Energy 207 (2020) 118265

Contents lists available at ScienceDirect

Energy

journal homepage: www.elsevier.com/locate/energy

A novel ensemble method for residential electricity demand forecasting based on a novel sample simulation strategy

Guoqiang Zhang *, Jifeng Guo

Northeast Forestry University Information and Computer Engineering Institute, Heilongjiang, Harbin, 150000, China

ARTICLE INFO

Article history:
Received 27 January 2020
Received in revised form
22 June 2020
Accepted 28 June 2020
Available online 6 July 2020

Keywords:
Electricity demand forecasting
Improved coupled generative adversarial stacked auto-encoder (ICoGASA)
Integrated forecast
Self-organizing map (SOM)
Memristor array (MA)

ABSTRACT

This paper presents a novel ensemble method of forecasting the residential electricity demand. Firstly, the time-series of the original input variables is filtered by unscented kalman filter (UKF), and then the incremental percentages of current and previous sample points are taken as new input features of the proposed method. Secondly, an improved coupled generative adversarial stacked auto-encoder (ICoGASA) consisting of three generative adversarial networks (GAN) is developed to generate more similar errors in weather forecast and lifestyles of different residents, with less noise. All of the three GANs are composed of two deep belief networks (DBNs), which serve as generator and discriminator, respectively. The three generators of GANs are used to simulate the samples with positive error, negative error and mixed error, respectively. Then the output of the three discriminators is integrated by memristor array (MA), and the integrated output of each ICoGASA are integrated by self-organizing map (SOM). Thirdly, the input weights of SOM are optimized by MA and a new weight updated strategy (WUS). Compared with other state-of-the-art ensemble methods, the scopes of the root mean square error (RMSE) are reduced by [8.295, 16.221] %, [15.507, 28.066] %, [20.494, 36.969] %, respectively.

1. Introduction

Electricity demand forecasting has become one of the research hotspots in security and generation scheduling of the electric power industry. Efficient electricity supply is import to promoting economic growth. Overestimation or underestimation may result in a series of economic and social consequences, such as, financial burden, electricity price rise, shortage of electricity supply and so on [1]. Subtle changes in electricity demand forecasting accuracy may have significant influences on operating costs in competitive electricity markets [2]. However, it is difficult to improve forecasting accuracy because of various influence factors of load such as, residents' lifestyle [3], meteorological factors [4–9] and electricity price [10].

So far, lots of methods have been widely developed to improve forecasting accuracy, including, regression methods [11] and time series methods [12,13], classical artificial neural networks (ANNs) [14], fuzzy logic inference [15], deep belief networks (DBN) [16], and support vector regression (SVR) [17]. Recently, numerous ensemble methods [18–20], based on artificial intelligence techniques, have been wildly applied to obtain higher forecasting accuracy. The simulation results all show that an ensemble model performs better than a single model in forecasting accuracy.

As is known to all, there is certain variability in the lifestyles of different residents [3]. Moreover, there may exist some errors in weather forecast because of complex climate changes [21]. Therefore, the coupled generative adversarial stacked Auto-encoder (CoGASA) [22] is proposed to solve the problems mentioned above. To simulate more realistic weather forecast errors and lifestyles of different residents, an improved CoGASA (ICoGASA) is developed. That is, another generative adversarial network (GAN) [23] is added in the classical CoGASA. Specifically, the ICoGASA consists of three GANs; all the three GANs are composed of two deep belief networks (DBNs) [24], which serve as generator and discriminator respectively. The three generators of the GANs are used to generate positive error, negative error and mixed error respectively, and generate more realistic synthesized samples (RSS) according to the training sample; while the discriminator model is used to distinguish RSS generated by generator model from real training sample.

More irregular fluctuations of the time-series will degrade the forecasting performance [25]. Therefore, the time-series of the

* Corresponding author.
E-mail addresses: limiyi061@qq.com, limiyi061@ieee.org (G. Zhang).

https://doi.org/10.1016/j.energy.2020.118265
0360-5442/

original input variables are smoothed and de-noised via unscented kalman filter (UKF) [26], and then the incremental percentages of current and previous sample points serve as new input features of the proposed method. Then the output of the three discriminators of each GAN is integrated by memristor array (MA) [27], and the integrated output of each ICoGASA is integrated by self-organizing map (SOM) [28]. The advantage of using SOM is that after each iteration we can identify the weight getting better or worse according to the learning mechanism of SOM. Therefore, MA composed of a set of memristors [29] that have a stronger ability of learning the historical behavior [27], is applied to optimize the input weights of SOM. Moreover, a new weight updated strategy (WUS) is proposed. That is, after each iteration, the weights (W_b) altering for the better are saved, while the ones (W_w) for the worse randomly choose their learning objects and learn from W_b. If W_w does not become better for N_w consecutive times, W_w is generated by chaotic tent map.

The contributions of this paper can be summarized as follows:

1) A novel input feature is applied to electricity demand forecasting, namely that the time-series of the original input variables are smoothed and de-noised by UKF, and then the incremental percentages of current and previous sample points serve as new input feature of the proposed method. The proposed input feature has two advantages, which play a significant role in improving forecasting performance [18,25] Firstly, smoothing and de-noising by UKF can decrease irregular fluctuations of the original load-series. Secondly, the incremental percentages establish a relationship between current data and historical data.
2) A novel sample simulation strategy, which is helpful to forecast the load of holidays (in the case of fewer training samples), is proposed. Namely that an improved coupled generative adversarial stacked Auto-encoder (ICoGASA) consisting of three GANs is developed to forecast residential electricity demand. That is, another generative adversarial network (GAN) is added in the classical CoGASA. All the three GANs are composed of two deep belief networks (DBNs), which serve as generator and discriminator respectively. The three generators (G) of the GANs are used to generate positive error, negative error and mixed error, respectively, and generate more realistic synthesized samples (RSS) according to the training sample; while the discriminator model (D) is used to distinguish RSS by generator model from real training sample. In case that the probability that the generated and real samples of correct data sources was reaches 0.5, the generated RSS with more realistic weather forecast errors and lifestyles of different residents is obtained. In other words, the model performance reaches the highest level [23]. Through the sample simulation strategy, the generalization and adaptability of the predicted model can be improved.
3) A novel ensemble method is proposed to forecast residential electricity demand. Specifically, the output of the three discriminators of each GAN is integrated by MA, the integrated result serves as the output of each ICoGASA, and then the output of each ICoGASA is integrated by SOM. The advantage of using SOM is that after each iteration we can identify the weight getting better or worse according to the learning mechanism of SOM. Through this strategy, the convergence speed of the predicted model can be improved by using optimization algorithms.
4) MA and a new weight updated strategy (WUS) are applied to optimize the weights of SOM. MA has a stronger ability of learning the historical behavior [27]. Moreover, WUS is helpful to improve the convergence ability and decrease model training time.

2. Acquisition of input features

The hourly temperature, wind speed, precipitation, relative humidity, electricity price, and manually operated appliances which can better distinguish different lifestyles of different residents (heating heat-pump, dishwasher, television, and clothes washer, etc.) are taken as original input variables. Then the time-series of the original input variables are smoothed and de-noised by UKF according to Refs. [26] the detailed steps as follows:

Step 1: Let X be the time-series of the original input variables, mean and covariance of X be $\overline{X}$ and P, respectively. Then the 2n+1 *sigma* sample points X_i and their corresponding weights ω can be obtained by the unscented transformation (UT):

$$\begin{cases} X^{(0)} = \overline{X}, i = 0 \\ X^{(i)} = \overline{X} + \left(\sqrt{(n+\lambda)P}\right)_i, i = 1, 2, \ldots, n \\ X^{(i)} = \overline{X} - \left(\sqrt{(n+\lambda)P}\right)_i, i = n+1, \ldots, 2n \\ \left(\sqrt{P}\right)^T \left(\sqrt{P}\right) = P \end{cases} \tag{1}$$

$$\begin{cases} \omega_m^{(0)} = \dfrac{\lambda}{n+\lambda} \\ \omega_c^{(0)} = \dfrac{\lambda}{n+\lambda} + \left(1 - \alpha^2 + \beta\right) \\ \omega_m^{(i)} = \omega_c^{(i)} = \dfrac{\lambda}{2(n+\lambda)}, i = 1, 2, \ldots, 2n \\ \lambda = \alpha^2(n+\kappa) - n \end{cases} \tag{2}$$

here λ and κ both mean scaling parameters, α means spread factor of the *sigma* sample points.

Step 2: Perform one step forecast of X_i, i = 1,2, …, 2n+1.

$$X^{(i)}\left(k+1|k\right) = f\left[k, X^{(i)}(k|k)\right] \tag{3}$$

Step 3: Calculate the system state and covariance matrix:

$$\widehat{X}\left(k+1|k\right) = \sum_{i=0}^{2n} \omega^{(i)} X^{(i)}(k+1|k) \tag{4}$$

$$P(k+1|k) = \sum_{i=0}^{2n} \omega^{(i)} \left[\widehat{X}(k+1|k) - X^{(i)}(k+1|k)\right] * \left[\widehat{X}\left(k+1|k\right) - X^{(i)}(k+1|k)\right]^T + Q \tag{5}$$

Step 4: Generate new *sigma* sample points by UT again:

$$X^{(i)}(k+1|k) = \begin{bmatrix} \widehat{X}(k+1|k) \\ \widehat{X}(k+1|k) + \sqrt{(n+\lambda)P(k+1|k)} \\ \widehat{X}(k+1|k) - \sqrt{(n+\lambda)P(k+1|k)} \end{bmatrix}^T \tag{6}$$

Step 5: Obtain the observed signal:

$$Z^{(i)}\left(k+1|k\right) = h\left[X^{(i)}(k+1|k)\right] \tag{7}$$

Step 6: Calculate the mean and covariance of system forecast:

$$\overline{Z}\left(k+1|k\right) = \sum_{i=0}^{2n} \omega^{(i)} Z^{(i)}(k+1|k) \tag{8}$$

$$P_{z_k z_k} = \sum_{i=0}^{2n} \omega^{(i)} \left[Z^{(i)}(k+1|k) - \overline{Z}(k+1|k) \right] * \left[Z^{(i)}(k+1|k) - \overline{Z}(k+1|k) \right]^T + R \tag{9}$$

$$P_{x_k z_k} = \sum_{i=0}^{2n} \omega^{(i)} \left[X^{(i)}(k+1|k) - \overline{Z}(k+1|k) \right] * \left[Z^{(i)}(k+1|k) - \overline{Z}(k+1|k) \right]^T \tag{10}$$

Step 7: Calculate the Kalman gain:

$$K(k+1) = P_{x_k z_k} P_{z_k z_k}^{-1} \tag{11}$$

Step 8: Update the covariance and system state:

$$\begin{cases} \widehat{X}(k+1|k+1) = \widehat{X}(k+1|k) + K(k+1)[Z(k+1) - \widehat{Z}(k+1|k)] \\ P(k+1|k+1) = P(k+1|k) - K(k+1) P_{z_k z_k} K^T(k+1) \end{cases} \tag{12}$$

Step 9: Repeat ***Steps1-9***, until all original input variables have been processed. Then the smoothed and de-noised time-series X'^N_i of the original input variables can be obtained. Here i = 1,2, …, n means the number of sample points in each original input variable; N means the type of the original input variables (1 represents temperature, 2 represents wind speed, etc.). The incremental percentages of current and previous sample points of X'^N_i can be obtained:

$$X''^N_i = \frac{X'^N_{i+1} - X'^N_i}{X'^N_i} \times 100\% \tag{13}$$

Step 10: Normalize X''^N_i into the scope of [0,1], according to Ref. [30], the normalization and de-normalization formulas are as follow:

$$X'''^N_i = \frac{X''^N_{i+1} - \min}{\max - \min} \tag{14}$$

$$X''^N_{i+1} = X'''^N_i (\max - \min) + \min \tag{15}$$

Here *max* and *min* mean the maximum and minimum of X''^N_i, respectively.

3. Methodologies

3.1. Principles of ICoGASA

CoGASA is chosen to address the issue of residential electricity demand forecasting for its superior performance of generating more realistic synthesized samples (RSS). A basic CoGASA is developed based on image processing. The main idea is to map a uniform random vector (Z) into two different domain images. Taking face recognition as an example, two different images with the same face but with different expressions (smiling and non-smiling) can be obtained after being processed by CoGASA. Nevertheless, as simulating time-series samples, there are three cases, positive error (the predicted value is more than the true value), negative error (contrary to positive error), positive and negative mixed error. It should be noted that a CoGASA can also realize the simulation of time series samples. However, using two GANs to simulate three different types of samples can result in large fluctuations in the weights of each GAN, which would spend more processing time and reduce the forecasting performance (test in ***Case VI***). To simulate more realistic weather forecast errors and lifestyles of different residents more efficiently, an extra GAN and MA are introduced into the classical CoGASA. Specifically, the improved CoGASA (ICoGASA) consists of one stacked auto-encoder, three GANs and one MA. The stacked auto-encoder transfers the real data from one domain to another with less noise. The three generators of GANs are applied to simulate the samples with positive error, negative error and mixed error, respectively; while the discriminator models (D) are utilized to distinguish simulated samples generated by the generator model from the real training sample. The MA is used to calculate the weights of each discriminator and integrate them.

The advantage of the stacked auto-encoder (SAE) should depend on its strong ability to reduce the noise level. Consequently, a lesser amount of noise can be induced in the Z space [22]. In other words, more similar samples can be generated in each generator as its output. The SAE process is shown in Fig. 1, the input load-series (D_1) is mapped to the hidden layer H_1 by the deterministic function f_1 and then the original inputs are reconstructed by the mapping function g_1. The training process of the second auto-encoder is the same as that of the first auto-encoder except that the input of the second auto-encoder is H_1. The reconstruction loss functions are as follows:

$$\begin{cases} L_1(D, D') = \sum_{j=1}^{M} \left(d_j - g_1(f_1(d_j')) \right)^2, \\ L_2(H_1, H_1') = \sum_{j=1}^{M} \left(h_{1j} - g_2(f_2(h_{1j}')) \right)^2 \end{cases} \tag{16}$$

Here M means the total number of the sample data.

Let S_1, S_2 and S_3 be the samples with positive error, negative error and mixed error, respectively. The generators in the three GANs can be constructed by Eq. (17), The detailed principle and process of GAN can be found in Section 3 Chapter 2.

$$\begin{cases} G_1(z) = G_{1k}(\ldots(G_{12}(G_{11}(z)))) \\ G_2(z) = G_{2k}(\ldots(G_{22}(G_{21}(z)))) \\ G_3(z) = G_{3k}(\ldots(G_{32}(G_{31}(z)))) \end{cases} \tag{17}$$

here G_{1k}, G_{2k} and G_{3k} represent the generators in the three GANs, respectively; k means the number of layers. And then, the simulated global features in the output of all the three generators can be generated by sharing the parameters in the weight sharing layers (the first and second layers) of the three generators. The final simulated samples can be obtained in the non-shared layer (the third layer) of each GAN. Similarly, the discriminators (GANs) can be modeled by Eq. (18). The detailed principle and process of GAN are described in Section 3 Chapter 2.

$$\begin{cases} D_1(G_1(z)) = D_{1k}(\ldots D_{12}((D_{11}(G_1(z))))) \\ D_2(G_2(z)) = D_{2k}(\ldots D_{22}((D_{21}(G_2(z))))) \\ D_3(G_3(z)) = D_{3k}(\ldots D_{32}((D_{31}(G_3(z))))) \end{cases} \tag{18}$$

To directly find out a corresponding vector z for fitting the input time-series, SAE is concatenated to the generators. The SAE part is independently trained with the input time-series. Once the training of the paired GANs and the SAE has been finished, the two parts would be integrated to directly map the input time-series into the new space Z. The integrated two parts can be modeled:

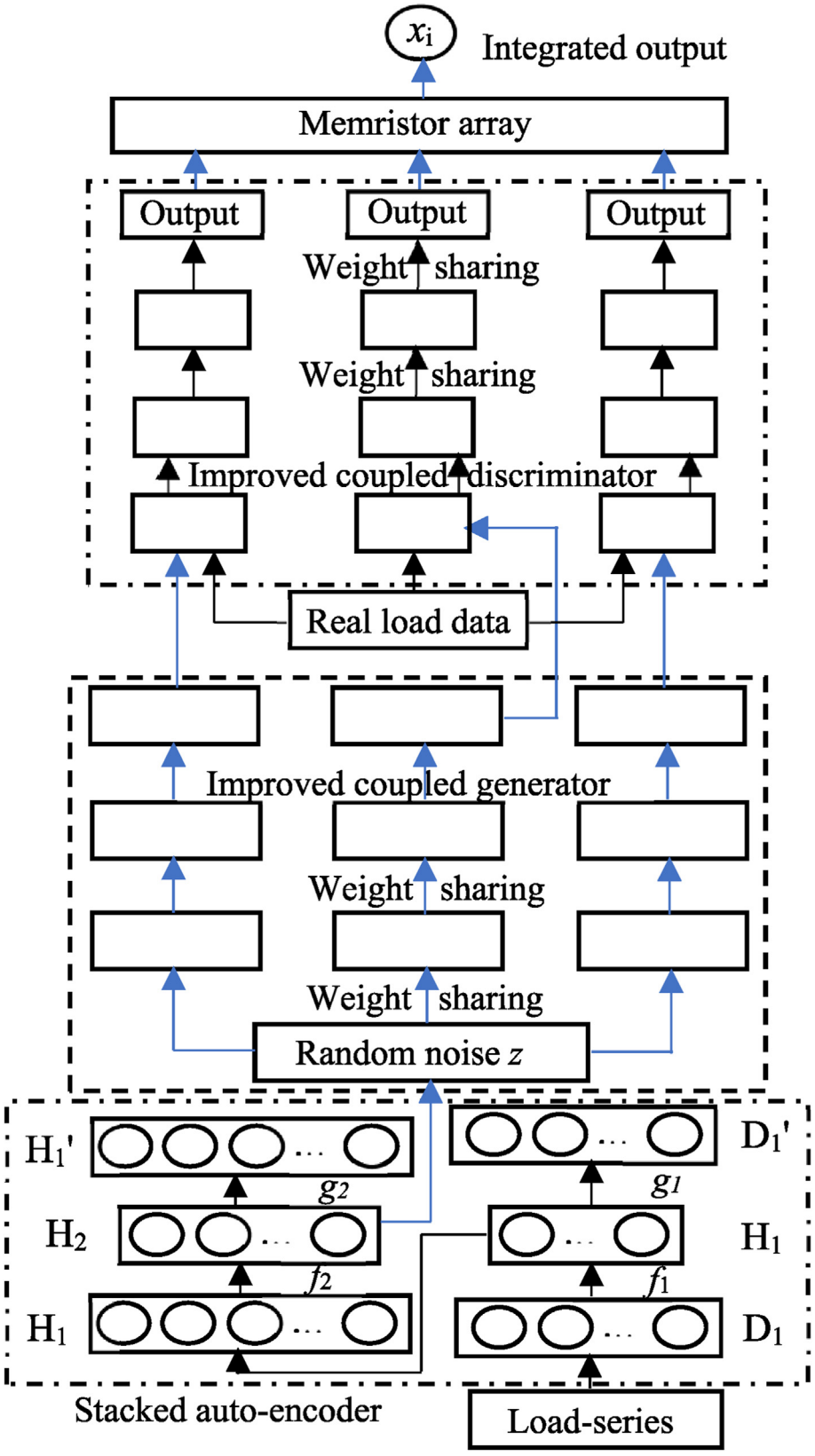

Fig. 1. The structure of ICoGASA.

$$\begin{cases} G_1(S_1) = G_{1j}(\ldots(G_{12}(G_{11}(E_{1k}(\ldots(E_{12}(E_{11}(S_1)))))))) \\ G_2(S_2) = G_{2j}(\ldots(G_{22}(G_{21}(E_{1k}(\ldots(E_{12}(E_{11}(S_2)))))))) \\ G_3(S_3) = G_{3j}(\ldots(G_{32}(G_{31}(E_{1k}(\ldots(E_{12}(E_{11}(S_3)))))))) \end{cases} \quad (19)$$

here G1 G2 and G3 represent the coupled generators with j layers. E means the SAE with k layers. The mean square error (MSE) is introduced to SAE to evaluate the reconstructed data of the load-series:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(S_i - \hat{S_i}\right)^2 \quad (20)$$

When the training process of all parts has been finished, three outputs from the three discriminators would be obtained. A MA is introduced to calculate the weights of each discriminator and integrate them. Consequently, an integrated output (x_i) is obtained. The calculation principle and process of MA are described in Section 3 Chapter 4.

3.2. Principles of GAN

As shown in Fig. 2, a classical GAN is composed of one generator model (G) and one discriminator model (D). In this paper, G is used to simulate more realistic weather forecast errors and lifestyles of different residents, and generate more realistic synthesized data $X_{\text{generated}} = G(z)$ with a distribution P_g according to a noise vector z (generated randomly). The purpose of G is to deceive D, so that D cannot correctly judge the realistic synthesized samples generated by G; while D is used to distinguish the realistic synthesized samples from real training sample. The loss function L, defined by Eq. (21), is introduced into D:

$$\begin{aligned} L = {} & E_{x\sim P_{data}}[\log p(s = real|x_{real})] + \\ & E_{z\sim P_{(z)}}\left[\log p\left(s = generated\middle|x_{generated}\right)\right] = \\ & E_{x\sim P_{data}}[\log(D(x))] + E_{z\sim P_{(z)}}[\log p(1 - D(G(z)))] \end{aligned} \quad (21)$$

Here P_{data} means the distribution of real training data, $E_{x\sim Pdata}$ means the expectation of x from P_{data}, $P(z)$ means the prior distribution of the randomly generated noise vector z, $E_{z\sim P(z)}$ means the expectation of sample z from noise, and $D(x)$ means the probability that x belongs to real training data rather than generated data. The training goal of G is to prevent D from correctly identifying generated sample by minimizing the second term in Eq. (21); while the training goal of D is to ensure real data source by maximizing L. Therefore, the objective function of GAN is defined as follows:

$$fun = \arg\min_{G}\max_{D} L(G, D) \quad (22)$$

The adversarial process between G and D is like a zero-sum game. If the probability that the generated and real samples belong to the correct data sources reaches 0.5, the model performance reaches the highest level [23]. In other words, the generated realistic synthesized samples (RSS) with more realistic weather forecast errors and lifestyles of different residents is obtained.

3.3. Principles of DBN

In this paper, G and D both are DBN. DBN, as a greedy layered deep learning model, is composed of some stacked restricted Boltzmann machines (RBMs). As shown in Fig. 3, a basic RBM contains n visible nodes v_i, m hidden nodes h_j, and a connected weight matrix w_{ij}.

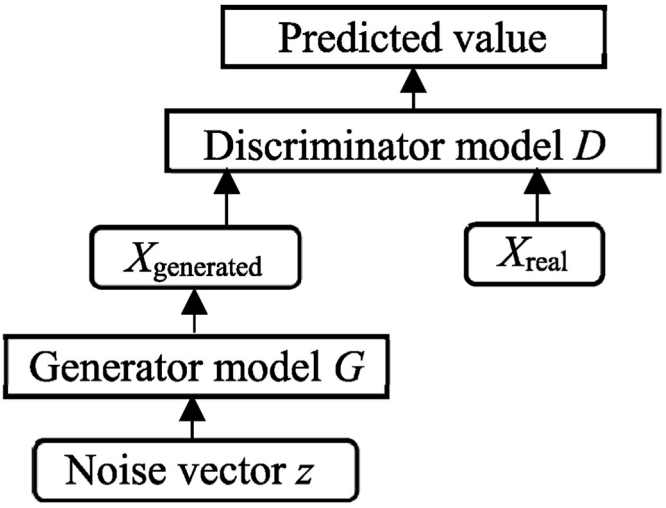

Fig. 2. The structure of GAN.

According to Ref. [31], an energy function (Eq. (23)) was introduced to evaluate a given state (v, h):

$$E(v, h|\theta) = -\sum_{i=1}^{n}\sum_{j=1}^{m} v_i w_{ij} h_j - \sum_{i=1}^{n} a_i v_i - \sum_{j=1}^{m} b_j h_j \quad (23)$$

Here b_j and a_i mean the biases of v_i and h_j, respectively. Conditional probability of v_i, and h_j can be obtained by Eq. (24). The contrastive divergence (CD) algorithm [32] was introduced to

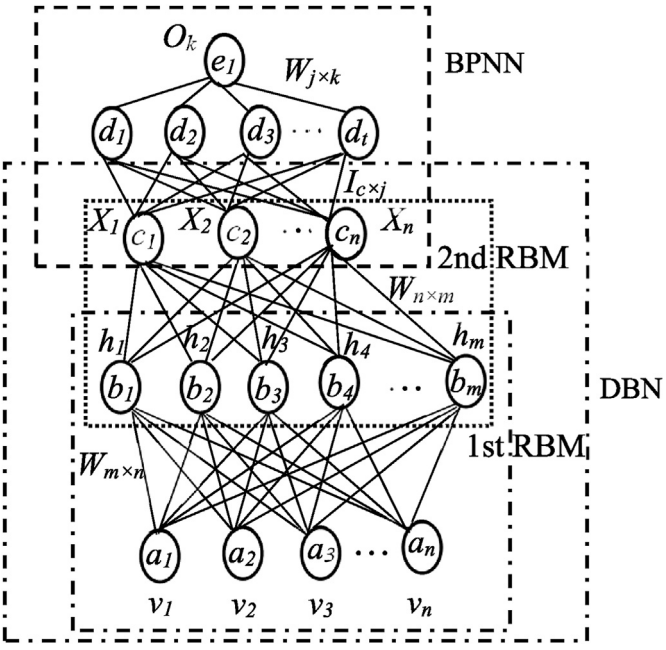

Fig. 3. The structure of DBN.

effectively train RBM to figure out an appropriate parameter $\theta(w_{ij}, a_i, b_j)$ that can effectively fit the training data. That is, $\theta(w_{ij}, a_i, b_j)$ is updated by Eq. (25).

$$\begin{aligned} P(h_i = 1|v) &= \sigma\left(\sum_{j=1}^{m} w_{ij}v_j + b_j\right) \\ P(v_i = 1|h) &= \sigma\left(\sum_{i=1}^{n} w_{ij}h_i + a_i\right) \\ \sigma(x) &= \frac{1}{1+e^{-x}} \end{aligned} \tag{24}$$

$$\begin{aligned} \Delta w_{ij} &= \varepsilon\left(\langle v_i h_j\rangle_{data} - \langle v_i h_j\rangle_{recon}\right) \\ \Delta b_i &= \varepsilon(\langle v_i\rangle_{data} - \langle v_i\rangle_{recon}) \\ \Delta a_j &= \varepsilon\left(\langle h_j\rangle_{data} - \langle h_j\rangle_{recon}\right) \end{aligned} \tag{25}$$

where $\langle\cdot\rangle_{data}$ and $\langle\cdot\rangle_{recon}$ represent the products of v_i, and h_j before and after reconstruction, respectively; and ε represents the learning rate.

A classical DBN is as shown in Fig. 3. In the bottom, after the visible nodes obtain a training sample, a learning feature generated in the first RBM contains the visible layer and the first hidden layer. By iteratively changing the learning feature, the first RBM can effectively fit the training data. This phase is the so-called alternating sampling process. As soon as the first RBM has been trained well, the second RBM (contains the first hidden layer and the hidden layer takes the output of the first hidden layer as its input) will be trained with the same training method using in the first RBM. The same process is repeated until the nth RBM contains the (n-1) hidden layer and the nth hidden layer finishes its train. So far, the layer-wise pre-training process has been finished. At the top of a classical DBN, a back propagation neural network (BPNN) takes the nth RBM's output ($o(k)$) as its input and starts the fine-tuning process. The training purpose is to improve the model performance by slightly adjusting the DBN parameters.

BPNN has the advantage of effectively adjusting weight matrices connecting neurons of different layers, which enables it to obtain a stronger approximation ability [33]. The structure of a classical BPNN is shown in Fig. 3. The next layer takes the output of the previous layer as its input. In this paper, BPNN takes the n-th RBM's output (X) as its input. The hidden neurons ($d_1, d_2, \ldots, d_t$) are used to adjust weight matrix (I_{cj}) connecting the input neurons ($c_1, c_2, \ldots, c_n$). The output layer's output (O_k) is the finally needed result. The training process of BPNN is composed of two phases, one is the neuron propagation phase and the other is the weight adjusting phase.

In the first phase, the input training samples are first propagated forward through the network so that the output activations can be realized. Then, the obtained output activations are propagated backward through the network via the selected training target. When the backward propagation process has been finished, the deltas of all neurons between different layers are obtained:

$$\begin{cases} Y_j = f\left(\sum_{c=1}^{n} I_{cj}X_c\right), (j = 1, 2, \ldots, t) \\ O_k = f\left(\sum_{j=1}^{t} W_{jk}Y_j\right), (k = 1, 2, \ldots, l) \end{cases} \tag{26}$$

Here I_{cj} means the weight matrix connecting the hidden neurons ($d_1, d_2, \ldots, d_t$) and input neurons ($c_1, c_2, \ldots, c_n$); W_{jk} means the weight matrix connecting the output neurons (e_1) and hidden neurons ($d_1, d_2, \ldots, d_t$). X represents the n-th RBM's output, namely, the input of BPNN; Y_j represents the output of the hidden neurons ($d_1, d_2, \ldots, d_t$). O_k means the finally needed output of BPNN. f(x) is usually taken as a sigmoid function defined as follows:

$$f(x) = \frac{1}{1+e^{-x}} \tag{27}$$

In the second phase, the gradient (Eq. (29)) of the weight can be calculated by multiplying the output activations and neurons'

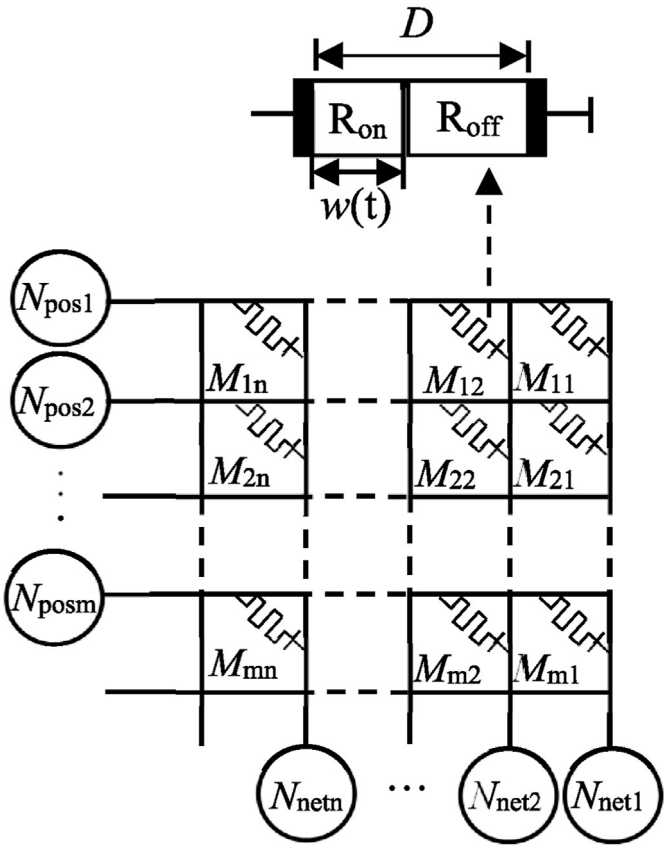

Fig. 4. The structure of MA.

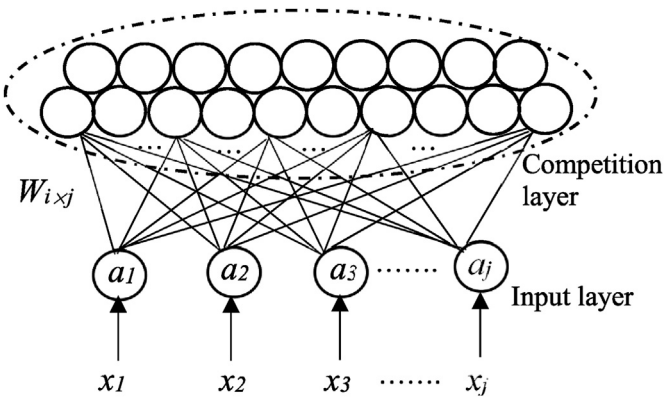

Fig. 5. The structure of SOM.

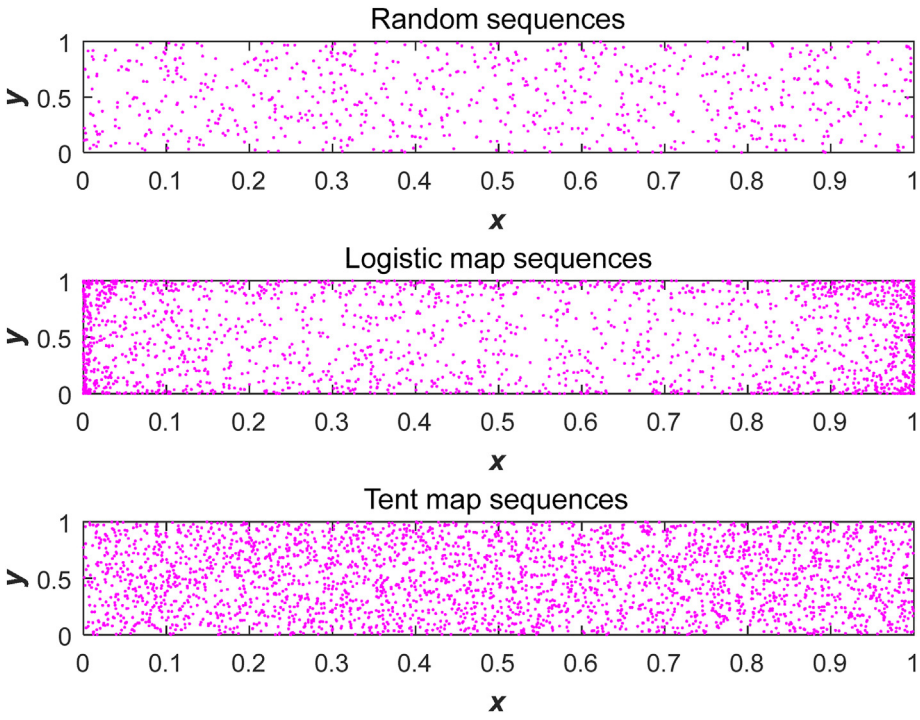

Fig. 6. Comparison of different sequences.

deltas. Next, a subtraction (Eq. (30)) of a learning rate of the gradient from the weight can be obtained. The learning rate has a great influence on the performance of BPNN. When the learning rate increases, the training speed is accelerated; when the learning rate decreases, the forecasting accuracy increases.

$$E = \frac{1}{2}(D-O)^2 = \frac{1}{2}\sum_{k=1}^{l}(D_k - O_k)^2 \tag{28}$$

Here O_k means the k-th output, D represents the desired output.

$$\begin{cases} \Delta I_{ij}^{h} = \eta \delta_j^{y} X_i, (i = 1,2,\ldots,n; j = 1,2,\ldots,t) \\ \Delta w_{jk}^{h+1} = \eta \delta_k^{0} Y_i, (j = 1,2,\ldots,t; k = 1,2,\ldots,l) \end{cases} \tag{29}$$

Here,

$$\begin{cases} \delta_i^0 (D_k - O_k) O_k (1 - O_k), (i = 1,2,\ldots,n; k = 1,2,\ldots,l) \\ \delta_j^y = \left[\sum_{k=1}^{l}(D_k - O_k) f'\left(\sum_{j=1}^{m} W_{jk} Y_j\right) W_{jk}\right] f'\left(\sum_{i=1}^{n} I_{jk} X_i\right), (j = 1,2,\ldots,t; k = 1,2,\ldots,l) \end{cases} \tag{30}$$

Repeat phases 1 and 2 till the training error (Eq. (28)) reaches the satisfied level.

3.4. Principles of MA

As shown in Fig. 4, a classical memristor is composed of one doped layer of thickness $w(t)$ and one un-doped layer. Both the two layers are in a semiconductor film of thickness D. The doped layer has low resistance R_{on} while the un-doped layer has much higher resistance R_{off}. The mathematic model of memristor can be defined as follows:

$$\begin{cases} v(t) = \left(R_{on}\frac{w(t)}{D} + R_{off}\left(1 - \frac{w(t)}{D}\right)\right) i(t) \\ \frac{dw(t)}{dt} = \mu_v \frac{R_{on}}{D} i(t) \\ w(t) = \mu_v \frac{R_{on}}{D} i(t) \end{cases} \tag{31}$$

$$M(q) = R_{off}\left(1 - \mu_v \frac{R_{on}}{D^2} q(t)\right) \tag{32}$$

Here $v(t)$ means an external bias, μ_v means average ion mobility.

Let the time difference of the action potential be different, and there will be different trends in the memristor synaptic weights. Therefore, the memristor array (MA) can be applied to optimize the

weight matrices of the artificial neural networks (ANNs).

3.5. Principles of integrating each ICoGASA by SOM

As shown in Fig. 5, there is an input layer with j nodes and a competition layer with i nodes in a basic SOM. When given an input vector $x = [x_1, x_2, x_3, \ldots, x_j]^T$, which represents the integrated output of each ICoGASA, each neuron at competition layer begins to compete, the winning neuron c can be calculated as follows:

$$\|x - w_c\| = \min_i \left\{ \sqrt{\sum_j (x_j - w_{ij})^2} \right\} \tag{33}$$

Adjust the weight matrix $w = [w_{i1}, w_{i2}, w_{i3}, \ldots, w_{ij}]^T$of competition layer nodes:

$$\begin{cases} w_i(t+1) = w_i(t) + h_{ci}(t)[x(t) - w_i(t)] \\ h_{ci}(t) = \alpha(t)\exp\left(-\dfrac{\|r_c - r_i\|^2}{2\sigma^2(t)} \right) \end{cases} \tag{34}$$

Repeat this competition process until the convergence criterion reaches the satisfied condition.

The advantage of using SOM is that after each iteration we can identify the weight getting better or worse according to Eq. (33). Therefore, the memristor array (MA) [27] composed of memristors [29] that have a stronger ability for learning the historical behavior is applied to optimize the input weights of SOM. Moreover, a new weight updated strategy (WUS) is proposed. That is, after each iteration the weights (W_b) altering for the better are saved, while the ones (W_w) for the worse randomly choose their learning objects and learn from W_b according to Eq. (35). If W_w does not become better for N_w consecutive times, W_w is generated by chaotic map sequences, which has been applied to many optimization fields [34]. To choose more suitable sequences to generate new weights (W_w), comparison is made among the tent map sequence (Eq. (36)), logistic map sequence (Eq. (37)) and random sequence, and the results are shown in Fig. 6.

$$W_{wi}(t+1) = W_{wi}(t) + C\left[W_{wi}(t) - W_{bj}(t)\right] \tag{35}$$

Here W_{bj} means the learning object which are randomly chosen by W_{wi}, C is a real number in [0,1], which is used to control the learning extent of W_{wi}.

$$W_{wi}(t+1) = \begin{cases} W_{wi}(t)/\lambda & 0 \le W_{wi} \le \lambda \\ (1 - W_{wi}(t))/(1-\lambda) & \lambda \le W_{wi} \le 1 \end{cases} \tag{36}$$

Here λ is in [0,1].

$$W_{wi}(t+1) = \mu * W_{wi}(t) * (1 - W_{wi}(t)) \tag{37}$$

Here μ is in [0,4].

As shown in Fig. 6, the ergodicity of random sequence is the worst, the ergodicity of logistic map sequence is relatively focused on the scopes of [0.9, 1] and [0, 0.1], while the ergodicity of tent map sequence can fill the entire two-dimensional space, which indicates that the new solutions generated by tent map sequence can avoid getting trapped in local minima with greater probability. Therefore, tent map sequence is chosen to generate new weights (W_w).

Table 1
The operating environment.

Terms	Details
Two CPUs	Intel Core i7-9750H
RAM	16 GB
Disk	2 TB
Operating system	Windows 10 Pro 64bit
Program language	MATLAB R2016 64bit

Table 2
Division of the dataset.

Cases	Training set (years)	Validation set (years)	Test set (years)
I	2016	2017	2018
	2015–2016		
	2014–2016		
	2013–2016		
	2012–2016		
	2011–2016		
	2010–2016		
	2009–2016		
II	2006–2010	2011	2012
III	2007–2011	2012	2013
IV	2008–2012	2013	2014
V	2009–2013	2014	2015
VI	2010–2014	2015	2016
VII	2011–2015	2016	2017
VIII	2012–2016	2017	2018
IX	2013–2017	2018	2019

Table 3
The best combination of neurons.

Cases	Serial number of sub-model	Generator	Discriminator	SOM
I	1	11,13,12	12,13,12	16
	2	13,11,11	11,12,11	
	3	15,14,15	14,15,15	
II	1	10,11,13	12,11,11	16
	2	12,12,11	10,13,12	
	3	15,15,16	16,14,15	
III	1	13,11,13	12,12,11	14
	2	12,11,12	11,12,13	
	3	16,15,16	16,16,15	
IV	1	13,13,12	11,11,13	15
	2	13,11,13	12,13,12	
	3	16,16,15	17,16,16	
V	1	14,12,13	13,12,13	16
	2	13,14,13	12,13,13	
	3	17,16,15	15,16,16	
VI	1	13,13,14	13,14,14	15
	2	14,13,13	14,14,13	
	3	16,16,15	15,16,17	
VII	1	14,13,12	14,13,13	15
	2	12,14,13	13,13,12	
	3	16,17,17	17,16,15	
VIII	1	16,16,15	15,16,16	18
	2	15,16,17	17,16,17	
	3	18,18,20	19,18,18	
IX	1	18,17,18	18,18,19	21
	2	17,18,18	19,18,19	
	3	20,21,21	21,20,21	

3.6. The electricity demand forecasting process of the proposed method

***Step* 1** Filter the time-series of the original input variables by UKF, then calculate the incremental percentages of current and previous sample points, and divide the calculated results into training set and test set;

***Step* 2** Set parameters. Let the maximum training times be T_{max}, the number of ICoGASA be N_j, the total nodes at the competition layer of SOM be k, and the dimension of input features be n_j. The p-th weight matrix of ICoGASA and SOM which are calculated by MA can be defined as follows:

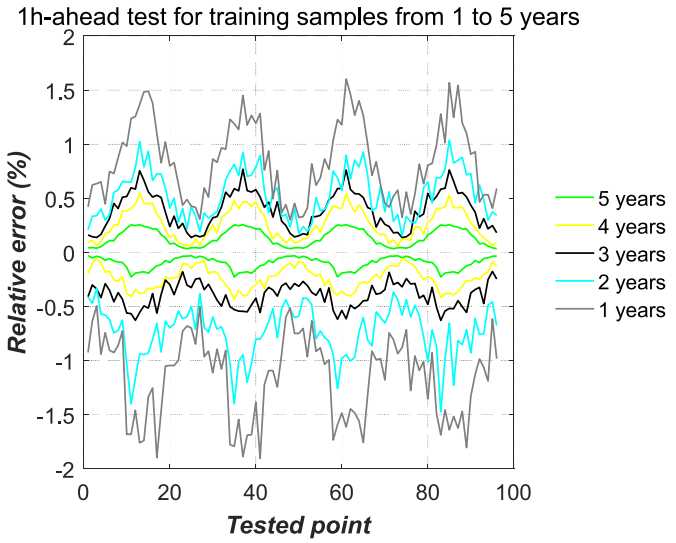

Fig. 7. 1h-ahead test for training samples from 1 to 5 years.

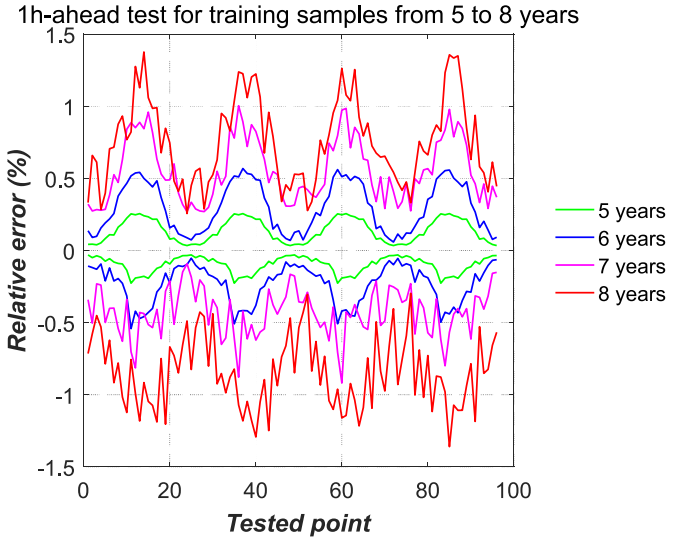

Fig. 8. 1h-ahead test for training samples from 5 to 8 years.

$$x^p = [w^p_{11}, w^p_{12}, \ldots w^p_{1c}, w^p_{21}, w^p_{22}, \ldots w^p_{2c}, \ldots, w^p_{r1}, w^p_{r2}, \ldots, w^p_{rc}],$$
$$r = 1, 2, \ldots, n, c = 1, 2, \ldots, k \tag{38}$$

For ICoGASA, r means the number of sub-ICoGASA, c means the number of the sub-DBNs of each ICoGASA; while for SOM, r and c mean the number of the nodes at input layer and competition layer, respectively.

Step 3 Train model. Generate positive error, negative error and mixed error, which are more similar to the real data through the stacked auto-encoder. Next, generate realistic synthesized samples (RSS) by generators of GANs according to the training sample; and distinguish RSD from real training sample with discriminator model. When this progress is finished, integrate the outputs of the discriminators of each sub-ICoGASA using MA.

Step 4 Integrate forecasting. The integrated output of each ICoGASA serves as the input of SOM. Optimize the weights of SOM using MA and WUS.

Step 5 Determine whether the end condition of the proposed algorithm is met or not. If the maximum number of iterations or the expected error is not met, turn to ***Step3***; otherwise, output the results for electricity demand forecasting.

3.7. Principles of VMD

The purpose of variational mode decomposition (VMD) is to extract better sub-sequences from the time-series variables that need to be processed, the detailed decomposition steps are as follows:

Step 1: The load-series (V_i) that need to be processed are decomposed into a series of subsequences, represented by $\{u_k(t)\}$, $k = 1, 2, \ldots K$. And then, the unilateral frequency spectrum (UFS) of $u_k(t)$ is calculated and shifted into a "baseband" by Eq. (39) and Eq. (40), respectively.

$$UFS = \left[\delta(t) + \frac{j}{\pi t} * u_k(t)\right] \tag{39}$$

$$baseband = \left[\delta(t) + \frac{j}{\pi t} * u_k(t)\right] e^{-j\omega_k t} \tag{40}$$

Step 2: Then the bandwidth is estimated by Eq. (41):

$$\begin{cases} \min\limits_{\{u_k\},\{\omega_k\}} \left\{ \sum\limits_k \left\| \partial_t \left[\delta(t) + \frac{j}{\pi t} * u_k(t) \right] e^{-j\omega_k t} \right\|_2^2 \right\} \\ s.t. \sum\limits_k u_k = f \end{cases} \tag{41}$$

Step 3: Eq. (41) can be converted into Eq. (42) via the Lagrangian multiplier $\lambda(t)$ and penalty factor α, The purpose of this process is to decide the number of subsequences of the time-series variables that need to be processed.

$$L(\{u_k\}, \{\omega_k\}, \lambda) = \alpha \sum_k \left\| \partial_t \left[\delta(t) + \frac{j}{\pi t} * u_k(t) \right] e^{-j\omega_k t} \right\|_2^2 + \left\| f(t) - \sum_k u_k(t) \right\|_2^2 + \langle \lambda(t), f(t) - \sum_k u_k(t) \rangle. \tag{42}$$

Step 4: Calculate the augmented Lagrangian saddle point:

$$u_k^{n+1} = \underset{u_k, u_k \in X}{\operatorname{argmin}} \left\{ \alpha \| j\omega[(1 + \operatorname{sgn}(\omega + \omega_k)) \hat{u_k}(\omega + \omega_k)] \|_2^2 + \left\| f(\hat{\omega}) - \sum_i \hat{u_i}(\omega) + \frac{\hat{\lambda(\omega)}}{2} \right\|_2^2 \right\} \tag{43}$$

Step 5: Calculate the non-negative frequency and its solution by Eqs. (44) and (45), respectively:

$$u_k^{n+1} = \underset{u_k, u_k \in X}{\operatorname{argmin}} \left\{ \int_0^\infty 4\alpha(\omega - \omega_k)^2 |\hat{u_k}(\omega)|^2 + 2 \left| f(\hat{\omega}) - \sum_i \hat{u_i}(\omega) + \frac{\hat{\lambda(\omega)}}{2} \right|^2 d\omega \right\} \tag{44}$$

$$u_k^{n+1}(\omega) = \frac{f(\hat{\omega}) - \sum\limits_{i \neq k} \hat{u_i}(\omega) + \frac{\hat{\lambda(\omega)}}{2}}{1 + 2\alpha(\omega - \omega_k)^2} \tag{45}$$

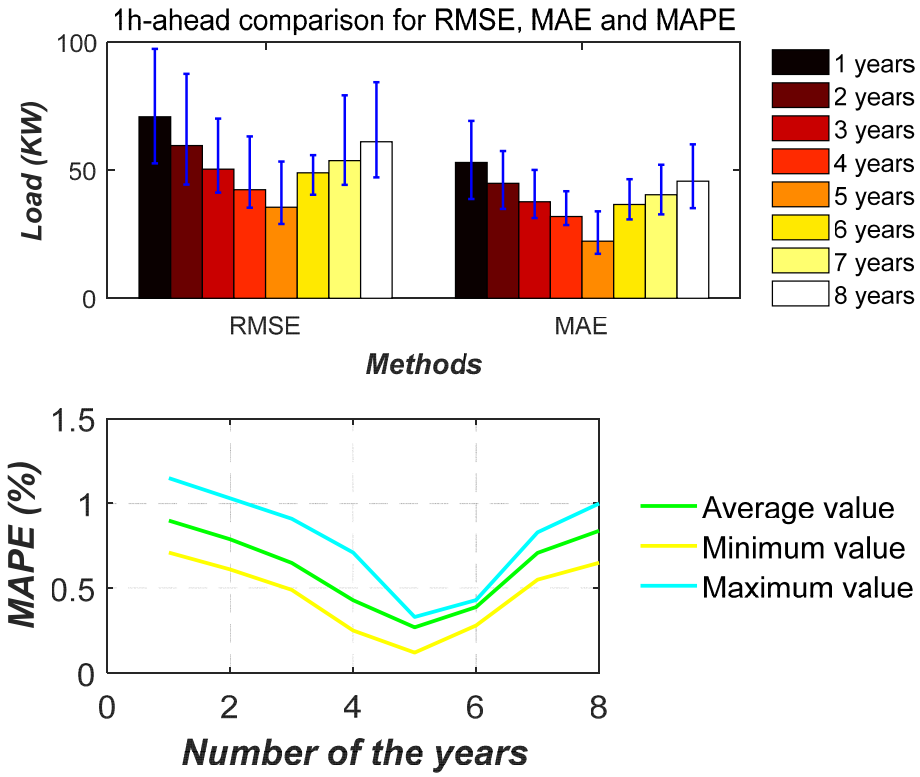

Fig. 9. 1h-ahead comparison for RMSE, MAE and MAPE.

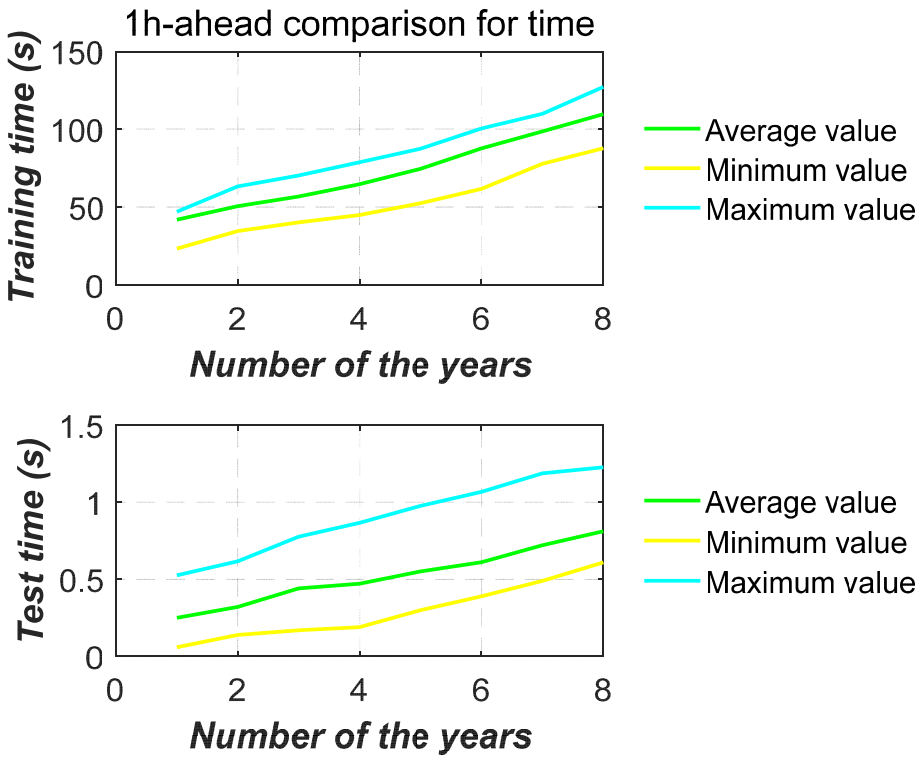

Fig. 10. 1h-ahead comparison for time.

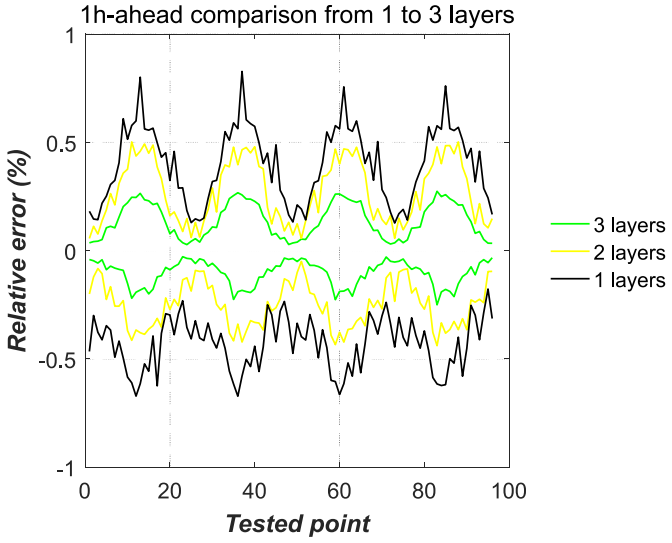

Fig. 11. 1h-ahead comparison from 1 to 3 layers.

Step 6: Calculate the updated Lagrangian multiplier $\lambda(t)$ and center frequency ω_k.

$$\omega_k^{n+1} = \frac{\int_0^\infty \omega |u_k(\omega)|^2 d\omega}{\int_0^\infty |u_k(\omega)|^2 d\omega} \tag{46}$$

$$\lambda^{n+1}(\omega) = \lambda^n(\omega) + \tau\left[f(\omega) - \sum_k u_k^{n+1}(\omega)\right] \tag{47}$$

Step 7: Repeat the same process until

$$\sum_k \left\|u_k^{n+1} - u_k^n\right\|_2^2 \Big/ \left\|u_k^n\right\|_2^2 < \varepsilon \tag{48}$$

When the end condition (Eq. (48)) is satisfied, the decomposed results (band-limited intrinsic mode functions, BLIMFs) from the time-series variables that need to be processed can be obtained.

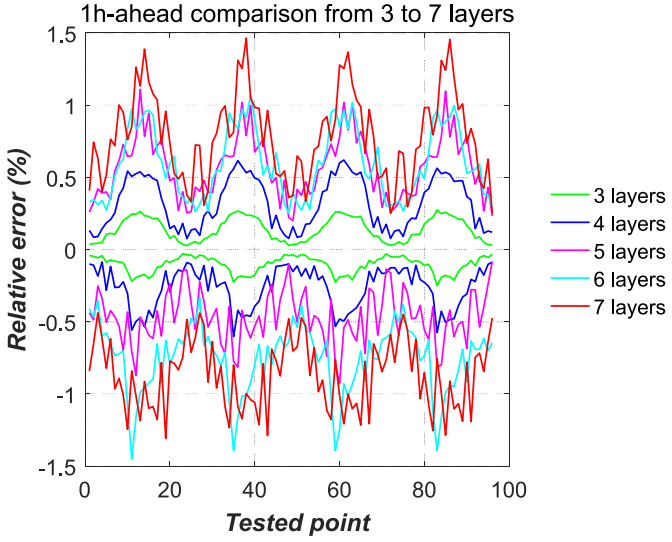

Fig. 12. 1h-ahead comparison from 3 to 7 layers.

3.8. Principles of EMD

The purpose of empirical mode decomposition (EMD) is to decompose the load-series that need to be processed into a series of intrinsic mode functions (IMFs), the detailed decomposition steps are as follows:

Step 1: Identify all minima and maxima of the load-series ($V(i)$). Apply cubic spline function to produce the lower envelope ($l(i)$) and upper envelope ($u(i)$), and then calculate the mean value $M(i)$ of the two envelopes:

$$M(i) = \frac{l(i) + u(i)}{2} \tag{49}$$

Step 2: Calculate the difference between $M(i)$ and $V(i)$:

$$H(i) = V(i) - M(i) \tag{50}$$

Step 3: Replace $V(i)$ with H(i) and repeat Steps1 and 2 until $H_k(i)$ becomes an IMF in the k-th iteration, the evaluation criterion is that the pre-determined threshold is more than D_k.

$$D_k = \frac{\sum_{i=0}^{T} |H_{(k-1)}(i) - H_k(i)|^2}{\sum_{i=0}^{T} |H_{(k-1)}(i)|^2} \tag{51}$$

Step 4: Calculate the difference between H(i) and the first IMF:

$$R_1(i) = V(i) - IFM_1(i) \tag{52}$$

Step 5: Take $R_1(i)$ as the time-series (V_i) to be processed, repeat Steps 1–4 until $R_n(i)$ becomes a monotone function or the pre-determined threshold is more than $R_n(i)$.

3.9. Principles of WT

A set of sub-subsequences with low frequency and high frequency can be obtained by wavelet transform (WT) with different mother wavelet functions. The load-series ($F_i(t)$) to be processed are expressed as follows:

$$F_i(t) = \sum_k \varsigma_{j_0}(k) 2^{\frac{j_0}{2}} \phi\left(2^{j_0}t - k\right) + \sum_k \sum_{j=j_0}^{\infty} \xi_j(k) 2^{\frac{j}{2}} \psi\left(2^j t - k\right), \quad I = 1, 2, \ldots, N \tag{53}$$

Here $\varsigma_{j_0}(k)$ and $\xi_j(k)$ represent the approximation and detail coefficients, respectively; $\psi(t)$ and $\phi(t)$ represent the selected mother wavelet functions (Coiflets (coif), Daubechies (db), etc.) and the corresponding scaling functions of them, respectively; An example of two-stage decomposition are as follows:

$$F_1(t) = F_1A_1(t) + F_1D_1(t) = F_1A_2(t) + F_1D_2(t) + F_1D_1(t) \tag{54}$$

Here $F_1D_2(t)$ and $F_1D_1(t)$ are the sub-subsequences with high frequency; $F_1A_2(t)$ means the sub-subsequences with low frequency, which reflects the general trend of $F_1(t)$.

4. Test cases

The data set used in this paper, consists of recorders of the weather factors, electricity price, and two Chinese residential buildings' data. One is in a northern city (Daqing, ***Case I-Case VII***) and the other is in a southern city (Sansha, ***Case VIII-Case IX***).

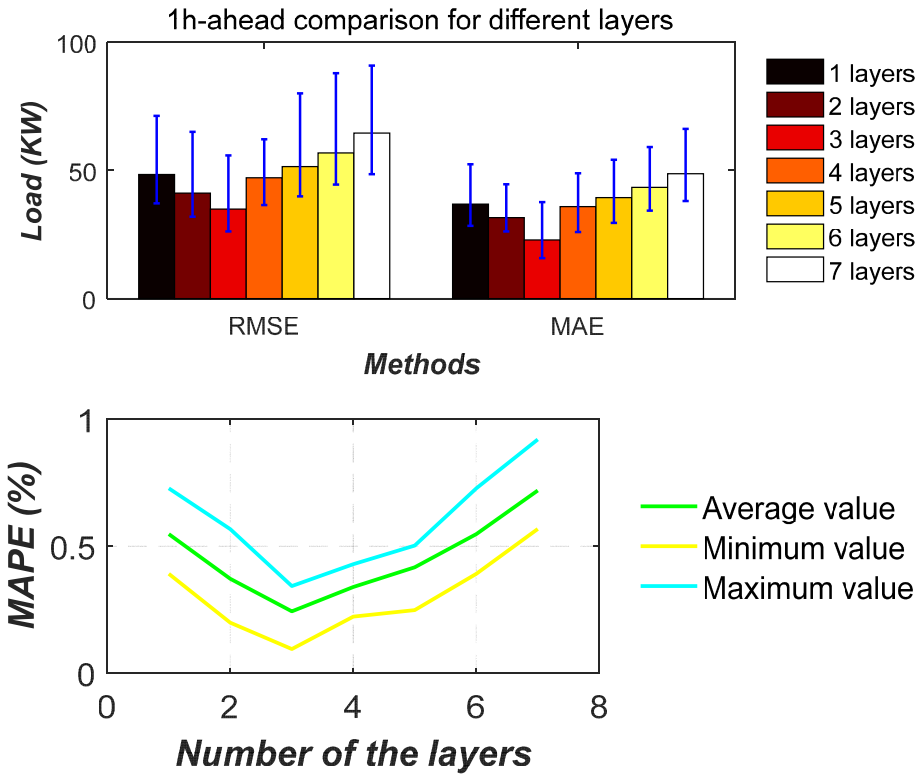

Fig. 13. 1h-ahead comparison for different layers.

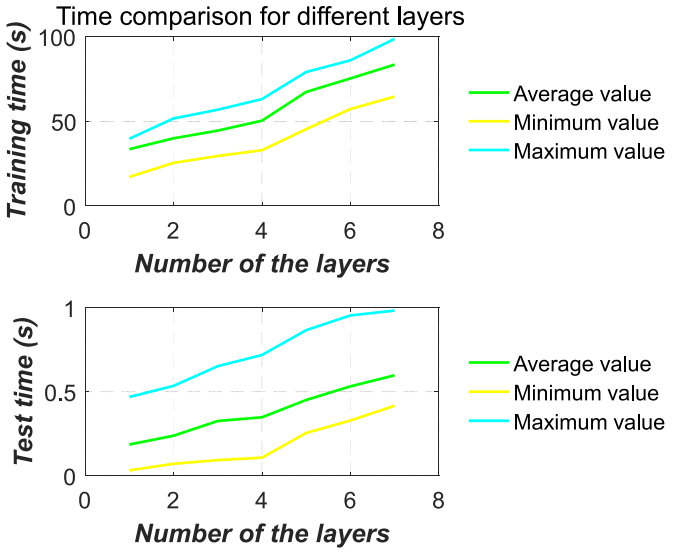

Fig. 14. Time comparison for different layers.

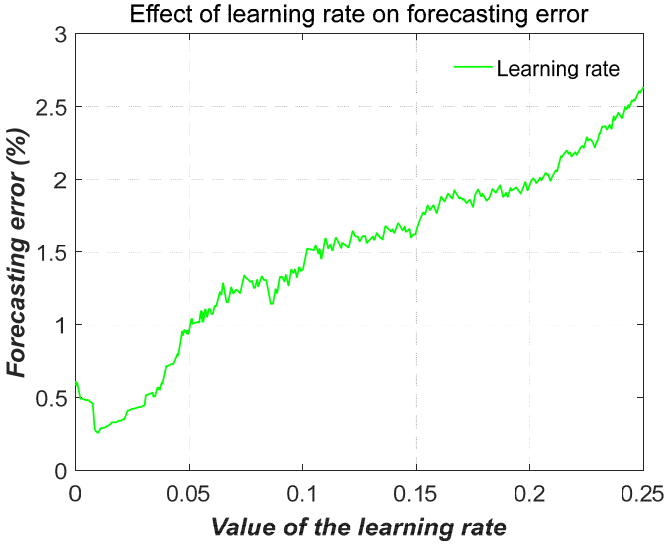

Fig. 15. Effect of learning rate on forecasting error.

Daqing City is located in the in the western part of Heilongjiang Province, with a total area of 21,219 square kilometers, a resident population of about 3.2 million, and a sex ratio of 101.76%. Daqing City is dominated by the continental monsoon climate and it belongs to the northern temperate zone, and the four seasons are quite distinctive here. The annual sunshine hours range between 2600 and 2900 h, and the annual rainfall ranges between 400 and 550 mm. Sansha is a city, located in the southern part of Hainan Province, and its land area is more than 20 square kilometers, with a resident population of more than 2500 people, and a sex ratio of 150.28%. Sansha City is dominated by the tropical maritime monsoon climate, and the hot weather continues throughout the

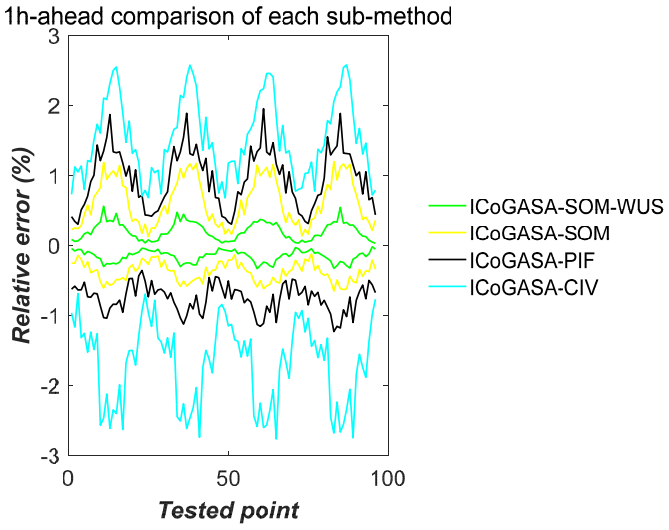

Fig. 16. 1h-ahead comparison of each sub-method.

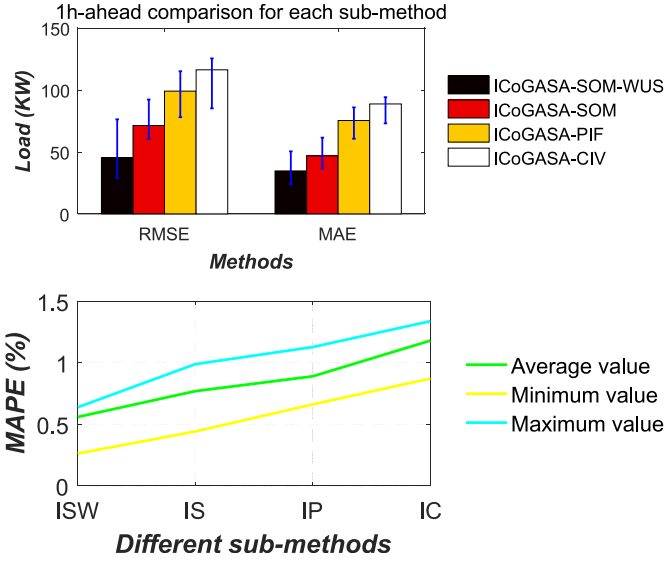

Fig. 17. 1h-ahead comparison for each sub-method.

year; its average annual rainfall ranges between 1500 and 1900 mm.

The data is gathered from the China's State Grid (which manages national electricity supply and sales.) and is sampled every 15 min. Table 1 shows the operating environment in each ***Case***. Table 2 shows the division of the dataset used in this paper. Table 3 illustrates the best combination of neurons in each ***Case***. The root mean square error (RMSE), mean absolute percentage error (MAPE), relative error (RE) and mean absolute error (MAE), defined by Eq. (55), are chosen to evaluate the forecasting performance of the methods set forth herein for comparison.

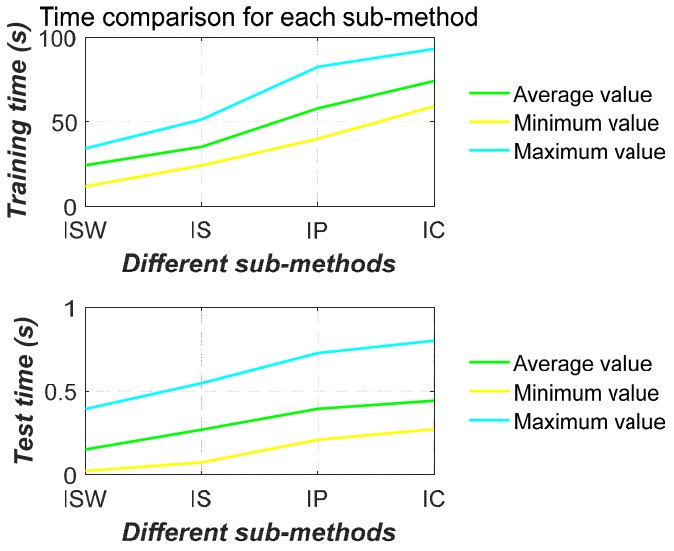

Fig. 18. Time comparison for each sub-method.

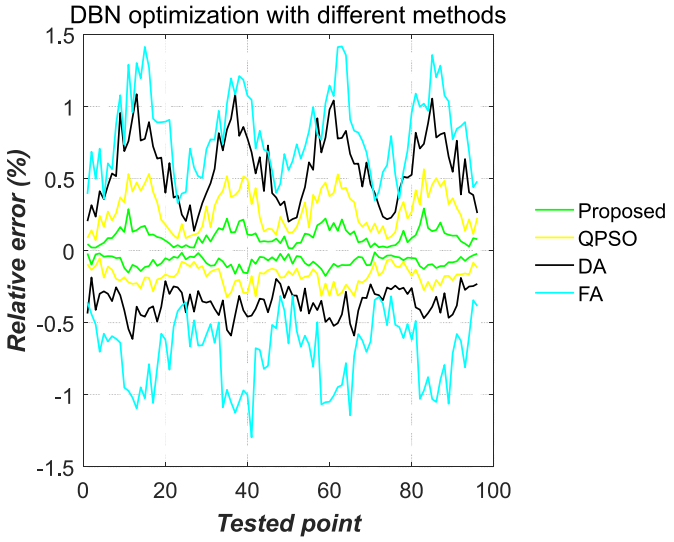

Fig. 19. DBN optimization with different methods.

$$\begin{cases} RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i^t - y_i\right)^2} \\ \text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i^t - y_i}{y_i}\right| \times 100\% \\ RE = \frac{y_i^t - y_i}{y_i} \times 100\% \\ MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i^t - y_i\right| \end{cases} \tag{55}$$

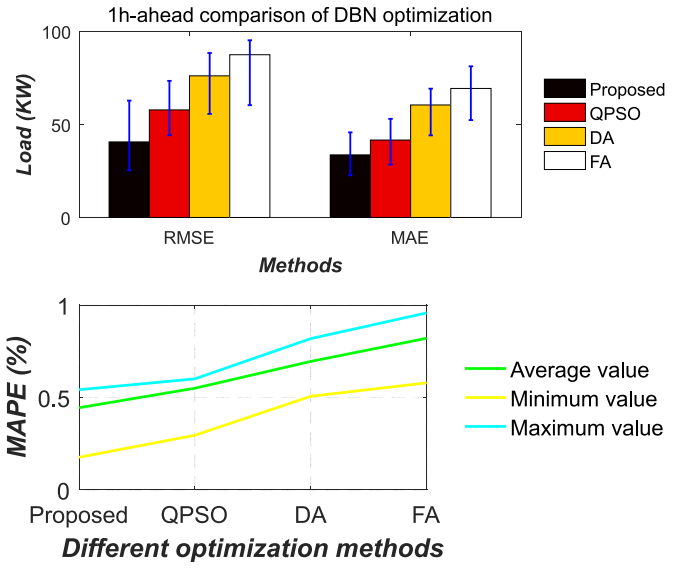

Fig. 20. 1h-ahead comparison of DBN optimization.

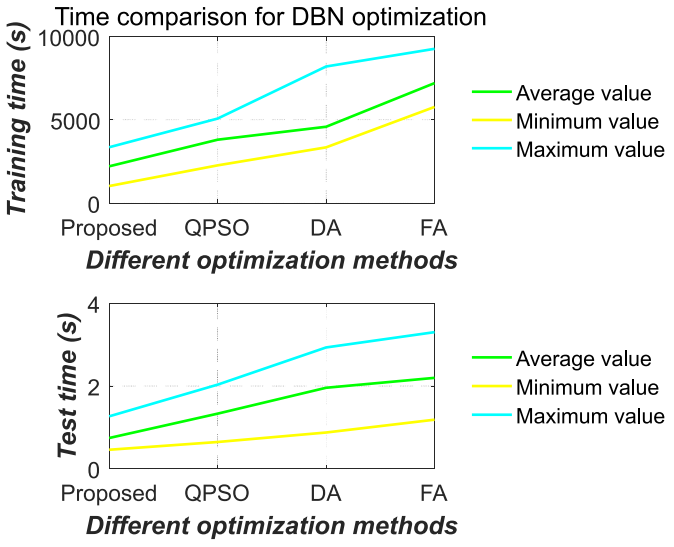

Fig. 21. Time comparison for DBN optimization.

where y_i^t represents forecasted value of the load series, y_i represents actual value of the load series, n means the total number of tested points.

Case I. This case verifies the maximum number of training samples. Firstly, the validation dataset and test dataset are fixed in 2017 and 2018, respectively. Secondly, the training set has been gradually increasing year by year since 2016. Figs. 7 and 8 compare the relative error of the proposed method with the same validation dataset and test dataset but with the different training sets. Fig. 9 compares the RMSE, MAPE and MAE of the proposed method with the same method in Figs. 7 and 8. As shown in Fig. 9, the bars mean the average of RMSE and MAE. The blue lines on either bar mean the minimum and maximum of each RMSE and MAE. It should be noted that the meanings of the bars and the blue lines on

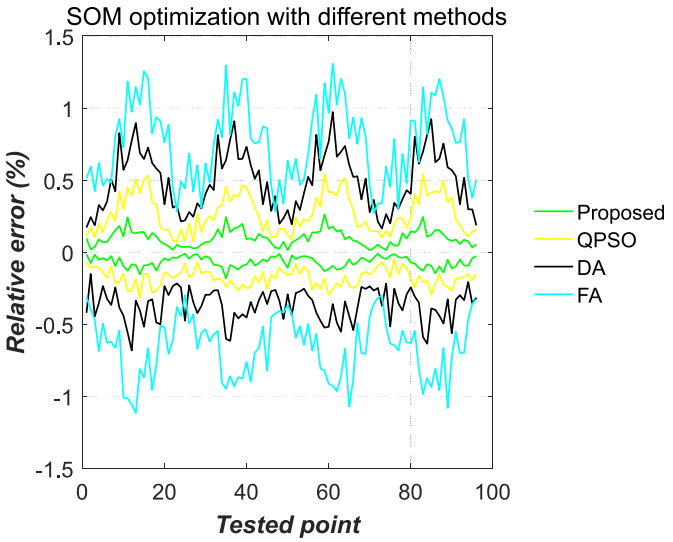

Fig. 22. SOM optimization with different methods.

them in the following figures of all ***Cases*** are the same as those in Fig. 9. The test results show that the scopes (maximum, average and minimum) of RE, RMSE, MAPE and MAE all reach the minimum value when the number of training samples is 5 years. Fig. 10 compare the training time and test time. As the number of training samples increase, the training time continues to be increased. When the number of training samples reaches 5 years, the maxima of the training time and test time get 87.537s and 0.975s, respectively, satisfying the demand for 1h-ahead load forecasting.

Case II. This case verifies the maximum number of layers. Figs. 11 and 12 compare the relative error of the proposed method with the different number of layers. Fig. 13 compares the RMSE, MAPE and MAE of the proposed method with the same method in Figs. 11 and 12. The test results show that the scopes (maximum, average and minimum) of RE, RMSE, MAPE and MAE all reach the minimum value when the number of layers is 3. Fig. 14 compares the training time and test time. As the number of the layer increases, the training time continues to be increased. When the number of layers reaches 3, the maxima of the training time and test time get 56.820s and 0.651s, respectively, which can meet the demand for 1h-ahead load forecasting. Therefore, the best number of layers is 3. According to a previous study [35–37], the highest value of the learning rate is usually no more than 0.2, and the lowest is generally no less than 0.001. Therefore, in this paper, the test range of the learning rate is set up from 0.0001 to 0.25, with an increasing step value of 0.0001.Fig. 15 makes a comparison on the effect of the learning rate on the forecasting error. Obviously, when the learning rate reaches 0.01, the minimum value of the forecasting error can be obtained.

Case III. This case verifies the forecasting performance of each sub-method. That is, the proposed sub-technologies are added one by one for comparison. In other words, the ICoGASA (described in Section 3 Chapter 1) with the conventional input variables (ICoGASA-CIV, abbreviated as IC), the ICoGASA with the proposed input feature (ICoGASA-PIF, abbreviated as IP, described in Section 2), the ICoGASAs with PIF integrated by SOM (ICoGASA- SOM, abbreviated

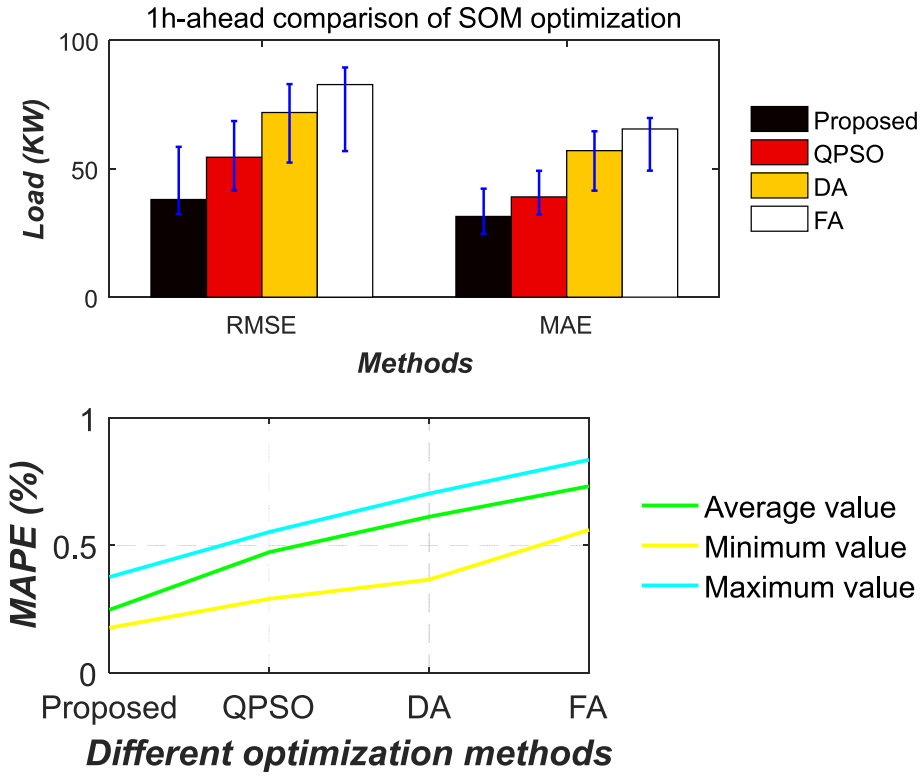

Fig. 23. 1h-ahead comparison of SOM optimization.

as IS, described in Section 3 Chapter 5), the ICoGASAs with PIF integrated by SOM and optimized by the proposed weight updated strategy (ICoGASA-SOM-WUS, abbreviated as ISW, described in Section 3 Chapter 5) are compared. Fig. 16 compares the relative error of the proposed sub-technologies. Obviously, the RE scope of the ICoGASA integrated all sub-technologies (ICoGASA-SOM-WUS) is best. Fig. 17 compares the RMSE, MAPE and MAE of the proposed sub-technologies. Remarkably, the scopes of the RMSE, MAPE and MAE of ICoGASA-SOM-WUS are the most optimal. Fig. 18 makes a comparison on the training time and test time of all sub-technologies. Although the maxima of the training time and test time reach 93.123s and 0.801s, respectively, the demand for 1h-ahead load forecasting still can be met.

Case IV. This case demonstrates the weights optimization effect of DBN and SOM with different swarm intelligence algorithms. Nowadays, a large number of swarm intelligence algorithms are applied to the field of time series forecasting. Unfortunately, due to the limitation of individual ability, only three representative methods are chosen for a comparative study with the proposed method. To be specific, the firefly algorithm (FA) [17], the quantum-

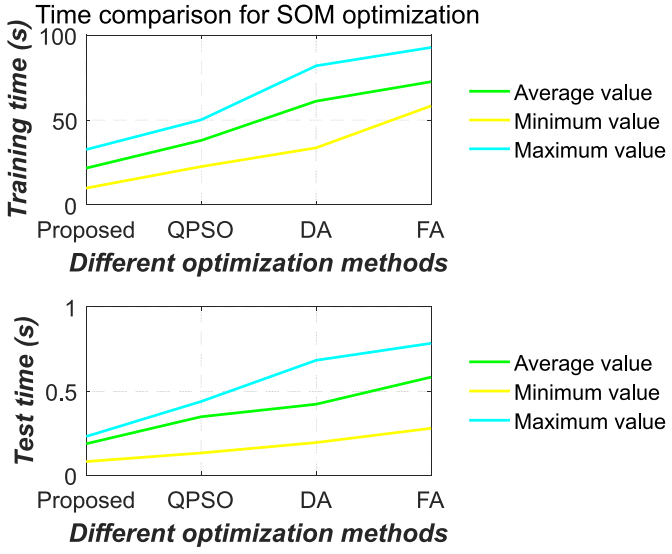

Fig. 24. Time comparison for SOM optimization.

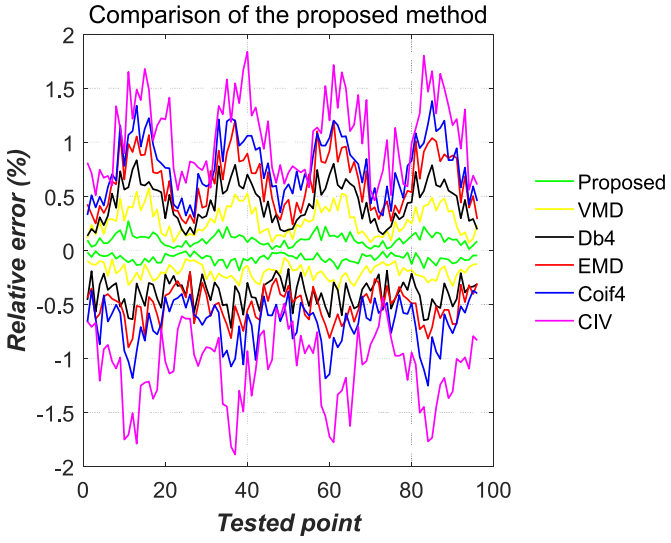

Fig. 25. Comparison of the proposed method.

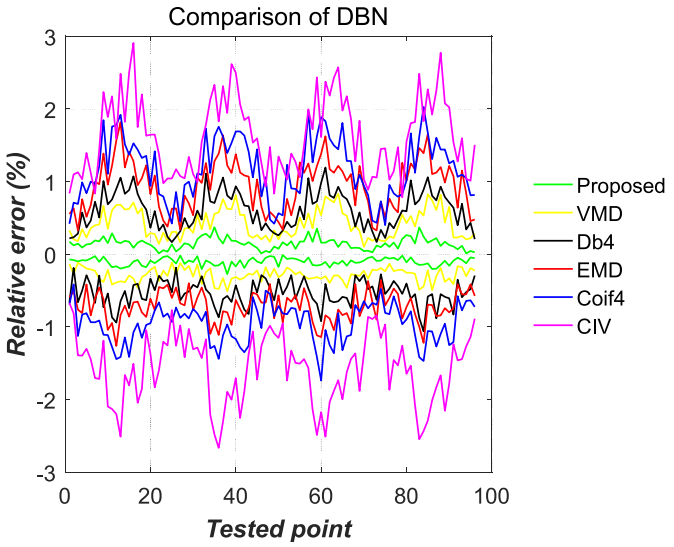

Fig. 26. Comparison of DBN

behaved particle swarm optimization (QPSO) [38], the dragonfly algorithm (DA) [39] and the proposed method are compared. Fig. 19 shows the weights optimization effect of DBN with all kinds of methods. The RE scopes of the proposed method, QPSO, DA and FA are [-0.176, 0.292] %, [-0.324, 0.565] %, [-0.612, 1.084] %, [-1.295, 1.416] %, respectively. Fig. 22 shows the weights optimization effect of SOM with all kinds of methods. The RE scopes of the proposed method, QPSO, DA and FA are [-0.181, 0.262] %, [-0.315, 0.544] %, [-0.682, 0.971] %, [-1.115, 1.307] %, respectively. A comprehensive comparison from Figs. 19 and 22 implies that the optimization effect of the proposed method is the most optimal; moreover, there is no much difference in the RE scope of optimizing DBN and SOM. As shown in Fig. 20, the RMSE averages of optimization DBN with the proposed method, QPSO, DA and FA reach 40.568, 57.804, 76.075, 87.436, respectively; the MAE averages reach 33.634, 41.584, 60.458, 69.295, respectively; The MAPE averages are 0.443%, 0.549%, 0.694%, and 0.820%, respectively. As shown in Fig. 23, the RMSE averages of optimization SOM with the proposed method, QPSO, DA and FA reach 38.075, 54.527, 71.967, 82.812, respectively;

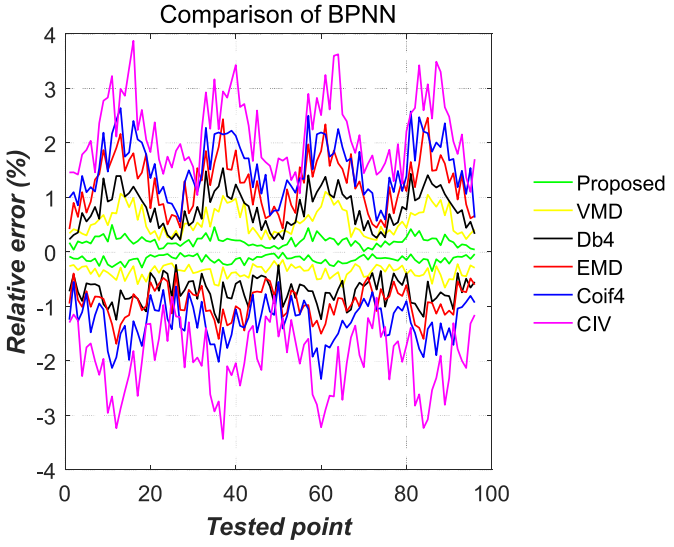

Fig. 27. Comparison of BPNN.

the MAE averages are 31.455, 39.044, 57.061, 65.495, respectively; The MAPE averages are 0.246%, 0.474%, 0.612%, and 0.732%, respectively. A comprehensive comparison from Figs. 20 and 23 implies that the optimization effect of the proposed method is the most optimal; moreover, there is no much difference in the averages of RMSE, MAE and MAPE of optimizing DBN and SOM. In other words, the forecasting effects of optimizing DBN and SOM are similar, which both can satisfy the demand for 1h-ahead load forecasting. Fig. 24 compares the training time and test time of SOM optimization. Although the maxima of the training time and test time reach 92.791s and 0.783s, respectively, the demand for 1h-ahead load forecasting still can be met. However, as shown in Fig. 21, the maxima of the training time and test time of DBN optimization reach 9267.925s and 3.304s, respectively. Clearly, the training time could not meet the demand for 1h-ahead load forecasting. Therefore, the weight of SOM is optimized, instead of DBN.

Case V. This case proves the forecasting performance with different input features. The original time-series variables are decomposed by empirical mode decomposition (EMD) [40], wavelet-based methods (Daubechies (Db4), Coiflets (Coif4)) [41] and variational mode decomposition (VMD) [42,43], respectively. After decomposition, a series of intrinsic mode functions (IMFs), a set of sub-subsequences with low frequency and high frequency (LFHF), and a series of band-limited intrinsic mode functions (BLIMFs) can be acquired, respectively. The IMFs, LFHF and BLIMFs, and conventional input variables (CIV) are tested on deep belief network (DBN) [16], back-propagation neural network (BPNN) [14], and with the proposed ensemble method, respectively. The detailed process of VMD, EMD and wavelet-based methods and VMD are described in Section 3 Chapters 7–9, respectively. Figs. 25–27 compare the relative error of the proposed method, DBN and BPNN, respectively. Fig. 28 makes a comparison on the RMSE and MAE of the proposed method, DBN and BPNN. Fig. 29 compares the MAPE of the proposed method, DBN and BPNN. The test results demonstrate that the scopes (maximum, average and minimum) of RE, RMSE, MAPE and MAE of the proposed input feature are the most optimal in the proposed ensemble method, DBN and BPNN. The training time and test time of different input features are compared in Fig. 30. Remarkably, the proposed input feature has the most optimal scope (maximum, average and minimum) of training time and test time in the proposed ensemble method, DBN and BPNN. Nevertheless, the proposed ensemble method spends more training time and test time than those of DBN and BPNN. Although the maxima of the training time and test time reach 108.877s and 1.731s, respectively, 1h-ahead load forecasting still can be satisfied.

Case VI. This case checks the forecasting performance of different sample simulation strategies. The proposed sample simulation strategy (ICoGASA), CoGASA and GAN are tested. Fig. 31 compares the relative error of the ICoGASA, CoGASA and GAN, respectively. Fig. 32 makes a comparison on the RMSE, MAE and MAPE of the sample simulation strategies mentioned above. The test results imply that the scopes (maximum, average and minimum) of RE, RMSE, MAPE and MAE of the proposed sample simulation strategy (ICoGASA) are the most optimal. The training time and test time of different sample simulation strategies are compared in Fig. 33. Obviously, the proposed sample simulation strategy (ICoGASA) has the most optimal scope (maximum, average and minimum) of training time and test time. The maxima of the training time and test time reach 112.805s and 0.898s, respectively, which meet the demand for 1h-ahead load forecasting.

Case VII. This case compares the forecasting effectiveness with different ensemble methods, which are used to integrate each sub-

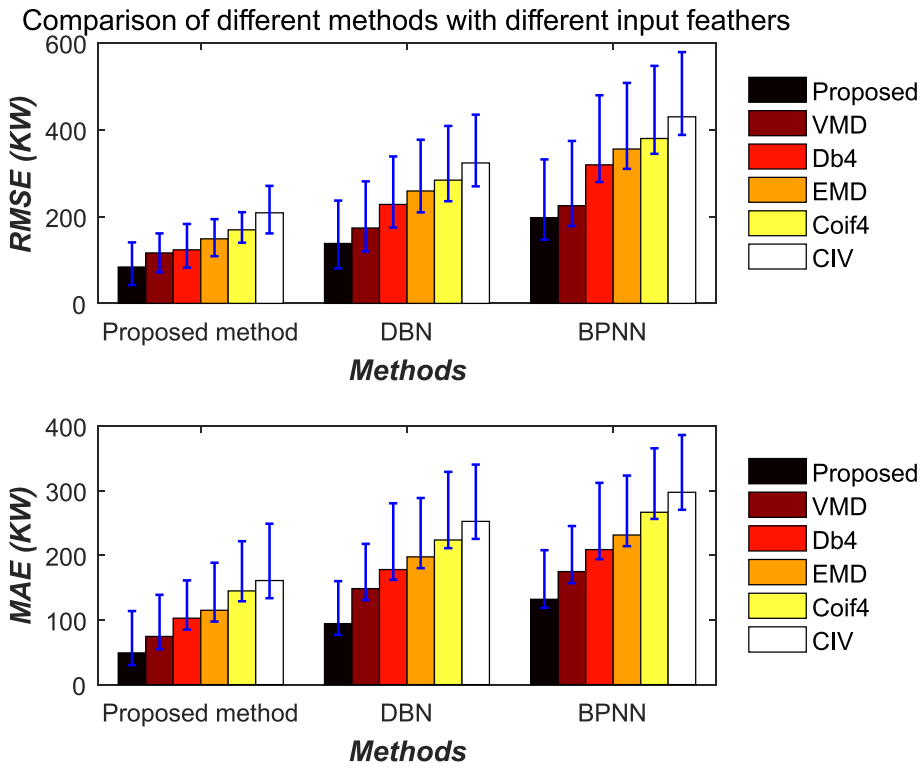

Fig. 28. Comparison of different methods with different input feathers.

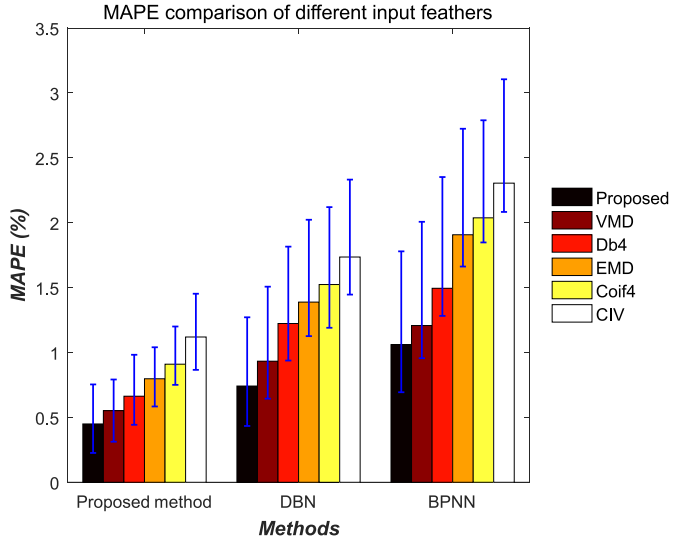

Fig. 29. MAPE comparison of different input feathers.

ICoGASA. The proposed ensemble method (SOM-WUS), partial least squares regression (PLSR) [18], adaptive boosting (AdaBoost) [44] and random forest (RF) [45] are tested. Fig. 34 compares the relative error of different ensemble methods. Fig. 35 makes a comparison on the RMSE, MAE and MAPE of the ensemble methods mentioned above. The test results imply that the scopes (maximum, average and minimum) of RE, RMSE, MAPE and MAE of the proposed ensemble method are the most optimal. The training time and test time of different ensemble methods are compared in Fig. 36. Obviously, the proposed sample simulation strategy (ICoGASA) has the most optimal scope (maximum, average and minimum) of training time and test time. The maxima of the training time and test time reach 105.342s and 0.665s, respectively, which meet the demand for 1h-ahead load forecasting.

Case VIII. This case compares the forecasting performance of different methods in forecasting the electricity demand of weekends (Saturdays and Sundays). The proposed method, DBN [16], SVR [17] and BPNN [14] are tested. Figs. A1 and A2 compare the relative error of different methods in Saturdays and Sundays. Figs. A3 and A4 compare the RMSE, MAE and MAPE of the methods above. The test results imply that the scopes (maximum, average and minimum) of RE, RMSE, MAPE and MAE of the proposed ensemble method are the most optimal, both on Saturdays and Sundays. Fig. A5 makes a comparison on the training time and test time of different methods. Apparently, the proposed ensemble method has the most optimal scope (maximum, average and minimum) of training time and test time, both on Saturdays and Sundays.

Case IX. This case compares the forecasting performance of different methods in forecasting the electricity demand of holidays. The proposed method, DBN [16], SVR [17] and BPNN [14] are tested. Fig. A6 compares the relative error of different methods on holidays. Fig. A7 makes a comparison on the RMSE, MAE and MAPE of the methods above. The test results show that the scopes (maximum, average and minimum) of RE, RMSE, MAPE and MAE of the proposed ensemble method are the most optimal. The training time and test time of different methods are compared in Fig. A8.

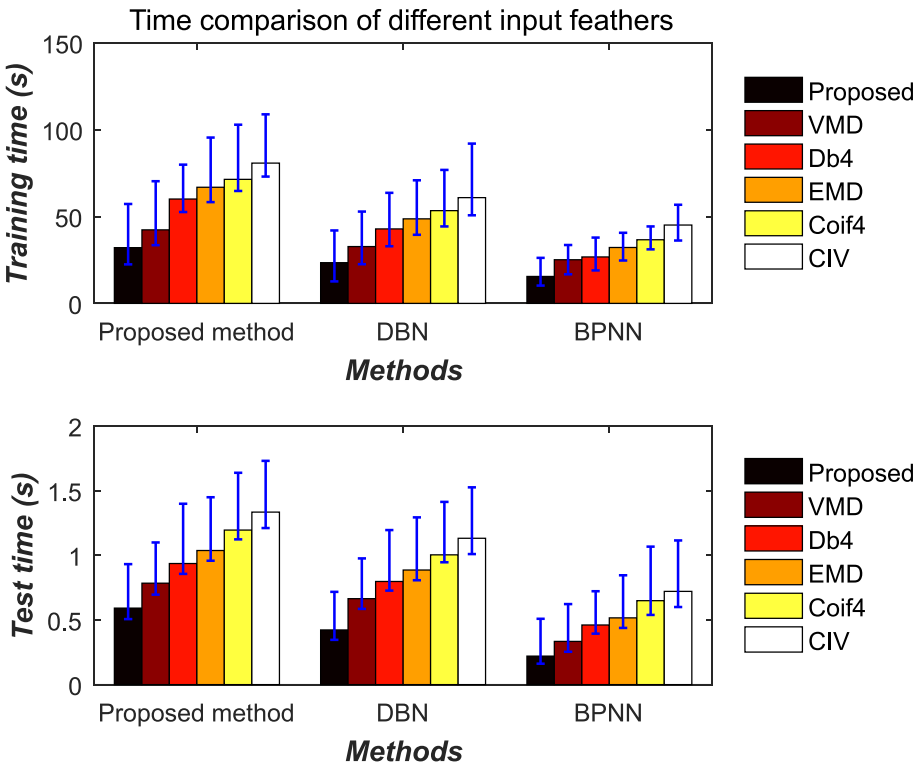

Fig. 30. Time comparison of different input feathers.

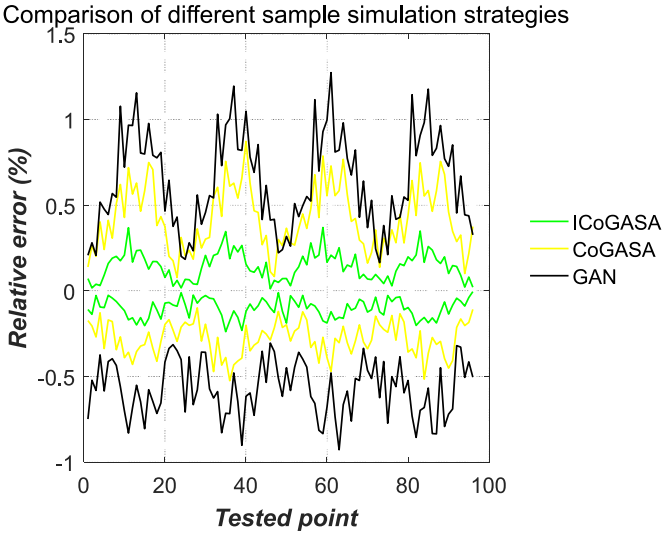

Fig. 31. Comparison of different sample simulation strategies.

Distinctly, the proposed ensemble method has the most optimal scope (maximum, average and minimum) of training time and test time.

5. Discussion and limitation

5.1. Discussion

The effectiveness of the proposed ensemble method in this paper has been tested in ***Cases I-IX***. The maximum number of training samples is tested in ***Case I***. Apparently, when the number of training samples is less than 5 years, the forecasting performance is enhanced with the increase of the number of training samples; when the number of training samples is more than 5 years, the forecasting performance is decreased with the increase of the number of training samples. The possible reason is that under-fitting occurs when the number of training samples is less than 5 years; while over-fitting occurs when the number of training samples is more than 5 years. Under-fitting usually refers to the poor performance of the forecasting model caused by insufficient training samples; while over-fitting usually refers to the poor performance of the forecasting model caused by introducing too many terms (samples or layers, etc.) [46,47]. The maximum number of layers is tested in ***Case II***. In evidence, when the number of layers is less than 3, the forecasting performance is increased with the number of layers; when the number of layers is more than 3, the forecasting performance is decreased with the increase of the number of training samples. The possible reason is that under-fitting occurs when the number of layers is less than 3; while over-fitting occurs when the number of layers is more than 3. The same phenomenon also exists in the learning rate. ***Case III*** verifies the forecasting performance of each sub-technology. Distinctly,

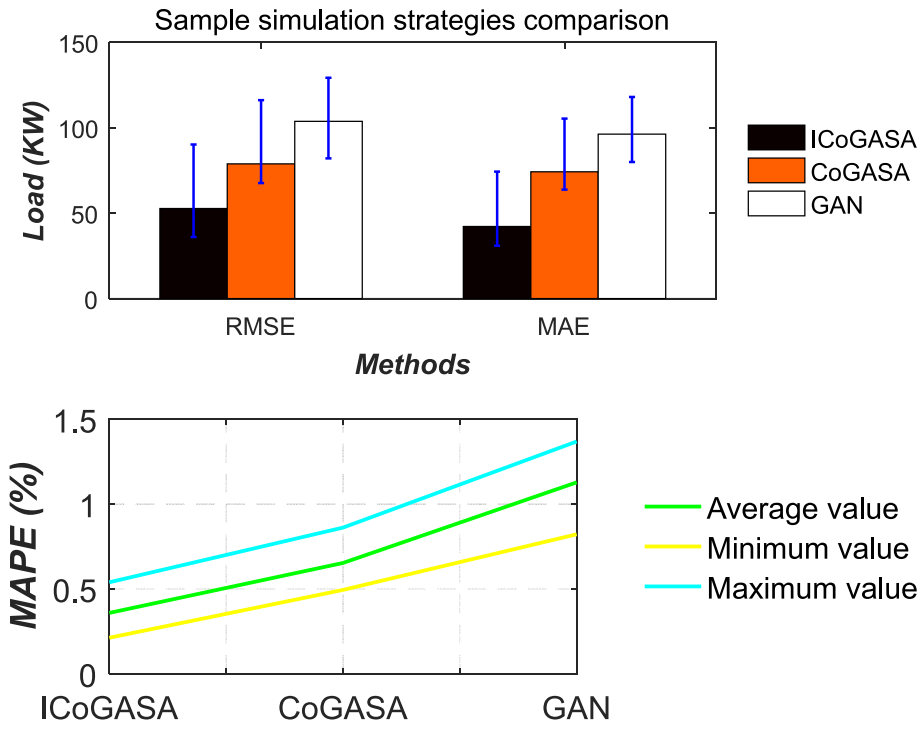

Fig. 32. Sample simulation strategies comparison.

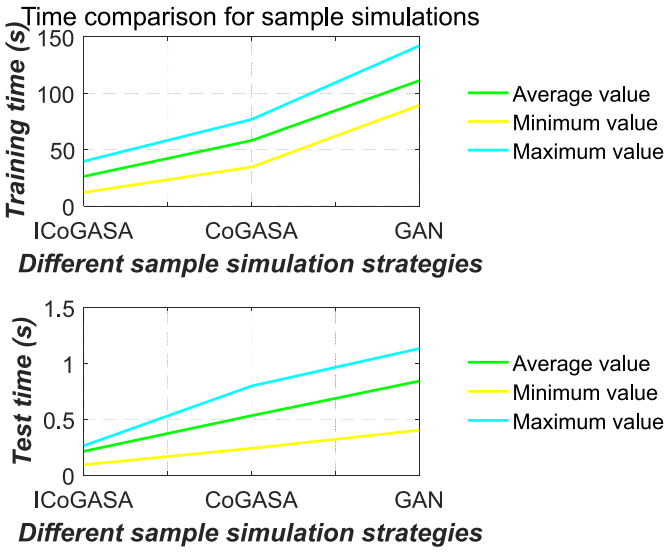

Fig. 33. Time comparison for sample simulations.

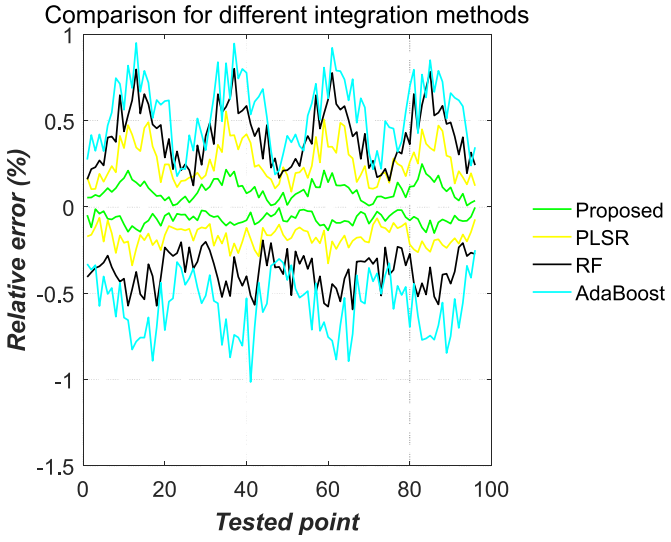

Fig. 34. Comparison for different integration methods.

when all sub-technologies are integrated, the forecasting performance reaches the highest level. The possible reason is that the proposed input feature has two advantages, which play a significant role in improving the forecasting performance. Firstly, smoothing and de-noising by UKF can decrease irregular fluctuations of the original load-series. Secondly, the incremental percentages establish a relationship between current and historical data. Moreover, the advantage of using SOM integration is that after each iteration we can identify the weight getting better or worse according to the learning mechanism of SOM (Eq. (33)). The proposed weight updated strategy (WUS) is helpful to improve the convergence ability and decrease model training time. MA has a stronger ability of learning the historical weights, which have a great influence on the forecasting effectiveness of the test model. ***Case IV*** demonstrates the weights optimization effect of DBN and SOM with different swarm intelligence algorithms. Remarkably, there is no much difference in the scopes of RE, RMSE, MAE and MAPE of optimizing DBN and SOM. But the maxima of the training time and test time of DBN optimization reach 9267.925s and 3.304s, respectively. Remarkably, the training time could not meet the requirement for 1h-ahead load forecasting. The possible reason is that too many weights are required for calculation. In other words, the proposed sub-model (ICoGASA) consists of three GANs. All of the three GANs are composed of two DBNs, which serve as generator and discriminator, respectively. The best number of layers of

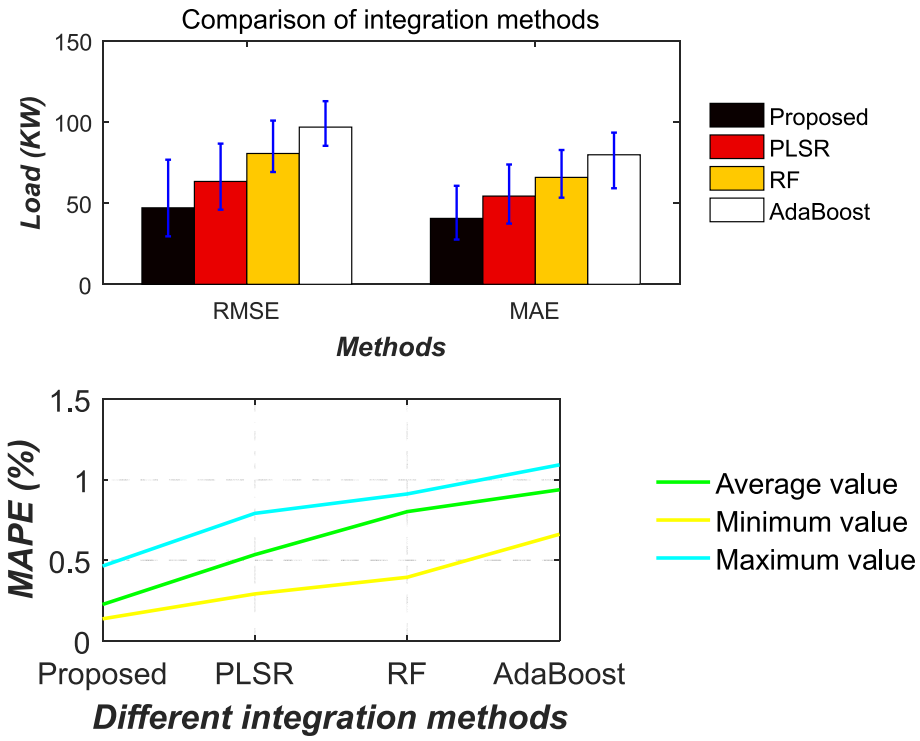

Fig. 35. Comparison of integration methods.

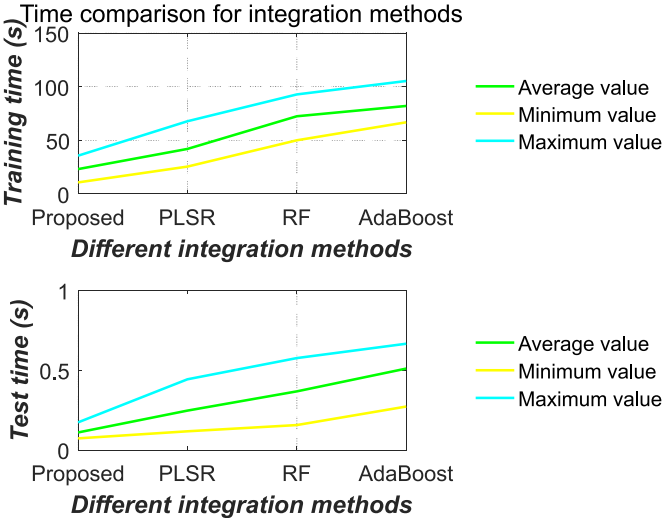

Fig. 36. Comparison of integration methods.

DBN is 3 (tested in ***CASE II***). Therefore, if we want to optimize the weights of DBNs, there will be 18 two-dimensional weight matrices to be calculated. If we optimize the SOM weights, we only need to deal with a two-dimensional matrix, which connects the competition layer and input layer. Therefore, the weight of SOM is optimized, instead of DBN. ***Case V*** proves the forecasting performance with different input features. Notably, the scopes (maximum, average and minimum) of RE, RMSE, MAE and MAPE of the proposed input feature are the most optimal in all different methods. The possible reason is that the proposed input feature has two advantages, which play a significant role in improving the forecasting performance. Firstly, smoothing and de-noising by UKF can decrease irregular fluctuations of the original load-series. Secondly, the incremental percentages establish a relationship between current and historical data. For the same test method, the proposed input feature requires less training and forecasting time in comparison with other test features; while for the same test input feature, the proposed method requires more training and forecasting time in comparison with a deep learning model (DBN) and a classical artificial neural network (BPNN). Moreover, for the same test input feature, DBN takes more training and forecasting time in comparison with BPNN. This may be caused by the complexity of forecasting models. ***Case VI*** demonstrates the forecasting performance of different sample simulation strategies. Remarkably, the scopes (maximum, average and minimum) of RE, RMSE, MAE, MAPE training time and test time of the proposed sample simulation strategy are the most optimal. The possible reason is that the proposed sample simulation strategy (ICoGASA) consisting of three GANs can generate more realistic weather forecast errors and lifestyles of different residents, namely, generate more realistic synthesized samples (RSS). Different from other test methods in this case, the three generators (G) of the GANs are used to generate positive error, negative error and mixed error to weather forecast and residents' lifestyles, respectively. In other words, the generator of one GAN is dedicated to generating only a kind of more realistic synthesized samples with positive error, negative error or mixed error. To a certain extent, this sample simulation strategy can avoid the negative impact of training samples containing different kinds of errors on the training model. It should be noted that a basic CoGASA can also realize the simulation of time series samples. However, using two GANs to simulate three different types of samples can result in large fluctuations in the weights of each GAN, which would spend more processing time and reduce the forecasting performance. ***Case VII*** compares the forecasting effectiveness with different ensemble methods, which are used to integrate each sub-ICoGASA. It is evident that the forecasting effectiveness of the proposed method is obviously better than that of the other test methods tested in this ***Case***. The possible reason is that the output

of each ICoGASA is integrated by SOM. The advantage of using SOM is that we can identify the weight getting better or worse according to the learning mechanism of SOM after each iteration. Through this strategy, the convergence speed and forecasting performance of the predicted model can be improved by using optimization algorithms. In this paper, MA and WUS are proposed to optimize the weight of SOM. MA has a stronger ability of learning the historical weights, which have a great influence on the forecasting performance of the forecasting model. Moreover, the proposed weight updated strategy (WUS) is helpful to improve the convergence ability and decrease model training time by letting different sub-models learn from each other. ***Case VIII*** compares the forecasting performance of different methods in forecasting the electricity demand of weekends (Saturdays and Sundays). It is worth noting that the forecasting effectiveness of the proposed method is apparently superior to that of the other test methods tested in this ***Case***. The possible reason is that the number of training samples of weekends is much smaller than that of normal days. While the proposed sample simulation strategy can solve the shortage of training samples of weekends, which have a great influence on the forecasting effectiveness of the forecasting model. Moreover, the proposed weight updated strategy (WUS) is helpful to improve the convergence ability and decrease model training time. ***Case IX*** compares the forecasting performance of different methods in forecasting the electricity demand of holidays. It is worth noting that the forecasting effectiveness of the proposed method is apparently superior to that of the other test methods tested in this ***Case***. The possible reason is the same as that in ***Case VIII***.

5.2. Limitation

Although the proposed ensemble method indicates better performance to forecast the future electricity demand for regular days and special days (weekends and holidays), there are still the following limitations to be improved in future studies. (1) The proposed ensemble method can only handle a maximum of 5 years of training samples. The ignored training samples may make contributions to enhancing forecasting performance. Therefore, an intelligent mechanism is needed to control the number of simulation samples. (2) In order to reduce a lot of running time, DBN weights are not optimized (tested in ***Case IV***). Integrating each sub-ICoGASA through SOM and optimizing the weight of SOM can compensate for the negative impact on ignoring the optimization of DBN weights. However, optimizing DBN weights makes a great contribution to improving the forecasting performance (tested in ***Case IV***). Therefore, a suitable and time-saving optimization algorithm should be required for optimizing DBN weights.

6. Conclusion

The incremental percentages of the load-series filtered by UKF from the original input variables, are applied as new input features to develop a novel ensemble method to perform hourly electricity demand forecasting. ICoGASA is developed to generate errors more similar to weather forecast and lifestyles of different residents more efficiently. All sub-ICoGASAs are integrated by SOM. MA and WUS are introduced to improve the forecasting performance of the integrated model (ICoGASA-SOM). The error ranges show the superior forecasting performance of the proposed model. Compared with other state-of-the-art ensemble methods (PLSR, RF and AdaBoost, tested in ***Case VII***), the scopes of positive values of RE are reduced by [2.295, 21.957] %, [8.604, 26.103] %, [12.701, 33.628] %, respectively; the scopes of negative values of RE are reduced by [1.159, 18.953] %, [9.375, 30.992] %, [15.805, 38.669] %, respectively; the scopes of RMSE are reduced by [8.295, 16.221] %, [15.507, 28.066] %, [20.494, 36.969] %, respectively; the scopes of MAE are reduced by [5.343, 13.707] %, [9.302, 21.196] %, [15.624, 31.639] %, respectively; the scopes of MAPE are reduced by [7.215, 14.367] %, [9.624, 25.491] %, [15.511, 29.481] %, respectively.

CRediT authorship contribution statement

Guoqiang Zhang: Conceptualization, Methodology, Software, Formal analysis, Data curation, Writing - original draft, Writing - review & editing, Visualization, Supervision, Project administration. **Jifeng Guo:** Validation, Investigation, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.energy.2020.118265.

References

[1] Varma R, Sushil. Bridging the electricity demand and supply gap using dynamic modeling in the Indian context. Energy Pol Sept. 2019;132:515–35. https://doi.org/10.1016/j.enpol.2019.06.014.
[2] Bunn DW. Forecasting loads and prices in competitive power markets. Proc IEEE Feb. 2000;88(2):163–9. https://doi.org/10.1109/5.823996.
[3] Kong WC, Dong ZY, Hill DJ, Luo FJ, Xu Y. Short-term residential load forecasting based on resident behaviour learning. IEEE Trans Power Syst Sept. 2017;33(1):1087–8. https://doi.org/10.1109/TPWRS.2017.2688178.
[4] Zachariadis T, Hadjinicolaou P. The effect of climate change on electricity needs -a case study from Mediterranean Europe. Energy Nov. 2014;76(1):899–910. https://doi.org/10.1016/j.energy.2014.09.001.
[5] Dadkhah M, Rezaee MJ, Chavoshib AZ. Short-term power output forecasting of hourly operation in power plant based on climate factors and effects of wind direction and wind speed. Energy Apr. 2018;148:775–88. https://doi.org/10.1016/j.energy.2018.01.163.
[6] Wang YP, Bielicki JM. Acclimation and the response of hourly electricity loads to meteorological variables. Energy Jan. 2018;142(1):473–85. https://doi.org/10.1016/j.energy.2017.10.037.
[7] Ahmed T, Vu DH, Muttaqi KM, Agalgaonkar AP. Load forecasting under changing climatic conditions for the city of Sydney, Australia. Energy Jan. 2018;142:911–9. https://doi.org/10.1016/j.energy.2017.10.070.
[8] Barman M, Dev Choudhury NB, Sutradhar S. A regional hybrid Goa-SVM model based on similar day approach for short-term load forecasting in Assam, India. Energy Feb. 2018;145:710–20. https://doi.org/10.1016/j.energy.2017.12.156.
[9] Xing YZ, Zhang S, Wen P, Shao LM, Rouyendegh BD. Load prediction in short-term implementing the multivariate quantile regression. Energy 2020;196(Apr):117035. https://doi.org/10.1016/j.energy.2020.117035.
[10] Zhang Y, Zhou Q, Sun CX, Lei SL, Liu YM, Song Y. RBF neural network and ANFIS-based short-term load forecasting approach in real-time price environment. IEEE Trans Power Syst Aug. 2008;23(3):853–8. https://doi.org/10.1109/TPWRS.2008.922249.
[11] Song KB, Baek YS, Hong DH, Jang G. Short-term load forecasting for the holidays using fuzzy linear regression method. IEEE Trans Power Syst Feb. 2005;20(1):96–101. https://doi.org/10.1109/TPWRS.2004.835632.
[12] Hussain A, Rahman M, Memon JA. Forecasting electricity consumption in Pakistan: the way forward. Energy Pol Mar. 2016;90:73–80. https://doi.org/10.1016/j.enpol.2015.11.028.
[13] Pappas SS, Ekonomou L, Karamousantas DC, Chatzarakis GE, Katsikas SK, Liatsis P. Electricity demand loads modeling using AutoRegressive Moving Average (ARMA) models. Energy Sept. 2008;33(9):1353–60. https://doi.org/10.1016/j.energy.2008.05.008.
[14] Hu YH, Li JG, Hong MN, Ren JZ, Lin RJ, Liu Y, Liu MR, Man Y. Short term electric load forecasting model and its verification for process industrial enterprises based on hybrid GA-PSO-BPNN algorithmdA case study of papermaking process. Energy Mar. 2019;170:1215–27. https://doi.org/10.1016/j.energy.2018.12.208.
[15] Azadeh A, Saberi M, Seraj O. An integrated fuzzy regression algorithm for energy consumption estimation with non-stationary data: a case study of Iran. Energy June. 2010;35(6):2351–66. https://doi.org/10.1016/j.energy.2009.12.023.
[16] Fu GY. Deep belief network based ensemble approach for cooling load

forecasting of air-conditioning system. Energy Apr. 2018;148:269–82. https://doi.org/10.1016/j.energy.2018.01.180.
[17] Barman M, Behari Dev Choudhury Nalin. Season specific approach for short-term load forecasting based on hybrid FA-SVM and similarity concept. Energy Mar. 2019;174:886–96. https://doi.org/10.1016/j.energy.2019.03.010.
[18] Li S, Goel L, Wang P. An ensemble approach for short-term load forecasting by extreme learning machine. Appl Energy Feb. 2016;170:22–9. https://doi.org/10.1016/j.apenergy.2016.02.114.
[19] Xiao J, Li YX, Xie L, Liu DH, Huang J. A hybrid model based on selective ensemble for energy consumption forecasting in China. Energy Sept. 2018;159:534–46. https://doi.org/10.1016/j.energy.2018.06.161.
[20] Li C, Tao Y, Ao WG, Yang S, Bai Y. Improving forecasting accuracy of daily enterprise electricity consumption using a random forest based on ensemble empirical mode decomposition. Energy Dec. 2018;165:1220–7. https://doi.org/10.1016/j.energy.2018.10.113.
[21] Fay D, Ringwood JV. On the influence of weather forecast errors in short-term load forecasting models. IEEE Trans Power Syst Aug. 2010;25(3):1751–8. https://doi.org/10.1109/TPWRS.2009.2038704.
[22] Kiasari MA, Moirangthem DS, Lee M. Coupled generative adversarial stacked Auto-encoder: CoGASA. Neural Network Jan. 2018;100:1–9. https://doi.org/10.1016/j.neunet.2018.01.002.
[23] Antipov G, Baccouche M, Dugelay JL. Face aging with conditional generative adversarial networks. In: 2017 IEEE international conference on image processing (ICIP); Feb. 2018. https://doi.org/10.1109/ICIP.2017.8296650.
[24] Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. Science July. 2006;313(5786):504–7. https://doi.org/10.1126/science.1127647.
[25] Li S, Wang P, Goel L. A novel wavelet-based ensemble method for short-term load forecasting with hybrid neural networks and feature selection. IEEE Trans Power Syst May. 2016;31(3):1788–98. https://doi.org/10.1109/TPWRS.2015.2438322.
[26] Wan EA, Van Der Merwe R. The unscented Kalman filter for nonlinear estimation. In: Proceedings of the IEEE 2000 adaptive systems for signal processing, communications, and control symposium (cat. No.00EX373); Aug. 2002. p. 234–63. https://doi.org/10.1109/ASSPCC.2000.882463.
[27] Hu M, Li H, Wu Q, Rose GS. Hardware realization of BSB recall function using memristor crossbar arrays. In: Proceedings of the 49th annual design automation conference; June. 2012. p. 498–503. https://doi.org/10.1145/2228360.2228448.
[28] Milosevic M, McConville KM, Sejdic E, Masani K, Kyan MJ, Popovic MR. Visualization of trunk muscle synergies during sitting perturbations using self-organizing maps (SOM). IEEE (Inst Electr Electron Eng) Trans Biomed Eng Sept. 2012;59(9):2516–23. https://doi.org/10.1109/TBME.2012.2205577.
[29] Strukov DB, Snider GS, Stewart DR, Williams RS. The missing memristor found. Nature May. 2008;453(7191):80–3. https://doi.org/10.1038/nature06932.
[30] Zhang WY, Hong WC, Dong YC, Tsai G, Sung JT, Fan GF. Application of SVR with chaotic GASA algorithm in cyclic electric load forecasting. Energy Aug. 2012;45:850–8. https://doi.org/10.1016/j.energy.2012.07.006.
[31] Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. Science July. 2006;313(5786):504–7. https://doi.org/10.1126/science.1127647.
[32] Hinton GE, Osindero S, WhyeTeh Y. A fast learning algorithm for deep belief nets. Neural Computation July. 2006;18(7):1527–54. https://doi.org/10.1162/neco.2006.18.7.1527.
[33] Nielsen RT. Theory of the backpropagation neural network. In: Ieee. Neural networks for perception; June. 1989. p. 65–93. https://doi.org/10.1016/b978-0-12-741252-8.50010-8.
[34] Talarposhti KM, Jamei MK. A secure image encryption method based on dynamic harmony search (DHS) combined with chaotic map. Optic Laser Eng June. 2016;81:21–34. https://doi.org/10.1016/j.optlaseng.2016.01.006.
[35] Li WQ, Chang L. A combination model with variable weight optimization for short term electrical load forecasting. Energy Dec. 2018;164:575–93. https://doi.org/10.1016/j.energy.2018.09.027.
[36] Guo ZF, Zhou K, Zhang XL, Yang SL. A deep learning model for short-term power load and probability density forecasting. Energy Oct. 2018;160:1186–200. https://doi.org/10.1016/j.energy.2018.07.090.
[37] Wu ZC, Zhao XC, Ma YQ, Zhao XY. A hybrid model based on modified multi-objective cuckoo search algorithm for short-term load forecasting. Appl Energy Mar. 2019;237:896–909. https://doi.org/10.1016/j.apenergy.2019.01.046.
[38] Niu WJ, Feng ZK, Cheng CT, Zhou JZ. Forecasting daily runoff by extreme learning machine based on quantum-behaved particle swarm optimization. J Hydrol Eng Mar. 2018;23(3):1–10. https://doi.org/10.1061/(ASCE)HE.1943-5584.0001625.
[39] Sureshkumar K, Ponnusamy V. Power flow management in micro grid through renewable energy sources using a hybrid modified dragonfly algorithm with bat search algorithm. Energy Aug. 2019;181:1166–78. https://doi.org/10.1016/j.energy.2019.06.029.
[40] Liang Y, Niu DX, Hong WC. Short term load forecasting based on feature extraction and improved general regression neural network model. Energy Oct. 2019;166:653–63. https://doi.org/10.1016/j.energy.2018.10.119.
[41] Bento PMR, Pombo JAN, Calado MRA, Mariano SJPS. Optimization of neural network with wavelet transform and improved data selection using bat algorithm for short-term load forecasting. Neurocomputing May. 2019;358:53–71. https://doi.org/10.1016/j.neucom.2019.05.030.
[42] Niu WJ, Feng ZK, Chen YB, Zhang HR, Cheng CT. Annual streamflow time series prediction using extreme learning machine based on gravitational search algorithm and variational mode decomposition. J Hydrol Eng Feb. 2020;25:1–10. https://doi.org/10.1061/(ASCE)HE.1943-5584.0001902.
[43] Feng ZK, Niu WJ, Tang ZY, Jiang ZQ, Xu Y, Liu Y, Zhang HR. Monthly runoff time series prediction by variational mode decomposition and support vector machine based on quantum-behaved particle swarm optimization. J Hydrol Jan. 2020;583:1–12. https://doi.org/10.1016/j.jhydrol.2020.124627.
[44] Xiao J, Li YX, Xie L, Liu DH, Huang J. A hybrid model based on selective ensemble for energy consumption forecasting in China. Energy Sept. 2018;159:534–46. https://doi.org/10.1016/j.energy.2018.06.161.
[45] Li C, Tao Y, Ao WG, Yang S, Bai Y. Improving forecasting accuracy of daily enterprise electricity consumption using a random forest based on ensemble empirical mode decomposition. Energy Dec. 2018;165:1220–7. https://doi.org/10.1016/j.energy.2018.10.113.
[46] Li Y, Wen Z, Cao YJ, Tan Y, Sidorov D, Panasetsky D. A combined forecasting approach with model self-adjustment for renewable generations and energy loads in smart community. Energy June. 2017;129:216–27. https://doi.org/10.1016/j.energy.2017.04.032.
[47] Singh N, Mohanty SR, Shukla RD. Short term electricity price forecast based on environmentally adapted generalized neuron. Energy Apr. 2017;125:127–39. https://doi.org/10.1016/j.energy.2017.02.094.