAPE is generally in the range of 4.0%–7.0%, with the maximum value reaching 7.33%.

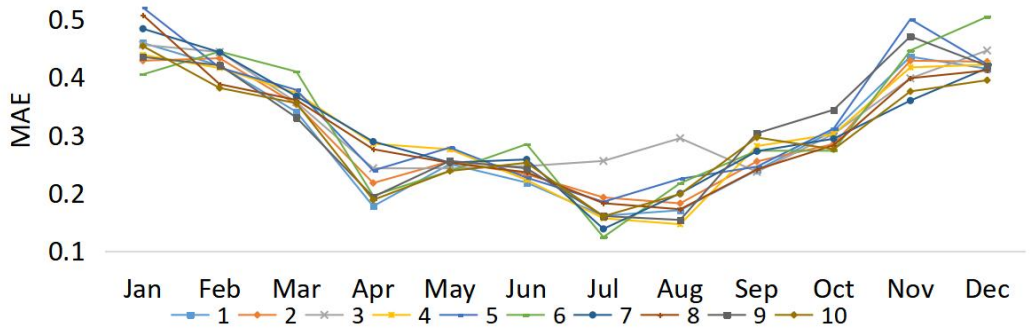


Fig. 3.6 Distribution of MAE in 10 locations in different periods.

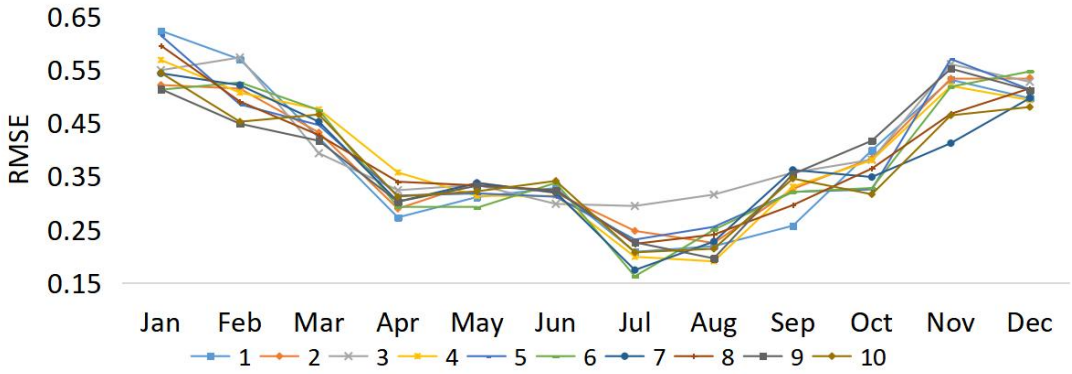


Fig. 3.7 Distribution of RMSE in 10 locations in different periods.

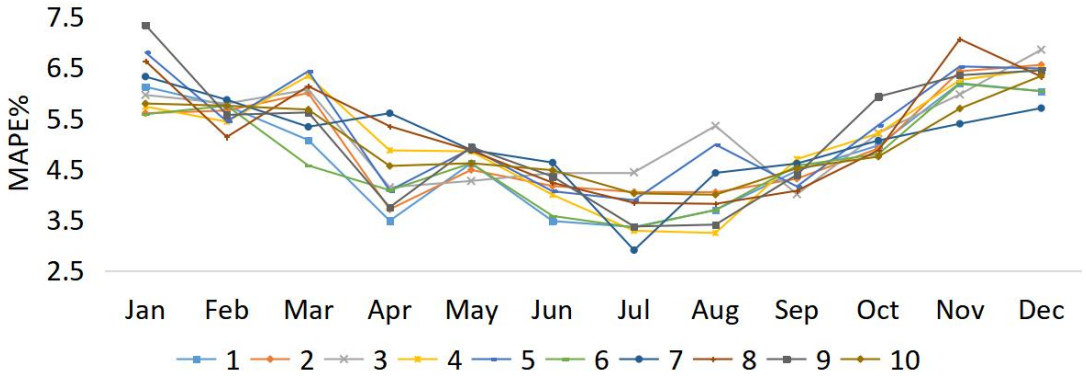


Fig. 3.8 Distribution of MAPE in 10 locations in different periods.

The monthly wind speed distribution is studied to investigate the cause of this phenomenon. With Location 1 (Wakkanai) taken as an example, the monthly instantaneous wind speed distribution in mid-2019 is organized into a box plot (Fig. 3.9). In Hokkaido, the average wind speed from November to March is relatively high, and the range of wind speed variation is relatively wide. The wind speed is mostly concentrated in the range of 6–10 m/s, whereas the average wind speed from July to September is relatively low, and the wind speed changes smoothly. The proposed wind speed forecasting system has high

stability and can deal with wind speed forecasting under different conditions better when concentrated near the average wind speed of 4–6 m/s. However, the specific forecasting conditions will be affected by the wind speed situation at the target location. The deviation range of MAPE is 3%–7%.

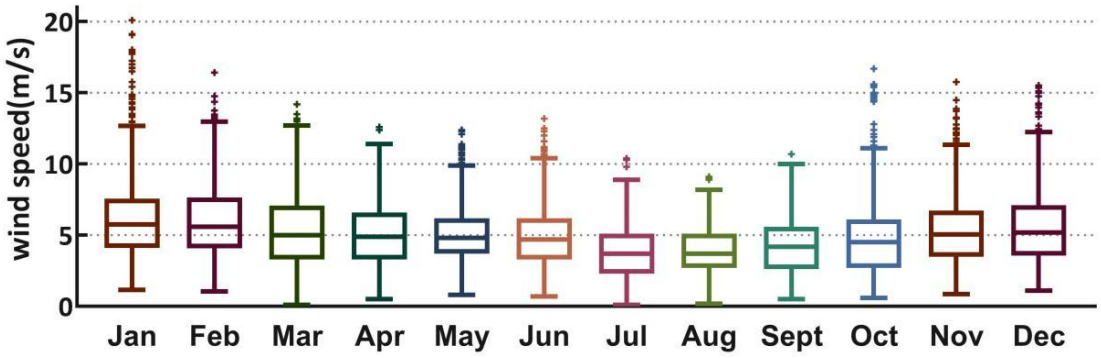


Fig. 3.9. Monthly instantaneous wind speed distribution of Wakkanai.

3.3.2 Experiment II: Accuracy comparison of different time intervals

In this section, the wind speed at the three time intervals of 10, 30, and 60 minutes is forecasted by using the data selection method in Section 3.1 to verify the accuracy of the proposed forecasting system in wind speed forecasting at different time intervals. Experiment 1 shows that the forecasting accuracy for 10-minute wind speed forecasting is greatly affected by wind conditions but is slightly affected by geographic location. Thus, only Wakkanai is selected as the forecast target location, and the selected forecast time periods are January, April, July, and October. The MAE and RMSE of the forecasted results are shown in Table A.2. The comparisons of MAE and RMSE are shown in Figs. 3.10 and 3.11. Fig. 3.12 shows a box plot chart of MAPE. As the time interval increases, the forecasting accuracy gradually decreases. Overall, the forecasting accuracy of the 30-minute interval can still be maintained at a good level. The RMSE is relatively stable and maintained in the range of 0.3 to 0.7. The MAPE can be maintained in the range of 4% to 9%. Similar to the forecasting results at a 10-minute interval, in July, when the wind speed is relatively stable, a good forecasting accuracy is achieved, and the minimum MAPE can reach 4.6%. In January, when the wind speed changes sharply, the forecasting accuracy is relatively poor but still acceptable. The MAPE is maintained in the range of 6% to 9%. The forecasting accuracy of the 60-minute wind speed is significantly reduced, and the RMSE is in the range of 0.5 to 0.8, with relatively large fluctuations. The MAPE is in the range of 6%–12%, and the highest is 11.6%. However, most cases are concentrated around 8%. The forecasting accuracy needs to be improved, but it is still within an

acceptable range.

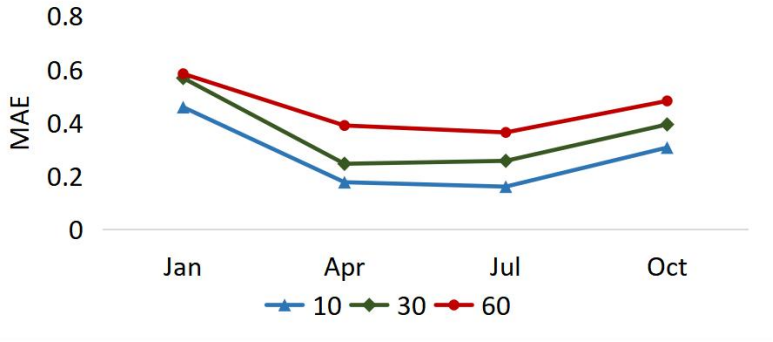


Fig. 3.10 Distribution of MAE of different time intervals.

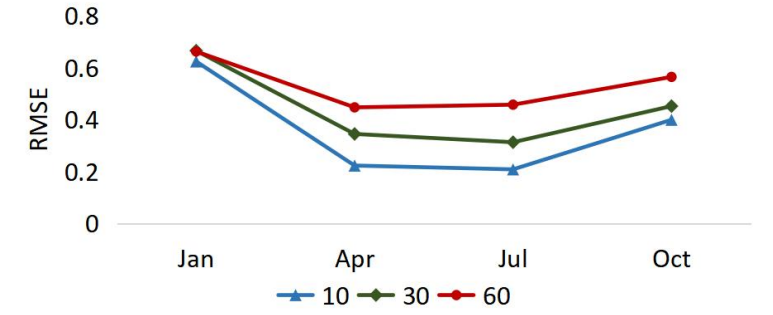


Fig. 3.11 Distribution of RMSE of different time intervals.

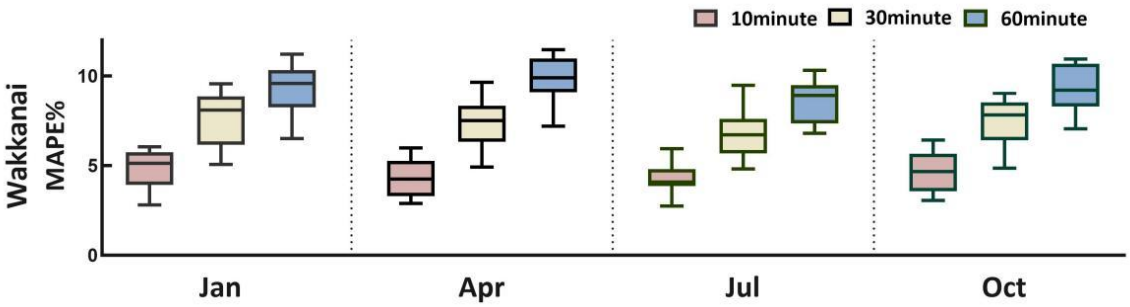


Fig. 3.12 Comparison of MAPE at different time intervals.

3.3.3 Experiment III: Accuracy comparison of different forecasting models

In this section, several typical single forecasting models, such as ARIMA[54]–[56], BPNN [35], Elman [57], ELM [58], and the widely used hybrid model EMD-RNN EEMD-LSTM, EWT-LSTM-Elman [59], and CEEMDAN-ENN [60], are selected.

The same historical wind speed data are used to train these forecasting models. For the

proposed forecasting system, the DAD method introduced in subsection 3.1.1 is used to extract data and train the forecasting model.

With Wakkanai taken as the forecasted target location and the wind speed at 10-minute intervals used as the forecast target, the wind speeds in January, April, July, and October 2019 are forecasted. Table A.3 and Fig. 3.13 show the comparison of the typical single forecasting model results and the proposed model, and Table A.4 and Fig. 3.14 show the comparison of hybrid forecasting model results. The results show the following:

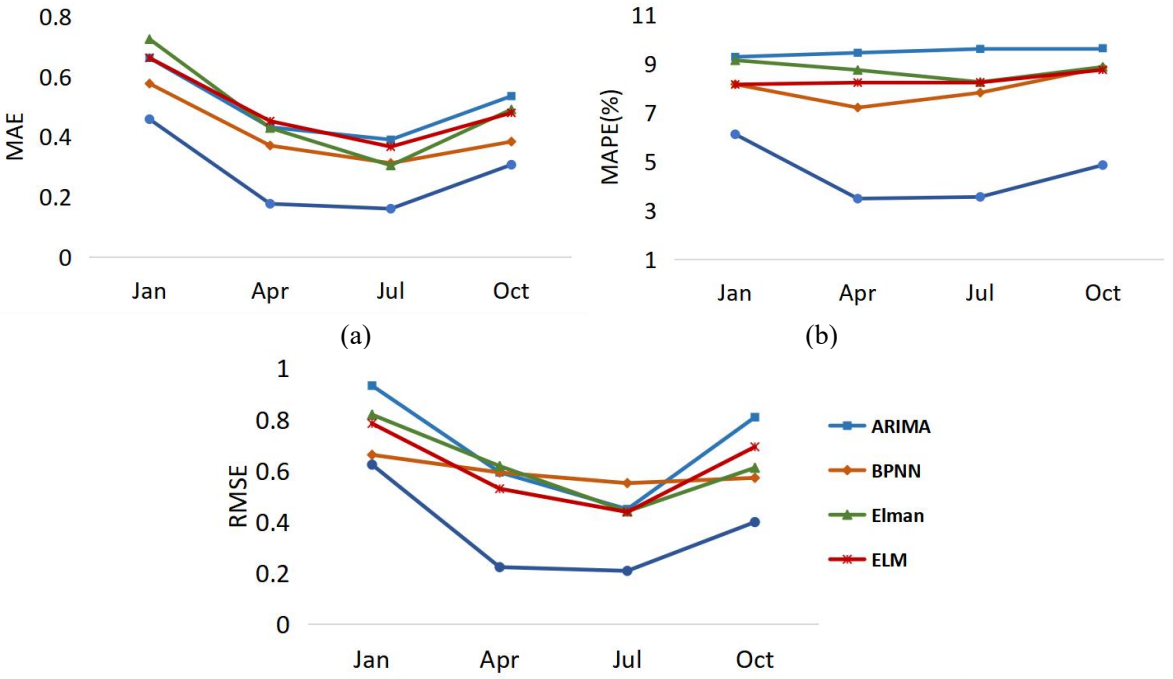
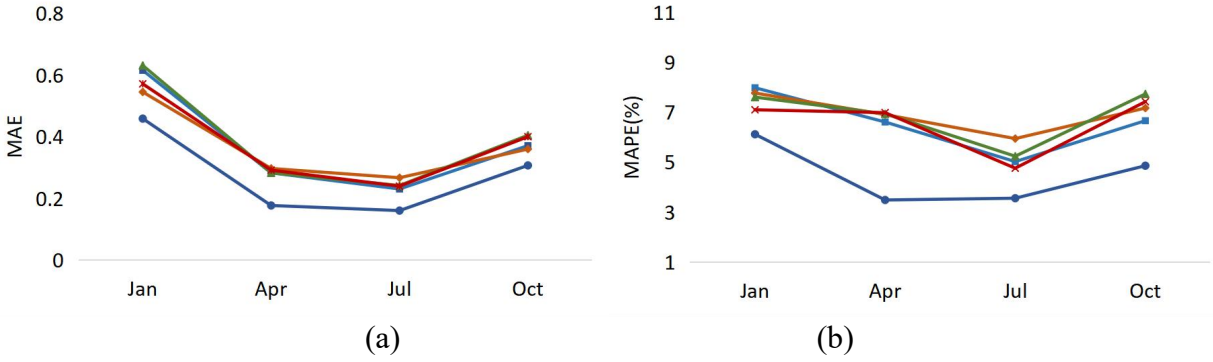
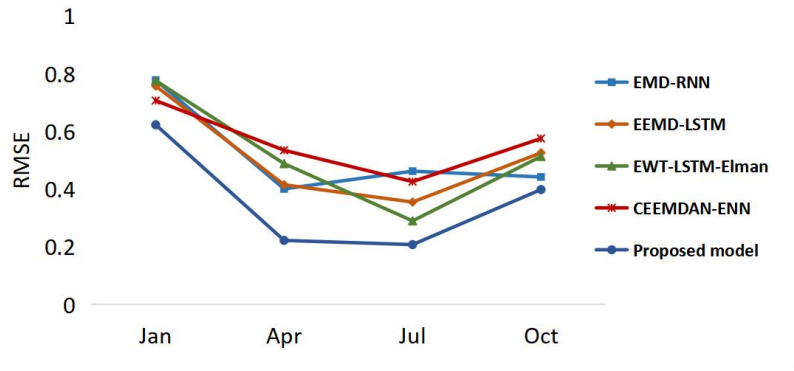


Fig. 3.13 Comparison of forecasting accuracy with typical single forecasting model.





(c)

Fig. 3.14 Comparison of forecasting accuracy with typical hybrid forecasting model.

(1) The classic single forecasting model has relatively poor versatility, and the forecasting accuracy varies greatly in different locations. For Wakkanai, which is the forecasted target location selected in this section, the fluctuation range of MAPE is not large and can basically be maintained in the range of 7%–9%. However, the overall accuracy is low. For forecasts in different seasons, January has the lowest accuracy, whereas July has the highest accuracy. Unlike the classic single forecasting model, the proposed forecasting system reaches the optimal value of MAE, MAPE, and RMSE in different periods.

(2) The overall forecasting accuracy of the hybrid forecasting model is higher than that of the single forecasting model, and the versatility of the model is relatively good. The results in Table A.4 show that the RMSE remains in the range of 0.4 to 0.8, and the stability is higher than that of the single model. However, due to the poor wind speed decomposition performance of the EMD-RNN model, the EWT-LSTM-Elman model cannot deal with high-frequency components well when the Elman neural network model is used. Thus, the forecasting accuracy is low, especially in January when the wind speed changes frequently and the average wind speed is high, and the MAPE of both models is in the range of 7% to 8%. By contrast, the proposed forecasting system has the highest stability and is least affected by external conditions. Although the forecasting accuracy of the CEEMDAN-ENN model can reach a high level under certain circumstances, the minimum MAPE is 4.7%, but compared with the proposed model, the RMSE is relatively divergent, and the stability is not as good as that of the proposed forecasting system.

To facilitate the observation of how much the forecasting accuracy of the proposed forecasting system has improved compared with that of other models, the forecasting result of Wakkanai is taken as an example and P_{MAE}, P_{MAPE}, and P_{RMSE} are calculated, as shown in Fig. 3.15.

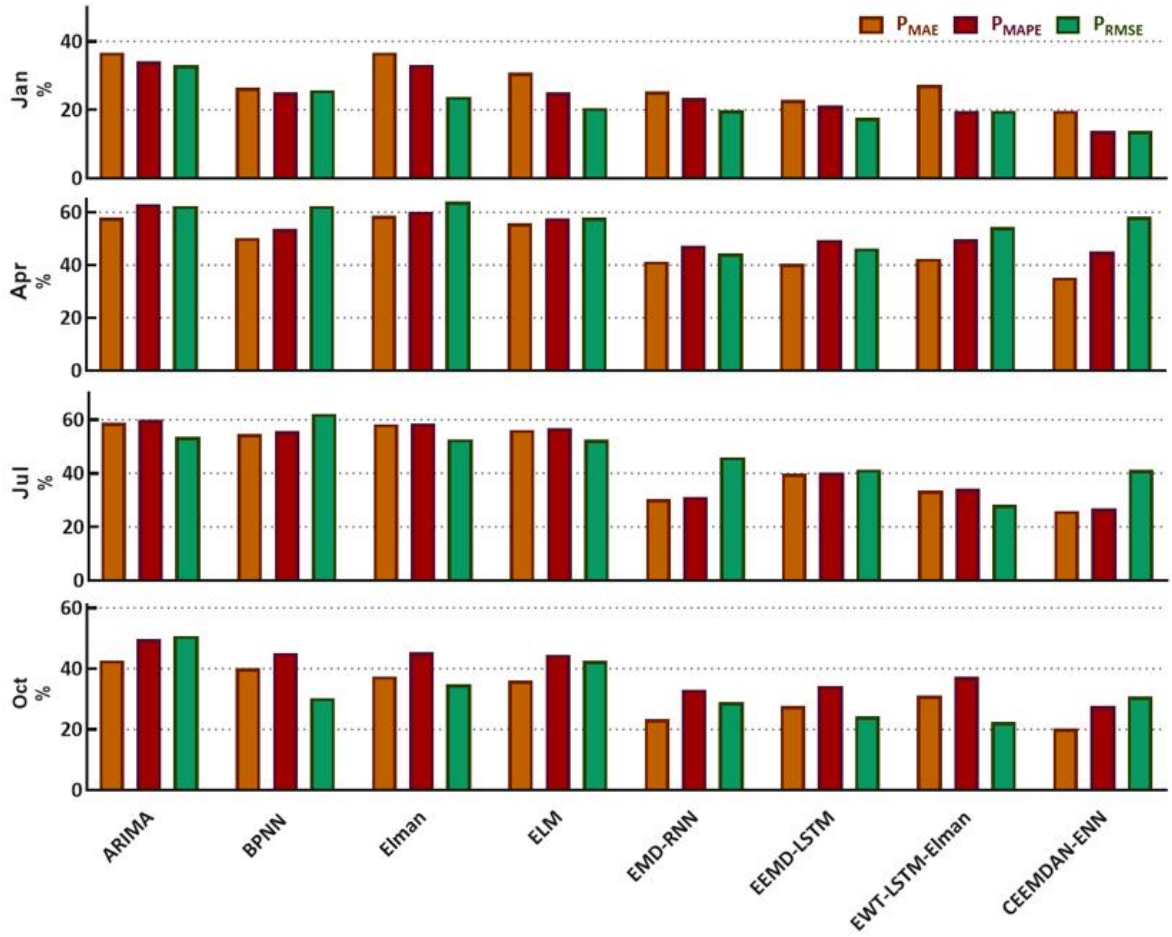


Fig. 3.15 Improved accuracy compared with other forecasting models.

(1) Compared with a single model, the proposed forecasting system has a greater improvement in P_{MAE}, P_{MAPE}, and P_{RMSE}. For example, in the wind speed forecast in January, for the ARIMA model, P_{MAE}, P_{MAPE}, and P_{RMSE} reached 36.76%, 34.12%, and 33.12%, respectively. For the BPNN model, P_{MAE}, P_{MAPE}, and P_{RMSE} reached 26.52%, 25.06%, and 25.78%, respectively.

(2) For other hybrid models, CEEMDAN-ENN has the highest forecasting accuracy, but the proposed forecasting system still improved compared with CEEMDAN-ENN. For example, in the forecast of wind speed in July, the values of P_{MAE}, P_{MAPE}, and P_{RMSE} are 25.99%, 26.82%, and 38.19%, respectively. The values of P_{MAE}, P_{MAPE}, and P_{RMSE} in the forecast of wind speed in January reached 19.71%, 13.82%, and 13.86%, respectively, indicating that the proposed method has better forecasting ability and versatility than the other hybrid forecasting models.

(3) Compared with other models, the proposed combined model improves the forecasting accuracy to a large extent, because the percentage of improvement in each season is more obvious than that of all comparison models.

Chapter 4. 24-hour ahead wind speed forecasting

Regarding the interval of wind speed forecasting, previous studies have proposed many forecasting models for ultra-short-term wind speed forecasting, which have achieved good forecasting accuracy. However, few studies have focused on short-term wind speed forecasts for 24 h. The 24-h wind speed forecast plays an important role in ensuring power system safety and regulating wind power generation.

However, unlike the ultra-short-term wind speed forecasts (10 min, 30 min, 1 h), there are more uncertainties in the future 24-hour wind speed forecasts, and it is difficult to find the wind speed variation pattern from the wind speed time series alone. Because of the large time span and long time interval, unlike in the short-term wind speed forecasting, no obvious correlation between the wind speed of the forecast target location and the surrounding locations, which means the DAD method proposed for short-term wind speed cannot be used for training data extraction, and only the data of the forecast target location can be used as training data. Meanwhile, the increase in output data makes the number of nodes in the output layer and the number of nodes in the hidden layer increase substantially, thus making the neural network more complex. Because the decomposition forecasting method is used in this study, the complexity of the neural network will greatly reduce the training efficiency of the forecasting model and also affect the accuracy of the forecasting model to some extent. Therefore, to solve the above problems, this study makes some adjustments in the data preprocessing module and the forecasting module to make the forecasting model more adaptable to the 24-hour wind speed forecasting.

4.1 Construction of the forecasting model

4.1.1 Data preprocessing module

Previous studies have shown that good data pre-processing techniques can effectively improve the accuracy of forecasting. For short-term wind speed forecasting, we use CEEMD to decompose the wind speed data, and train and forecast the neural network for the decomposed subseries separately. However, for the 24-hour wind speed forecasting, we use the cyclic forecasting method (Subsection 4.2.2), which increases the number of forecasting steps, greatly reduces the training efficiency of the neural network, and

increases the probability of errors during the training process. Therefore, for 24-hour wind speed forecasting, in the data preprocessing module, this study proposes the CEEMD combined with the PE method to process the decomposed wind speed subseries, retaining the subseries with relatively high complexity and merging the subseries with low complexity to simplify the training process.

The flow of the CEEMD-PE [77] algorithm is shown in Fig. 4.1.

(1) CEEMD decomposes the original wind speed time series to obtain IMF.

(2) The PE algorithm analyzes each IMFn component and calculates the H_p of each component. If $H_p>\theta$ (threshold value), then the complexity is high, and the original IMFn is retained; otherwise, the complexity is low, and the low-complexity IMFn is reconstructed.

(3) A set of k IMFs is generated.

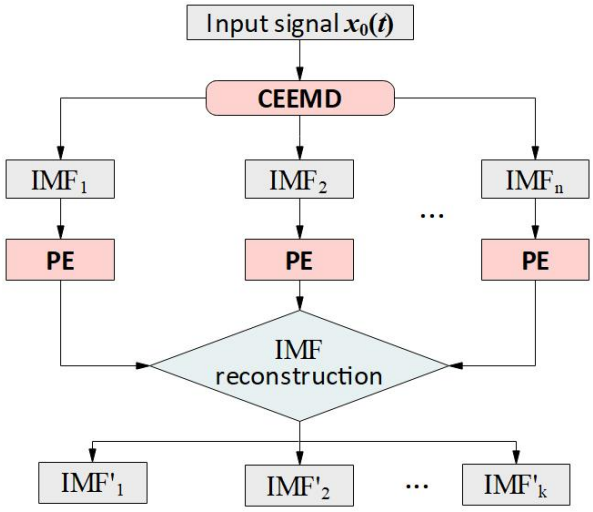


Fig. 4.1 Flow of CEEMD-PE.

4.1.2 Forecasting module

For 24-hour wind speed forecasting, the same as ultra-short-term wind speed forecasting, the LSTM network is still used as the main forecasting model. However, the large increase in the number of neurons in the neural network makes the initial parameter setting more inefficient and error-prone, and thus, a combination of two optimization algorithms, GA and ACO, is proposed in this study to determine the initial parameters of the neural network.

ACO has several shortcomings. For instance, this algorithm lacks an initial pheromone, can easily fall into local optimal solution, and has a low solution speed. Meanwhile, despite its fast global search capabilities, GA uses less feedback information in

the system and has low solution efficiency. Therefore, in this article, we fuse these two algorithms. Specifically, the rapid search ability and global convergence characteristics of GA are used to generate an initial pheromone distribution, and then the parallelism, positive feedback mechanism, and high solving efficiency of ACO are used to optimize the internal initial parameters of RNN. This hybrid algorithm outperforms GA in terms of solving efficiency and ACO in terms of time efficiency. We call this hybrid algorithm ACO-GA hereinafter [77].

The specific process of ACO-GA is as follows and illustrated in Fig. 4.2:

(1) The GA coding is selected.

(2) The initial population is generated.

(3) Selection, crossover, and mutation operations are performed.

(4) The optimal population is generated.

(5) According to the optimal population generated in step (4), the initial distribution of the pheromone is formed, the initial parameters of ACO are set, and the Tabu list for each ant is established.

(6) The selection probability of each ant is calculated, and the residual pheromone is updated.

(7) The pheromone is partially updated. After n times, m ants traverse all nodes (i.e., complete a cycle), and then perform a cycle on each path.

(8) Steps (6) and (7) are repeated until all ants have constructed a path, and the pheromone is updated globally.

(9) The Tabu list of all ants is cleared, and steps (6) to (8) are repeated until the set number of cycles is reached or when the termination condition is met.

(10) The termination conditions are assessed, and the result is outputted.

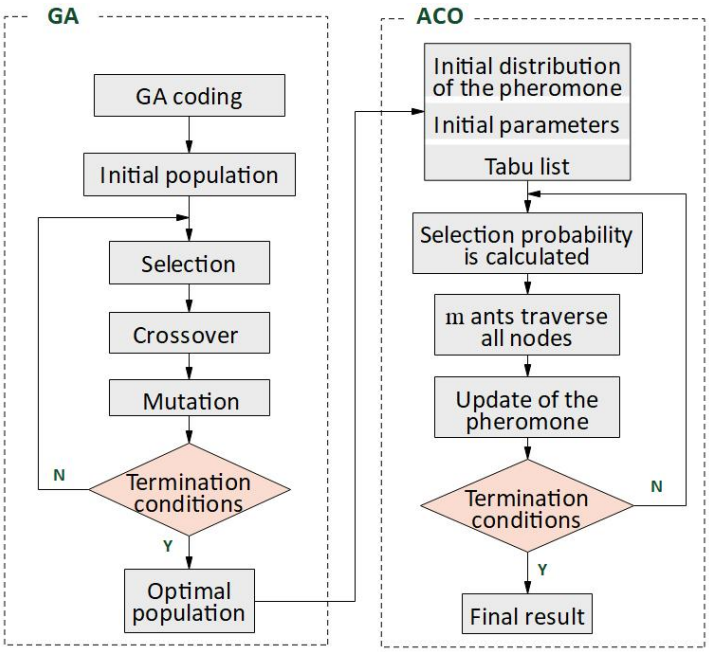


Fig. 4.2 Flowchart of ACO-GA.

We use ACO-GA to find the optimal initial weight and initial threshold. The specific method is illustrated in Fig. 4.3. ACO-GA uniformly divides the domain of weights and thresholds and divides them into several uniform sub-regions. The points on the boundary of these sub-regions are equivalent to the candidate weights. At the initial moment, the pheromone content of each boundary point is the same. Each ant passes through each weighted sub-area only once and records the corresponding label. These labels represent a combination of sub-regions, which are equivalent to a set of weights and domain values of the BP neural network. The error value is obtained according to the output sample, and the pheromone is updated according to the size of the error value.

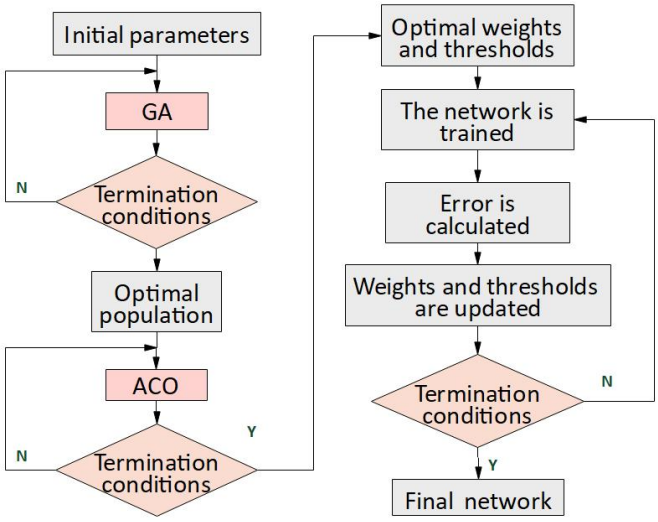


Fig. 4.3 Flow of ACO-GA-LSTM.

4.2 Forecast preparation

4.2.1 Data description

We chose Wakkanai, Matsumae, and Nemuro, which are rich in wind energy resources as shown in Fig. 4.4, as our target forecasting locations. The wind speed data in these locations as published on the official website of the Japan Meteorological Agency were used as training and testing data. We trained and evaluated our proposed hybrid forecasting model based on these data.

Fig. 4.4 Distribution of forecast target locations.

To evaluate the versatility of our hybrid forecasting model, we chose the wind speeds recorded in 2018 and 2019 as our data basis. Given that winters in Hokkaido are relatively long, we chose January, April, July, and October 2019 as our forecasting periods and the hourly average 24 h-ahead wind speed as our forecast target. We then assessed the forecasting accuracy of our hybrid forecasting model for Wakkanai, Matsumae, and Nemuro. The databases and testing data selection methods for each forecast target period are shown in Table 4.1.

Table 4.1 Principles of data selection.

Target time period	Data Base Time Period	Number of data	Testing data Time Period	Number of data

Jan	2018.10.3~ 2019.1.31	2904	2019.1.1~ 2019.1.31	744
Apr	2019.1.1~ 2019.4.30	2880	2019.4.1~ 2019.4.30	720
Jul	2019.4.2~ 2019.7.31	2904	2019.7.1~ 2019.7.31	744
Oct	2019.7.3~ 2019.10.31	2904	2019.10.1~ 2019.10.31	744

Given that the proposed hybrid forecasting model has a multi-step forecasting format, the data used for training this model slightly differ at each step. The training data selection method is illustrated with the January 2019 forecast as an example. When the forecast target date is set to January 2, 2019, the training data are selected as shown in Fig. 4.5, where n is the number of forecast steps, which is set to 6 in this paper as will be specified in subsection 4.2.2. The data 90 days prior the forecasting period are selected for the training data in consideration of their timeliness and training efficiency. We try to ensure the adequacy of the data, and eventually determine via trial and error that these 90-day data are the most suitable for this study.

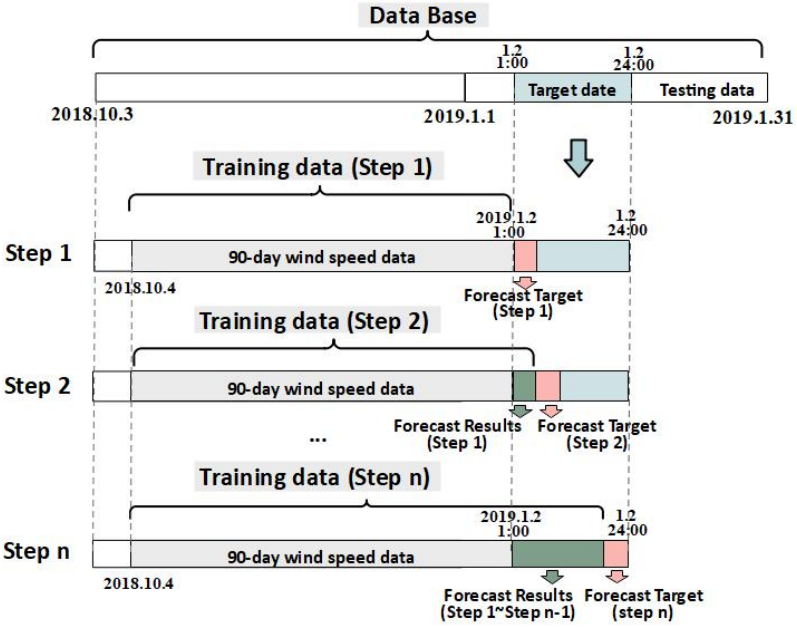


Fig. 4.5 Training data extraction method

The forecasting models employed in each step of the forecasting process are independent of one another and need to be trained and forecasted separately. The specific process is described as follows:

Step 1: The training data for the current step (Fig. 4.5) are decomposed by using CEEMD-PE, and the neural network is trained with each component of the decomposition. The forecasting model for this step is eventually obtained.

Step 2: The dataset containing the forecasting results and training data for Step 1 are used in this step for training and forecasting.

Step 3: Step 2 is repeated until the wind speed is forecasted for 24 hours.

Following the abovementioned data selection principle, we selected the training and verification data to be forecasted for the aforementioned locations and then forecasted the wind speed in 2019.

4.2.2 Parameter setting of the model

CEEMD: Using the method in subsection 3.2.2, the effect of different *NE* and *Nstd* values on forecast accuracy is investigated in order to achieve the best decomposition. By taking Wakkanai as an example for illustration, we used our proposed forecasting model with different *NE* and *Nstd* values to forecast the 24-hour wind speed for January, April, July, and October 2018 and then calculate the average 24-hour MAPE. The results are shown in Table 4.2.

Table 4.2 The average MAPE corresponding to different settings of *NE* and *Nstd.*

MAPE(%)		*Nstd*				
	NE	0.1	0.2	0.3	0.4	0.5
	100	35.24	34.53	35.08	34.71	35.33
	200	36.10	35.53	35.72	33.51	35.64
Jan	300	35.84	32.49	32.79	32.74	32.56
	400	34.64	32.41	32.71	32.92	32.61
	500	33.97	32.41	32.17	32.38	32.77
	600	35.76	36.42	34.62	35.02	36.12
	100	33.99	35.46	35.11	34.04	35.18
	200	34.54	34.17	34.19	33.61	34.43
Apr	300	34.29	33.01	32.19	32.24	32.99
	400	34.12	32.01	31.96	31.87	32.04
	500	34.69	31.68	31.03	31.31	30.83
	600	34.68	35.12	33.86	34.07	35.21
	100	34.58	34.81	33.91	33.61	35.05
	200	34.86	34.21	34.21	33.68	33.74
Jul	300	33.58	32.45	32.46	31.77	31.67
	400	33.97	31.92	30.44	30.22	31.93
	500	33.39	31.16	30.23	30.97	30.82
	600	33.72	33.88	33.41	33.43	35.01
Oct	100	35.74	34.66	34.63	34.31	34.81

200	34.42	34.75	33.43	33.66	33.71
300	34.23	32.14	32.83	32.64	33.21
400	35.21	32.29	31.95	32.22	31.35
500	34.13	32.68	31.72	31.56	31.07
600	35.77	34.86	33.33	33.63	35.77

As shown in Table 4.2, even if the forecast target period is set to different months, the best forecasting results are obtained when the *NE* value ranges between 400 and 500 and when the *Nstd* value is set to either 0.3 or 0.4. Therefore, we combined the actual decomposition efficiency and eventually set *NE* and *Nstd* to 500 and 0.3, respectively.

PE: The key parameters in the PE algorithm include time series length N, time delay t, and embedding dimension m. Among these parameters, m ranged from 4 to 8. If the value of m is too small, then the reconstructed signal loses the main information and the algorithm loses its meaning; otherwise, the efficiency of the algorithm is greatly reduced, and the algorithm becomes unable to reflect the changes in the details of the signal. Following Feng et al.[61],[62], to obtain a stable PE value, we performed a trial and error and set m to 4 and t to 3. Setting a different threshold value θ leads to different reconstruction results. Based on the theory of other researchers[63] and by considering the effect of reconstruction, we finally set θ to 0.7.

The decomposition and reconstruction results obtained by taking the wind speed time series of Wakkanai from January to March 2019 as example are shown in Fig. 4.6. By reconstructing the PE algorithm, the simple IMFs were merged, whereas the feature quantity in a single IMF was increased to facilitate feature extraction. The reconstruction also reduced the number of IMFs and the amount of consumed computing resources, thereby greatly improving forecasting efficiency.

ACO-GA: Setting the parameters of ACO and GA is a critical step, but a fixed method for parameter setting is unavailable. Moreover, different parameters must be set to deal with various problems. Therefore, in this article, we drew on the findings of other scholars[26], [48], [49] and used trial and error to finalize these parameters in the forecasting process. The parameters were set as shown in Table 4.3.

Table 4.3 Parameter setting of ACO and GA.

ACO		**GA**	
The number of ants	60	Population	80
Maximum number of iterations	500	Maximum number of iterations	500
The information heuristic factor α	1	Crossover probability	0.8
The expected heuristic factor β	1.5	Mutation probability	0.05
The volatilization coefficient ρ	0.6		

The intensity of information Q	100

LSTM: In LSTM, input_size is the number of input layer nodes, lstm_size is the number of nodes in each hidden layer, and output_size is the number of output layer nodes. We must first determine input_size and output_size. The setting of input_size and output_size will directly affect the forecasting efficiency (the forecasting will be divided into several steps) and accuracy.

The principle for setting input_size has been explained in detail in our previous paper[75]. In sum, while considering the continuity of the time series, the wind speed data showing a high correlation with the wind speed time series to be forecasted are preferred. The specific method is briefly illustrated by using the wind speed time series of Wakkanai for October–December 2018. V_t is assumed to be the wind speed time series to be forecasted, V_{t-i} is the wind speed time series before hour i of the forecast time point, and V_{t+i} is the wind speed time series after hour i of the forecast period. The Pearson correlation coefficients between V_t and each time series are shown in Table 4.4.

Table 4.4 Correlation coefficient between each wind speed time series.

Correlation coefficient	V_t	V_{t+1}	V_{t+2}	V_{t+3}	V_{t+4}
V_{t-1}	0.6871	0.6632	0.5122	0.3875	0.3133
V_{t-2}	0.6502	0.6391	0.5020	0.3644	0.2919
V_{t-3}	0.6176	0.5977	0.4712	0.3511	0.2804
V_{t-4}	0.5144	0.5125	0.4418	0.3241	0.2553
V_{t-5}	0.4923	0.4601	0.4482	0.3406	0.2414
V_{t-6}	0.4279	0.4331	0.3914	0.3102	0.2591
V_{t-7}	0.3672	0.3402	0.3104	0.2892	0.2302
V_{t-8}	0.3719	0.3413	0.3032	0.2812	0.2472
V_{t-9}	0.3420	0.3210	0.3041	0.2612	0.2227
V_{t-10}	0.3471	0.3122	0.2981	0.2531	0.1892
V_{t-11}	0.3345	0.3010	0.2922	0.2782	0.1421
V_{t-12}	0.3102	0.2891	0.2698	0.2413	0.2007
V_{t-13}	0.2881	0.2803	0.2485	0.2124	0.1372

The above table shows that the correlation between each wind speed time series V_{t-1}–V_{t-13} and V_t gradually decreases as the time point moves away from the time point to be forecasted. The significant correlation between V_{t-13} and V_t disappears at the node 13 hours before the time point to be forecasted. The correlation of each wind speed time series after the forecast time point V_{t+1}–V_{t+4} with the wind speed time series before the forecast time point V_{t-1}–V_{t-13} also significantly decreases along with an increasing time spacing. Meanwhile, at the node 4 hours after the forecast time point, the correlation between these time series and V_{t+4} disappears. Therefore, the wind speed data within 13 hours before the

forecast target period should be prioritized when selecting the input data. When forecasting the wind speed for the next 24 hours, the first 5 hours of the forecast are relatively reliable.

The method for selecting the input and output data of the proposed forecasting model is shown in Fig. 4.7, where n_1 and n_2 denote input_size and output_size, respectively. Given that our forecasting target is the hourly 24 h-ahead average wind speed, we must perform multiple cycle forecasts when n_2 is less than 24. To consider computational efficiency and forecast accuracy and combined with the above analysis regarding the correlation between the wind speed time series, n_1 and n_2 are eventually set to 12 and 4, respectively. In this case, the wind speed forecast for 24 hours needs to be carried out in six steps.

For setting the number of hidden layer nodes, given the lack of a unified setting standard, we used a single-layer hidden layer structure following our previous experience and then calculated the number of hidden layer nodes in the interval [75] by using the empirical formula shown in Equation (4-1)[64]–[66]. Afterward, we combined the forecast accuracy of each component after decomposing wind speed with overall calculation efficiency, and we eventually determined the number of hidden layer nodes to be 20.

$$N_h = \frac{N_s}{\alpha \cdot (N_i + N_o)} \qquad (4-1)$$

where N_i is the number of nodes in the input layer, N_o is the number of nodes in the output layer, N_s is the number of samples in the training set, and α is an arbitrary value variable that may range from 2 to 10 [67],[68].

The other related parameters are set as shown in Table 4.5.

When using LSTM for the numerical simulation, the initial values of the weights and thresholds in the network should be set. The setting of these initial values directly affects the network training efficiency. We used the ACO-GA algorithm to select the weights and thresholds of LSTM. The process is described in detail in section 4.12.

Table 4.5 Initial parameter setting of LSTM.

Parameters	value
batch_size	120
epochs	50
dropout	0.2
lstm_size	20

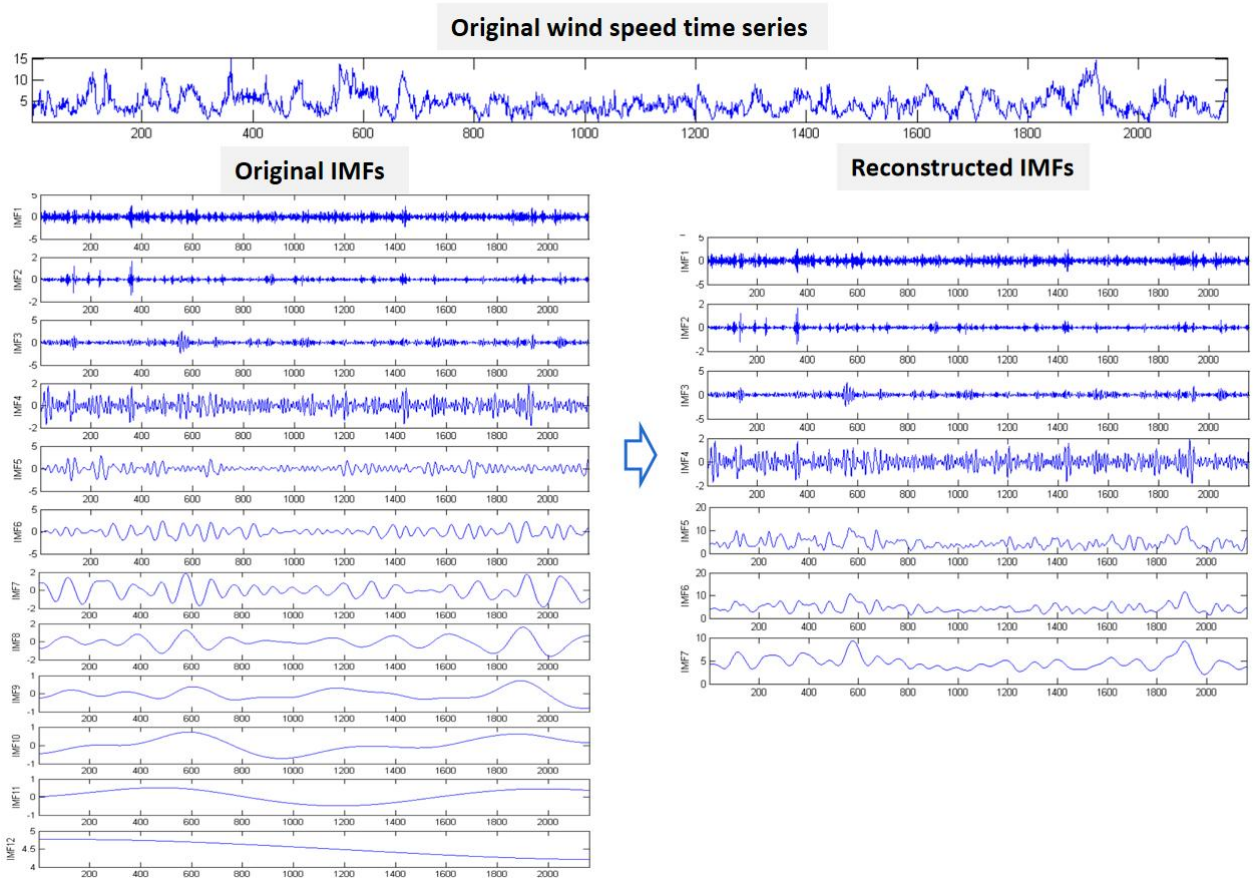


Fig. 4.6 Decomposition and reconstruction of the wind speed time series of Wakkanai from January to March in 2019.

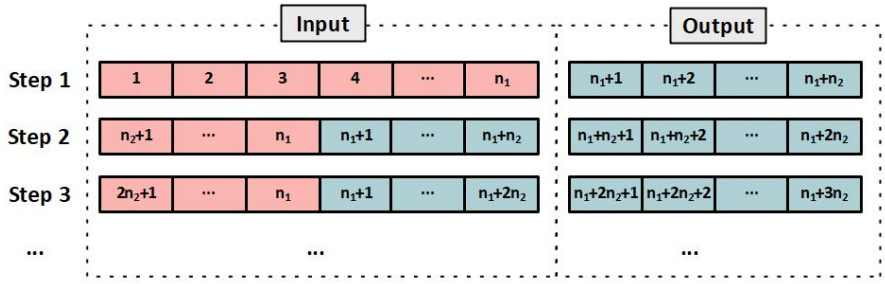


Fig. 4.7 The selection of input and output data.

4.3 Validation of the proposed method by numerical calculations

To analyze the versatility and practicality of our proposed hybrid forecasting model, we conducted some experiments to test its forecasting performance across different conditions. We selected different types of forecasting models and used the same historical wind speed data as verification data to further evaluate forecasting accuracy.

4.3.1 Experiment I: Forecasting wind speed in four seasons across different locations

We used Wakkanai, Matsumae, and Nemuro as our forecast locations, 24 h-ahead hourly average wind speed as our forecast target, and January, April, July, and October 2019 as our forecast periods. The wind speed characteristics across different seasons are shown in Table 4.6. We used RMSE, MAE, and MAPE of our forecast results to calculate the average error obtained at each step as shown in Fig. 4.8 and Table A.5.

Table 4.6 Characteristics of wind speed for each season across the three locations.

Month	Location	Max (m/s)	Min (m/s)	Mean (m/s)	Standard Deviation (m/s)
Jan	1	18.4	0.3	6.17	2.98
	2	15.2	0.1	6.81	3.39
	3	18.2	0.5	6.37	3.17
Apr	1	14.9	0.2	4.68	2.64
	2	11.1	0.1	4.16	2.32
	3	15.6	0.6	5.02	2.22
Jul	1	10.4	0.1	3.78	1.91
	2	10.6	0.1	3.55	1.38
	3	9.4	0.3	3.69	1.69
Oct	1	16.7	0.6	5.61	2.81
	2	14.8	0.2	5.53	2.86
	3	17.8	0.5	5.56	2.64

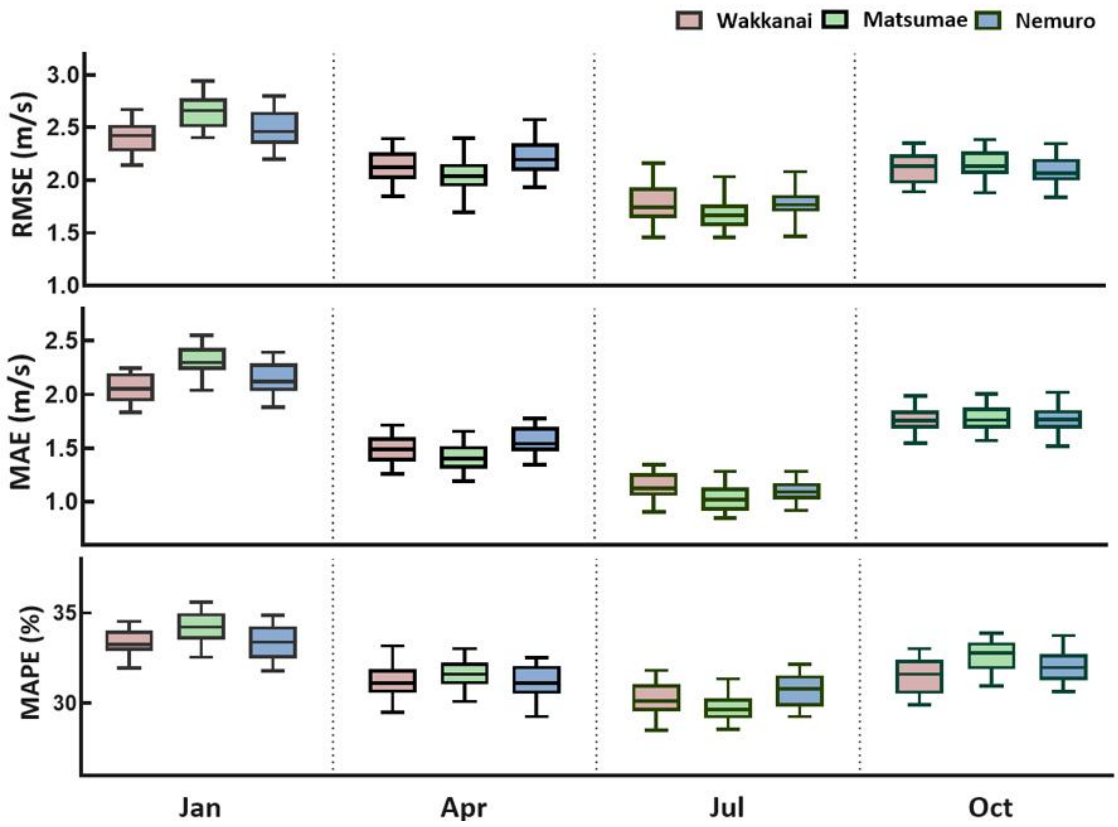


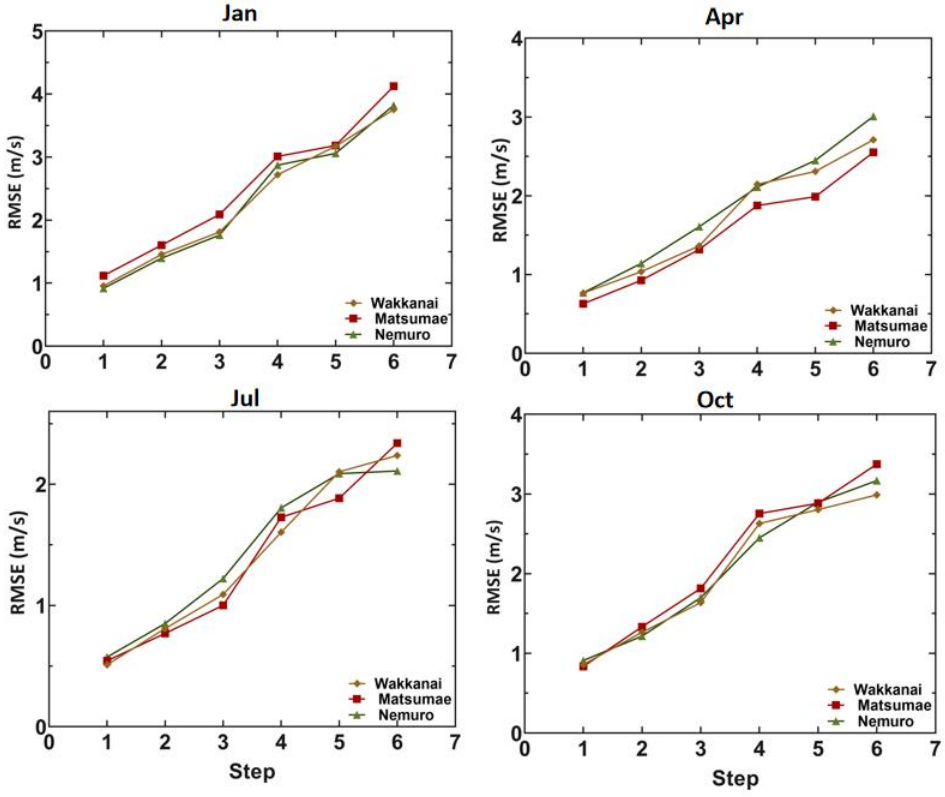
Fig. 4.8 Distribution of errors for each season across the three locations.

Fig. 4.8 shows that for the 24 h-ahead wind speed forecast, RMSE stays between 1.5 and 3.0, whereas MAE ranges from 1.0 to 2.5. While these values change along with seasons, the overall fluctuation is small. The maximum MAE and RMSE values are 2.58 and 2.93, respectively, both of which are observed in the January forecast for Matsumae. Meanwhile, the minimum MAE and RMSE values are 0.82 and 1.46, respectively, both of which are observed in the July forecast for Matsumae. In sum, the forecast results for January and October are relatively poor as reflected in the high MAE and RMSE values recorded during these periods. July achieves the best forecast given the low MAE and RMSE values obtained across the three locations during this period. The forecast accuracy for April lies in between. Based on the changes in MAE and RMSE, we preliminarily concluded that our proposed forecasting system can effectively forecast the wind speed for each season and that the stability of its forecasts does not significantly change along with the seasons. Fig. 4.8 shows some seasonal variations in MAPE, MAE, and RMSE. The minimum MAPE value (28% to 32%) is recorded in July. The forecasting accuracy for April is unstable with a MAPE ranging from 29% to 35%. Meanwhile, the forecasting accuracy for January and October is relatively poor with a MAPE ranging from 29% to 36%.

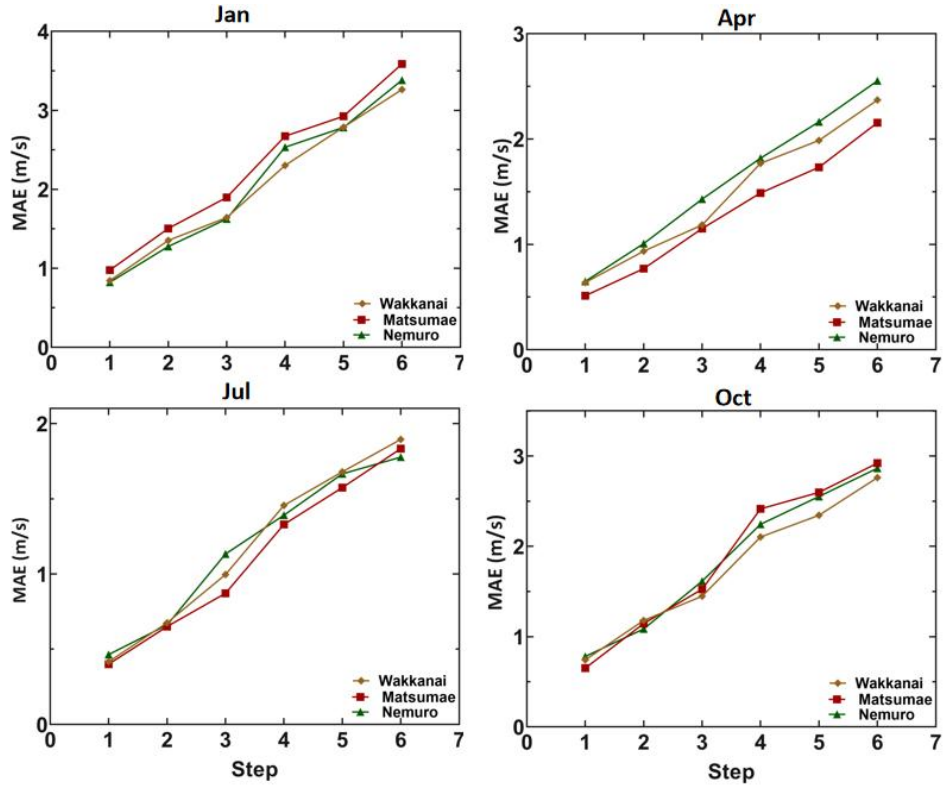
Table A.5 and Fig. 4.9 show that the forecasting accuracy decreases in each forecasting step. In the first two steps, MAPE ranges between 11% and 21%, whereas

MAE and RMSE range between 0.4 and 1.6. These two sets of values are relatively close, thereby suggesting a relatively stable forecast and good forecasting effect. However, MAPE significantly increases in the subsequent forecasting steps. In the fifth and sixth steps, MAPE ranges between 42% and 53% and peaks at 53.22%. Meanwhile, MAE and RMSE range from 1.5 to 3.6 and from 1.8 to 4.2, respectively. Therefore, the stability and forecasting accuracy of our hybrid forecasting model are reduced in the next few forecasting steps, thereby explaining the low overall accuracy of its 24 h-ahead wind speed forecasting.

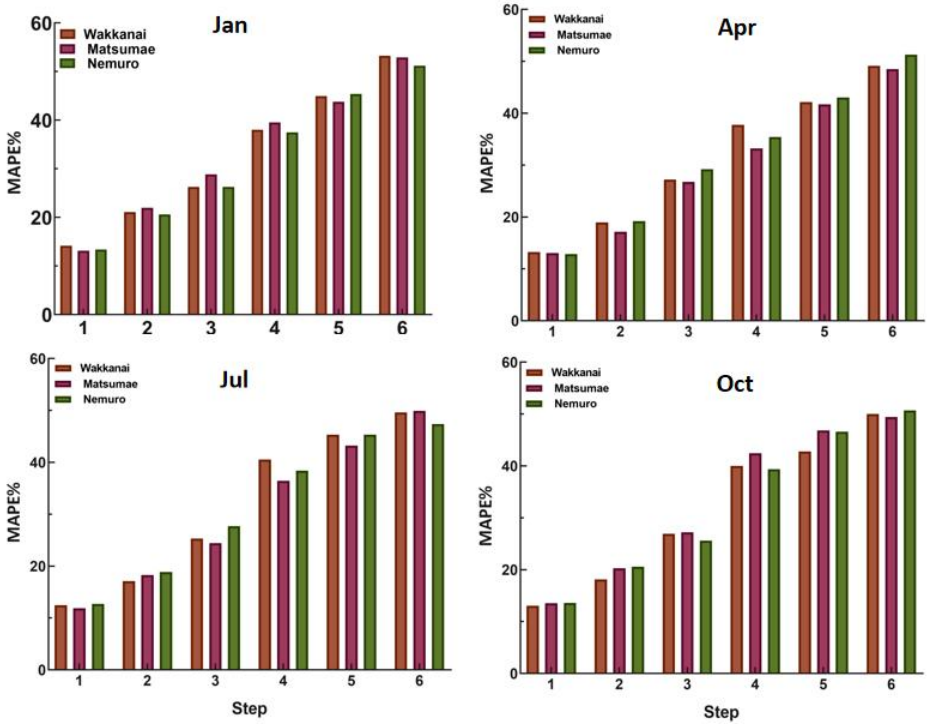
Table A.5 and Fig. 4.9 show the changes in the forecasting accuracy of our model across different seasons. The forecasting effect for January and October, during which large changes in wind speed are recorded, is relatively poor, whereas the forecasting effect for April and July, during which small changes in wind speed are observed, is relatively good. Small deviations are also observed in the forecasting results across different locations, thereby suggesting that the accuracy of our hybrid forecasting model is not significantly affected by forecasting location.



(a)


Jan
Apr
Jul
Oct
MAE (m/s)
Step
Wakkanai
Matsumae
Nemuro

(b)


Jan
Apr
Jul
Oct
MAPE%
Step
Wakkanai
Matsumae
Nemuro

(c)

Fig. 4.9 Comparison of forecasting errors for each season across the three locations.

4.3.2 Experiment II: Comparison of the accuracy of different forecasting models at each hour ahead

We then examined the forecasting accuracy of our proposed system at different times over the next 24 h. In the previous experiment, we found that in terms of 24 h-ahead wind speed forecasting, our proposed system has better universality compared with other forecasting models and that its forecasting results are only related to wind conditions and are not affected by geographic locations. Therefore, in Experiment II, we chose Wakkanai as our forecast target and January, which has a lower forecast accuracy compared with the other months, as our forecast period. We also compared several typical single forecasting models, such as ARIMA[54]–[56],[69], BPNN[35], and Elman[57], as well as some widely used mixed models, such as EMD-RNN, EEMD-LSTM, and EWT-LSTM [59], with our forecasting system. The parameter setting for these models is shown in Table 4.7. Those parameters whose values were not mentioned in the literature were set to default.

Table 4.7 Experimental parameter setting for different models.

Model	Parameter	Value
ARIMA	Autoregressive term (p)	4
	Moving average number (q)	5
	Difference times (d)	1
BPNN	Maximum number of iteration times	500
	Learning rate	0.01
	Training accuracy goal	0.005
	Number of input layer nodes	6
	Number of hidden layer nodes	12
	Number of output layer nodes	1
Elman	Maximum number of iteration times	500
	Number of input layer nodes	12
	Number of hidden layer nodes	20
	Number of output layer nodes	4
EEMD	Ensemble number (*NE*)	500
	Amplitude of white noise (*Nstd*)	0.3
EWT	Expected number of the filter bank	6

We calculated the hourly average and 24-hour total average along with the MAE, MAPE, and RMSE values of the forecasts and then calculated the improvement percentages P_{RMSE}, P_{MAE}, and P_{MAPE} of our hybrid forecasting model compared with the

other models. Fig. 4.10 compares the forecasting accuracy and percentage improvement of all forecasting models for each hour, whereas Table A.6 and Fig. 4.11 compare their 24-hour forecasting accuracy and percentage improvement.

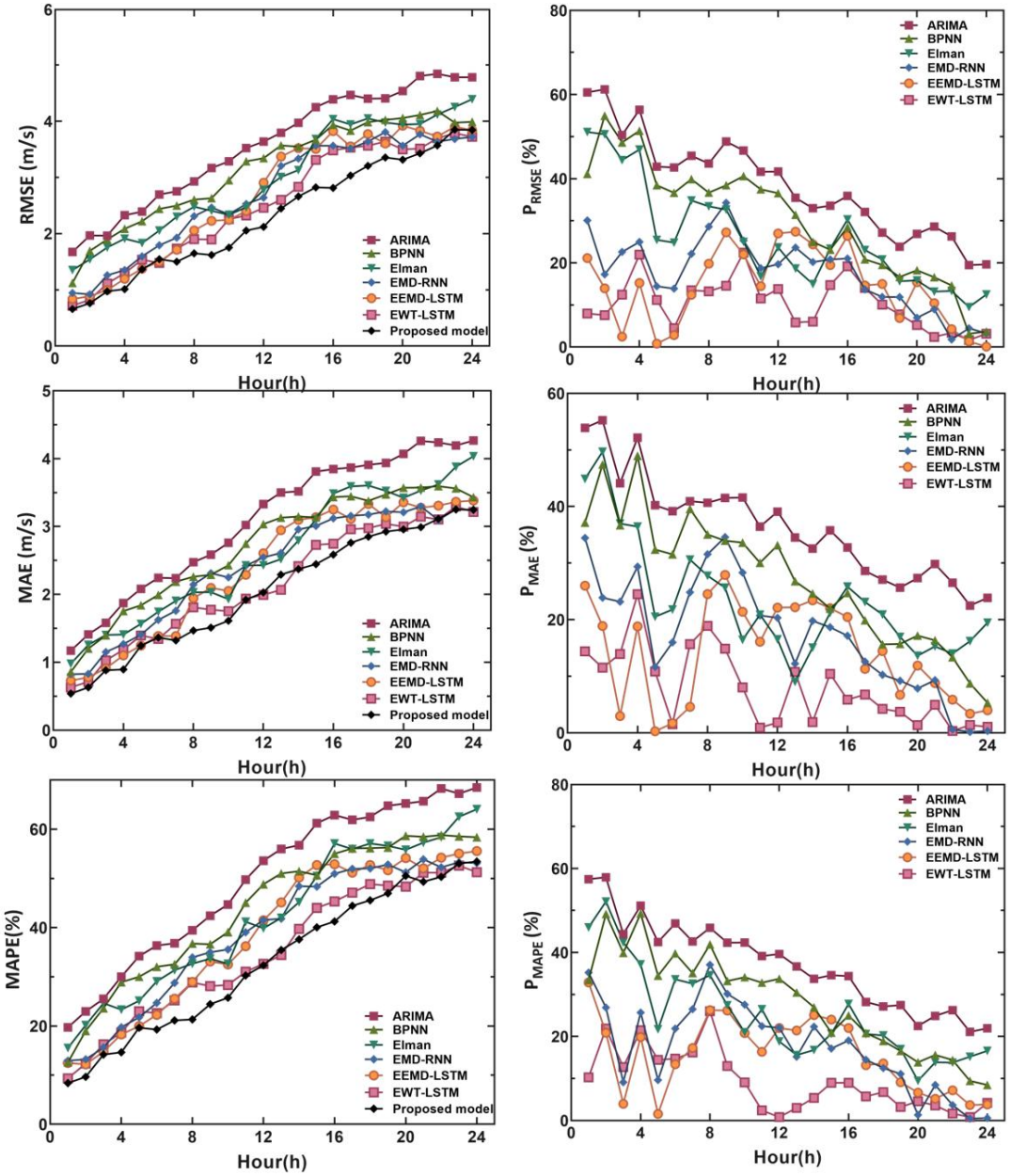


Fig. 4.10 Error distribution of different forecasting models in each hour ahead.

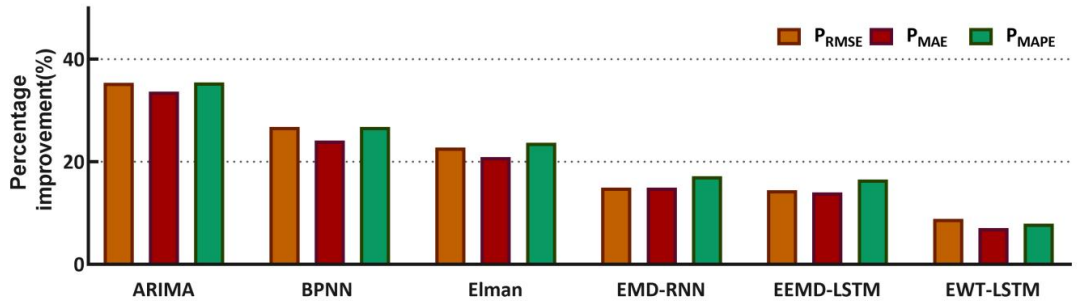


Fig. 4.11 Percentage improvement in the accuracy of 24-hour wind speed forecasting.

(1) The changes in MAE and MAPE indicate that our hybrid forecasting system generally has higher accuracy than any single forecasting model, whereas the changes in RMSE indicate that our model has better stability than the other models.

(2) Each forecasting model shows a decreasing accuracy from 1 h-ahead to 24 h-ahead. Although each model achieved a good forecasting effect for the first 8 hours, compared with the other models, our hybrid forecasting model achieves the highest forecasting accuracy for the first 7 to 8 hours with a MAPE of below 20%. By taking 8 h-ahead as an example, the P_{RMSE}, P_{MAE}, and P_{MAPE} of ARIMA are 43.63%, 40.67%, and 42.64%, respectively, which are 13.25%, 18.91%, and 16.23% of EWT-LSTM.

(3) After the first 8 hours, the EEMD-LSTM and EWT-LSTM models outperformed our proposed model in terms of accuracy. However, Table A.6 and Fig. 4.11 show that overall, our model has a higher forecasting accuracy than the other models.

(4) Fig. 4.10 show that RMSE increases along with forecasting time, but compared with those of the other models, the RMSE of our proposed model stays small for each period. From 1 h-ahead to 8 h-ahead, the RMSE of our hybrid forecasting model changes between 0.6 and 1.5, whereas from 9 h-ahead to 16 h-ahead, this value only changes between 1.5 and 2.8. These ranges are smaller than those observed for the other forecasting models. However, from 17 h-ahead to 24 h-ahead, the RMSE of our proposed model changes between 3.0 and 4.0, but this range is still smaller than those observed for the other models. In sum, our proposed model is more stable than the other forecasting models.

Chapter 5. 1-hour ahead wind power forecasting

The size of wind power is influenced by wind speed, geography, wind turbine type, and related parameters, etc. It is difficult to have an accurate grasp of the specific wind power at a certain moment in the future, which is the main reason why large-scale wind power grid connections will have a negative effect on the stable operation of the power grid. Therefore, make wind power forecasting for wind farms is important. High-accuracy wind power forecasting helps the power system make reasonable dispatching decisions to ensure the safe and stable operation of the power grid, and also helps wind farms to arrange equipment maintenance reasonably to ensure the economic benefits of wind power generation.

There are two main methods of wind power forecasting, one is based on power curve forecasting and the other is direct forecasting of output power. In this chapter, for ultra-short-term wind power forecasting after one hour, we propose a forecasting method based on the power curve, a direct forecasting method for output power, and a hybrid forecasting method by combining the previously proposed ultra-short-term wind speed forecasting model and using the core forecasting methods used in the wind speed forecasting model. The forecasting accuracy of the proposed wind power forecasting model is also verified using the wind power data provided on the official website of an experimental wind farm named Sotavento in Spain.

5.1 Forecasting model based on wind speed forecasting

Because wind speed forecasting models with high accuracy have been proposed, the method of calculating the forecast wind power directly using the forecast wind speed is considered first when constructing the wind power forecasting model. The calculation of wind power from wind speed is carried out using wind speed-power characteristic curves, and the standard wind speed-power characteristic curves are often provided by the manufacturers of wind turbines, which are obtained under specific experimental operating conditions.

The cut-in wind speed is 2 m/s, the rated wind speed is 9 m/s, and the cut-out wind speed is 25 m/s. The relationship between wind speed and wind power generation when the wind speed is in the range of 2–9 m/s is shown in Equation (5-1), where Cp (=0.593) is the

Betz coefficient, S is the contact area between the windmill blade and the air (m^2), ρ is the air density (kg/m^3), and V is the wind speed (m/s). Although different wind turbines have dissimilar contact areas S between wind turbine blades and air, the S of the wind turbines at the locations to be forecasted in this paper is assumed to be the same to facilitate the evaluation.[70][71]

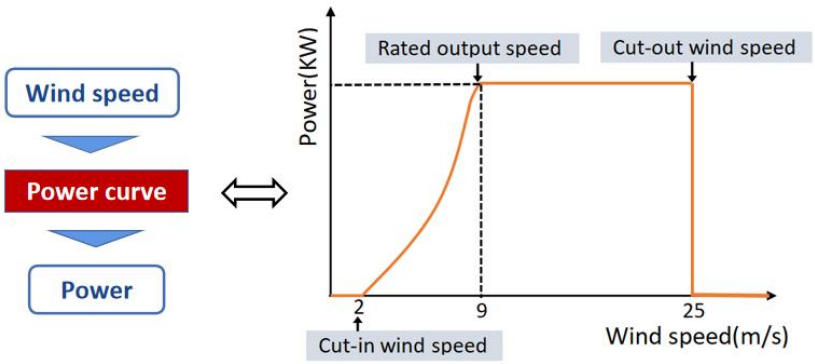


Fig. 5.1 Power curve of the more common wind turbines.

$$P = \frac{1}{2} C_p (\rho S) v^3 \qquad (5-1)$$

Based on this wind speed-power characteristic curve, we propose a wind power forecasting model based on wind speed forecasting. The specific process is shown in Fig. 5.2.

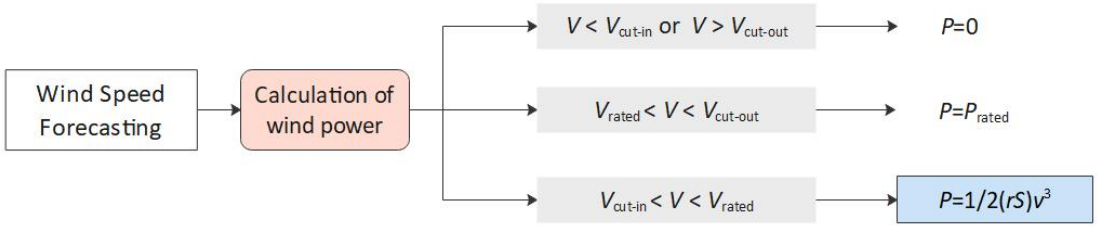


Fig. 5.2 Flowchart of wind power forecasting based on wind speed forecasting.

(1) Predict the wind speed at the time of Forecasting

(2) Determine the magnitude of the wind speed:

- If $V < V_{\text{cut-in}}$ or $V > V_{\text{cut-out}}$, P is set to 0 ;
- If $V_{\text{rated}} < V < V_{\text{cut-out}}$, P is set to P_{rated} ;
- If $V_{\text{cut-in}} < V < V_{\text{rated}}$, P uses Equation (5-1) to calculate the wind power.

The wind speed forecasting method used here is the ultra short-term wind speed

forecasting model proposed in chapter 3. Because the wind power forecasting target is a Sotavento wind farm in Spain, only the wind speed and wind power data of the plant are available from its official website but not the wind speed data of the surrounding area, the short-term wind speed forecasting model proposed in chapter 3 is used without the DAD method, and only the historical wind speed data of the plant are used as training data and validation data to train the forecasting model.

However, the actual operating wind farm is affected by the topography, unit distribution location, and wake effect, which makes the operation of each unit more complicated, i.e., the actual operating environment of wind turbines is significantly different from the design environment. Fig. 5.3 shows the comparison between the standard wind speed-power characteristic curve and the actual wind power. It can be seen that the actual wind speed and power pairs operate in a wider interval and are not in a simple one-to-one correspondence. This result indicates that if the standard wind speed-power characteristic curve is directly used for wind power forecasting, there will be some forecasting error, and therefore, it is necessary to further optimize the forecasting model to improve the forecasting accuracy.

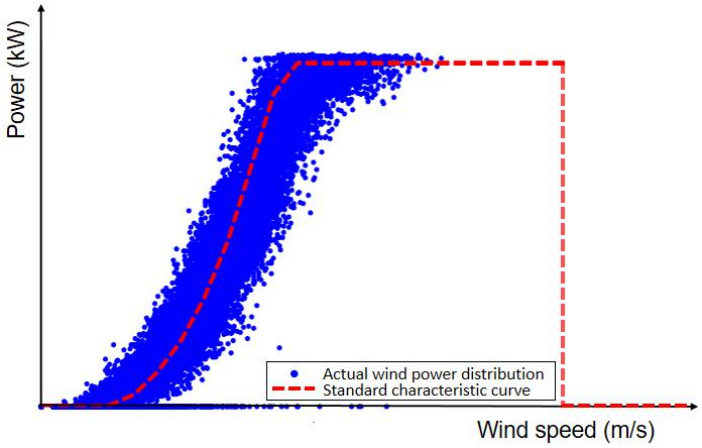


Fig. 5.3 Comparison of actual wind power and standard characteristic curves.

5.2 Direct forecasting model of wind power

5.2.1 Forecasting model based on LSTM network

In the section of wind speed forecasting model construction, it has been proved that LSTM neural network has a good forecasting effect for time series forecasting, and by combining with GA algorithm, it can improve the training efficiency of neural network and enhance the forecasting accuracy to some extent. Therefore, in this section, the LSTM

neural network optimized with GA used in wind speed forecasting is still followed for wind power forecasting. Since the wind power forecasting in this study is the hourly average wind power after one hour, the time scale is the same as the wind speed forecasting after one hour in the short-term wind speed forecasting in section 3.1, and so the parameter setting method mentioned in section 3.1 (3.2.2) is used in the parameter setting. To make the forecasting model suitable for wind power forecasting, some adjustments are made in the selection of input data and the processing of training data for the network based on the forecasting model mentioned in section 3.1.

5.2.1.1 Selection of input data

Unlike wind speed forecasting, wind speed and wind power data can be used as input data for wind power forecasting because of the high correlation between wind speed and its corresponding wind power data. Therefore, finding wind speed and wind power time series that exhibit a high correlation with the wind power time series to be forecast is the key to determine the composition of the input data.

The method of determining input data items in subsection 4.2.2 is used to determine the composition of the input data. The time series of wind speed and the wind power for the Sotavento wind farm from January to February 2020 are used as an example to briefly explain the specific method. It is assumed that P_t is the time series of wind power to be forecasted, and P_{t-i} and V_{t-i} are the time series of wind power and wind speed before the ith hour of the forecast time point, respectively. the Pearson correlation coefficients between P_t and each time series are shown in Table 5.1.

Table 5.1 Correlation coefficient between wind power series and each time series.

Correlation coefficient i	P_{t-i}	V_{t-i}
1	0.7834	0.6589
2	0.6916	0.5874
3	0.6589	0.4212
4	0.4989	0.3411
5	0.3012	0.2127
6	0.1932	0.1159

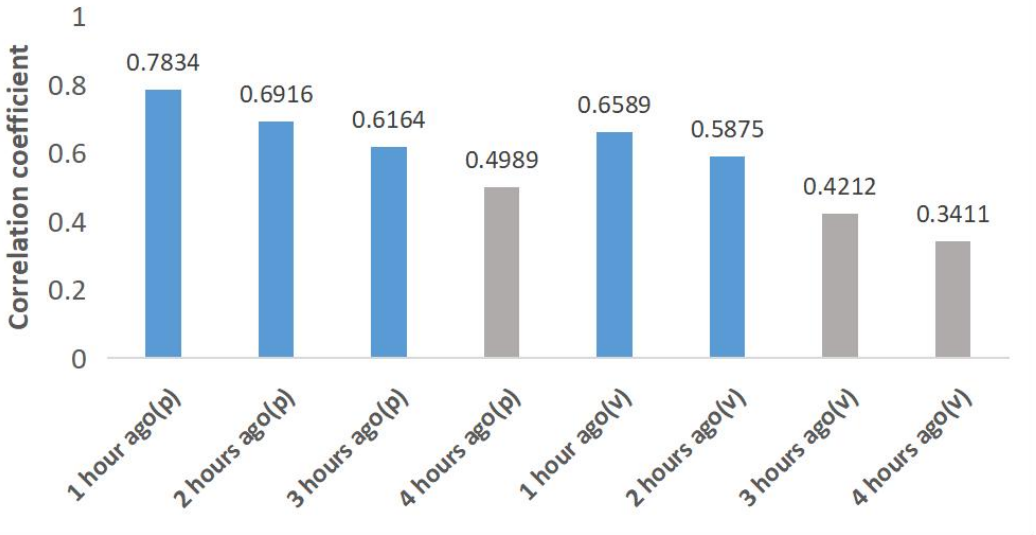


Fig. 5.4 Correlation coefficient between wind power series and each time series.

The figure shows that a correlation between P_{t-1}–P_{t-3}, V_{t-1}–V_{t-2}, and P_t can be observed, and the correlation coefficients reach above 5.5, so the wind power 3 hours before the time point to be forecast and the wind speed 2 hours before can be selected as the input data. However, because the wind speed and wind power correspond to each other, the simultaneous input of a set of wind speeds and wind power is beneficial to the extraction of their features using the neural network [70]. Therefore, the input data are finally determined as shown in Fig. 5.5, which are the wind power and wind speed data of 3 hours before the time point to be forecast.

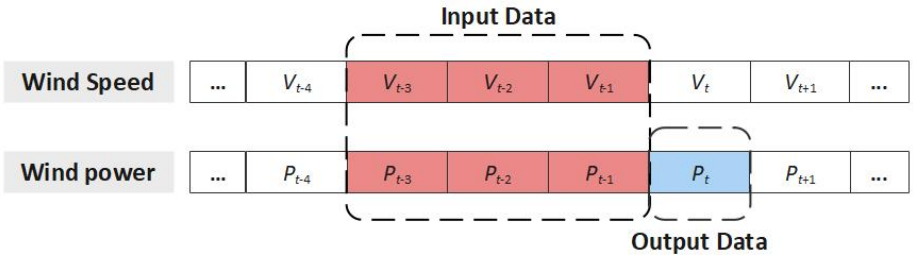


Fig. 5.5 Composition of input data and output data.

5.2.1.2 Processing of training data

Unlike the previous wind speed forecasting, when performing wind power forecasting, the only data currently available and used as training data are the wind speed and wind power data provided on the official website of the Sotavento wind farm, due to the limited conditions. Applying the previously proposed DAD method for data extraction is challenging because the data from surrounding locations are not available and the amount of available data provided on the official website is limited. Hence, to utilize the currently

available training data as efficiently as possible while ensuring forecasting accuracy, the k-means based group training method is proposed. The essence of this method is to classify data units (data groups containing input and output data) made from historical data, and then use all data units in each category as training data and train the forecasting model separately.

The objective function of k-means is shown in Equation (5-2).

$$d = min \sum_{i=1}^{n} min_j \left\| \boldsymbol{P_i} - \boldsymbol{P_j} \right\|^2 \qquad (5-2)$$

where $\boldsymbol{P}$:($P_{t\text{-}1}$, $P_{t\text{-}2}$, $P_{t\text{-}3}$, $V_{t\text{-}1}$, $V_{t\text{-}2}$, $V_{t\text{-}3}$) is the data unit consisting of the input data, i is the number of data, and j is the number of clustering centers. The size of i is determined by the amount of training data, and the size of j is the number of groupings, which is determined in this paper by the elbow method [34] in the grouping process.

5.2.1.3 Flow of the forecasting model

The flow of the forecasting is shown in Fig. 5.6.

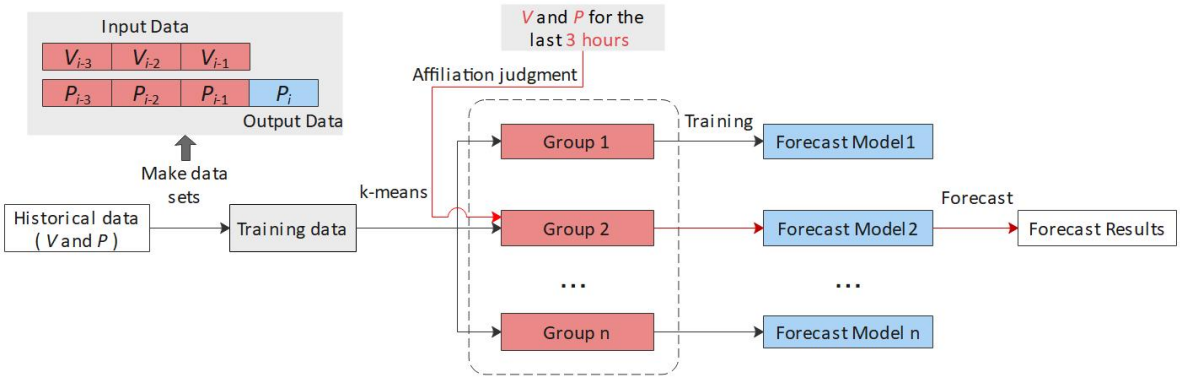


Fig. 5.6 Flowchart of wind power forecasting.

(1) The training data are formed by making all historical data of wind speed and its corresponding wind power into data units containing input data and output data (3 wind speed and 4 wind power data).

(2) The training data are divided into several groups with the same characteristics by the k-means method.

(3) Before making forecasts, the wind speed and wind power of the 3 hours before the time to be forecast are used as the judgment benchmark to determine which group the forecasting target belongs to.

(4) The forecasting model is obtained by training the forecasting model with that group of

data, and the forecasting model is used to forecast the data to be forecast, and the forecasting results are obtained.

5.2.2 Forecasting model based on CEEMD and LSTM

Drawing on the wind speed Forecasting model proposed in this paper, we try to use CEEMD to pre-process wind power data to improve the training effect of the Forecasting model and thus the Forecasting accuracy, and we propose a wind power Forecasting model based on CEEMD and LSTM.

5.2.2.1 Selection of input data

To decompose the training data using CEEMD, it must be ensured that the training data do not consist of discrete data sets but rather of a single and continuous time series. Thus, in case the training data cannot contain any data other than wind power, which limits the form of the input data to consist of wind power only. From the analysis of subsection 5.2.1.1, it can be seen that the strongest correlation with the output power is the wind power three hours before the forecasting time point, thereby setting the input data composition as shown in Fig. 5.7.

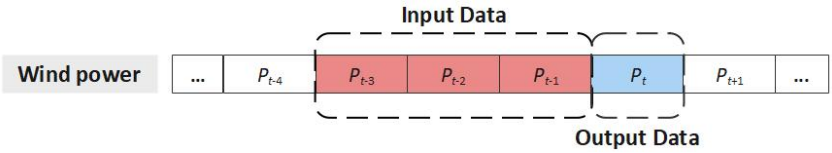


Fig. 5.7 Composition of input data and output data.

5.2.2.2 Processing of training data

In this session, the wind power time series is decomposed using CEEMD to obtain IMFs, and then each IMF is grouped separately using the method of 5.2.1.2. The training data are obtained and the forecasting model is trained, as shown in Fig. 5.8

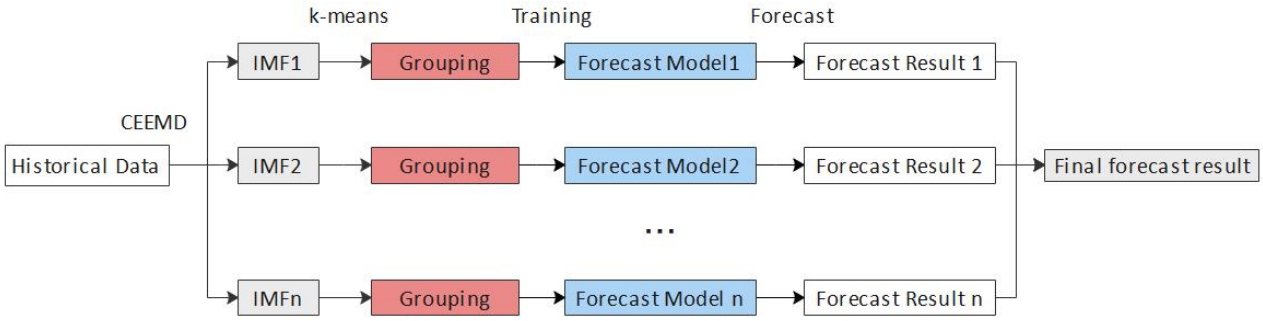


Fig. 5.8 Structure of wind power forecasting model.

The training of the forecasting model by this method will cause the workload to be very huge, which will directly affect the forecasting efficiency. Moreover, too many times of network training and forecasting will increase the probability of errors, and the final results will affect the accuracy of the forecasting model to some extent when they are superimposed. Therefore, we consider using the PE method mentioned in subsection 2.2.3 to simplify the forecasting model.

The parameter setting method in subsection 4.2.2 is used here, with m set to 4, t set to 3, and the threshold θ set to 0.7. Therefore, for the decomposed IMF, if H_p>0.7, the complexity is considered high and the grouping training process in subsection 5.2.1.2 is retained, while when H_p<0.7, the complexity of the IMF is measured as low, the grouping training process is canceled, and the original IMF data are used to directly train the neural network for training.

5.2.2.3 Flow of the forecasting model

The overall flow of the forecasting model is shown in Fig. 5.9,

(1) Decompose the historical data into several eigenmode functions IMFs by CEEMD.

(2) The complexity of each IMF is determined by using permutation entropy (PE).

- In the case of H_p≥0.7, the IMFs are grouped by the k-means method and a forecasting model is trained for each group.
- In the case of H_p<0.7, the forecasting models are trained directly using historical data as training data.

(3) Determine which group the decomposed forecasting object belongs to, use the forecasting model corresponding to that group and allow them to make forecasts, and combine the forecasting results of each IMF to obtain the final location

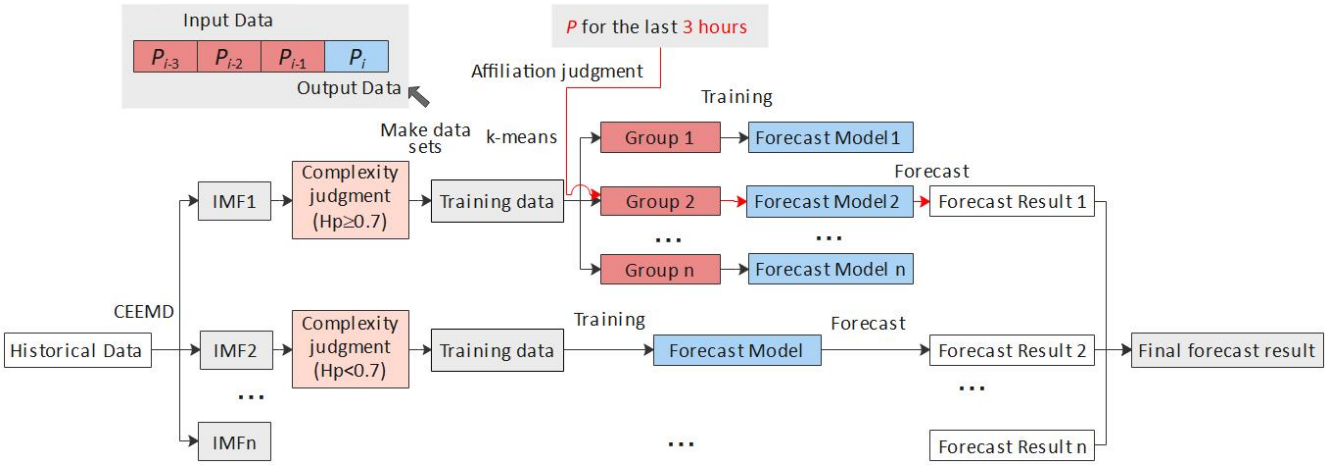


Fig. 5.9 Specific flowchart of wind power forecasting model.

5.2.3 Hybrid forecasting model

Three wind power forecasting methods are proposed, namely, forecasting based on wind speed forecasting, a forecasting model based on the LSTM network, and a forecasting model based on CEEMD and LSTM. It is verified by numerical calculations that all three methods were verified to show that the complete wind power forecasting with relatively good average forecasting accuracy. However, for wind power forecasting at individual time points, the forecasting accuracy of the three methods varies, and there is often a situation that one method has high accuracy while the other methods have low accuracy. The specific forecasting accuracy differences will be explained in detail in 5.2.2.

In the process of numerical simulation, we observed a certain inheritance effect in the forecasting accuracy of various forecasting methods within the adjacent hours, i.e., if the accuracy of a certain forecasting method is higher in consecutive hours, it is likely to have higher forecasting accuracy in the next hour's forecasting, whereas if the forecasting accuracy of this method is lower in consecutive hours, it is likely to have lower accuracy in the next hour's forecasting. If the forecasting accuracy of this method is low in consecutive hours, the forecasting result of the next hour will also be lower [72]–[74].

Based on the above two analyses, a linear combination model based on the three forecasting methods is proposed, as shown in Equation (5-3), to ensure that the forecasting method with the highest possible accuracy among the three methods in different situations can be used.

$$F(x) = \omega_1 \cdot f_1(x_1) + \omega_2 \cdot f_2(x_2) + \omega_3 \cdot f_3(x_3) \qquad (5-3)$$

where x is the input data vector, $f_1(x)$ is the forecasting model based on wind speed forecasting, $f_2(x)$ is the forecasting model based on the LSTM network, $f_3(x)$ is the wind power forecasting model based on CEEMD, and LSTM, ω is the weighting factor, and ω is determined by the forecasting accuracy of the three methods for n hours before the time to be forecast. The specific calculation is shown in Equation (5-4).

$$\omega_1 = \frac{n_1}{n}, \omega_2 = \frac{n_2}{n}, \omega_3 = \frac{n_3}{n} \quad (5-4)$$

n_1, n_2, and n_3 are the number of hours in n hours when the highest forecasting accuracy is obtained using methods 1, 2, and 3 ($n=n_1+n_2+n_3$)

The trial-and-error method found that the best forecasting accuracy could be obtained when n=6, so n was set to 6 in this study, so $\omega_1=n_1/6$, $\omega_2=n_2/6$, $\omega_3=n_3/6$, and $n_1+n_2+n_3=6$.

5.3 Evaluate the effectiveness of the proposed method by numerical calculation

5.3.1 Forecast preparation

The Sotavento experimental wind farm, which was founded in 1997 by Sotavento Galicia, S.A., was created to ensure that sufficient wind speed and wind power data are available for training. It was created with the following objectives in mind:

To build, operate, and manage facilities that use renewable energy as the main source of energy, encourage research in the field of renewable energy and energy efficiency, and provide equipment and data support for research in the field of wind energy.

Fig. 5.10 Location of the target wind farm.

To evaluate the forecasting accuracy and generalizability of the proposed wind power forecasting model, October 2020, January 2021, April, and July were selected as the forecasting target periods, and the hourly average wind power was chosen as the forecasting target.

The characteristics of wind speed and wind power for the four months of the wind farm are shown in Tables 5.2 and 5.3.

Table 5.2 Characteristics of wind speed for four months at the wind farm.

Wind speed (m/s)	Oct. 2020	Jan. 2021	Apr. 2021	Jul. 2021
Max	16.44	18.09	14.26	18.14
Min	0	0.11	0.04	0.03
Mean	6.10	7.18	5.49	5.09
Standard deviation	2.43	3.56	2.40	3.62

Table 5.3 Characteristics of wind power for four months at the wind farm.

Wind power (MWh)	Oct. 2020	Jan. 2021	Apr. 2021	Jul. 2021
Max	8.752	130.112	82.952	80.696
Min	0	0	0	0
Mean	26.207	28.913	14.433	11.081
Standard deviation	23.150	30.739	17.645	12.542

The table shows that the wind speed characteristics of the Sotavento wind farm are significantly similarity to those of the Hokkaido sites. In April and July, the wind speed and wind power were relatively small, and the average wind speed and average wind power in

July were the lowest among the four months. The fluctuation of wind speed in April and July is relatively flat as shown by the magnitude of the standard deviation. In contrast, the wind speed and wind power loss in October and January are relatively large and fluctuate widely, and the maximum average wind speed and average wind power were observed in January.

The database and testing data selection methods for each forecast target period refer to the introduction of subsection 3.2.1, and the specific selection methods are shown in Table 5.4, where incoherent wind speed or wind power data are complemented by interpolation.

Table 5.4 Principles of data selection.

Target time period	Data Base Time Period	Number of data	Testing data Time Period	Number of data
Jan	2020.11.3~ 2021.1.31	2184	2021.1.1~ 2021.1.31	744
Apr	2021.1.31~ 2021.4.30	2160	2021.4.1~ 2021.4.30	720
Jul	2021.5.2~ 2021.7.31	2184	2021.7.1~ 2021.7.31	744
Oct	2020.8.2~ 2020.10.31	2184	2020.10.1~ 2020.10.31	744

5.3.2 Experiment I: Comparison of Single forecasting model and hybrid forecasting model

We compared the proposed single forecasting models f_1, f_2, f_3, and the combined forecasting model F under the same forecasting conditions (Subsection 5.2.3).

The hourly average wind power for October 2020, January, April, and July 2021 was forecast by the four methods, and the RMSE, MAE, and MAPE of the forecasting results are shown in Table A.7, and the comparison of monthly average RMSE, MAE, and MAPE is shown in Fig.5.11, the distribution of MAPE for each method is shown in Fig. 5.12.

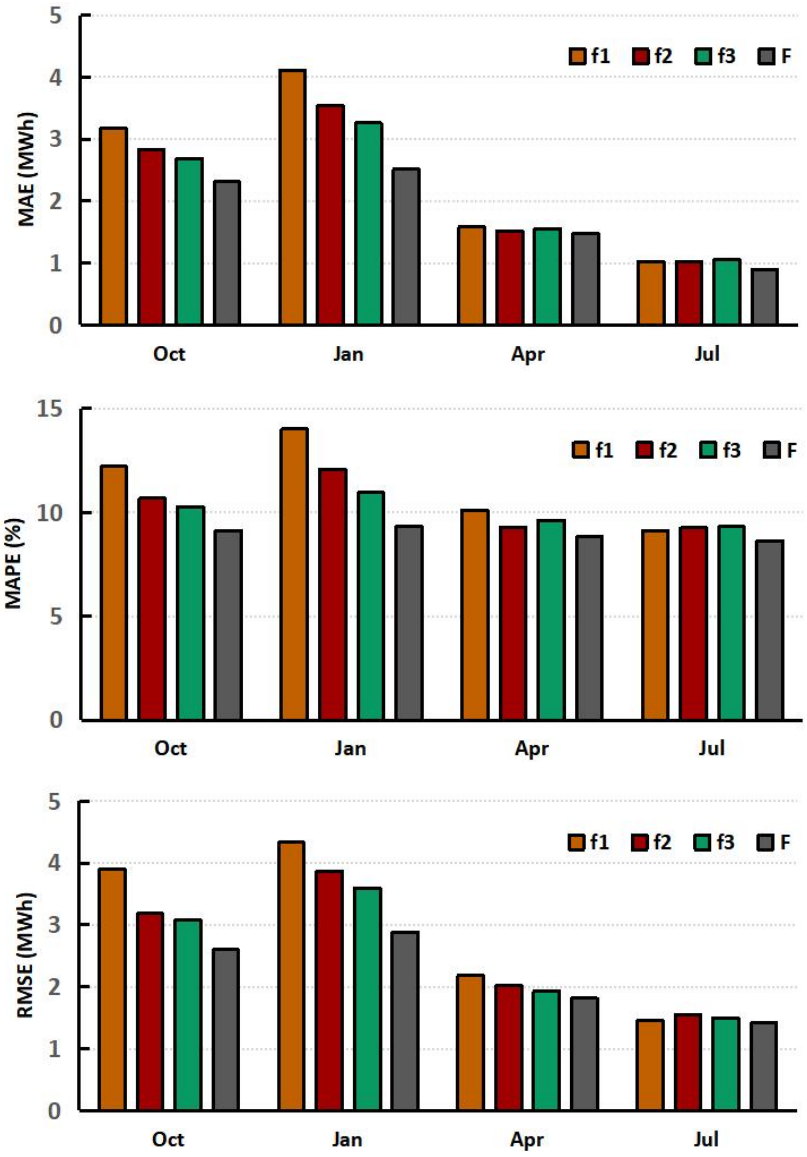


Fig. 5.11 Comparison of the accuracy of the proposed single method and the hybrid method.

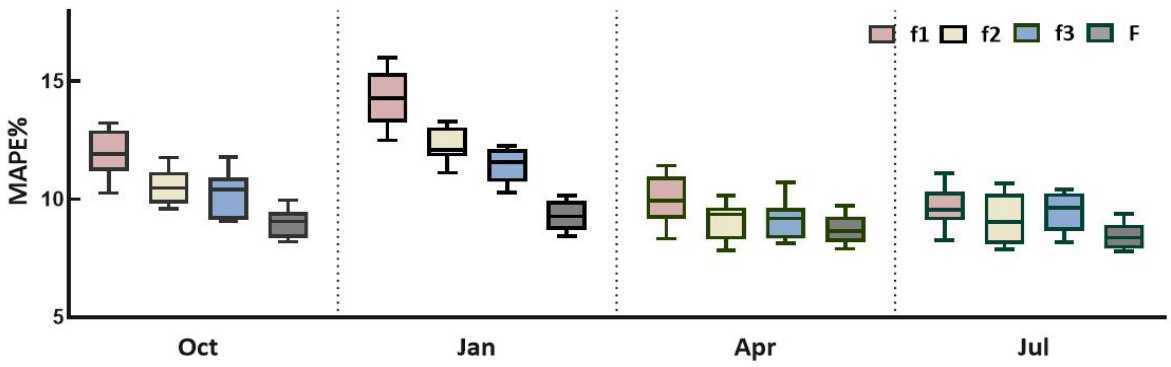


Fig. 5.12 Distribution of MAPE for the proposed single method and the hybrid method.

Fig. 5.11 shows that for future one-hour wind power forecasts, the hybrid forecasting method F obtained the highest forecast accuracy in all four months compared to the single forecasting method, with the RMSE remaining between 1.5 and 3.0 and the MAE between 0.8 and 2.5. Although the values vary from month to month, the overall difference is not significant, indicating that the hybrid forecasting method is less affected by climatic conditions and has better generalizability.

In contrast, the three single forecasting models are affected by climatic conditions, with relatively poor forecasting results in October and January, as reflected by the higher MAE and RMSE values in these two periods. The best forecasting results were obtained in July, with the MAE and RMSE values of all three methods at a low level. The forecasting accuracy in April was somewhere in between.

Table A.7 shows that although the forecasting accuracies of the three single methods have nearly the same trend with the change of seasons, the accuracies are different in the forecasting of each month. f_3 had the best overall performance but is very close to the accuracy of f_2 in the forecasting of July, but lower than that of f_1. f_2 has better forecasting accuracy in the forecasting of April.

Fig. 5.12 shows that compared to the direct forecasting method f_2 and f_3, the MAPE of f_1 based on wind speed forecasting is more divergent, especially in January and October when wind speed and wind power fluctuate more. However, the distribution of MAPE of f_1 is relatively converged in July when the fluctuations of wind speed and wind power are more stable. And the distribution of MAPE of the proposed hybrid forecasting method is relatively converged in all months of forecasting. This proves that the forecast stability of the hybrid forecasting method is higher than that of the single forecasting method.

Thus, it can be concluded tentatively that the proposed hybrid forecasting model can effectively forecast wind power in each season, and the accuracy of its forecasting does not change significantly with the change of seasons.

5.3.2 Experiment II: Comparison of other forecasting models with hybrid forecasting model

In Experiment 2, the hourly average wind powers in October 2020, January 2021, April, and July were selected as our forecast target period. Similar to that in subsection 4.3.2, typical single forecasting models, ARIMA [54]–[56],[69], BPNN[35], and Elman[57], and the widely used hybrid models, EMD-RNN, EEMD-LSTM, and EWT-LSTM[59] were selected for comparison with our proposed hybrid forecasting model. The parameter settings of these forecasting models are shown in Table 5.5. The parameter values that are not mentioned in the literature are set as default values. The RMSE, MAE, and MAPE of

the forecasting results for each method are shown in Table A.8, and a comparison of the RMSE, MAE, and MAPE for each method for four months is shown in Figs. 5.13 and 5.14. The comparison of the percent lift P_{RMSE}, P_{MAE}, and P_{MAPE} is shown in Fig. 5.15.

Table 5.5 Experimental parameter setting for different models.

Model	Parameter	Value
ARIMA	Autoregressive term (p)	4
	Moving average number (q)	5
	Difference times (d)	1
BPNN	Maximum number of iteration times	500
	Learning rate	0.01
	Training accuracy goal	0.005
	Number of input layer nodes	6
	Number of hidden layer nodes	12
	Number of output layer nodes	1
Elman	Maximum number of iteration times	500
	Number of input layer nodes	6
	Number of hidden layer nodes	12
	Number of output layer nodes	1
EEMD	Ensemble number (*NE*)	500
	Amplitude of white noise (*Nstd*)	0.3
EWT	Expected number of the filter bank	6

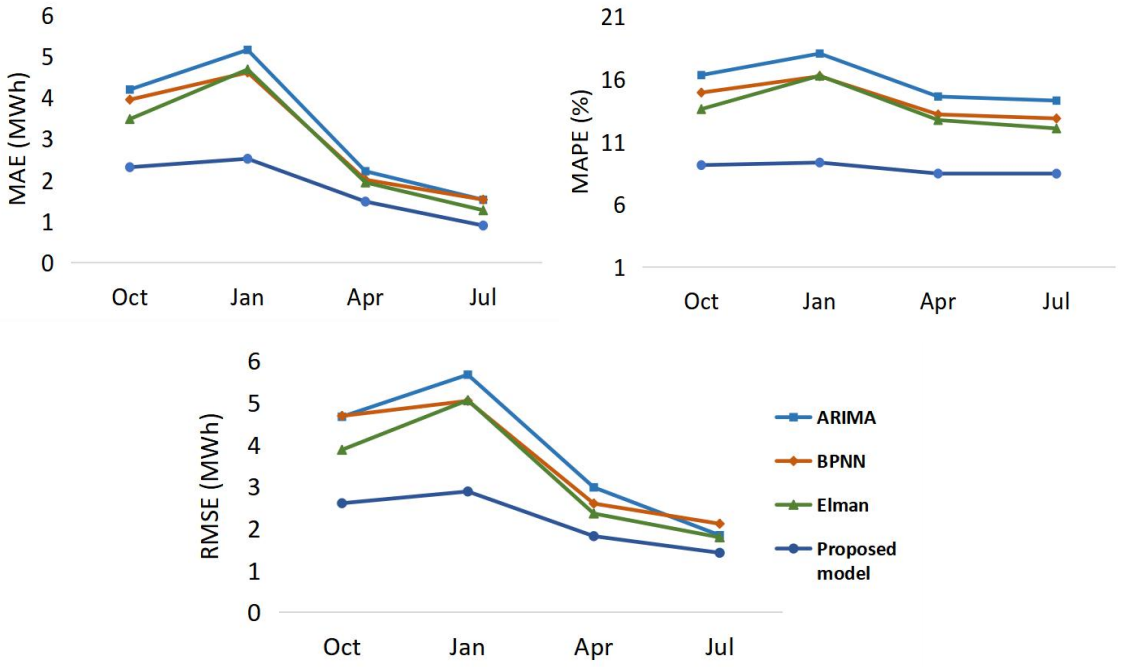


Fig. 5.13 Comparison of forecasting accuracy with typical single forecasting model.

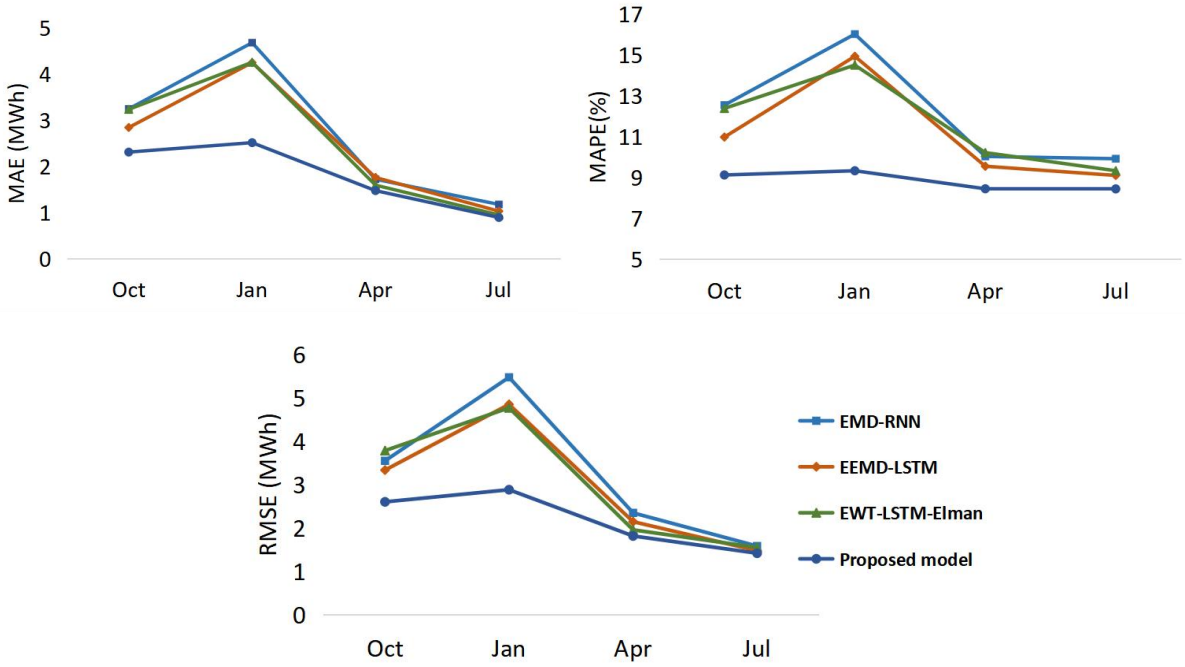


Fig. 5.14 Comparison of forecasting accuracy with typical hybrid forecasting model.

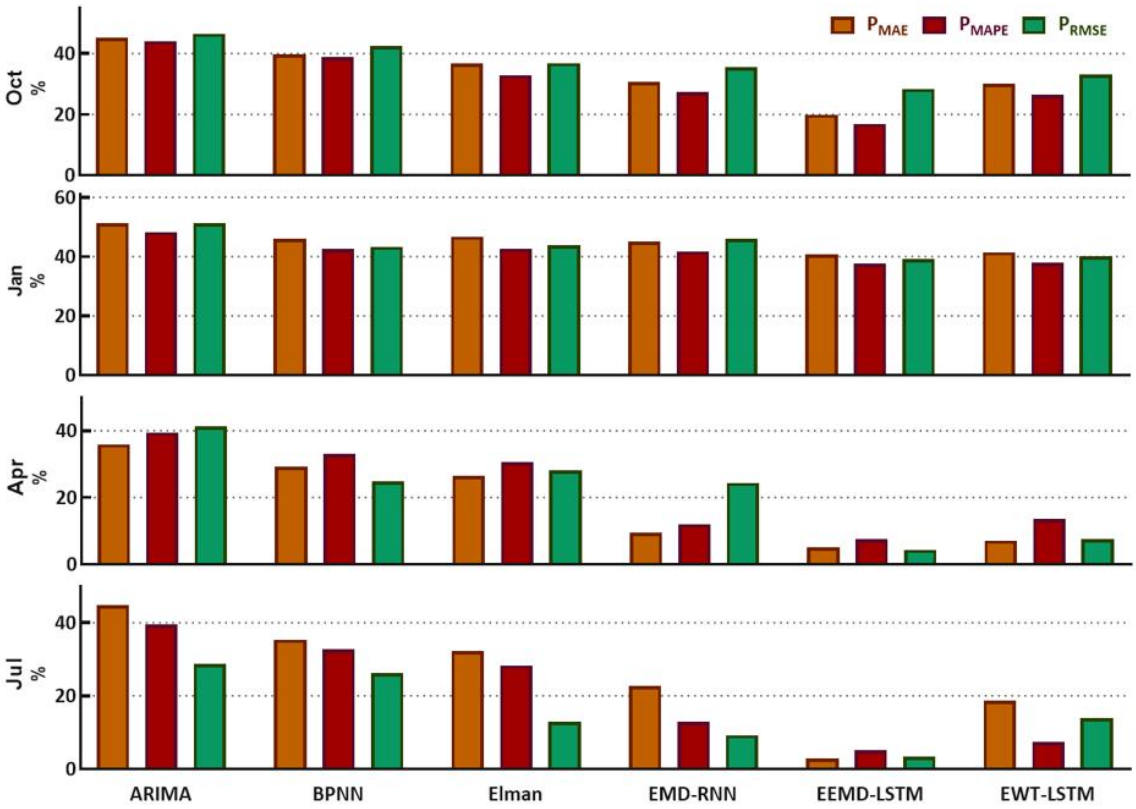


Fig. 5.15 Improved accuracy compared with other forecasting models.

(1) The generality of the single forecasting model is relatively poor, and the forecast

accuracy varies greatly in months. The lowest accuracy is obtained in January, while the highest accuracy is obtained in July. From the perspective of RMSE, the stability of the single forecasting models is generally low, and the RMSE fluctuates in the range of 1.7 to 5.7 with a large variation.

(2) The overall forecasting accuracy of the hybrid forecasting model is higher than that of the single forecasting model, and the model has better generality. The results in Table A.8 show that the RMSE stays in the range of 1.5 to 5.5, and the stability is higher than that of the single model. Among them, the EEMD-LSTM model has the best forecast accuracy, but the RMSE of the EEMD-LSTM model fluctuates more compared with the proposed hybrid forecasting model, which is seen to have lower forecasting stability than the proposed forecasting model.

(3) Among all models, the proposed hybrid forecasting model achieved the best forecasting results, and the accuracy of the four months did not differ much. The MAPE remained in the range of 8.5% to 10%. In terms of RMSE, the proposed forecasting model maintained a low value in the four-month forecast, and the overall body measurement stability is good.

(4) Fig. 5.15 shows that the proposed forecasting model has improved in RMSE, MAE, and MAPE compared to other forecasting models, and the accuracy improvement is more obvious for a single forecasting model compared to the accuracy improvement for a hybrid forecasting model. In addition, the accuracy of the proposed forecasting method does not vary considerably in each month of forecasting, while the other methods are more influenced by meteorological conditions, which makes the proposed method improve the accuracy in forecasting the months of October and January when the wind speed and wind power vary considerably as compared with other methods.

Chapter 6. Conclusion and Perspectives

6.1 Conclusion of Researches

This thesis introduces the forecasting method for wind speed and wind power generation based on data decomposition and deep learning neural network and verifies the effectiveness of the derived method by numerical simulation of the measured data.

The first chapter describes the background of the research. The development status of wind power generation is introduced and then, the current research status of wind speed and wind power forecasting is introduced, highlighting the description of the mainstream forecasting methods and the current development trends, and finally, the main research content of this paper is outlined.

Chapter 2 introduces the methods used in constructing the forecasting models in this thesis, specifically the main methods used in the four stages of data selection, data decomposition, forecasting model construction, and forecasting model improvement.

Chapters 3 and 4 presents the construction methods of wind speed forecasting models based on CEEMD decomposition and LSTM networks and proposes ultra-short-term (after 10 minutes, after 30 minutes, and 1 hour ahead) and short-term (24 hours ahead) wind speed forecasting models by adjusting the selection of training data, the method for data decomposition, and the optimization of neural networks, respectively. Finally, several locations in Hokkaido are used as forecasting target locations, and numerical simulations confirm that the proposed forecasting models have higher forecasting accuracy and better forecasting stability performance compared with other forecasting methods.

Chapter 5 presents the construction method for the wind power forecasting model. In this chapter, based on the wind speed forecasting model proposed in Chapter 3 and the main forecasting tools used in the forecasting model, three forecasting methods are proposed for wind power forecasting, namely, wind power forecasting model based on wind speed forecasting, wind power forecasting model based on LSTM network, and wind power forecasting model based on CEEMD and LSTM. Moreover, a hybrid forecasting model based on the three methods is proposed after analyzing the difference in forecasting accuracy of the three methods in different cases. Then, the numerical simulation is carried out with an experimental wind farm named sotavento in Spain as the target site for forecasting. The forecasting results show that the proposed hybrid model has a higher forecasting accuracy than the single forecasting model, is less affected by meteorological conditions, and has better generalizability. Compared with other forecasting methods, the

proposed method has higher forecasting accuracy and forecasting stability.

The results of numerical simulations indicate that the proposed wind speed and wind power forecasting method has relatively high forecasting accuracy and that the combination of data decomposition, training data pre-processing, and neural network parameter optimization allows the forecasting model to be adjusted flexibly for different forecasting conditions, which greatly enhances the practicality of the forecasting model. The forecasting method introduced in this study can be a good complement to the current research on wind power forecasting and lay a solid foundation for future research work.

6.2 Perspectives

The proposed forecasting method achieves good forecasting accuracy under achievable forecasting conditions but still has room for improvement in the use of data, selection of neural networks, improvement of forecasting efficiency, and implementation of longer time scale forecasting. Follow-up research work that can be carried out is as listed below.

(1) In the construction of the ultra-short-term wind speed forecasting model, the data area division (DAD) method is used to extract the wind speed data of the neighboring locations of the forecasting site as training data to train the neural network, and a good training effect is achieved. The historical data of the forecast locations were used as training data. In subsequent studies, wind farms where wind speed data from surrounding locations are available can be selected as the research target to verify whether the DAD method can be applied to wind power forecasting.

(2) The neural network used in this thesis is a modified form of the RNN LSTM network. From the introduction of Chapter 5, it can be seen that the hybrid forecasting model can improve the accuracy and forecasting stability of the forecasting model to some extent, and thus, in the subsequent research, we can try to propose a hybrid model using other neural networks (such as Elman, RNN, and Convolutional Neural Networks) at the same time to further improve the forecasting accuracy of the forecasting model.

(3) A hybrid forecasting model using three forecasting methods simultaneously is proposed for wind power forecasting, which makes the forecasting process more complicated and greatly reduces the forecasting efficiency. In subsequent research, the internal parameter adjustment of a neural network can be used to improve the training efficiency for different forecasting methods.

(4) For wind speed forecasting, ultra-short-term and short-term wind speed forecasting models are proposed, and the effectiveness of the forecasting methods is verified. However, for wind power forecasting, only the ultra-short-term forecasting model after 1 hour is

proposed in this thesis, and forecasting for longer time scales has not been discussed, which will be carried out in the subsequent research work.

References

[1] “Annual Global Wind Reports,” *Global Wind Energy Council,* 2020. [Online]. Available: https://gwec.net/category/annual-global-wind-reports/.

[2] “Wind power industry related reports,” *FD.BJX.COM.CN*, 2020. [Online]. Available: https://fd.bjx.com.cn/.

[3] J. Zhao, J. Wang, Z. Su, “Power generation and renewable potential in China,” *Renew Sustain Energy Rev.*, vol. 40, pp. 727–40, 2014.

[4] G.M. Joselin Herbert, S. Iniyanb., “A review of wind energy technologies,” *Renew Sustain Energy Rev.*, vol. 11, no. 6, pp. 1117–1145, 2007.

[5] “The world's cumulative wind power installed capacity rankings in 2020,” *FD.BJX.COM.CN*, 2020. [Online]. Available: https://news.bjx.com.cn/html/20210616/1158420.shtml.

[6] M. Imal, M. Sekkeli, C. Yildiz, and F Kececioglu, “Wind energy potential estimation and evaluation of electricity generation in Kahramanmaras,” *Energy Educ. Sc. Technol Part A-Energy Sci. Res.,* vol. 30, pp. 661–72 , 2012.

[7] “GWEC | A quick overview of global wind power development in 2019,” *FD.BJX.COM.CN*, 2020. [Online]. Available:https://news.bjx.com.cn/html/20200331/1059654.shtml.

[8] “Global Offshore Wind Report 2020,” *Global Wind Energy Council,* 2020. [Online]. Available: https://gwec.net/global-offshore-wind-report-2020/.

[9] J. G. Slootweg, W. L. Kling, “The impact of large scale wind power generation on power system oscillations,” *Electric Power Syst. Res.*, vol. 67, no. 1, pp. 9–20, 2003.

[10] A. Tascikaraoglu, M. Uzunoglu, “A review of combined approaches for forecasting of short-term wind speed and power,” *Renew Sustain Energy Rev.*, vol. 34, pp. 243–254, 2014.

[11] J. Jung and R. P. Broadwater, “Current status and future advances for wind speed and power forecasting,” *Renew Sustain Energy Rev.* vol.31, pp. 762–777 , 2014.

[12] S. S. Soman, H. Zareipour, O. Malik, and P. Mandal, “A review of wind power and wind speed forecasting methods with different time horizons,” *North American power symposium* , 2010.

[13] E. P. James, S. G. Benjamin, and M. Marquis, “Offshore wind speed estimates from a high resolution rapidly updating numerical weather forecasting model forecast dataset,” *Wind Energy*, vol. 21, pp. 264–84, 2018.

[14] D. J. Allen, A. S. Tomlin, C. Bale, A. Skea, S. Vosper, M. L. Gallani, et al, “A

boundary layer scaling technique for estimating near-surface wind energy using numerical weather prediction and wind map data," *Appl Energy*, 2017.

[15] H. Tom, C. Peter, "Correction and downscaling of NWP wind speed forecasts," *Meteorol Appl*. vol. 14, pp. 105–16, 2010.

[16] C. Lynch, M. J. Omahony, and T. Scully, "Simplified method to derive the Kalman filter covariance matrices to predict wind speeds from a NWP model," *Energy Procedia*, vol. 62, pp. 676–685, 2014.

[17] H. Liu, X. Mi, Y. Li, "Comparison of two new intelligent wind speed forecasting approaches based on Wavelet Packet Decomposition, Complete Ensemble Empirical Mode Decomposition with Adaptive Noise and Artificial Neural Networks," *Energy Conversion and Management*, vol. 155, no. 1, pp. 188–200, 2018.

[18] Z. Guo, W. Zhao, H. Lu, and J. Wang, "Multi-step forecasting for wind speed using a modified EMD-based artificial neural network model," *Renew Energy*, vol. 37, pp. 241–249, 2012.

[19] H. Wang, G. Wang, G. Li, J. Peng, Y. Liu, "Deep belief network based deterministic and probabilistic wind speed forecasting approach," *Apply Energy*, vol. 182, pp. 80–93, 2016.

[20] G. X. Silva, P. M. Fonte, J. C. Quadrado, "Radial basis function networks for wind speed prediction," *Proc. 5th WSEAS Int. Conf. Artificial Intelligence*, *Knowledge Engineering and Data Bases*, *Madrid*, *Spain*, pp. 286–290, 2006.

[21] D. Wang, H. Luo, O. Grunder, Y. Lin, "Multi-step ahead wind speed forecasting using an improved wavelet neural network combining variational mode decomposition and phase space reconstruction," *Renew Energy*, vol. 113, pp. 1345–58, 2017.

[22] O. B. Shukur, M.H. Lee, "Daily wind speed forecasting through hybrid KF-ANN model based on ARIMA," *Renew Energy*, vol. 76, pp. 637–47, 2015.

[23] C. Zhang, H. Wei, J. Zhao, "Short-term wind speed forecasting using empirical mode decomposition and feature selection," *Renew Energy*, vol, 96, pp. 727–37, 2016.

[24] H. Liu, C. Chen, H. Tian, "A hybrid model for wind speed prediction using empirical mode decomposition and artificial neural networks," *Renew Energy*, vol. 48, no. 6, pp. 545–56, 2012.

[25] H. Liu, C. Chen, H. Tian, and Y. Li, "A hybrid model for wind speed forecasting using empirical mode decomposition and artificial neural networks," *Renew Energy*, vol. 48, pp. 545–556, 2012.

[26] Z. Xi , A. Xu , Y. Kou, Z. Li, and A. Yang, "Air Combat Maneuver Trajectory Prediction Model of Target Based on Chaotic Theory and IGA-VNN," *Mathematical Problems in Engineering*, 2020.

[27] L. Xiao, W. Shao, F. Jin, and Z. Wu, "A self-adaptive kernel extreme learning machine for short-term wind speed forecasting," Applied Soft Computing, vol.99, 2021.

[28] P. Ramírez, J. A. Carta, "The use of wind probability distributions derived from the maximum entropy principle in the analysis of wind energy. A case study", *Energy*

Conversion and Management, vol. 47, no.15, 16, pp. 2564–2577, 2006.

[29] C. Tian, Y. Hao, J. Hu, "A novel wind speed forecasting system based on hybrid data preprocessing and multi-objective optimization," *Appl Energy*, 2018.

[30] T. Ouyang, X. Zha, and L. Qin, "A combined multivariate model for wind power forecasting," *Energy Convers Manage*, vol. 144, pp. 361–373, 2017.

[31] Z. Yang, J. Wang, "A hybrid forecasting approach applied in wind speed forecasting based on a data processing strategy and an optimized artificial intelligence algorithm," *Energy*, vol. 160,pp. 87–100, 2018.

[32] G. Li, J. Shi, "On comparing three artificial neural networks for wind speed forecasting," *Appl Energy*, vol. 87, pp. 2313–2320, 2010.

[33] J. Song, J. Wang, H. Lu, "A novel combined model based on advanced optimization algorithm for short-term wind speed forecasting," *Appl Energy*, vol. 215, pp. 643–58, 2018.

[34] A. Likas, N. V. lassis, J. J. Verbeek, "The global k-means clustering algorithm," *Pattern Recognition*, vol. 36, no. 2, pp. 451–461, 2003.

[35] S. Wang, N. Zhang, L. Wu, Y. Wang, "Wind speed forecasting based on the hybrid ensemble empirical mode decomposition and GA-BP neural network method," *Renewable Energy*, vol. 94, pp. 629–636, 2016.

[36] R. Niazy, C. Beckmann, J. Brady, S. Smith, "Performance evaluation of ensemble empirical mode decomposition," *Adv Adapt Data Anal 2009*. vol. 1, pp. 231–242, 2009.

[37] D. Liu, D. Niu, H. Wang, L. Fan, "Short-term wind speed forecasting using wavelet transform and support vector machines optimized by genetic algorithm," *Renewable Energy*, vol. 62, pp. 592–7, 2014.

[38] Z. Wu, N. E. Huang, "Ensemble empirical mode decomposition: a noise-assisted data analysis method," *Advances in Adaptive Data Analysis*, vol. 1, no. 1, pp. 1-41, 2009.

[39] C. Bandt, and B. Pompe, "Permutation entropy: a natural complexity measure for time series," *Phys. Rev. Lett.*, vol. 88, 2002.

[40] F. Feng, A. Si, and G. Rao, "Early fault diagnosis of bearing based on wavelet correlation permutation entropy," *J. Mechanical Engineering*, vol. 48, pp. 73-79, 2012. (in Chinese)

[41] M. Riedl, A. Müller, and N. Wessel, "Practical considerations of permutation entropy," *Eur. Phys. J. Spec. Top*, vol. 222, pp. 249–262, 2013.

[42] T. G. Barbounis, J. B. Theocharis, "Locally recurrent neural networks for long-term wind speed and power forecasting," *Neurocomputing*, vol. 69, pp. 466–496, 2006.

[43] H. Shao, X. Deng, and Y. Jiang,"A novel deep learning approach for short-term wind power forecasting based on infinite feature selection and recurrent neural network,"*J. Renewable Sustainable Energy*, vol. 10, 2018.

[44] S. Hochreiter, J. Schmidhuber, "Long short-term memory," *Neural computation*, 1997.

[45] R. Yu, J. Gao, M. Yu, W. Lu, T. Xu, M. Zhao, "LSTM-EFG for wind power forecasting based on sequential correlation features," *Futur Gener Comput Syst 2019*, vol. 93, pp. 33–42, 2019.

[46] M. Srinivas, and L. M. Patnik, "Genetic Algorithms: A Survey," Computer, vol. 6, pp. 17–26 , 1994.

[47] M. Dorigo and G. D. Caro, "Ant Colony Optimization: A New Metaheuristic," *in Proc. of the Congress on Evolutionary Computing*, 1999.

[48] B. Liu, H. A. Abbass, and B. McKay, "Classification Rule Discovery with Ant Colony Optimization," *IEEE Computational Intelligence Bulletin*, vol. 3, pp. 31–35, 2004.

[49] J. Mohammadhassani, A. Dadvand, and S. Khalilarya, "forecasting and reduction of diesel engine emissions using a combined ANN–ACO method," *Applied Soft Computing*, vol. 34, pp. 139–150, 2015.

[50] A. P. Marugán, F. P. G. Márquez, J. M. P. Perez, D. Ruiz-Hernández, "A survey of artifcial neural network in wind energy systems," *Appl Energy*, vol. 228, pp.1822–1836, 2018.

[51] Z. Yang, J. Wang, "A combination forecasting approach applied in multistep wind speed forecasting based on a data processing strategy and an optimized artifcial intelligence algorithm," *Appl Energy*, vol. 230, pp.1108–25, 2018.

[52] E. Cadenas, W. Rivera, "Wind speed forecasting in three different regions of Mexico, using a hybrid ARIMA-ANN model," *Renew Energy*, vol. 35, pp. 2732–8, 2010.

[53] J.R. Yeh, J.S. Shieh, N.E. Huang, "Complementary ensemble empirical mode decomposition: A novel noise enhanced data analysis method," *Advances in Adaptive Data Analysis*, vol. 02, no. 02, pp. 135–156, 2010.

[54] J. L. Torres, A. Garcıa, M. D. Blas, and A. D. Francisco, "Forecast of hourly average wind speed with ARMA models in Navarre (Spain)," *Sol. Energy*, vol. 79, pp. 65–77, 2005.

[55] V. Radziukynas, A. Klementavicius, "Short-term wind speed forecasting with ARIMA model," *ABT-ISC on P and EE of RTU*, pp. 145–9, 2014.

[56] R. G. Kavasseri, K. Seetharaman, "Day-ahead wind speed forecasting using f-ARIMA models," *Renew Energy*, vol. 34, pp. 1388–93, 2009.

[57] H. Liu, H. Tian, X. Liang, Y. Li, "Wind speed forecasting approach using secondary decomposition algorithm and Elman neural networks,". Appl Energy, vol. 157, pp. 183–94, 2015.

[58] J. Wang, J. Hu, "A robust combination approach for short-term wind speed forecasting and analysis - Combination of the ARIMA (Autoregressive Integrated Moving Average), ELM (Extreme Learning Machine), SVM (Support Vector Machine) and LSSVM (Least Square SVM) forecasts using a GPR (Gaussian Process Regression) model," *Energy*, vol. 93, pp. 41–56, 2015.

[59] H. Liu, X. Mi, Y. Li, "Wind speed forecasting method based on deep learning strategy using empirical wavelet transform, long short term memory neural network and Elman neural network," *Energy Convers Manage*, vol. 156, pp. 498-514, 2018.

[60] W. Zhang, Z. Qu, K. Zhang, W. Mao, Y. Ma, X. Fan, "A combined model based on CEEMDAN and modified flower pollination algorithm for wind speed forecasting," *Energy Convers Manage*， vol. 136, pp. 439–51, 2017.

[61] F. Feng, A. Si, and G. Rao, "Early fault diagnosis of bearing based on wavelet correlation permutation entropy," *J. Mechanical Engineering*, vol. 48, pp. 73-79, 2012. (in Chinese)

[62] M. Riedl, A. Müller, and N. Wessel, "Practical considerations of permutation entropy," *Eur. Phys. J. Spec. Top*, vol. 222, pp. 249–262, 2013.

[63] Z. Tian, S. Li, and Y. Wang, "A forecasting approach using ensemble empirical mode decomposition-permutation entropy and regularized extreme learning machine for short-term wind speed," *Wind Energy*, vol. 23, pp. 177-206, 2020.

[64] L. Chen, X. Lai, "Comparison between ARIMA and ANN models used in short-term wind speed forecasting," *BT-A-PP and EEC*, pp. 1–4, 2011.

[65] H. H. H. Aly, "A proposed intelligent short-term load forecasting hybrid models of ANN, WNN and KF based on clustering techniques for smart grid," *Int J Electric Power Syst. Res.*, 2020.

[66] L. Wang, Z. Wang, H. Qu, S. Liu, "Optimal forecast combination based on neural networks for time series forecasting," *Appl. Soft Comput.*, vol. 66, pp. 1–17, 2018.

[67] S. Salcedo-Sanz, A Pastor-Sánchez, L. Prieto, A. Blanco-Aguilera, R. García-Herrera, "Feature selection in wind speed prediction systems based on a hybrid coral reefs optimization - Extreme learning machine approach," *Energy Convers Manage*, vol. 87, pp. 10–8, 2014.

[68] X. Zhao, C. Wang, J. Su, J. Wang, "Research and application based on the swarm intelligence algorithm and artifcial intelligence for wind farm decision system," *Renew Energy*, vol. 134, pp.681–97, 2019.

[69] H. Liu, H. Tian, Y. Li, "An EMD-recursive ARIMA method to predict wind speed for railway strong wind warning system," *J Wind Eng Ind Aerodyn*, vol. 141, pp. 27–38, 2015.

[70] H. Wang, G. Li, G. Wang, J. Peng, H. Jiang, Y. Liu, "Deep learning based ensemble approach for probabilistic wind power forecasting," *Appl Energy*, vol. 188, pp. 56–70, 2017.

[71] J. Zhang, J. Yan, D. Infeld, Y. Liu, L.Fue-sang, "Short-term forecasting and uncertainty analysis of wind turbine power based on long short-term memory network and Gaussian mixture model," *J Appl Energy*, vol. 241, pp. 229–44, 2019.

[72] N. Sun, J. Zhou, L. Chen, B. Jia, M. Tayyab, T. Ping, "An adaptive dynamic short-term wind speed forecasting model using secondary decomposition and an improved regularized extreme learning machine," *Energy*, vol. 165, pp. 939–57, 2018.

[73] K. Zhang, Z. Qu, J. Wang, W. Zhang, F. Yang, "A novel hybrid approach based on cuckoo search optimization algorithm for short-term wind speed forecasting," *Environ Prog Sustain Energy*, vol. 36, 2017.

[74] Y. Huang, L. Yang, Y. Yang, and Y. Dong, "A novel hybrid approach based on

dynamic adaptive variable-weight optimization for short-term wind speed forecasting," *J. Renewable Sustainable Energy*, vol. 12, 2020.

[75] Z. Liu, R. Hara, and H. Kita, "Wind Speed Forecast System using Recurrent Neural Network," Joint Technical Meeting on Power Engineering & Power Systems Engineering, PE-19–189,PSE-19 -201, 198–203 (2019).(in Japanese)

[76] Z. Liu, R. Hara, H. Kita, "Hybrid forecasting system based on data area division and deep learning neural network for short-term wind speed forecasting", *Energy Conversion and Management*, vol. 238, no. 15, pp. 114–136, 2021.

[77] Z. Liu, R. Hara, H. Kita, "24h-ahead wind speed forecasting using CEEMD-PE and ACO-GA-based deep learning neural network", *Journal of Renewable and Sustainable Energy*, vol. 13, no. 46, pp. 101–115, 2021.

Acknowledgment

My deepest gratitude to those who helped and supported me in making this research possible.

I would like to thank Associate Professor Ryoichi Hara for his constant guidance and encouragement in my studies and research. This research could not have been completed without his considerable efforts in weekly meetings to guide the direction of my research, provide ideas to solve difficulties and correct errors that occurred during the research process. In addition to the research aspect, Prof. Hara's erudition, humor, and approachability are also examples that I have learned from.

I am also deeply grateful to Professor Hiroyuki Kita for his valuable advice on the research direction of this study, his patient guidance on the presentation shape of the research results, and his advice on the presentation style and content of the presentation in the presentation practice, which improved my ability to give scientific presentations in conferences. Professor Kita also encouraged me in all aspects of my life in Japan and I am very grateful to him for his support and help.

I would also like to express my sincere thanks to Assistant Professor Shiho Ishikawa, Technical Staff Ko Rinoie, Secretary Rie Arima, Chiemi Ozaki and Midori Takahasi for the help they gave me. My sincere thanks to all the members of the Power and Energy Systems Laboratory, especially to Tsubasa Ichikawa, my tutor when I first came to Sapporo, and to local Japanese friends including Keisuke Izumiya, Shuhei Kamiyama, and Yushi Kida, who have helped me at different times in my life and my research.

I am grateful to the Ministry of Education, Culture, Sports, Science, and Technology (MEXT) of Japan for providing me with financial support for tuition and living expenses since October 2017, which has allowed me to devote myself to my research work without facing the stress of living.

Finally, and most importantly, I would like to thank my parents for their constant love and support. Without their encouragement, I would not have had the courage to pursue my PhD. I dedicate this thesis to them.

Zhuoyi Liu

February 2022

Appendix A: Detailed Calculated Results in Numerical Tests

Table A.1 Forecast accuracy of different locations and different periods (wind speed).

Locations		1	2	3	4	5
Jan	MAE (m/s)	0.4585	0.4280	0.4554	0.4379	0.5195
	RMSE (m/s)	0.6233	0.5215	0.5493	0.5685	0.6145
	MAPE (%)	6.1160	5.5197	5.9051	5.7018	6.7265
Feb	MAE (m/s)	0.4195	0.4322	0.4428	0.4154	0.4160
	RMSE (m/s)	0.5697	0.5147	0.5732	0.5072	0.4850
	MAPE (%)	5.7734	5.6510	5.7722	5.3628	5.4349
Mar	MAE (m/s)	0.3397	0.3524	0.3563	0.3721	0.3776
	RMSE (m/s)	0.4224	0.4322	0.3933	0.4756	0.4461
	MAPE (%)	5.0368	5.9311	5.9994	6.3376	6.4091
Apr	MAE (m/s)	0.1771	0.2171	0.2424	0.2849	0.2387
	RMSE (m/s)	0.2227	0.2891	0.3238	0.3572	0.3137
	MAPE (%)	3.4580	3.7372	4.1021	4.8524	4.1009
May	MAE (m/s)	0.2497	0.2544	0.2421	0.2751	0.2782
	RMSE (m/s)	0.3109	0.3381	0.3340	0.3131	0.3179
	MAPE (%)	4.6770	4.4950	4.3270	4.7720	4.9402
Jun	MAE (m/s)	0.2178	0.2321	0.2462	0.2224	0.2263
	RMSE (m/s)	0.3281	0.3186	0.2981	0.3136	0.3116
	MAPE (%)	3.4168	4.2014	4.3916	4.0103	3.9997
Jul	MAE (m/s)	0.1606	0.1922	0.2551	0.1561	0.1847
	RMSE (m/s)	0.2081	0.2474	0.2942	0.1988	0.2311
	MAPE (%)	3.3331	4.0364	4.4927	3.2846	3.9551
Aug	MAE (m/s)	0.1701	0.1820	0.2943	0.1460	0.2243
	RMSE (m/s)	0.2187	0.2247	0.3156	0.1902	0.2550
	MAPE (%)	3.6372	4.0689	5.3377	3.2608	4.9829
Sep	MAE (m/s)	0.2393	0.2543	0.2363	0.2808	0.2453
	RMSE (m/s)	0.2571	0.3262	0.3568	0.3304	0.3207
	MAPE (%)	4.5054	4.3351	3.9605	4.6963	4.1705
Oct	MAE (m/s)	0.3070	0.2843	0.3018	0.3014	0.3108
	RMSE (m/s)	0.3988	0.3836	0.3812	0.3802	0.3253
	MAPE (%)	4.9011	4.8518	5.2358	5.1665	5.2966
Nov	MAE (m/s)	0.4949	0.4280	0.3976	0.4164	0.4995
	RMSE (m/s)	0.5314	0.5334	0.5605	0.5196	0.5703
	MAPE (%)	6.1775	6.4359	6.0433	6.2986	6.5360

Dec	MAE (m/s)	0.4141	0.4262	0.4456	0.4212	0.4216
	RMSE (m/s)	0.4965	0.5351	0.5274	0.4932	0.5130
	MAPE (%)	6.0393	6.4716	6.8401	6.5408	6.4071

Locations		6	7	8	9	10
Jan	MAE (m/s)	0.4044	0.4831	0.5066	0.4343	0.4530
	RMSE (m/s)	0.5127	0.5434	0.5955	0.5137	0.5443
	MAPE (%)	5.6281	6.3686	6.5672	7.3334	5.7401
Feb	MAE (m/s)	0.4437	0.4425	0.3874	0.4201	0.3809
	RMSE (m/s)	0.5262	0.5213	0.4905	0.4487	0.4525
	MAPE (%)	5.7044	5.9386	5.1983	5.5680	5.8191
Mar	MAE (m/s)	0.4091	0.3666	0.3602	0.3299	0.3547
	RMSE (m/s)	0.4744	0.4525	0.4274	0.4167	0.4660
	MAPE (%)	4.6228	5.3917	6.1228	5.5742	5.6271
Apr	MAE (m/s)	0.1952	0.2885	0.2753	0.1932	0.1880
	RMSE (m/s)	0.2425	0.3024	0.3398	0.2723	0.2018
	MAPE (%)	4.0399	5.5705	5.3642	3.8030	4.5353
May	MAE (m/s)	0.2378	0.2523	0.2517	0.2552	0.2379
	RMSE (m/s)	0.2908	0.3372	0.3328	0.3321	0.3217
	MAPE (%)	4.5515	4.8709	4.8005	4.9569	4.5753
Jun	MAE (m/s)	0.2840	0.2577	0.2356	0.2424	0.2519
	RMSE (m/s)	0.3377	0.3196	0.3238	0.3228	0.3413
	MAPE (%)	3.5614	4.5829	4.2572	4.3195	4.4589
Jul	MAE (m/s)	0.1238	0.1380	0.1822	0.1601	0.1597
	RMSE (m/s)	0.1621	0.1739	0.2232	0.2259	0.2071
	MAPE (%)	3.3957	2.8916	3.7694	3.3308	4.0216
Aug	MAE (m/s)	0.2168	0.1992	0.1718	0.1534	0.1981
	RMSE (m/s)	0.2495	0.2273	0.2404	0.1957	0.2139
	MAPE (%)	3.7635	4.3800	3.7793	3.3321	3.9537
Sep	MAE (m/s)	0.2727	0.2719	0.2404	0.3028	0.2956
	RMSE (m/s)	0.3206	0.3619	0.2958	0.3542	0.3457
	MAPE (%)	4.5815	4.5817	4.1256	4.3090	4.5701
Oct	MAE (m/s)	0.2736	0.2934	0.2822	0.3433	0.2752
	RMSE (m/s)	0.3280	0.3486	0.3645	0.4168	0.3162
	MAPE (%)	4.7693	5.0125	4.8538	5.9785	4.7609
Nov	MAE (m/s)	0.4456	0.3592	0.3978	0.4698	0.3750
	RMSE (m/s)	0.5183	0.4122	0.4673	0.5521	0.4645
	MAPE (%)	6.1467	5.3917	6.9815	6.3727	5.6925
Dec	MAE (m/s)	0.5037	0.4151	0.4112	0.4192	0.3943
	RMSE (m/s)	0.5469	0.4968	0.5151	0.5116	0.4801
	MAPE (%)	6.0770	5.6745	6.2500	6.5137	6.2792

Table A.2 Forecast accuracy at different time intervals.

	Jan			Apr		
	10	30	60	10	30	60
MAE (m/s)	0.4585	0.5675	0.5836	0.1771	0.2462	0.3897
RMSE (m/s)	0.6233	0.6647	0.6617	0.2227	0.3443	0.4466

	Jul			Oct		
	10	30	60	10	30	60
MAE (m/s)	0.1606	0.2572	0.3637	0.3070	0.3936	0.4818
RMSE (m/s)	0.2081	0.3125	0.4569	0.3988	0.4514	0.5638

Table A.3 Comparison of forecasting accuracy with typical single model.

	Jan			Apr		
	MAE (m/s)	MAPE (%)	RMSE (m/s)	MAE (m/s)	MAPE (%)	RMSE (m/s)
ARIMA	0.6622	9.2890	0.9321	0.4318	9.4578	0.5933
BPNN	0.5769	8.1650	0.6616	0.3707	7.2106	0.5926
Elman	0.7249	9.1513	0.8186	0.4293	8.7521	0.6176
ELM	0.6625	8.1591	0.7840	0.4521	8.2349	0.5292
Proposed model	**0.4585**	**6.1188**	**0.6233**	**0.1771**	**3.4817**	**0.2227**

	Jul			Oct		
	MAE (m/s)	MAPE (%)	RMSE (m/s)	MAE (m/s)	MAPE (%)	RMSE (m/s)
ARIMA	0.3903	9.6159	0.4491	0.5353	9.6440	0.8087
BPNN	0.3121	7.8228	0.5509	0.3838	8.8544	0.5716
Elman	0.3047	8.2564	0.4405	0.4908	8.8776	0.6110
ELM	0.3668	8.2442	0.4381	0.4795	8.7584	0.6929
Proposed model	**0.1606**	**3.5519**	**0.2081**	**0.3070**	**4.8564**	**0.3988**

Table A.4 Comparison of forecasting accuracy with typical hybrid forecasting model.

	Jan			Apr		
	MAE (m/s)	MAPE (%)	RMSE (m/s)	MAE (m/s)	MAPE (%)	RMSE (m/s)
EMD-RNN	0.6149	7.9848	0.7782	0.2829	6.6105	0.4011
EEMD-LSTM	0.5448	7.7728	0.7577	0.2970	6.9082	0.4155
EWT-LSTM-Elman	0.6307	7.6084	0.7764	0.2829	6.9394	0.4887
CEEMDAN-ENN	0.5711	7.1008	0.7072	0.2916	6.9850	0.5350
Proposed model	**0.4585**	**6.1188**	**0.6233**	**0.1771**	**3.4817**	**0.2227**

	Jul			Oct		
	MAE (m/s)	MAPE (%)	RMSE (m/s)	MAE (m/s)	MAPE (%)	RMSE (m/s)
EMD-RNN	0.2306	5.0210	0.4626	0.3709	6.6625	0.4425
EEMD-LSTM	0.2670	5.9450	0.3555	0.3598	7.1698	0.5262
EWT-LSTM-Elman	0.2418	5.2369	0.2904	0.4047	7.7360	0.5135
CEEMDAN-ENN	0.2397	4.7566	0.4264	0.3999	7.4260	0.5758
Proposed model	**0.1606**	**3.5519**	**0.2081**	**0.3070**	**4.8564**	**0.3988**

Table A.5 Forecasting errors for each season across the three locations.

Month	Location	1-step			2-step		
		RMSE (m/s)	MAE (m/s)	MAPE (%)	RMSE (m/s)	MAE (m/s)	MAPE (%)
Jan	1	0.9569	0.8444	14.1811	1.4561	1.3529	21.1204
	2	1.1186	0.9781	13.1702	1.6020	1.5044	21.9623
	3	0.9182	0.8230	13.3942	1.3968	1.2773	20.6046
Apr	1	0.7657	0.639	13.2531	1.0389	0.9354	18.9793
	2	0.6294	0.5125	13.0761	0.9281	0.7704	17.1517
	3	0.7698	0.6495	12.8527	1.1417	1.0075	19.2440
Jul	1	0.5115	0.4166	12.4448	0.8097	0.6753	17.1344

	2	0.5435	0.4005	11.8821	0.7704	0.6512	18.2703
	3	0.5748	0.4645	12.7005	0.8515	0.6610	18.8810
	1	0.8602	0.7413	13.0336	1.2654	1.1793	18.1422
Oct	2	0.8371	0.6507	13.5519	1.3334	1.1446	20.2895
	3	0.9120	0.7801	13.5843	1.2170	1.0854	20.5770

Month	Location	3-step			4-step		
		RMSE (m/s)	MAE (m/s)	MAPE (%)	RMSE (m/s)	MAE (m/s)	MAPE (%)
	1	1.8130	1.6424	26.2721	2.7211	2.3037	38.0442
Jan	2	2.0913	1.8989	28.8693	3.0112	2.6732	39.5431
	3	1.7634	1.6257	26.2923	2.8725	2.5348	37.5198
	1	1.3643	1.1847	27.2111	2.1470	1.7696	37.7734
Apr	2	1.3187	1.1502	26.7806	1.8773	1.4889	33.2272
	3	1.6094	1.4302	29.2477	2.1113	1.8191	35.4002
	1	1.0917	0.9963	25.3463	1.6052	1.4561	40.5487
Jul	2	1.0010	0.8719	24.4328	1.7276	1.3292	36.4291
	3	1.2223	1.1334	27.7128	1.8053	1.3907	38.4176
	1	1.6382	1.4481	26.9530	2.6302	2.1031	40.0202
Oct	2	1.8161	1.5246	27.2303	2.7552	2.4154	42.4811
	3	1.6992	1.6142	25.6001	2.4516	2.2440	39.3763

Month	Location	5-step			6-step		
		RMSE (m/s)	MAE (m/s)	MAPE (%)	RMSE (m/s)	MAE (m/s)	MAPE (%)
	1	3.1714	2.7885	44.9695	3.7559	3.2642	53.2253
Jan	2	3.1840	2.9258	45.7721	4.1252	3.5875	52.8933
	3	3.0597	2.7844	45.4123	3.8222	3.3837	51.2127
	1	2.3102	1.9870	42.1811	2.7106	2.3709	49.1531
Apr	2	1.9899	1.7321	41.7443	2.5514	2.1553	48.5327
	3	2.4493	2.1642	43.0720	3.0091	2.5526	51.2922
	1	2.1029	1.6797	45.3574	2.2382	1.8945	49.6585
Jul	2	1.8852	1.5745	43.2583	2.3392	1.8330	49.9285
	3	2.0878	1.6659	45.3428	2.1102	1.7769	47.4106
Oct	1	2.8036	2.3434	42.8283	2.9888	2.7614	50.0104

2	2.8831	2.5972	46.8421	3.3748	2.9214	49.4401
3	2.8962	2.5523	46.5922	3.1684	2.8641	50.7342

Table A.6 The 24-hour average error of each forecasting model.

	RMSE (m/s)	MAE (m/s)	MAPE (%)
ARIMA	3.5726	3.0918	49.8522
BPNN	3.1522	2.7032	43.9442
Elman	2.9878	2.5932	42.1504
EMD-RNN	2.7121	2.4123	38.8255
EEMD-LSTM	2.6968	2.3852	38.5503
EWT-LSTM	2.5302	2.1601	34.9540
Proposed model	**2.3067**	**2.0509**	**31.8723**

Table A.7 Comparison of the accuracy of the proposed single method and the hybrid method.

	Oct. 2020			Jan. 2021		
	MAE (MWh)	MAPE (%)	RMSE (MWh)	MAE (MWh)	MAPE (%)	RMSE (MWh)
f_1	3.1721	12.2135	3.6213	4.1201	14.0188	4.7050
f_2	2.8902	10.6859	3.2726	3.5560	12.0918	4.0571
f_3	2.6566	10.2822	3.1539	3.2135	10.9522	3.5577
Proposed model	**2.3087**	**9.1219**	**2.6017**	**2.5131**	**9.3312**	**2.8821**

	Apr. 2021			Jul. 2021		
	MAE (MWh)	MAPE (%)	RMSE (MWh)	MAE (MWh)	MAPE (%)	RMSE (MWh)
f_1	1.5965	10.1191	2.0715	1.0431	9.1334	1.4145
f_2	1.4480	9.284	1.7587	0.9349	9.2577	1.4899
f_3	1.5509	9.62312	1.9524	1.1144	9.3171	1.5563
Proposed model	**1.4745**	**8.8311**	**1.8151**	**0.8932**	**8.6422**	**1.4171**

Table A.8 The accuracy of the proposed hybrid forecasting model is compared with other forecasting methods.

	Oct. 2020			Jan. 2021		
	MAE (MWh)	MAPE (%)	RMSE (MWh)	MAE (MWh)	MAPE (%)	RMSE (MWh)
ARIMA	4.1926	16.3224	4.6664	5.1557	18.0434	5.6717
BPNN	3.9476	14.9221	4.6863	4.6068	16.2312	5.0465
Elman	3.4766	13.6016	3.8774	4.6810	16.2736	5.0625
EMD-RNN	3.2443	12.5512	3.5482	4.6769	16.0312	5.4744
EEMD-LSTM	2.8411	10.9820	3.3349	4.2461	14.9421	4.8493
EWT-LSTM	3.2334	12.3874	3.7839	4.2533	15.0122	4.7705
Proposed model	**2.3087**	**9.1243**	**2.6017**	**2.5131**	**9.3312**	**2.8821**

	Apr. 2021			Jul. 2021		
	MAE (MWh)	MAPE (%)	RMSE (MWh)	MAE (MWh)	MAPE (%)	RMSE (MWh)
ARIMA	2.2104	14.6081	2.9802	1.5174	14.2802	1.8369
BPNN	2.0045	13.1744	2.5935	1.5215	12.8551	2.1109
Elman	1.9384	12.7303	2.3553	1.2639	12.0555	1.7870
EMD-RNN	1.6212	10.0341	2.3478	1.0755	9.9221	1.5824
EEMD-LSTM	1.5597	9.5523	2.1446	1.0296	9.1023	1.4767
EWT-LSTM	1.5932	10.2210	1.9567	0.9482	9.3329	1.5507
Proposed model	**1.4745**	**8.8311**	**1.8151**	**0.8932**	**8.6422**	**1.4171**

Appendix B: Program code of the forecasting model.

The program code of the forecasting system presented in this thesis is attached here. This program was developed and executed under the execution environment of MATLAB R2019a, but it can also be run on MATLAB R2018a and later versions.

● DAD/distanceab

```
function [distance_aB]=distanceab(a,B)

%% Calculate the distance between two points
number_B = size(B,1);
distance_aB = [];
for i = 1:number_B
    if all(B(i,:))==0
        distance_aB(i,1)=0;
       % distance_aB(i,2)=0;
    else

    %distance_ab(i,1)=distance(coordinate_a(1),
coordinate_a(2),coordinate_b(i,1),coordinate_b
(i,2));

    distance_aB(i,1) =
distance(a(1),a(2),B(i,1),B(i,2),6371);
    end
End

function [ dist ] = distence( vecA, vecB )
    dist = (vecA-vecB)*(vecA-vecB)';
end
```

● DAD/kMeans

```
%[plain] view plain copy
%% The core procedure of kMeans, iteratively
solving the clustering center
    function [ centroids ] = kMeans( dataSet, k )
        [m,n] = size(dataSet);
        %Initialize the clustering center
        centroids = randCent(dataSet, k);
        subCenter = zeros(m,2);
       %Make a matrix of m*2, the first column
stores the category, the second column stores the
distance
        change = 1;%Determining whether to change
        while change == 1
            change = 0;
            %Calculate the distance for each set of
data
            for i = 1:m
                minDist = inf;
                minIndex = 0;
                for j = 1:k
                    dist= distence(dataSet(i,:),
centroids(j,:));
                    if dist < minDist
                        minDist = dist;
                        minIndex = j;
                    end
                end
                if subCenter(i,1) ~= minIndex
                    change = 1;
                    subCenter(i,:)=[minIndex,
minDist];
                end
            end
            %Recalculating clustering centers for
k classes

            for j = 1:k
                sum = zeros(1,n);
```

```
            r = 0;%Quantity
            for i = 1:m
                if subCenter(i,1) == j
                    sum = sum + dataSet(i,:);
                    r = r+1;
                end
            end
            centroids(j,:) = sum./r;
        end
    end
    c=[1 1 1;1 0 1;0 1 0;0 0 1;1 1 0;1 0 0;0 1 1;
0.667 0.667 1;
      1 0.5 0;0.5   0 0;0.5 0.5 0.5;0.47  0.53
0.6;];
     assignin('base','subCenter',subCenter)
    %% Complete the drawing
    hold on
    for i = 1:m
      for j = 1:k
        switch subCenter(i,1)

            case (j)
                plot(dataSet(i,1),
dataSet(i,2),'LineStyle','none','Marker','o','
MarkerSize',8,...

'MarkerFace',c(j,:),'MarkerEdge',[0,0,0],'Line
Width',2);  AA(i,:,j)=dataSet(i,:);
assignin('base','Group',AA)

        end
      end


plot(centroids(:,1),centroids(:,2),'+k','LineW
idth',2);
    xlabel('风速平均值 m/s');
    ylabel('风速标准差');
    end
end
```

● DAD/randCent

```
%% Get Random Center
function [ centroids ] = randCent( dataSet, k )
%Get the number of columns
     [m,n] = size(dataSet);
    centroids = zeros(k, n);
    for j = 1:n
        minJ = min(dataSet(:,j));
        rangeJ =
max(dataSet(:,j))-min(dataSet(:,j));
        % Generate random numbers on the interval
        centroids(:,j) = minJ+rand(k,1)*rangeJ;
    end
    en
```

DAD/main

```
clc,clear
%% step1
% Enter the serial number of the location to be
predicted
No=7;
A = xlsread('DADdata.xlsx');
centroids = kMeans1(A, 6);
group_i = subCenter(No,1);

%% step2
G=Group(:,:,group_i);
[d_aB]=distanceab(G(No,3:4),G(:,3:4));
place_i=find(all(d_aB~=0,2));
d_aB(all(d_aB==0,2),:)=[];
distanceaB=[place_i,d_aB];
Train50=distanceaB;Train100=distanceaB;

%Within 50km, directly as training data
Train50(all(Train50(:,2)>50,2),:)=[];
%Within 50km-100km, proceed to step3
Train100(all(Train100(:,2)>100|Train100(:,2)<5
0,2),:)=[];
```

```
%% step3

a=[]; % Time series of wind speed at the location
to be predicted
b=[]  % Data matrix within 50km-100km
delaymax=5;
m=length(a);
k=length(Train100(:,1));

for j=1:k
    for i=0:2*delaymax

cor(i+1,j)=corrcoef(a(delaymax+1:m-delaymax+1,:
),b(i:m+1,j));
    end
end
[b,d]=max(cor,[],2);
delay=d-1; %Delay time number
```

● GA/data_process

```
function [in,out]=data_process(data,num)
% Using 1-num as input and num+1 as output
n=length(data)-num;
for i=1:n
    x(i,:)=data(i:i+num);
end
in=x(:,1:end-1);
out=x(:,end);
```

● GA/my_fitness

```
function y = my_fitness(population)
% population is a random number [0,1] matrix, the
following operation changes the range to [-1,1]
%population = 2 * (population - 0.5);
p=size(population,1);
P=[population(:,1)*(100-50)+50,population(:,2)
*(200-100)+100,population(:,3)*(300-100)+100,p
opulation(:,4)*(0.9-0.1)+0.1];
for i=1:p
    y(i,:) =
lstm_fitness(round(P(i,1)),round(P(i,2)),round
(P(i,3)),population(i,4));
End
```

● GA/my_ga

```
function [best_fitness, elite, generation,
last_generation] = my_ga( ...
    number_of_variables, ...
    fitness_function, ...
    population_size, ...
    parent_number, ...
    mutation_rate, ...
    maximal_generation, ...
    minimal_cost ...
```

```
cumulative_probabilities =
cumsum((parent_number:-1:1) /
sum(parent_number:-1:1));

% Optimal adaptation
best_fitness = ones(maximal_generation, 1);
% Elite
elite = zeros(maximal_generation,
number_of_variables);
% Number of children
child_number = population_size -
parent_number; % Number of children per generation
% Initializing the population
population = rand(population_size,
number_of_variables);
%{
population=[floor(unifrnd(1,10,population_size,
1))*10,...

floor(unifrnd(50,200,population_size,1)),...

floor(unifrnd(100,300,population_size,1)),...

floor(unifrnd(1,9,population_size,1))/10];
%}
last_generation = 0; % Record the number of
generations when the loop is jumped out


% The latter code is in the for loop
for generation = 1 : maximal_generation
    % Calculate the fitness of all individuals
(matrix of population_size*1)
    cost = feval(fitness_function, population);

     % Sorting the fitness function values from
smallest to largest
    [cost, index] = sort(cost);
    population =
population(index(1:parent_number), :);
```

```
    best_fitness(generation) = cost(1); % Record
the optimal adaptation of this generation
    elite(generation, :) = population(1, :); %
Record the optimal solution for this generation
(elite)

    if best_fitness(generation) < minimal_cost
        last_generation = generation;
        break;
        last_generation = generation;
    end


    for child = 1:2:child_number

        mother = find(cumulative_probabilities >
rand, 1);
        father = find(cumulative_probabilities >
rand, 1);


        %  Randomly determine a chromosome
crossing point
        crossover_point =
ceil(rand*number_of_variables);

        mask1 = [ones(1, crossover_point), zeros(1,
number_of_variables - crossover_point)];
        mask2 = not(mask1);

        mother_1 = mask1 .* population(mother, :);
        mother_2 = mask2 .* population(mother, :);

        father_1 = mask1 .* population(father, :);
        father_2 = mask2 .* population(father, :);

        % Get the next generation
        population(parent_number + child, :) =
mother_1 + father_2;
        population(parent_number+child+1, :) =
mother_2 + father_1;

    end
   % Start of chromosomal variation
    mutation_population =
population(2:population_size, :);
    number_of_elements = (population_size - 1) *
number_of_variables;
    number_of_mutations =
ceil(number_of_elements * mutation_rate);

    mutation_points = ceil(number_of_elements *
rand(1, number_of_mutations));

    %  The selected genes are all replaced by a
random number, completing the mutation
    mutation_population(mutation_points) =
rand(1, number_of_mutations);

    population(2:population_size, :) =
mutation_population;
end
```

● GA/test_ga

```
[best_fitness, elite, generation,
last_generation] = my_ga(4, 'my_fitness', 5, 2,
0.05, 5, 0.05);

disp(last_generation);
%i_begin = last_generation - 9;
i_begin=1;
disp(best_fitness(i_begin:last_generation,:));
my_elite = elite(i_begin:last_generation,:);
%my_elite = 2 * (my_elite - 0.5);
my_elite=[round(elite(:,1)*(100-50)+50),round(
elite(:,2)*(200-100)+100),round(elite(:,3)*(30
0-100)+100),elite(:,4)*(0.9-0.1)+0.1];
disp(my_elite);
```

```
% Evolution of the optimal adaptation
figure
loglog(1:generation, best_fitness(1:generation),
'linewidth',2)
xlabel('Generation','fontsize',15);
ylabel('Best Fitness','fontsize',15);
set(gca,'fontsize',15,'ticklength',get(gca,'ti
cklength')*2);
% Evolution of the optimal solution
figure
semilogx(1 : generation, 2 * elite(1 :
generation, :) - 1)
xlabel('Generation','fontsize',15);
ylabel('Best Solution','fontsize',15);
set(gca,'fontsize',15,'ticklength',get(gca,'ti
cklength')*2);
```

● ACO/yiqunjulei

```
clc;
clear;
X = load('data.txt');
[N,n]=size(X);     % N = number of test samples;
n = number of attributes of test samples;
K = 4;             % K = Number of groups; ;
R = 100;           % R = Number of Ants;
t_max = 1000;      % t_max =Maximum number of
iterations;

c = 10^-2;
%Pheromone matrix, N*K matrix with initial value
of 0.01 (number of samples * number of clusters)
tau = ones(N,K) * c;
q = 0.9;               % Threshold q
rho = 0.1;             % Evaporation rate
best_solution_function_value = inf;
tic
t = 1;

while ((best_solution_function_value>=19727))

   % path identifier character: identifies the
path of each ant
   solution_string = zeros(R,N+1);
   for i = 1 : R
      r = rand(1,N);
      for g = 1 : N
         if r(g) < q
            tau_max = max(tau(g,:));
            Cluster_number =
find(tau(g,:)==tau_max);
            solution_string(i,g) =
Cluster_number(1);
         else
            sum_p = sum(tau(g,:));
            p = tau(g,:) / sum_p;
            for u = 2 : K
               p(u) = p(u) + p(u-1);
            end
            rr = rand;
            for s = 1 : K
               if (rr <= p(s))
                  Cluster_number = s;
                  solution_string(i,g) =
Cluster_number;
                  break;
               end
            end
         end
      end
   end

   % Calculate the clustering center
   weight = zeros(N,K);
      for h = 1:N
         Cluster_index = solution_string(i,h);
         weight(h,Cluster_index) = 1;
      end

      cluster_center = zeros(K,n);  % clustering
centers (number of clusters K centers)
      for j = 1:K
         for v = 1:n
            sum_wx = sum(weight(:,j).*X(:,v));
            sum_w = sum(weight(:,j));
            if sum_w==0
              cluster_center(j,v) =0
               continue;
            % The number of samples in this class
is not zero, then the value of the cluster center
is taken as the average of the values of the sample
attributes
            else
            cluster_center(j,v) = sum_wx/sum_w;
            end
         end
      end
```

```
    % Calculate the sum of the mean squared
deviation of each attribute of each sample point
to its corresponding cluster center, and store the
value in the last bit of solution_string
     F = 0;
     for j= 1:K
        for ii = 1:N
           Temp=0;
           if solution_string(i,ii)==j;
              for v = 1:n
                 Temp =
((abs(X(ii,v)-cluster_center(j,v))).^2)+Temp;
              end
              Temp = sqrt(Temp);
           end
          F = (Temp)+F;
        end
     end

      solution_string(i,end) = F;

   end
   [fitness_ascend,solution_index] =
sort(solution_string(:,end),1);
   solution_ascend =
[solution_string(solution_index,1:end-1)
fitness_ascend];

   pls = 0.1;
   L = 2;
   % Localized optimization search procedure
   solution_temp = zeros(L,N+1);
   k = 1;
   while(k <= L)
         solution_temp(k,:) =
solution_ascend(k,:);
         rp = rand(1,N);
         for i = 1:N
            if rp(i) <= pls
               current_cluster_number =
setdiff([1:K],solution_temp(k,i));
               rrr=randint(1,1,[1,K-1]);
               change_cluster =
current_cluster_number(rrr);
               solution_temp(k,i) =
change_cluster;
            end
         end

       % Calculate temporary clustering centers
         solution_temp_weight = zeros(N,K);
         for h = 1:N
            solution_temp_cluster_index =
solution_temp(k,h);

solution_temp_weight(h,solution_temp_cluster_i
ndex) = 1;
         end

         solution_temp_cluster_center =
zeros(K,n);
         for j = 1:K
            for v = 1:n
               solution_temp_sum_wx =
sum(solution_temp_weight(:,j).*X(:,v));
               solution_temp_sum_w =
sum(solution_temp_weight(:,j));
               if solution_temp_sum_w==0

solution_temp_cluster_center(j,v) =0;
               continue;
               else

solution_temp_cluster_center(j,v) =
solution_temp_sum_wx/solution_temp_sum_w;
               end
            end
        end
```

```
        % Calculate the sum of mean squared
differences Ft for each attribute of each sample
point to its corresponding provisional clustering
center.
        solution_temp_F = 0;
        for j= 1:K
            for ii = 1:N
                st_Temp=0;
                if solution_temp(k,ii)==j;
                    for v = 1:n
                        st_Temp =
((abs(X(ii,v)-solution_temp_cluster_center(j,v)
)).^2)+st_Temp;
                    end
                    st_Temp = sqrt(st_Temp);
                end
                solution_temp_F =
(st_Temp)+solution_temp_F;
            end
        end
       solution_temp(k,end) = solution_temp_F;

        if solution_temp(k,end) <=
solution_ascend(k,end)
            solution_ascend(k,:) =
solution_temp(k,:);
        end

        if
solution_ascend(k,end)<=best_solution_function
_value
            best_solution =
solution_ascend(k,:);
        end
     k = k+1;
     end

   % Update the message count matrix with the best
L paths
    tau_F = 0;
    for j = 1:L
      tau_F = tau_F + solution_ascend(j,end);
    end
    for i = 1 : N
      tau(i,best_solution(1,i)) = (1 - rho) *
tau(i,best_solution(1,i)) + 1/ tau_F;

    end
    t=t+1
    best_solution_function_value =
solution_ascend(1,end);
    best_solution_function_value
end
toc;
clc
t
time
cluster_center
best_solution = solution_ascend(1,1:end-1);
IDY=ctranspose(best_solution)
best_solution_function_value =
solution_ascend(1,end)
%分类结果显示
plot3(cluster_center(:,1),cluster_center(:,2),
cluster_center(:,3),'o');grid;box
title('ACO 结果(R=100,t=1000)')
xlabel('X')
ylabel('Y')
zlabel('Z')
YY=[1 2 3 4];
index1 = find(YY(1) == best_solution)
index2 = find(YY(2) == best_solution)
index3 = find(YY(3) == best_solution)
index4 = find(YY(4) == best_solution)
line(X(index1,1),X(index1,2),X(index1,3),'line
style','none','marker','*','color','g');
line(X(index2,1),X(index2,2),X(index2,3),'line
style','none','marker','*','color','r');
line(X(index3,1),X(index3,2),X(index3,3),'line
style','none','marker','+','color','b');
```

```
line(X(index4,1),X(index4,2),X(index4,3),'line
style','none','marker','s','color','b');
rotate3d
```


● PE/pec

```
function [pe hist] = pec(y,m,t)

%  Input:   y: time series;
%          m: order of permuation entropy
%          t: delay time of permuation entropy,

% Output:
%          pe:    permuation entropy
%          hist:  the histogram for the order
distribution

ly = length(y);
permlist = perms(1:m);
c(1:length(permlist))=0;

 for j=1:ly-t*(m-1)
     [a,iv]=sort(y(j:t:j+t*(m-1)));
     for jj=1:length(permlist)
         if (abs(permlist(jj,:)-iv))==0
             c(jj) = c(jj) + 1 ;
         end
     end
 end

hist = c;

c=c(find(c~=0));
p = c/sum(c);
pe = -sum(p .* log(p));
```

● Ultra short-term wind speed forecast/ceemd

```
function [modes its]=ceemd(x,Nstd,NR,MaxIter)
%   Input:
%   x: signal to decompose
%   Nstd: noise standard deviation
%   NR: number of realizations
%   MaxIter: maximum number of sifting iterations
allowed.

%  Output:
%  modes: contain the obtained modes in a matrix
with the rows being the modes
%   its: contain the sifting iterations needed for
each mode for each realization (one row for each
realization)

x=x(:)';
desvio_x=std(x);
x=x/desvio_x;

modes=zeros(size(x));
temp=zeros(size(x));
aux=zeros(size(x));
acum=zeros(size(x));
iter=zeros(NR,round(log2(length(x))+5));

for i=1:NR
   white_noise{i}=randn(size(x));%creates the
noise realizations
end;

for i=1:NR

modes_white_noise{i}=emd(white_noise{i});%calc
ulates the modes of white gaussian noise
end;

for i=1:NR %calculates the first mode
   temp=x+Nstd*white_noise{i};
   [temp, o,
it]=emd(temp,'MAXMODES',1,'MAXITERATIONS',MaxI
ter);
   temp=temp(1,:);
   aux=aux+temp/NR;
   iter(i,1)=it;
end;

modes=aux; %saves the first mode
k=1;
aux=zeros(size(x));
acum=sum(modes,1);

while
nnz(diff(sign(diff(x-acum))))>2 %calculates the
rest of the modes
   for i=1:NR
      tamanio=size(modes_white_noise{i});
      if tamanio(1)>=k+1
         noise=modes_white_noise{i}(k,:);
         noise=noise/std(noise);
         noise=Nstd*noise;
         try
            [temp, o,
it]=emd(x-acum+std(x-acum)*noise,'MAXMODES',1,
'MAXITERATIONS',MaxIter);
            temp=temp(1,:);
         catch
            it=0;
            temp=x-acum;
         end;
      else
         [temp, o,
it]=emd(x-acum,'MAXMODES',1,'MAXITERATIONS',Ma
xIter);
         temp=temp(1,:);
      end;
      aux=aux+temp/NR;
   iter(i,k+1)=it;
   end;
```

```
    modes=[modes;aux];
    aux=zeros(size(x));
    acum=zeros(size(x));
    acum=sum(modes,1);
    k=k+1;
end;
modes=[modes;(x-acum)];
[a b]=size(modes);
iter=iter(:,1:a);
modes=modes*desvio_x;
its=iter;
```

- Ultra short-term wind speed forecast/main1_ceemd

```
%% Data Loading
s=std(f);
aver=mean(f);
u=emd(f);%ceemd
Nstd = 0.3;
NR = 1;
MaxIter = 500;

% [u ,~]=eemd(f,Nstd,NR,MaxIter);%eemd
% [u ,~]=ceemd(f,Nstd,NR,MaxIter);%ceemd

K=size(u,1);

figure
subplot(K+1,1,1)
plot(f)
ylabel('原始')

for i=2:K+1
    subplot(K+1,1,i)
    plot(u(i-1,:))
    ylabel(['IMF',num2str(i-1)])
end
ylabel('res')
save ceemd_data pre_i pre_n tran_n in_n out_n A
f u s aver -append
```

● Ultra short-term wind speed forecast/main2_LSTM

```
clc;clear;format compact;
%%
load ceemd_data
data=sum(u);%All sequences added up
[x,y]=data_process(data,in_n);%Previous in_n
moments Predict the next moment
[xs,mappingx]=mapminmax(x',0,1);x=xs';
[ys,mappingy]=mapminmax(y',0,1);y=ys';

n=size(x,1);
%m=round(n*0.9);%Training the first 90% and
making predictions for the last 10%

m=tran_n-in_n-out_n+1;
XTrain=x(1:m,:)';
XTest=x(m+1:end,:)';
YTrain=y(1:m,:)';
YTest=y(m+1:end,:)';

numFeatures = size(XTrain,1);
numResponses = 1;
numHiddenUnits = 50;
layers = [
    sequenceInputLayer(numFeatures)
    lstmLayer(numHiddenUnits)
    fullyConnectedLayer(numResponses)
    regressionLayer];
options = trainingOptions('adam', ...
    'MaxEpochs',200, ...
    'MiniBatchSize',200, ...
    'GradientThreshold',1, ...
    'InitialLearnRate',0.001, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropPeriod',125, ...
    'LearnRateDropFactor',0.2, ...
    'Verbose',0, ...
'Plots','training-progress');
net =
trainNetwork(XTrain,YTrain,layers,options);
numTimeStepsTest = size(XTest,2);
for i = 1:numTimeStepsTest
    [net,YPred(:,i)] =
predictAndUpdateState(net,XTest(:,i),'Executio
nEnvironment','cpu');

YPred(:,i)=sim(LSTM,[s;YPred(:,i);y(m+i,:)']);
end

% inverse normalization
predict_value=mapminmax('reverse',YPred,mappin
gy);
true_value=mapminmax('reverse',YTest,mappingy)
;
save 结果/lstm predict_value true_value
delete(findall(0));
%%
load 结果/lstm
disp('结果分析')
rmse=sqrt(mean((true_value-predict_value).^2))
;
disp(['根均方差(RMSE)：',num2str(rmse)])

mae=mean(abs(true_value-predict_value));
disp(['平均绝对误差（MAE)：',num2str(mae)])

mape=mean(abs(true_value-predict_value)./true_
value);
disp(['平均相对百分误差（MAPE)：
',num2str(mape*100),'%'])

fprintf('\n')

figure
plot(true_value,'-*','linewidth',1)
hold on
plot(predict_value,'--.','linewidth',1)
legend('实际值','预测值')
```

```
grid on
title('LSTM')
```


● Ultra short-term wind speed forecast/main3_ceemd_lstm

```
clear;format compact;
rng('default')
%% Data pre-processing
load ceemd_data
imf=u;

c=size(imf,1);
pre_result=[];
true_result=[];
%% Modeling of each component
for i=1:c
disp(['对第',num2str(i),'个分量建模'])
[x,y]=data_process(imf(i,:),in_n);
% Normalization
[xs,mappingx]=mapminmax(x',0,1);x=xs';
[ys,mappingy]=mapminmax(y',0,1);y=ys';
% Segmentation Data
n=size(x,1);

m=tran_n-in_n-out_n+1;
XTrain=x(1:m,:)';
XTest=x(m+1:end,:)';
YTrain=y(1:m,:)';
YTest=y(m+1:end,:)';

numFeatures = size(XTrain,1);
numResponses = 1;
numHiddenUnits = 50;
layers = [ ...
    sequenceInputLayer(numFeatures)
    lstmLayer(numHiddenUnits)
    fullyConnectedLayer(numResponses)
    regressionLayer];
options = trainingOptions('adam', ...
    'MaxEpochs',200, ...
    'GradientThreshold',1, ...
    'InitialLearnRate',0.005, ...
```

```
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropPeriod',125, ...
    'LearnRateDropFactor',0.2, ...
    'Verbose',0, ...
'Plots','training-progress');
net_inf{i} =
trainNetwork(XTrain,YTrain,layers,options);
numTimeStepsTest = size(XTest,2);
for j = 1:numTimeStepsTest
    [net_inf{i},YPred(:,j)] =
predictAndUpdateState(net_inf{i},XTest(:,j),'E
xecutionEnvironment','cpu');

YPred(:,j)=sim(LSTM,[s;YPred(:,j);y(m+j,:)']);
end

% inverse normalization
pre_value=mapminmax('reverse',YPred,mappingy);
true_value=mapminmax('reverse',YTest,mappingy)
;
pre_result=[pre_result;pre_value];
true_result=[true_result;true_value];
delete(findall(0));
end

%% Sum of predicted results of each component
true_value=sum(true_result);
predict_value=abs(sum(pre_result));
save 结果/ceemd_lstm predict_value true_value
net_inf

%%
load 结果/ceemd_lstm
disp('结果分析')
rmse=sqrt(mean((true_value-predict_value).^2))
;
disp(['根均方差(RMSE)：',num2str(rmse)])

mae=mean(abs(true_value-predict_value));
disp(['平均绝对误差（MAE)：',num2str(mae)])

mape=mean(abs(true_value-predict_value)./true_
value);
disp(['平均相对百分误差（MAPE)：
',num2str(mape*100),'%'])

fprintf('\n')
figure
plot(true_value,'-*','linewidth',1)
hold on
plot(predict_value,'--.','linewidth',1)
legend('实际值','预测值')
grid on
title('CEEMD-LSTM')
```

● Ultra short-term wind speed forecast/main

```
clc;clear;
format compact;
close all;
tic
%%
global pre_i
global pre_n
global tran_n
global in_n
global out_n
global A
global f

 pre_i=26065; %The serial number of the wind speed
to be predicted (starting from the serial number)
pre_n=4320;  %Number of predicted wind speeds
tran_n=8648; %Number of training data
in_n=4;      %Number of input layer nodes
out_n=1;     %Number of output layer nodes
loc_n=3;     %number of locations

%Data determined using DAD
 A=xlsread('DATA.xls');

 %f=A(per_i-tran_n:per_i,6:6+loc_n);A(per_i-tr
an_n:per_i,2);%When there is another location

  f=A(pre_i-tran_n:pre_i+pre_n-1,2);
  run('main1_ceemd.m')
  run('main2_lstm.m');
  %run('main3_ceemd_lstm');
 toc
```

● Ultra short-term wind speed forecast/loop_main1_ceemd

```
s=std(f);
aver=mean(f);
u=emd(f);%emd

Nstd = 0.3;
NR = 1;
MaxIter = 500;

K=size(u,1);
figure
subplot(K+1,1,1)
plot(f)
ylabel('原始')

for i=2:K+1
    subplot(K+1,1,i)
    plot(u(i-1,:))
    ylabel(['IMF',num2str(i-1)])
end
ylabel('res')
save loop_ceemd_data pre_i pre_en pre_n tran_n
in_n out_n A f u s aver -append
```

- Ultra short-term wind speed forecast/loop_main2_LSTM

```
format compact;
%%
load loop_ceemd_data
data=sum(u);
[x,y]=data_process(data,in_n);

[xs,mappingx]=mapminmax(x',0,1);x=xs';
[ys,mappingy]=mapminmax(y',0,1);y=ys';

n=size(x,1);
m=n-pre_en;
XTrain=x(1:m,:)';
XTest=x(m+1:end,:)';
YTrain=y(1:m,:)';
YTest=y(m+1:end,:)';

numFeatures = size(XTrain,1);
numResponses = 1;
numHiddenUnits = 50;
layers = [
    sequenceInputLayer(numFeatures)
    lstmLayer(numHiddenUnits)
    fullyConnectedLayer(numResponses)
    regressionLayer];
options = trainingOptions('adam', ...
    'MaxEpochs',200, ...
    'MiniBatchSize',100, ...
    'GradientThreshold',1, ...
    'InitialLearnRate',0.001, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropPeriod',125, ...
    'LearnRateDropFactor',0.2, ...
    'Verbose',0, ...
'Plots','training-progress');

net =
trainNetwork(XTrain,YTrain,layers,options);

numTimeStepsTest = size(XTest,2);
for i = 1:numTimeStepsTest
    [net,YPred(:,i)] =
predictAndUpdateState(net,XTest(:,i),'Executio
nEnvironment','cpu');

YPred(:,i)=sim(LSTM,[s;YPred(:,i);y(m+i,:)']);
end

%%Save all the neural networks
net_loop{vn}=net;

predict_value=mapminmax('reverse',YPred,mappin
gy);
true_value=mapminmax('reverse',YTest,mappingy)
;

delete(findall(0));
%%
% Error in calculating the output predicted wind
speed
rmse=mean((true_value(pre_en)-abs(predict_valu
e(pre_en))).^2);
mae=mean(abs(true_value(pre_en)-abs(predict_va
lue(pre_en))));
mape=mean(abs(true_value(pre_en)-abs(predict_v
alue(pre_en)))./true_value(pre_en));
```

● Ultra short-term wind speed forecast/loop_main3_ceemd_lstm

```
format compact;
rng('default')
%% Data pre-processing
load loop_ceemd_data
imf=u;

c=size(imf,1);
pre_result=[];
true_result=[];
%% Modeling of each component
for i=1:c
disp(['对第',num2str(i),'个分量建模'])
[x,y]=data_process(imf(i,:),4);

[xs,mappingx]=mapminmax(x',0,1);x=xs';
[ys,mappingy]=mapminmax(y',0,1);y=ys';

n=size(x,1);

m=n-pre_en;
XTrain=x(1:m,:)';
XTest=x(m+1:end,:)';
YTrain=y(1:m,:)';
YTest=y(m+1:end,:)';

numFeatures = size(XTrain,1);
numResponses = 1;
numHiddenUnits = 50;
layers = [ ...
    sequenceInputLayer(numFeatures)
    lstmLayer(numHiddenUnits)
    fullyConnectedLayer(numResponses)
    regressionLayer];
options = trainingOptions('adam', ...
    'MaxEpochs',200, ...
    'GradientThreshold',1, ...
    'InitialLearnRate',0.005, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropPeriod',125, ...
    'LearnRateDropFactor',0.2, ...
    'Verbose',0, ...
'Plots','training-progress');
net =
trainNetwork(XTrain,YTrain,layers,options);
net_inf{i}=net;
numTimeStepsTest = size(XTest,2);
for j = 1:numTimeStepsTest
    [net_inf{i},YPred(:,j)] =
predictAndUpdateState(net_inf{i},XTest(:,j),'E
xecutionEnvironment','cpu');

YPred(:,j)=sim(LSTM,[s;YPred(:,j);y(m+j,:)']);
end


pre_value=mapminmax('reverse',YPred,mappingy);
true_value=mapminmax('reverse',YTest,mappingy)
;
pre_result=[pre_result;pre_value];
true_result=[true_result;true_value];
delete(findall(0));
end

net_loop{vn}=net_inf;

%% The results of each component prediction are
summed
true_value=sum(true_result);
predict_value=sum(pre_result);

%%
%Error in calculating the output predicted wind
speed
rmse=mean((true_value(pre_en)-abs(predict_valu
e(pre_en))).^2);
mae=mean(abs(true_value(pre_en)-abs(predict_va
lue(pre_en))));
```

```
mape=mean(abs(true_value(pre_en)-abs(predict_v
alue(pre_en))).\true_value(pre_en));
```


● Ultra short-term wind speed forecast/main4_compare

```
clc;clear;
format compact;
close all;
tic

%%
global pre_i
global pre_n
global pre_en
global tran_n
global in_n
global out_n
global A
global f

pre_i=26065;
pre_en=50;
pre_n=3;
tran_n=3000;
in_n=4;
out_n=1;
loc_n=3;
predict_v=zeros(1,pre_n);
true_v=zeros(1,pre_n);
accuracy=zeros(3,pre_n);

A=xlsread('DATA.xls');

for vn=1:pre_n

  %save loop_ceemd_data vn -append
  %f=A(per_i-tran_n:per_i,6:6+loc_n);A(per_i-t
ran_n:per_i,2);

  f=A(pre_i-tran_n+vn-1:pre_i+out_n+vn-2,2);
```

```
  disp(['对第',num2str(vn),'个风速进行建模并预测
'])
  run('loop_main1_ceemd.m');
  %run('loop_main2_lstm.m');
  run('loop_main3_ceemd_lstm');

  predict_v(1,vn)=predict_value((pre_en));
  true_v(1,vn)=true_value(pre_en);
  accuracy(1,vn)=rmse;
  accuracy(2,vn)=mae;
  accuracy(3,vn)=mape;
end
save 结果/loop_ceemd_lstm predict_v true_v
accuracy net_loop

%%
load 结果/loop_ceemd_lstm
disp('结果分析')
RMSE=sqrt(mean(accuracy(1,:)));
disp(['根均方差(RMSE)：',num2str(RMSE)])

MAE=mean(accuracy(2,:));
disp(['平均绝对误差（MAE)：',num2str(MAE)])

MAPE=mean(accuracy(3,:));
disp(['平均相对百分误差（MAPE)：
',num2str(MAPE*100),'%'])

fprintf('\n')

figure
plot(true_v,'-*','linewidth',1)
hold on
plot(predict_v,'-s','linewidth',1)
legend('实际值','预测值')
grid on
title('CEEMD-LSTM')
toc
```

- Ultra short-term wind speed forecast/main4_compare

```
clc;clear;close all
%%
lstm=load('结果/lstm.mat');%1
ceemd_lstm=load('结果/ceemd_lstm.mat');%2
%% 1
disp('1-结果分析-lstm')
result(lstm.true_value,lstm.predict_value)
fprintf('\n')
%% 2
disp('2-结果分析-ceemd-lstm')
result(ceemd_lstm.true_value,ceemd_lstm.predic
t_value)
fprintf('\n')

 %%
figure
plot(lstm.true_value)
hold on;grid on
plot(lstm.predict_value)
plot(ceemd_lstm.predict_value)
legend('真实值','lstm 预测值','ceemd-lstm 预测值')
title('各算法预测效果对比')
ylabel('值')
xlabel('测试集样本')
```

● Ultra short-term wind speed forecast/sd_months

```
clc,clear;

C=xlsread('DATA');

p=floor(size(C(:,2),1)/12);

for i=1:12

    f(i,:)=C(1+p*(i-1):p*i,2);
    g(i,:)=std(f(i,:));
    m(i,:)=mean(f(i,:));

D(1+p*(i-1):p*i,1)=C(1+p*(i-1):p*i,2)/m(i,:)*m
1(i,:);


end
```

● 24-hour wind speed forecast/data_process24

```
function [in,out]=data_process24(data,nin,nout)
% Using 1--nin as input and nin+1--nout as output
n=length(data)-nin-nout+1;
for i=1:n
    x(i,:)=data(i:i+nin+nout-1);
end
in=x(:,1:end-nout);
out=x(:,end-nout+1:end);
```

● 24-hour wind speed forecast/data_process24ture

```
function [S]=data_process24ture(data)
% Using 1--nin as input and nin+1--nout as output
[m n]=size(data);
data1=[data(1,:),data(2:m,n)'];
N=m+n-24;
for i=1:N
    S(:,i)=data1(i:i+23)';
End
```

● 24-hour wind speed forecast/main1_ceemd24

```
s=std(f);
aver=mean(f);
u=emd(f);%emd

Nstd = 0.3;
NR = 1;
MaxIter = 500;


K=size(u,1);

figure
subplot(K+1,1,1)
plot(f)
ylabel('原始')

for i=2:K+1
    subplot(K+1,1,i)
    plot(u(i-1,:))
    ylabel(['IMF',num2str(i-1)])
end
ylabel('res')

save ceemd_data24 pre_i pre_n tran_n in_n out_n
A f u s aver -append
```

● 24-hour wind speed forecast/main2_lstm24

```
clc;clear;format compact;
%%
load ceemd_data24
data=sum(u);
[x,y]=data_process24(data,in_n,out_n);
[xs,mappingx]=mapminmax(x',0,1);x=xs';
[ys,mappingy]=mapminmax(y',0,1);y=ys';

n=size(x,1);
m=n-pre_n;
XTrain=x(1:m,:)';
XTest=x(m+out_n:end+out_n-24,:)';
YTrain=y(1:m,:)';
YTest=y(m+out_n:end,:)';

%[YTest24]=data_process24ture(y(m+1:end,1)');
%YTest24=y(m+1:end,:)';
%true_value24
YTest24=mapminmax('reverse',YTest,mappingy);
[true_value24]=data_process24ture(YTest24);

numFeatures = size(XTrain,1);
numResponses = size(YTrain,1);
numHiddenUnits = 100;
layers = [
    sequenceInputLayer(numFeatures)
    lstmLayer(numHiddenUnits)
    fullyConnectedLayer(numResponses)
    regressionLayer];
options = trainingOptions('adam', ...
    'MaxEpochs',200, ...
    'MiniBatchSize',200, ...
    'GradientThreshold',1, ...
    'InitialLearnRate',0.005, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropPeriod',125, ...
    'LearnRateDropFactor',0.2, ...
    'Verbose',0, ...
    'Plots','training-progress');
net =
trainNetwork(XTrain,YTrain,layers,options);
delete(findall(0));
numTimeStepsTest = size(XTest,2);

 XTest24(:,:,1)=XTest;
for z= 1:24/numResponses
   disp(['第',num2str(z),'步建模并预测'])

    for j = 1:numTimeStepsTest
      [net,YPred(:,j,z)] =
predictAndUpdateState(net,XTest24(:,j,z),'Exec
utionEnvironment','cpu');

     for i=1:numResponses

YPred(i,j,z)=sim(LSTM,[s;YPred(i,j,z);y(m+out_
n+z-1+(z-1)*4,i)']);%y(m+out_n+j-1,i)'
YTest24(i,j)
     end

    end

XTest24(:,:,z+1)=[XTest24(numResponses+1:end,:,
z);YPred(:,:,z)];
end

predict_value24=mapminmax('reverse',YPred(:,:,
1),mappingy);
for i=1:24/numResponses-1

predict_value24=[predict_value24;mapminmax('re
verse',YPred(:,:,i+1),mappingy)];
end

disp('结果分析')
for i=1:24
rmse(i)=sqrt(mean((true_value24(i,:)-predict_v
alue24(i,:)).^2));
```

```
mae(i)=mean(abs(true_value24(i,:)-predict_valu
e24(i,:)));
mape(i)=mean(abs(true_value24(i,:)-predict_val
ue24(i,:))./true_value24(i,:));
disp(['第',num2str(i),'小时的根均方差(RMSE):
',num2str(rmse(i)),'  平均绝对误差（MAE):
',num2str(mae(i)),'  平均相对百分误差（MAPE):
',num2str(mape(i)*100),'%'])

end
fprintf('\n')
for i=1:24/numResponses
rmse_step(i)=mean(rmse(1+(i-1)*numResponses:i*
numResponses));
mae_step(i)
=mean(mae(1+(i-1)*numResponses:i*numResponses))
;
mape_step(i)=mean(mape(1+(i-1)*numResponses:i*
numResponses));
disp(['第',num2str(i),'步的根均方差(RMSE):
',num2str(rmse_step(i)),'  平均绝对误差（MAE):
',num2str(mae_step(i)),'  平均相对百分误差（MAPE):
',num2str(mape_step(i)*100),'%'])

end

figure(2)
plot(mape,'-*','linewidth',1)
%legend('')
grid on
title('MAPE%')
figure(3)
plot(rmse,'r-*','linewidth',1)
%legend('')
grid on
title('RMSE')
figure(4)
plot(mae,'g-*','linewidth',1)
%legend('')
grid on
title('MAE')
%%
```

● 24-hour wind speed forecast/main3_ceemd_lstm24

```
clc;clear;format compact;
load ceemd_data24

imf=u;

c=size(imf,1);
[x0,y0]=data_process24(imf(1,:),in_n,out_n);

n=size(x0,1);
m=n-pre_n;

pre_result24=zeros(24,n-m-23);
true_result24=zeros(24,n-m-23);


%% Modeling of each component
for i=1:c
disp(['对第',num2str(i),'个分量建模'])
[x,y]=data_process24(imf(i,:),in_n,out_n);
%归一化
[xs,mappingx]=mapminmax(x',0,1);x=xs';
[ys,mappingy]=mapminmax(y',0,1);y=ys';

%划分数据
XTrain=x(1:m,:)';
XTest=x(m+out_n:end+out_n-24,:)';
YTrain=y(1:m,:)';
YTest=y(m+out_n:end,:)';

YTest24=mapminmax('reverse',YTest,mappingy);
[true_value24]=data_process24ture(YTest24);

numFeatures = size(XTrain,1);
numResponses = size(YTrain,1);
numHiddenUnits = 50;
layers = [ ...
    sequenceInputLayer(numFeatures)
    lstmLayer(numHiddenUnits)
    fullyConnectedLayer(numResponses)
    regressionLayer];
options = trainingOptions('adam', ...
    'MaxEpochs',200, ...
    'GradientThreshold',1, ...
    'InitialLearnRate',0.005, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropPeriod',125, ...
    'LearnRateDropFactor',0.2, ...
    'Verbose',0, ...
'Plots','training-progress');
net_imf{i} =
trainNetwork(XTrain,YTrain,layers,options);
delete(findall(0));
numTimeStepsTest = size(XTest,2);

XTest24(:,:,1)=XTest;
for j= 1:24/numResponses

   disp(['第',num2str(j),'步建模并预测'])
    for l = 1:numTimeStepsTest

      [net_imf{i},YPred(:,l,j)] =
predictAndUpdateState(net_imf{i},XTest24(:,l,j)
,'ExecutionEnvironment','cpu');
      for z=1:numResponses

YPred(z,l,j)=sim(LSTM,[s;YPred(z,l,j);y(m+out_
n+j-1+(j-1)*4,z)']);%y(m+out_n+j-1,i)'
YTest24(i,j)
      end

    end

XTest24(:,:,j+1)=[XTest24(numResponses+1:end,:,
j);YPred(:,:,j)];
end

predict_value24=mapminmax('reverse',YPred(:,:,
1),mappingy);
```

```
for o=1:24/numResponses-1

predict_value24=[predict_value24;mapminmax('re
verse',YPred(:,:,o+1),mappingy)];
end

% The results of each component prediction are
summed
pre_result24=pre_result24+predict_value24;
true_result24=true_result24+true_value24;

end
save 结果/ceemd_lstm24 pre_result24
true_result24

disp('结果分析')
for i=1:24
rmse(i)=sqrt(mean((true_result24(i,:)-pre_resu
lt24(i,:)).^2));
mae(i)=mean(abs(true_result24(i,:)-pre_result2
4(i,:)));
mape(i)=mean(abs(true_result24(i,:)-pre_result
24(i,:))./true_result24(i,:));
disp(['第',num2str(i),'小时的根均方差(RMSE):
',num2str(rmse(i)),'  平均绝对误差（MAE):
',num2str(mae(i)),'  平均相对百分误差（MAPE):
',num2str(mape(i)*100),'%'])
end
fprintf('\n')
for i=1:24/numResponses
rmse_step(i)=mean(rmse(1+(i-1)*numResponses:i*
numResponses));
mae_step(i)
=mean(mae(1+(i-1)*numResponses:i*numResponses))
;
mape_step(i)=mean(mape(1+(i-1)*numResponses:i*
numResponses));
disp(['第',num2str(i),'步的根均方差(RMSE):
',num2str(rmse_step(i)),'  平均绝对误差（MAE):
',num2str(mae_step(i)),'  平均相对百分误差（MAPE):
',num2str(mape_step(i)*100),'%'])

end

%%
figure(2)
plot(mape,'-*','linewidth',1)
grid on
title('MAPE%')
figure(3)
plot(rmse,'r-*','linewidth',1)
grid on
title('RMSE')
figure(4)
plot(mae,'g-*','linewidth',1)
grid on
title('MAE')
Toc
```

● 24-hour wind speed forecast/main24

```
clc;clear;
format compact;
close all;
tic

%%
global pre_i
global pre_n
global tran_n
global in_n
global out_n
global A
global f

pre_dn=20;          % Forecast days
tran_dn=90;         % days of training data
pre_i=3500;
in_n=12;
out_n=4;
pre_n=pre_dn*24;
tran_n=tran_dn*24;
loc_n=3;

A=xlsread('稚内 1.xlsx');
f=A(pre_i-tran_n:pre_i+pre_n-1,1);

  run('main1_ceemd24.m');
  run('main2_lstm24.m');
  %run('main3_ceemd_lstm24.m');

  save 结果/lstm24 predict_value24 true_value24
Toc
```

● wind power forecast/data_conversion

```
clear;
close all;
clc;
A=xlsread('DATA');

%%Set cut-in air speed and rated air speed
D=A;D(D(:,3)==0,:)=[];
x=D(1:2000,1);
y=D(1:2000,3);


xcutin=A;xcutin(xcutin(:,3)>0,:)=[];
Vcutin=max(xcutin(:,1));
Vrated=20;
```

● wind power forecast/data_processVP

```
function [out]=data_processVP(data,num)

% Use 1-num as input and num+1 as output
n=length(data)-num+1;
for i=1:n
   x(i,:)=data(i:i+num-1);
end
out=x;
```

● wind power forecast/VP

```
function [Pout]=VP(Vin)
%% Data Loading
global A N
D=A;D(D(:,3)==0,:)=[];
x=D(1:end,1);
y=D(1:end,3);
n=size(x,1);

xcutin=A;xcutin(xcutin(:,3)>0,:)=[];

Vcutin=3;
%Vcutin=max(xcutin(:,1));
Vrated=20;
% Training the first 90% and making predictions
for the last 10%
m=round(n*0.9);
XTrain=x(1:m,:)';
XTest=x(m+1:end,:)';
YTrain=y(1:m,:)';
YTest=y(m+1:end,:)';

n=length(Vin);
for i=1:n
  if Vin(:,i)<=Vcutin
     Pout(:,i)=0;
  else
     [netvp,Pout(:,i)] =
predictAndUpdateState(netvp,Vin(:,i));
  end
End
```

- wind power forecast/main1_ceemd

```
s=std(fv);
aver=mean(fv);
u1=emd(fv);
u2=emd(fp);%emd

Nstd = 0.3;
NR = 1;
MaxIter = 500;

% [u ,~]=eemd(f,Nstd,NR,MaxIter);%eemd
% [u ,~]=ceemd(f,Nstd,NR,MaxIter);%ceemd

K1=size(u1,1);
K2=size(u2,1);

figure(1)
subplot(K1+1,1,1)
plot(fv)
ylabel('原始')

for i=2:K1+1
    subplot(K1+1,1,i)
    plot(u1(i-1,:))
    ylabel(['IMF',num2str(i-1)])
end
ylabel('res')

figure(2)
subplot(K1+1,1,1)
plot(fp)
ylabel('原始')

for i=2:K2+1
    subplot(K2+1,1,i)
    plot(u2(i-1,:))
    ylabel(['IMF',num2str(i-1)])
end
ylabel('res')

save ceemd_data_F pre_i pre_n tran_n in_n out_n Prated
ns no A fv fp u1 u2 s aver -append
```

● wind power forecast/main2_f1

```
clc;clear;format compact;
rng('default')
%% Data pre-processing
load ceemd_data_F
load result

imf=u1;
c=size(imf,1);
pre_result=[];
true_result=[];

disp(['f1 建模'])
fprintf('\n')
%% Modeling of each component
for i=1:c
disp(['对第',num2str(i),'个分量建模'])
[x,y]=data_process(imf(i,:),in_n);
%Normalization
[xs,mappingx]=mapminmax(x',0,1);x=xs';
[ys,mappingy]=mapminmax(y',0,1);y=ys';
%Segmentation data
n=size(x,1);
%m=round(n*0.8);
m=n-pre_n-no;%Predicting the last 6 figures
%m=tran_n-in_n-out_n+1;
XTrain=x(1:m,:)';
XTest=x(m+1:end,:)';
YTrain=y(1:m,:)';
YTest=y(m+1:end,:)';

numFeatures = size(XTrain,1);
numResponses = 1;
numHiddenUnits = 50;
layers = [ ...
    sequenceInputLayer(numFeatures)
    lstmLayer(numHiddenUnits)
    fullyConnectedLayer(numResponses)
    regressionLayer];
options = trainingOptions('adam', ...
    'MaxEpochs',200, ...
    'GradientThreshold',1, ...
    'InitialLearnRate',0.005, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropPeriod',125, ...
    'LearnRateDropFactor',0.2, ...
    'Verbose',0, ...
'Plots','training-progress');
net_inf{i} =
trainNetwork(XTrain,YTrain,layers,options);
numTimeStepsTest = size(XTest,2);
for j = 1:numTimeStepsTest
    [net_inf{i},YPred(:,j)] =
predictAndUpdateState(net_inf{i},XTest(:,j),'Execut
ionEnvironment','cpu');
    YPred(:,j)=sim(LSTM,[s;YPred(:,j);y(m+j,:)']);
end

% inverse normalization
pre_value=mapminmax('reverse',YPred,mappingy);
true_value=mapminmax('reverse',YTest,mappingy);
pre_result=[pre_result;pre_value];
true_result=[true_result;true_value];
delete(findall(0));
end
%% The results of each component prediction are summed
f1true_value_all=sum(true_result);
f1predict_value_all=abs(sum(pre_result));

%f1true_value_y=f1true_value_all(:,end-no-ns+1:end-
no)
f1true_value_y=Vtrue_value_y;
f1predict_value_y=f1predict_value_all(:,end-no-ns+1:
end-no);
%f1true_value_c=f1true_value_all(:,end-no+1,end)
f1true_value_c=Vtrue_value_c;
f1predict_value_c=f1predict_value_all(:,end-no+1:en
d);
```

```
f1Ppredict_value=VP(f1predict_value_all(:,end-no-ns
-in_n+2:end));
f1Ppredict_value_y=f1Ppredict_value(:,1:ns);
f1Ppredict_value_c=f1Ppredict_value(:,ns+1:end);
f1Ptrue_value_y=Ptrue_value_y;
f1Ptrue_value_c=Ptrue_value_c;
save 结果/f1 net_inf f1predict_value_y
f1predict_value_c f1Ppredict_value_y
f1Ppredict_value_c
%%

load 结果/f1
fprintf('\n')
disp('f1 V 预测的结果分析')
rmse=sqrt(mean((f1true_value_y-f1predict_value_y).^
2));
mae=mean(abs(f1true_value_y-f1predict_value_y));
mape=mean(abs(f1true_value_y-f1predict_value_y)./f1
predict_value_y);
disp(['V 的根均方差(RMSE)：',num2str(rmse),' 平均绝对误差
 (MAE)：',num2str(mae),' 平均相对百分误差 (MAPE)：
',num2str(mape*100),'%'])
fprintf('\n')

disp('f1 P 预测的结果分析')
rmse1=sqrt(mean((f1Ptrue_value_y-f1Ppredict_value_y)
.^2));
mae1=mean(abs(f1Ptrue_value_y-f1Ppredict_value_y));

[h l]=size(f1Ptrue_value_y);
for i=1:h
    for j=1:l
        if f1Ptrue_value_y(i,j)<Prated

mape0(i,j)=abs(f1Ptrue_value_y(i,j)-f1Ppredict_valu
e_y(i,j))./Prated;
        else

mape0(i,j)=abs(f1Ptrue_value_y(i,j)-f1Ppredict_valu
e_y(i,j))./f1Ptrue_value_y(i,j);
        end
        f1mape(i,j)=mape0(i,j);
    end
end
 mape1=mean(mape0,2);

 disp(['P 的根均方差(RMSE)：',num2str(rmse1),' 平均绝对误
差 (MAE)：',num2str(mae1),' 平均相对百分误差 (MAPE)：
',num2str(mape1*100),'%'])
fprintf('\n')

save 结果/f1 mape1 f1mape -append

figure(2)
plot(f1true_value_y,'-*','linewidth',1)
hold on
plot(f1predict_value_y,'--.','linewidth',1)
legend('实际值','预测值')
grid on
title('f1 VP(v)')

figure(3)
plot(f1Ptrue_value_y,'-*','linewidth',1)
hold on
plot(f1Ppredict_value_y,'--.','linewidth',1)
legend('实际值','预测值')
grid on
title('f1 VP(p)')
```

● wind power forecast/main3_f2

```
clear;close all
%%
load ceemd_data_F
load result
%data=sum(u2);%All sequences added up
[x0,y0]=data_process(fv,in_n);
[x,y]=data_process(fp,in_n);
%Previous in_n moments Predict the next moment
%Normalization
[xs,mappingx]=mapminmax([x0,x]',0,1);x=xs';
[ys,mappingy]=mapminmax(y',0,1);y=ys';
%Segmentation data
n=size(x,1);
%m=round(n*0.8);
%m=n-ns-1;%
m=n-pre_n-no;
XTrain=x(1:m,:)';
XTest=x(m+1:end,:)';
YTrain=y(1:m,:)';
YTest=y(m+1:end,:)';

fprintf('\n')
disp(['f2 建模'])
fprintf('\n')

numFeatures = size(XTrain,1);
numResponses = 1;
numHiddenUnits = 50;
layers = [ ...
    sequenceInputLayer(numFeatures)
    lstmLayer(numHiddenUnits)
    fullyConnectedLayer(numResponses)
    regressionLayer];
options = trainingOptions('adam', ...
    'MaxEpochs',200, ...
    'GradientThreshold',1, ...
    'InitialLearnRate',0.005, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropPeriod',125, ...
    'LearnRateDropFactor',0.2, ...
    'Verbose',0, ...
'Plots','training-progress');
net = trainNetwork(XTrain,YTrain,layers,options);
delete(findall(0));
numTimeStepsTest = size(XTest,2);
for i = 1:numTimeStepsTest
    [net,YPred(:,i)] =
predictAndUpdateState(net,XTest(:,i),'ExecutionEnvi
ronment','cpu');
     YPred(:,i)=sim(LSTM,[s;YPred(:,i);y(m+i,:)']);
end

% inverse normalization
f2Ppredict_value=abs(mapminmax('reverse',YPred,mapp
ingy));
f2Ptrue_value=mapminmax('reverse',YTest,mappingy);

f2Ppredict_value_y=f2Ppredict_value(:,end-no-ns+1:e
nd-no);
f2Ppredict_value_c=f2Ppredict_value(:,end-no+1:end)
;
f2Ptrue_value_y=Ptrue_value_y;
f2Ptrue_value_c=Ptrue_value_c;
save 结果/f2 f2Ppredict_value_y f2Ppredict_value_c
%%

load 结果/f2
fprintf('\n')

disp('f2 P 预测的结果分析')
rmse1=sqrt(mean((f2Ptrue_value_y-f2Ppredict_value_y)
.^2));
mae1=mean(abs(f2Ptrue_value_y-f2Ppredict_value_y));

[h l]=size(f2Ptrue_value_y);
for i=1:h
    for j=1:l
        if f2Ptrue_value_y(i,j)<Prated
```

```
mape0(i,j)=abs(f2Ptrue_value_y(i,j)-f2Ppredict_valu
e_y(i,j))./Prated;
       else

mape0(i,j)=abs(f2Ptrue_value_y(i,j)-f2Ppredict_valu
e_y(i,j))./f2Ptrue_value_y(i,j);
       end
       f2mape(i,j)=mape0(i,j);
    end
end
 mape1=mean(mape0,2);

 disp(['P 的根均方差(RMSE):',num2str(rmse1),' 平均绝对误
差 (MAE): ',num2str(mae1),' 平均相对百分误差 (MAPE):
',num2str(mape1*100),'%'])
fprintf('\n')

save 结果/f2 f2mape -append

figure(4)
plot(f2Ptrue_value_y,'-*','linewidth',1)
hold on
plot(f2Ppredict_value_y,'--.','linewidth',1)
legend('实际值','预测值')
grid on
title('f2 LSTM(p)')
```


● wind power forecast/main4_f3

```
clear;format compact;close all;tic
rng('default')
%%
load ceemd_data_F
load result
imf=u2;

c=size(imf,1);
pre_result=[];
true_result=[];

fprintf('\n')
disp(['f3 建模'])
fprintf('\n')
%%
for i=1:c
disp(['对第',num2str(i),'个分量建模'])
[x,y]=data_process(imf(i,:),in_n);
%
[xs,mappingx]=mapminmax(x',0,1);x=xs';
[ys,mappingy]=mapminmax(y',0,1);y=ys';
%
n=size(x,1);

m=n-pre_n-no;
XTrain=x(1:m,:)';
XTest=x(m+1:end,:)';
YTrain=y(1:m,:)';
YTest=y(m+1:end,:)';

numFeatures = size(XTrain,1);
numResponses = 1;
numHiddenUnits = 50;
layers = [ ...
    sequenceInputLayer(numFeatures)
    lstmLayer(numHiddenUnits)
    fullyConnectedLayer(numResponses)
    regressionLayer];
```

```
options = trainingOptions('adam', ...
    'MaxEpochs',200, ...
    'GradientThreshold',1, ...
    'InitialLearnRate',0.005, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropPeriod',125, ...
    'LearnRateDropFactor',0.2, ...
    'Verbose',0, ...
'Plots','training-progress');
net_inf{i} =
trainNetwork(XTrain,YTrain,layers,options);
delete(findall(0));
numTimeStepsTest = size(XTest,2);
for j = 1:numTimeStepsTest
    [net_inf{i},YPred(:,j)] =
predictAndUpdateState(net_inf{i},XTest(:,j),'Execut
ionEnvironment','cpu');
    YPred(:,j)=sim(LSTM,[s;YPred(:,j);y(m+j,:)']);
end

pre_value=mapminmax('reverse',YPred,mappingy);
true_value=mapminmax('reverse',YTest,mappingy);
pre_result=[pre_result;pre_value];
true_result=[true_result;true_value];
End

f3Ptrue_value=sum(true_result);
f3Ppredict_value=abs(sum(pre_result));
f3Ppredict_value_y=f3Ppredict_value(:,end-no-ns+1:e
nd-no);
f3Ppredict_value_c=f3Ppredict_value(:,end-no+1:end)
;
f3Ptrue_value_y=Ptrue_value_y;
f3Ptrue_value_c=Ptrue_value_c;

save 结果/f3 f3Ppredict_value_y f3Ppredict_value_c
 %%

load 结果/f3
```



```
fprintf('\n')

disp('f3 P 预测的结果分析')
rmse1=sqrt(mean((f3Ptrue_value_y-f3Ppredict_value_y)
.^2));
mae1=mean(abs(f3Ptrue_value_y-f3Ppredict_value_y));

[h l]=size(f3Ptrue_value_y);
for i=1:h
    for j=1:l
        if f3Ptrue_value_y(i,j)<Prated

mape0(i,j)=abs(f3Ptrue_value_y(i,j)-f3Ppredict_valu
e_y(i,j))./Prated;
        else

mape0(i,j)=abs(f3Ptrue_value_y(i,j)-f3Ppredict_valu
e_y(i,j))./f3Ptrue_value_y(i,j);
        end
        f3mape(i,j)=mape0(i,j);
    end
end
 mape1=mean(mape0,2);

 disp(['P 的根均方差(RMSE)：',num2str(rmse1),' 平均绝对误
差（MAE)： ',num2str(mae1),' 平均相对百分误差（MAPE)：
',num2str(mape1*100),'%'])
fprintf('\n')

save 结果/f3 f3mape -append

figure(5)
plot(f3Ptrue_value_y,'-*','linewidth',1)
hold on
plot(f3Ppredict_value_y,'--.','linewidth',1)
legend('实际值','预测值')
grid on
title('f3 CEEMD-LSTM(p)')

%toc
```

● wind power forecast/mainF

```
clc;clear;
format compact;
close all;
tic

%%
global pre_i
global pre_n
global tran_n
global in_n
global out_n
global Prated
global ns
global no
global A
global fv
global fp


pre_i=6492;  %The serial number of the wind speed to be
predicted (starting from the serial number)
pre_n=30;    %Verify the number of wind speeds
tran_n=5000; %Number of training data
in_n=6;     %输入层节点个数
out_n=1;     %Number of input layer nodes
Prated=17;   %Rated Power(MW)
ns=3;       %Number of discriminations for accuracy
no=1;        %Number of predicted results per output

A=xlsread('DATA.xlsx');

fv=A(pre_i-tran_n:pre_i+no-1,1);
fp=A(pre_i-tran_n:pre_i+no-1,3);

Ptrue_value_y=fp(end-ns-no+1:end-no,:)';
Vtrue_value_y=fv(end-ns-no+1:end-no,:)';
Ptrue_value_c=fp(end-no+1:end,:)';
Vtrue_value_c=fv(end-no+1:end,:)';

save result Ptrue_value_y Vtrue_value_y Ptrue_value_c
Vtrue_value_c %-append

M=zeros(3,ns); %Discriminant and output matrix

run('main1_ceemd.m');
run('main2_f1.m');
run('main3_f2.m');
run('main4_f3.m');

load('.\结果\f1.mat ')
load('.\结果\f2.mat ')
load('.\结果\f3.mat ')

disp(['f1 ',num2str(ns),'个相对百分误差（MAPE):
',num2str(f1mape*100),' (%)'])
disp(['f2 ',num2str(ns),'个相对百分误差（MAPE):
',num2str(f2mape*100),' (%)'])
disp(['f3 ',num2str(ns),'个相对百分误差（MAPE):
',num2str(f3mape*100),' (%)'])

M(1,:)=f1mape(end-ns+1:end);
M(2,:)=f2mape(end-ns+1:end);
M(3,:)=f1mape(end-ns+1:end);
[Pmin,Pi]=min(M);

%%Set the weight w
w1=length(find(Pi==1))/ns;
w2=length(find(Pi==2))/ns;
w3=length(find(Pi==3))/ns;

Pnextpred=f1Ppredict_value_c*w1+f2Ppredict_value_c*
w2+f3Ppredict_value_c*w3;
PPrmse=sqrt(mean((Pnextpred-Ptrue_value_c).^2));
PPmae=mean(abs(Pnextpred-Ptrue_value_c));
PPmape=mean(abs(Pnextpred-Ptrue_value_c)./Ptrue_val
ue_c);
fprintf('\n')
disp('最终 P 预测的结果分析')
```

```
disp(['P 的根均方差(RMSE)： ',num2str(PPrmse),' 平均绝对
误差（MAE)： ',num2str(PPmae),' 平均相对百分误差（MAPE)：
',num2str(PPmape*100),'%'])
fprintf('\n')

Toc
```


- wind power forecast/loop_mainF

```
clc;clear;
format compact;
close all;
tic

%%
global pre_i
global pre_n
global tran_n
global in_n
global out_n
global Prated
global ns
global no
global A
global fv
global fp


pre_i=6392;
pre_n=30;
tran_n=5000;
in_n=6;
out_n=1;
Prated=17;
ns=8;
no=1;
loop_n=8;    %Number of cycles

A=xlsread('DATA.xlsx');

for lp=1:loop_n

save result lp -append

fv=A(pre_i-tran_n+lp-1:pre_i+no+lp-2,1);
fp=A(pre_i-tran_n+lp-1:pre_i+no+lp-2,3);
```

```
Ptrue_value_y=fp(end-ns-no+1:end-no,:)';
Vtrue_value_y=fv(end-ns-no+1:end-no,:)';
Ptrue_value_c=fp(end-no+1:end,:)';
Vtrue_value_c=fv(end-no+1:end,:)';

save result Ptrue_value_y Vtrue_value_y Ptrue_value_c
Vtrue_value_c -append

M=zeros(3,ns); % Discriminant and output matrix

run('main1_ceemd.m');
run('main2_f1.m');
run('main3_f2.m');
run('main4_f3.m');

load('.\结果\f1.mat ')
load('.\结果\f2.mat ')
load('.\结果\f3.mat ')

disp(['f1 ',num2str(ns),'个相对百分误差（MAPE）:
',num2str(f1mape*100),'%'])
disp(['f2 ',num2str(ns),'个相对百分误差（MAPE）:
',num2str(f2mape*100),'%'])
disp(['f3 ',num2str(ns),'个相对百分误差（MAPE）:
',num2str(f3mape*100),'%'])

M(1,:)=f1mape(1:ns);
M(2,:)=f2mape(1:ns);
M(3,:)=f1mape(1:ns);
[Pmin,Pi]=min(M);

%% set weight w
w1=length(find(Pi==1))/ns;
w2=length(find(Pi==2))/ns;
w3=length(find(Pi==3))/ns;

Pnextpred(lp)=f1Ppredict_value_c*w1+f2Ppredict_valu
e_c*w2+f3Ppredict_value_c*w3;
Pnexttrue(lp)=Ptrue_value_c;
save result Pnextpred Pnexttrue -append

end

PPrmse=sqrt(mean((Pnextpred-Pnexttrue).^2));
PPmae=mean(abs(Pnextpred-Pnexttrue));
PPmape=mean(abs(Pnextpred-Pnexttrue)./Pnexttrue);
fprintf('\n')
disp('最终 P 预测的结果分析')
disp(['P 的根均方差(RMSE)：',num2str(PPrmse),' 平均绝对
误差（MAE）：',num2str(PPmae),' 平均相对百分误差（MAPE）:
',num2str(PPmape*100),'%'])
fprintf('\n')

toc
```

- Other forecasting models/BPNN

```
clc
clear all
close all
%Prediction code for bp neural network
tic
% Load output and input data
p=[];%
t=[];%Output data, 15 groups in total, 1 output per
group
p0=[];%Enter verification data
t0=[];

%Normalization of the data, using the mapminmax
function, to normalize the values to between [-1.1]

[p1,ps]=mapminmax(p,0,1);
[t1,ts]=mapminmax(t,0,1);
%Determine training data, test data
[trainsample.p,valsample.p,testsample.p]
=dividerand(p1,0.8,0.2,0);
[trainsample.t,valsample.t,testsample.t]
=dividerand(t1,0.8,0.2,0);

[testsample.p,ps]=mapminmax(p0,0,1);
[testsample.t,ts]=mapminmax(t0,0,1);


%Build BP neural network with back propagation
algorithm, using newff function

%TF1 = 'tansig';TF2 = 'logsig';
%TF1 = 'logsig';TF2 = 'purelin';
%TF1 = 'logsig';TF2 = 'logsig';
%TF1 = 'purelin';TF2 = 'purelin';
%TF1='tansig';TF2='tansig';TF3='tansig';
%TF4='purelin';
%net=newff(minmax(p),[15,15,15,1],{TF1 TF2 TF3
TF4},'traingdm');

for i=1:30

TF1='tansig';TF2='purelin';
net=newff(minmax(p),[15,1],{TF1 TF2},'traingdm');

%Setting of network parameters
net.trainParam.epochs=10000;%Training times setting
net.trainParam.goal=1e-10;%Training goal setting
net.trainParam.lr=0.01;%Learning Rate Settings
net.trainParam.mc=0.9;%Setting of momentum factor
net.trainParam.show=25;%Number of intervals displayed

% Specify training parameters
 net.trainFcn = 'traingd';
%net.trainFcn = 'traingdm';
% net.trainFcn = 'traingda';
% net.trainFcn = 'traingdx';
%net.trainFcn = 'trainrp';
% Conjugate gradient algorithm
% net.trainFcn = 'traincgf';
% net.trainFcn = 'traincgp';
% net.trainFcn = 'traincgb';
% (preferred algorithm for large networks)
%net.trainFcn = 'trainscg';
% net.trainFcn = 'trainbfg';
% net.trainFcn = 'trainoss';
% (Preferred algorithm for medium-sized networks)
%net.trainFcn = 'trainlm';
% net.trainFcn = 'trainbr';

net.trainFcn='trainlm';
[net,tr]=train(net,trainsample.p,trainsample.t);
% of the trained data, according to the results obtained
by BP
[normtrainoutput,trainPerf]=sim(net,trainsample.p,[]
,[],trainsample.t);
```

```
[normvalidateoutput,validatePerf]=sim(net,valsample.
p,[],[],valsample.t);
[normtestoutput,testPerf]=sim(net,testsample.p,[],[]
,testsample.t);
% Inverse normalize the obtained results to obtain its
fitted data
trainoutput=mapminmax('reverse',normtrainoutput,ts)
;
validateoutput=mapminmax('reverse',normvalidateoutp
ut,ts);
testoutput=mapminmax('reverse',normtestoutput,ts);
% The inverse normalization of the normal input data
is processed to obtain its formal value
trainvalue=mapminmax('reverse',trainsample.t,ts);
validatevalue=mapminmax('reverse',valsample.t,ts);
testvalue=mapminmax('reverse',testsample.t,ts);

%To make a prediction, enter the data to be predicted
pnew
%pnew=[[1.70000000000000 0.700000000000000
1.60000000000000 2.50000000000000 -0.100000000000000
1007.60000000000 94]]';
%pnewn=mapminmax(pnew,0,1);
%anewn=sim(net,pnewn);
%anew=mapminmax('reverse',anewn,ts)
%绝对误差的计算
errors=testvalue-testoutput;

Emae=mean(abs(errors))
Emape=Emae/mean(testvalue)

errors0=trainvalue-trainoutput;

Emae0=mean(abs(errors0))
Emape0=Emae0/mean(trainvalue)

if Emape<=0.3
    break;
end

end

figure,plot(1:length(errors),errors,'-b')
title('誤差変化図')

figure
plot(1:644,[trainoutput
validateoutput],'b-',1:644,[trainvalue
validatevalue],'g--',645:664,testoutput,'m*-',645:6
64,testvalue,'ro-');
title('o 実測値，*予測値')

xlabel('時間/h');
ylabel('風速m/s');

legend('学習データの予測値','学習データの実測値','検証デー
タの予測値','検証データの実測値',0);
axis([0 664,-inf,inf]) %xmin is x min, xmax is x max,
ymin, ymax are similar

figure
plot(1:20,testoutput,'m*-',1:20,testvalue,'ro-');

title('o 実測値，*予測値')

xlabel('時間/h');
ylabel('風速m/s');

legend('検証データの予測値','検証データの実測値',0);

figure
figure,hist(errors);% Frequency Histogram
%figure,normplot(errors);%Q-Q diagram
%[muhat,sigmahat,muci,sigmaci]=normfit(errors);%Par
ameter estimation Mean, variance, 0.95 confidence
interval of mean, 0.95 confidence interval of variance
%[h1,sig,ci]= ttest(errors,muhat);%Hypothesis
testing
```

```
%figure, parcorr(errors);%Plotting bias correlation
diagrams


Toc
```


- Other forecasting models/RNN

```
% implementation of RNN
clc
clear
close all
%% training dataset generation
binary_dim = 8;

largest_number = 2^binary_dim-1;
binary = cell(largest_number,1);
int2binary = cell(largest_number,1);
for i = 1:largest_number+1
    binary{i} = dec2bin(i-1, 8);
    int2binary{i} = binary{i};
end

%% input variables
alpha = 0.1;
input_dim = 2;
hidden_dim = 16;
output_dim = 1;

%% initialize neural network weights
synapse_0 = 2*rand(input_dim,hidden_dim) - 1;
synapse_1 = 2*rand(hidden_dim,output_dim) - 1;
synapse_h = 2*rand(hidden_dim,hidden_dim) - 1;

synapse_0_update = zeros(size(synapse_0));
synapse_1_update = zeros(size(synapse_1));
synapse_h_update = zeros(size(synapse_h));

%% train logic
for j = 0:19999
    % generate a simple addition problem (a + b = c)
    a_int = randi(round(largest_number/2)); % int
version
    a = int2binary{a_int+1}; % binary encoding
```

```
    b_int = randi(floor(largest_number/2)); % int
version
    b = int2binary{b_int+1}; % binary encoding

    % true answer
    c_int = a_int + b_int;
    c = int2binary{c_int+1};

    % where we'll store our best guess (binary encoded)
    d = zeros(size(c));

    if length(d)<8
        pause;
    end

    overallError = 0;

    layer_2_deltas = [];
    layer_1_values = [];
    layer_1_values = [layer_1_values; zeros(1,
hidden_dim)];

    for position = 0:binary_dim-1
        X = [a(binary_dim - position)-'0' b(binary_dim
- position)-'0'];   %  X is the input
        y = [c(binary_dim -
position)-'0']';                              % Y is the
label, used to calculate the final error

        layer_1 = sigmoid(X*synapse_0 +
layer_1_values(end, :)*synapse_h);
        % layer_1 ------------------> hidden layer
(S(t))
        % output layer (new binary representation)
        layer_2 = sigmoid(layer_1*synapse_1);

        % Calculate the error and back propagate based
on the error
        layer_2_error = y - layer_2;

        layer_2_deltas = [layer_2_deltas;
layer_2_error*sigmoid_output_to_derivative(layer_2)]
;

        % Overall error (error has positive and negative,
use absolute value)
        overallError = overallError +
abs(layer_2_error(1));

        % decode estimate so we can print it out
        d(binary_dim - position) = round(layer_2(1));

        % store hidden layer so we can use it in the next
timestep
        layer_1_values = [layer_1_values; layer_1];
    end

    future_layer_1_delta = zeros(1, hidden_dim);

    for position = 0:binary_dim-1
        X = [a(position+1)-'0' b(position+1)-'0'];

        layer_1 = layer_1_values(end-position, :);
        prev_layer_1 =
layer_1_values(end-position-1, :);

        % error at output layer
        layer_2_delta =
layer_2_deltas(end-position, :);
        % error at hidden layer
        layer_1_delta =
(future_layer_1_delta*(synapse_h') +
layer_2_delta*(synapse_1')) ...
                        .*
sigmoid_output_to_derivative(layer_1);

        synapse_1_update = synapse_1_update +
(layer_1')*(layer_2_delta);
        synapse_h_update = synapse_h_update +
(prev_layer_1')*(layer_1_delta);
```

```
        synapse_0_update = synapse_0_update +
(X')*(layer_1_delta);

        future_layer_1_delta = layer_1_delta;
    end

    synapse_0 = synapse_0 + synapse_0_update * alpha;
    synapse_1 = synapse_1 + synapse_1_update * alpha;
    synapse_h = synapse_h + synapse_h_update * alpha;

    synapse_0_update = synapse_0_update * 0;
    synapse_1_update = synapse_1_update * 0;
    synapse_h_update = synapse_h_update * 0;

    if(mod(j,1000) == 0)
        err = sprintf('Error:%s\n',
num2str(overallError)); fprintf(err);
        d = bin2dec(num2str(d));
        pred = sprintf('Pred:%s\n',dec2bin(d,8));
fprintf(pred);
        Tru = sprintf('True:%s\n', num2str(c));
fprintf(Tru);
        out = 0;
        size(c)
        sep = sprintf('-------------\n');
fprintf(sep);
    end

end
```


- Other forecasting models/Elman

```
clc;
clear all
close all
nntwarn off;

%% Data Loading

load data;
a=data;

%% Select training data and test data

for i=1:6
    p(i,:)=[a(i,:),a(i+1,:),a(i+2,:)];
end
% Training data input
p_train=p(1:5,:);
% Training data output
t_train=a(4:8,:);
% Test data input
p_test=p(6,:);
% Test Data Output
t_test=a(9,:);

% Transpose to fit the network structure

p_train=p_train';
t_train=t_train';
p_test=p_test';


%% Network building and training

nn=[4 10 1];
for i=1:4
    threshold=[0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1;0 1];
```

```
net=newelm(threshold,[nn(i),3],{'tansig','purelin'})
;
    % Set network training parameters
    net.trainparam.epochs=1000;
    net.trainparam.show=20;
    % Initialize the network
    net=init(net);
    %  Elman network training
    net=train(net,p_train,t_train);

    y=sim(net,p_test);
    error(i,:)=y'-t_test;
end

%% The prediction effect of the network with different
number of hidden layer neurons is observed by graphing

plot(1:1:3,error(1,:),'-ro','linewidth',2);
hold on;
plot(1:1:3,error(2,:),'b:x','linewidth',2);
hold on;
plot(1:1:3,error(3,:),'k-.s','linewidth',2);
hold on;
plot(1:1:3,error(4,:),'c--d','linewidth',2);
title('Elman 预测误差图')
set(gca,'Xtick',[1:3])
legend('7','11','14','18','location','best')
xlabel('时间点')
ylabel('误差')
hold off;
```

- Other forecasting models/ARIMA

```
%Original data
y=[ ];
Data=y';
step=6;
TempData=SourceData;
TempData=detrend(TempData);%Go to trendline
TrendData=SourceData-TempData;%Trend Functions

H=adftest(TempData);
difftime=0;
SaveDiffData=[];
while ~H
SaveDiffData=[SaveDiffData,TempData(1,1)];
TempData=diff(TempData);
difftime=difftime+1;
H=adftest(TempData);
end
%Model ranking or identification
u = iddata(TempData);
test = [];
%for p = 1:5
for p = 1:5
for q = 1:5
m = armax(u,[p q]);
AIC = aic(m);%armax(p,q),calculate AIC
test = [test;p q AIC];
end
end
test1=[];

for p=1:5
for q=1:5
for k = 1:size(test,1)
if test(k,3) == min(test(:,3)) %Select the model with
the lowest AIC value
p_test = test(k,1);
q_test = test(k,2);
```

```
m = armax(u,[p q]);
AIC = aic(m);
m1 = armax(u,[p_test q_test]);
BIC = aic(m1);
test1=[test1;p q p_test q_test AIC BIC ];

break;
end
end

end
end

%1st order forecast
TempData=[TempData;zeros(step,1)];
n=iddata(TempData);
%p_test=1;q_test=2;
m = armax(u,[p_test q_test]);

P1=predict(m,n,1);
PreR=P1.OutputData;
PreR=PreR';

if size(SaveDiffData,2)~=0
for index=size(SaveDiffData,2):-1:1
PreR=cumsum([SaveDiffData(index),PreR]);
end
end
%Predict trends and return results
mp1=polyfit([1:size(TrendData',2)],TrendData',1);
xt=[];
for j=1:step
xt=[xt,size(TrendData',2)+j];
end
TrendResult=polyval(mp1,xt);
PreData=TrendResult+PreR(size(SourceData',2)+1:size
(PreR,2));
tempx=[TrendData',TrendResult]+PreR;
```

```
figure(1)
subplot(2,2,1),plot(tempx,'r');%Red Line Forecast
Value;
hold on
grid;
title('预测值');
xlabel('时间/h');ylabel('风速 m/s');
subplot(2,2,2),plot(Data,'b');%% Blue line actual
value
hold on
grid;
title('实际值');
xlabel('时间/h');ylabel('风速 m/s');

t=tempx';
erro=abs(Data-t);
subplot(2,2,3),plot(erro,'g');
hold on
grid;
title('绝对误差值');
xlabel('时间/h');ylabel('风速 m/s');
```

- Other forecasting models/EWT

```
function y = EWT_v(ewt,boundaries)
% Inputs:
%   ewt: component obtained by the EWT transform
%   boundaries: the corresponding boundaries provided
by the EWT transform

% Outputs:
%   Hilb: structure containing the instantaneous
components (Hilb{i}{1} is
%   the instantaneous amplitude of the component i and
Hilb{i}{2} is the
%   instantaneous frequency of component i)

%% compute gamma to estimate the support sizes
gamma=pi;
boundaries=[0  boundaries  pi];
for k=1:length(boundaries)-1

r=(boundaries(k+1)-boundaries(k))/(boundaries(k+1)+
boundaries(k));
    if r<gamma
       gamma=r;
    end
end

%% Calculate the instantaneous amplitude and frequency
of each component
Hilb=cell(length(ewt),2);
for i=1:length(ewt)
    ht=hilbert(ewt{i});

    % Instantaneous amplitude
    Hilb{i}{1}=abs(ht);

    %Phase unwrapping, take its derivative and clean the
outliers
Hilb{i}{2}=IFcleaning(diff(unwrap(angle(ht))),(1-ga
mma)*boundaries(i),(1+gamma)*boundaries(i+1));

%     figure;
%     subplot(211);plot(Hilb{i}{1});
%     subplot(212);plot(Hilb{i}{2});
end
```