Hindawi
Complexity
Volume 2020, Article ID 1209547, 20 pages
https://doi.org/10.1155/2020/1209547

Research Article

Short-Term Load Forecasting with Improved CEEMDAN and GWO-Based Multiple Kernel ELM

Taiyong Li,[1] **Zijie Qian**,[1] **and Ting He**[2]

[1]*School of Economic Information Engineering, Southwestern University of Finance and Economics, Chengdu 611130, China*
[2]*Architectural Design Institute, Nuclear Power Institute of China, Chengdu 610213, China*

Correspondence should be addressed to Taiyong Li; litaiyong@gmail.com

Received 15 October 2019; Revised 7 January 2020; Accepted 31 January 2020; Published 25 February 2020

Academic Editor: Pietro De Lellis

Short-term load forecasting (STLF) is an essential and challenging task for power- or energy-providing companies. Recent research has demonstrated that a framework called "decomposition and ensemble" is very powerful for energy forecasting. To improve the effectiveness of STLF, this paper proposes a novel approach integrating the improved complete ensemble empirical mode decomposition with adaptive noise (ICEEMDAN), grey wolf optimization (GWO), and multiple kernel extreme learning machine (MKELM), namely, ICEEMDAN-GWO-MKELM, for STLF, following this framework. The proposed ICEEMDAN-GWO-MKELM consists of three stages. First, the complex raw load data are decomposed into a couple of relatively simple components by ICEEMDAN. Second, MKELM is used to forecast each decomposed component individually. Specifically, we use GWO to optimize both the weight and the parameters of every single kernel in extreme learning machine to improve the forecasting ability. Finally, the results of all the components are aggregated as the final forecasting result. The extensive experiments reveal that the ICEEMDAN-GWO-MKELM can outperform several state-of-the-art forecasting approaches in terms of some evaluation criteria, showing that the ICEEMDAN-GWO-MKELM is very effective for STLF.

1. Introduction

Accurate load forecasting plays a significant role for the participants in the electricity industry because it can provide a safe and reliable automatic management for a smart grid. According to the length of the period involved, the tasks of load forecasting can be divided into three groups: long-term forecasting, mid-term forecasting, and short-term forecasting. Among them, short-term load forecasting (STLF) has become the hottest research topic in load forecasting because it can not only increase the scheduling efficiency but also reduce the costs of operations [1, 2].

In the last decades, time series-based models, such as random walk, moving average (MA), autoregressive integrated MA (ARIMA), ARIMA with explanatory variable (ARIMAX), and generalized autoregressive conditional heteroskedasticity (GARCH), are widely used in load forecasting [3, 4]. Lee and Ko embedded a lifting scheme into ARIMA models to improve the forecasting ability, and the simulation results verified the effectiveness of STLF [3]. Cui and Peng introduced temperature into ARIMA to propose an improved ARIMAX to deal with the mutation data structures [4]. However, because these time series-based models are usually built on the assumption that the load data are with the characteristics of linearity and stationarity, which are not always met in practical load data, the forecasting accuracy is limited. In fact, recent research has demonstrated that the load data are usually nonlinear and nonstationary. Therefore, it is necessary to use other models instead of time series-based models to improve load forecasting accuracy. Artificial intelligence (AI) models are thought of as having the capacity to capture intrinsic features of complicated signals. Therefore, they have become more and more popular in energy forecasting. Typical AI models include support-vector regression (SVR) and its extension least-squares SVR (LSSVR) [5, 6], artificial neural network (ANN) [7–9], extreme learning machine (ELM) [10], sparse Bayesian learning (SBL) [11, 12], deep learning [13] (stacked

denoising autoencoders (SDAs) [14], deep belief network (DBN) [15], convolutional neural network (CNN) [16], and long short-term memory (LSTM)) [17]), and nature-inspired optimization algorithms [18, 19]. For example, Chen et al. put forward a new SVR model that used the temperature before demand response as additional input variables for STLF [6]. Kulkarni et al. proposed a spiking neural network to forecast short-term load with consideration of weather variables [8]. Yoem and Kwak used an ELM with knowledge representation for STLF, and the experimental results indicated good performance of the approach [10]. Han et al. presented time-dependency CNN and cycle-based LSTM for STLF by mapping the load data as pixels and rearranging them into a 2-D image, and the extensive experiments demonstrated that the proposed models were superior to the compared models in terms of computation complexity [13]. Some other research aims to forecast load data accurately using hybrid models [20, 21].

As far as energy time series forecasting is concerned, recent studies have shown that a framework called "decomposition and ensemble" is able to improve the forecasting performance significantly. The main idea of this framework is to decompose the raw energy data series into several simpler components, then handle each component individually, and finally integrate the result from each component as the final forecasting result. This framework is a typical form of the strategy of "divide and conquer" that is widely used in energy price forecasting [22–24], wind speed forecasting [25, 26], load forecasting [27, 28], biosignal processing [29, 30], fault diagnosis [31], image processing [32–35], and so on. There are many types of decomposition methods that can be applied to decomposing energy time series. Among them, the members of the family of empirical mode decomposition (EMD), i.e., ensemble EMD (EEMD), complete EEMD (CEEMD), CEEMD with adaptive noise (CEEMDAN), and the improved CEEMDAN (ICEEMDAN), have become very popular in signal decomposition [36–39]. With them, the complicated raw energy time series can be decomposed into several components, some high-frequency components and some low-frequency ones, making it easier to forecast each component individually than to forecast the raw energy time series directly. The existing research has revealed that the ICEEMDAN is better than the other decomposition methods in the family of EMD [39]. Theoretically, any regression methods in AI can be applied to forecasting each component. ELM has shown its superpower in the tasks of both classification and regression in recent years [40]. In particular, ELM is able to achieve satisfactory results with time series forecasting [41–43]. When kernel trick is introduced into ELM, it has the ability to improve the performance of STLF further [44, 45]. However, the existing kernel ELM usually uses one kernel or a simple combination of several kernels to construct the kernel matrix as the input of ELM, and the parameters in the single kernel or the weights of the several kernels are optimized by algorithms or specified by users. The number and types of single kernels in such ELM may limit forecasting effectiveness. An ideal way is to construct the kernel matrix using multiple kernels built by different types of kernels, and both the parameters and the weights can be optimized adaptively. Some nature-inspired algorithms, such as ant colony optimization [46, 47], particle swarm optimization (PSO) [48, 49], and grey wolf optimization (GWO) [50, 51], have the potential to optimize the parameters and the weights for the involved kernels simultaneously. Recent research has demonstrated that GWO outperforms some other nature-inspired algorithms in numerical optimization [52, 53].

Inspired by the power of ICEEMDAN in signal decomposition, GWO in numerical optimization, and multiple kernel ELM (MKELM) in regression or prediction, this paper proposes a novel approach that integrates ICEEMDAN, GWO, and MKELM, so-called ICEEMDAN-GWO-MKELM, for STLF. Specifically, the proposed ICEEMDAN-GWO-MKELM is composed of three stages. First, the raw load series is decomposed into a couple of components, some showing high-frequency characteristics while others showing low-frequency ones. Second, an independent MKELM model is built on each component. To improve the representation ability of the kernel matrix, GWO is applied to optimizing both the parameters and the weight of each base kernel simultaneously. Finally, the predicted results of all the individual components are simply accumulated as the final forecasting result.

The novelty of this paper lies in the following two aspects: (1) it is the first time that the combination of ICEEMDAN, GWO, and ELM is used for energy forecasting, especially for STLF. (2) A novel multiple kernel learning (MKL) framework optimizing both the parameters and the weight of every single kernel with GWO is proposed. We conduct extensive experiments to compare the proposed ICEEMDAN-GWO-MKELM with many state-of-the-art approaches, and the results demonstrate that the ICEEMDAN-GWO-MKELM is able to outperform the compared approaches in most cases.

The remaining part of this paper is structured as follows. Section 2 briefly describes ICEEMDAN, GWO, ELM, and MKL. The proposed ICEEMDAN-GWO-MKELM is methodologically formulated in detail in Section 3. To evaluate the proposed method, we conduct extensive experiments, and the results are analyzed and discussed in Section 4. Finally, the paper is concluded in Section 5.

2. Methods

2.1. The Improved Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (ICEEMDAN). The ICEEMDAN is a signal decomposition algorithm proposed by Colominas et al. based on CEEMDAN in 2014 [39]. It is a new member of the family of EMD.

The EMD is a signal decomposition technique that can decompose a data series into a set of components with different frequencies, including intrinsic mode functions (IMFs) and a residue [36]. The signal must satisfy the following two characteristics: (1) in all data, the number of extreme values (maximum and minimum) and zero crossings are equal or different at most by one; (2) the local mean must be zero at any time.

The EMD algorithm is described as follows:

Step 1: connect the local maxima as well as the minima of the raw data by two cubic spines to form one upper envelope and one lower envelope, respectively

Step 2: average the upper envelope and the lower envelope, and subtract the mean value from the original signal to obtain a new sequence

Step 3: examine whether the new sequence from Step 2 meets the two characteristics listed above; if not, take the new sequence as raw data and go to Step 1; otherwise, take the new sequence as one IMF and subtract it from the original sequence to obtain one residue (the local mean)

Step 4: examine whether the residue from Step 3 is a monotonic function or its value is less than a predetermined threshold; if not, take the residue as raw data and go to Step 1; otherwise, end the decomposition and output all IMFs and one residue

However, EMD is prone to mode-mixing problems during the decomposition process [37]. The main form of the problem is that an IMF contains signals with large scale differences. The main reason for this problem is as follows: in the case of an abnormal event, there will be an uneven distribution of extreme points in the signal, which will affect the shape of the envelope. The IMF thus contains the intrinsic mode of the original signal and the intrinsic mode of the adjacent time scale brought by the anomalous event.

In order to overcome the shortcomings of EMD, Wu and Huang put forward the ensemble EMD (EEMD) method, which is an integrated, straightforward, and adaptive decomposition method for nonlinear and nonstationary time series [37]. Similarly, EEMD decomposes the original data into a couple of IMFs and one residue and each decomposed component has the same length as the original data. In EEMD, a certain amount of different Gaussian white noises are firstly added to the original signal, as shown in the following equation:

$$x^{(i)} = x + \omega^{(i)}, \quad i = 1, 2, \ldots, N, \tag{1}$$

where x denotes the original series, $w^{(i)}$ is the i-th Gaussian white noise to be added, and $x^{(i)}$ is the corresponding processed signal.

Then, each processed signal is subjected to EMD processing to get IMFs and one residue, respectively. At last, the IMFs and residues from different processed signals are integrated and averaged correspondingly to obtain the final decomposition results:

$$c_k = \frac{1}{N}\sum_{i=1}^{N} c_k^{(i)}, \quad k = 1, 2, \ldots, K, \tag{2}$$

where c_k is the k-th final decomposition result, $c_k^{(i)}$ is the k-th decomposition result from $x^{(i)}$, K is the final number of IMFs after averaging, determined by the length M of the original sequence of the time series, $K = \lfloor \log_2 M \rfloor - 1$, and N is the realization number of decomposition.

The method causes the extreme point distribution state in the original signal to be changed so that the processed signals can be continuously distributed on different time scales and reduce the probability of generating mode-mixing problems.

EEMD can reduce the mode-mixing problem to a certain extent. However, because of the added white noise, after a finite number of averaging calculations, the error is not completely eliminated, which will affect the accuracy of the reconstruction sequence and the accuracy of the prediction. Although the error can be reduced by the increase of the average number of integrations, the amount of calculation and the calculation time are greatly increased. Based on this phenomenon, Torres et al. designed CEEMDAN [38].

In CEEMDAN, Gaussian white noise is added to each decomposition layer to obtain one IMF and corresponding residual signal. Given the operator $E_k(\cdot)$ that produces the k-th IMF by EMD, the CEEMDAN can be simply described as follows: (1) like the EEMD, the CEEMDAN decomposes the original data and gets the first IMF c_1 and residue r_1; (2) the following k-th IMF ($k \geq 2$) and residue can be obtained by

$$\begin{aligned} c_k &= \frac{1}{N}\sum_{i=1}^{N} E_1\left(r_{k-1} + p_{k-1}E_{k-1}\left(\omega^{(i)}\right)\right), \\ r_k &= r_{k-1} - c_k, \end{aligned} \tag{3}$$

where $w^{(i)}$ is the i-th Gaussian white noise to be added, $E_k[w^{(i)}]$ is the k-th IMF by decomposing $w^{(i)}$ using EMD, and p_k is the signal-to-noise ratio of the additional noise and the original signal.

The algorithm terminates when the residual r_k satisfies the iterative termination condition, and finally CEEMDAN can get several IMFs and compute one residue as follows:

$$R = x - \sum_{k=1}^{K} r_k, \tag{4}$$

where R is the residue obtained from CEEMDAN, x is the original signal, and K is the number of IMFs.

However, CEEMDAN still suffers from two main problems: (1) residual noise contained in models and (2) spurious mode problem. An improved CEEMDAN (ICEEMDAN) was proposed to solve these problems. Let $M(\cdot)$ be the operator for generating local mean and $E_k(\cdot)$ be the operator that produces the k-th IMF by EMD [39]. The specific decomposition process is as follows:

Step 1: add $E_1[w^{(i)}]$ to the original signal x, $x^{(i)} = x + p_0 E_1(w^{(i)})(i = 1, 2, \ldots, N)$, where $w^{(i)}$ is the i-th white noise to be added, p_0 is the signal-to-noise ratio, and N is the amount of white noise added.

Step 2: calculate the local mean of $x^{(i)}$ by EMD and get the first residue $r_1 = (1/N)\sum_{i=1}^{N} M(x^{(i)})$; then, obtain the first IMF $c_1 = x - r_1$.

Step 3: recursively use formulas $c_k = r_{k-1} - r_k (k \geq 2)$ to get the k-th IMF c_k, where

$$r_k = \frac{1}{N}\sum_{i=1}^{N} M\left(r_{k-1} + p_{k-1}E_k\left(w^{(i)}\right)\right). \quad (5)$$

The results show that the residual noise problem in the IMF is greatly reduced and also the mean value problem caused by the different numbers of IMFs generated by EEMD is also solved.

2.2. Grey Wolf Optimization (GWO). Grey wolf optimization (GWO) algorithm is a new type of swarm intelligence optimization algorithm proposed by Mirjalili et al. in 2014 [50]. It is derived from the simulation of the predation behavior of the grey wolf population and achieves optimization through wolf group's tracking, encirclement, pursuit, and attack. The advantages of the GWO algorithm include simple principle, fewer parameters to be adjusted, easy implementation, and strong global search ability.

The wolves are divided into four levels: α represents the dominant wolf in the group and it is at the first level, β represents the subordinate wolf at the second level who helps α make decision, δ represents the wolf following the instructions of α and β, and ω represents the wolves at the lowest level. In GWO, the pursuit behavior is performed by α, β, and δ, and ω follows the first three to track and coffer the prey and finally completes the predation task. Suppose the number of grey wolves is N and the dimension of search space is d, the position of the i-th grey wolf in the d-th dimension space can be expressed as $x_i = (x_{i1}, x_{i2}, \ldots, x_{id})$. According to the fitness function of a specific optimization problem, the optimal individual is recorded as α and the corresponding individuals ranked second and third are recorded as β and δ with the remaining individuals recorded as ω. The position of the prey means the global optimal solution of the optimization problem. Three definitions to be used in the GWO are given as follows.

Definition 1. Distance between grey wolf and prey:

$$\vec{D} = \left|\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)\right|, \quad (6)$$

where $\vec{X}_p(t)$ indicates the position of the t-th generation prey, $\vec{X}(t)$ indicates the position of the t-th generation grey wolf individual, and C is a swing factor determined by

$$\vec{C} = 2 \cdot \vec{r}_1, \quad (7)$$

where $\vec{r}_1$ is a random vector between 0 and 1.

Definition 2. Encircling prey.

In nature, grey wolves always hunt preys by encircling them. Also, the mathematical model is given as follows:

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D}, \quad (8)$$

where $\vec{A}$ is the convergence factor determined by

$$\vec{A} = 2\vec{a} \cdot \vec{r}_2 - \vec{a}, \quad (9)$$

where $\vec{r}_2$ is a random vector between 0 and 1 and $\vec{a}$ decreases linearly from 2 to 0 as the iteration number increases.

Definition 3. Hunting and capturing prey.

An issue with Definition 2 is that the position of the prey (the optimal solution) in practical optimization problems is unknown. So, in order to simulate the behavior of hunting prey, three types of wolves, namely, α, β, and δ, are defined based on their distance to the prey, and they have the clearest understanding of the position of the prey. The closer the distance, the more the wolves understand the position of the prey. We can use their positions to find the prey and lead the rest ω wolves to update their own locations. The mathematical expression of hunting prey can be explained as follows:

$$\vec{D}_\alpha = \left|\vec{C}_1 \cdot \vec{X}_\alpha(t) - \vec{X}(t)\right|, \quad (10)$$

$$\vec{D}_\beta = \left|\vec{C}_1 \cdot \vec{X}_\beta(t) - \vec{X}(t)\right|, \quad (11)$$

$$\vec{D}_\delta = \left|\vec{C}_1 \cdot \vec{X}_\delta(t) - \vec{X}(t)\right|, \quad (12)$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot \vec{D}_\alpha, \quad (13)$$

$$\vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot \vec{D}_\beta, \quad (14)$$

$$\vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot \vec{D}_\delta, \quad (15)$$

$$\vec{X}(t+1) = \frac{\left(\vec{X}_1 + \vec{X}_2 + \vec{X}_3\right)}{3}. \quad (16)$$

During the hunting, it first calculates the distance between individuals within the group and α, β, and δ from equations (10)–(15) and comprehensively determines the direction in which the individual moves toward the prey by using equation (16).

Finally, wolves (searching agents) finish hunting when they capture the prey and the algorithm terminates.

The main idea of GWO can be described on the basis of the following definitions: randomly generate a population of grey wolves in the problem space; evaluate each individual wolf based on its distance to the prey according to Definition 1, and nominate α, β, and δ wolves and then update the location of each wolf by Definition 3; repeat the operations of evaluation and update the positions of wolves until the wolves capture the prey [50].

2.3. Extreme Learning Machine (ELM). The ELM is a new single-hidden-layer feedforward neural network (SLFN) [54]. This machine randomly initializes the linking weights and the bias, and only the number of hidden-layer nodes needs to be determined by users. ELM can get unique optimal output weights by only one-step calculation and thus gets high training speed. ELM has been proved to perform well in both regression and classification problems.

Given a dataset $(\mathbf{x}_i, \mathbf{t}_i)$ of size N, where $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T \in \mathbb{R}^n$ and $\mathbf{t}_i = [t_{i1}, t_{i2}, \ldots, t_{im}]^T \in \mathbb{R}^m$, an ELM regression model with $\widetilde{N}$ hidden-layer nodes and activation functions G can be expressed as

$$\sum_{i=1}^{\widetilde{N}} \boldsymbol{\beta}_i G\left(\mathbf{a}_i \cdot \mathbf{x}_j + b_i\right) = \mathbf{o}_j, \quad j = 1, \ldots, N, \tag{17}$$

where $\mathbf{a}_i = [a_{i1}, a_{i2}, \ldots, a_{in}]^T$ is a vector of weights of the i-th hidden-layer node and the input node, $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{im}]^T$ is another vector of weights between the i-th hidden-layer node and the output node, b_i is the bias of the i-th hidden-layer node, $\mathbf{a}_i \cdot \mathbf{x}_j$ is the inner product between $\mathbf{a}_i$ and $\mathbf{x}_j$, $\widetilde{N}$ is the number of hidden-layer nodes, and $\mathbf{o}_j$ is the output when input is $\mathbf{x}_j$. When the model fits the N samples exactly, we can get

$$\sum_{j=1}^{\widetilde{N}} \left\| \mathbf{o}_j - \mathbf{t}_j \right\| = 0, \tag{18}$$

$$\sum_{i=1}^{\widetilde{N}} \boldsymbol{\beta}_i G\left(\mathbf{a}_i \cdot \mathbf{x}_j + b_i\right) = t_j, \quad j = 1, \ldots, N. \tag{19}$$

Equation (19) can be written as

$$\begin{aligned} \mathbf{H}\boldsymbol{\beta} &= \mathbf{T}, \\ \mathbf{H}\left(\mathbf{a}_1, \ldots, \mathbf{a}_{\widetilde{N}}, b_1, \ldots, b_{\widetilde{N}}, \mathbf{x}_1, \ldots, \mathbf{x}_N\right) &= \begin{bmatrix} G\left(\mathbf{a}_1 \cdot \mathbf{x}_1 + b_1\right) & \cdots & G\left(\mathbf{a}_{\widetilde{N}} \cdot \mathbf{x}_1 + b_{\widetilde{N}}\right) \\ \vdots & \ddots & \vdots \\ G\left(\mathbf{a}_1 \cdot \mathbf{x}_N + b_1\right) & \cdots & G\left(\mathbf{a}_{\widetilde{N}} \cdot \mathbf{x}_N + b_{\widetilde{N}}\right) \end{bmatrix}_{N \times \widetilde{N}}, \\ \boldsymbol{\beta} &= \begin{bmatrix} \boldsymbol{\beta}_1^T \\ \vdots \\ \boldsymbol{\beta}_{\widetilde{N}}^T \end{bmatrix}_{\widetilde{N} \times m}, \\ \mathbf{T} &= \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix}_{N \times m}, \end{aligned} \tag{20}$$

where $\mathbf{H}$ is a hidden-layer output matrix, and the output weight can be obtained by solving the following linear system:

$$\left\| \mathbf{H}\widehat{\beta} - \mathbf{T} \right\| = \min_{\boldsymbol{\beta}} \left\| \mathbf{H}\boldsymbol{\beta} - \mathbf{T} \right\|, \tag{21}$$

and the solution is

$$\widehat{\beta} = \mathbf{H}^{\dagger}\mathbf{T}, \tag{22}$$

where $\mathbf{H}^{\dagger}$ is Moore–Penrose pseudoinverse of hidden-layer output matrix $\mathbf{H}$.

It is proved that equation (22) is the unique minimum norm least-squares solution of equation (21), which can be shown as [54]

$$\begin{aligned} &\left\| \mathbf{H}\widehat{\boldsymbol{\beta}} - \mathbf{T} \right\| = \left\| \mathbf{H}\mathbf{H}^{\dagger}\mathbf{T} - \mathbf{T} \right\| = \min_{\boldsymbol{\beta}} \left\| \mathbf{H}\boldsymbol{\beta} - \mathbf{T} \right\|, \\ &\left\| \widehat{\boldsymbol{\beta}} \right\| = \left\| \mathbf{H}^{\dagger}\mathbf{T} \right\| \le \left\| \boldsymbol{\beta} \right\|, \\ &\forall \boldsymbol{\beta} \in \left\{ \boldsymbol{\beta} : \left\| \mathbf{H}\boldsymbol{\beta} - \mathbf{T} \right\| \le \left\| \mathbf{H}z - \mathbf{T} \right\|, \forall z \in \mathbf{R}^{\widetilde{N} \times N} \right\}. \end{aligned} \tag{23}$$

When the numbers of the hidden-layer nodes and the samples are identical, the network can approximate the samples very well, but in practice, the number of hidden nodes is usually less than the number of training samples, so the data samples may have multicollinearity problems. When solving the Moore–Penrose pseudoinverse $\mathbf{H}^{\dagger} = \mathbf{H}^{\dagger}(\mathbf{H}\mathbf{H}^T)^{-1}$, the existence of multicollinearity may make $\mathbf{H}\mathbf{H}^T$ singular. Each time the model is modeled by ELM, the obtained matrix $\mathbf{H}^{\dagger}$ is different and the hidden output weight $\widehat{\boldsymbol{\beta}}$ of the hidden layer is also inconsistent. These reasons finally cause the output of ELM prone to random fluctuations, and the model's stability and generalization ability are not ideal.

2.4. Multiple Kernel Learning (MKL). To further enhance the generalization ability and stability of ELM, Huang et al. introduced the kernel function into ELM by comparing the principles of ELM and support-vector machine (SVM) and proposed kernel extreme learning machine (KELM) [40].

The objective of ELM is to minimize not only the sum of training error but also the norm of weights, which can be presented as

$$\begin{aligned} &\text{minimize}: \quad L_{\text{ELM}} = \frac{1}{2}\left\| \boldsymbol{\beta} \right\|^2 + \frac{C}{2}\sum_{i=1}^{N} \left\| \boldsymbol{\xi}_i \right\|^2 \\ &\text{s.t.}: \quad \mathbf{h}\left(\mathbf{x}_i\right)\boldsymbol{\beta} = \mathbf{t}_i^T - \boldsymbol{\xi}_i^T, \quad i = 1, 2, \ldots, N, \end{aligned} \tag{24}$$

where $\boldsymbol{\xi}_i = [\xi_{i1}, \xi_{i2}, \ldots, \xi_{im}]^T$ is the error of the m output nodes when the input is $\mathbf{x}_i$, C is the penalty coefficient used to weigh the ratio between structural risk and empirical risk, $\boldsymbol{\beta}$ is the output weight vector between the hidden layer and the output layer, $\mathbf{t}_i$ is the true value with respect to $\mathbf{x}_i$, N is the

amount of samples, and $\mathbf{h}(\mathbf{x}_i)$ is the output vector of the hidden layer by input $\mathbf{x}_i$.

According to the KKT theorem, equation (24) is equal to the dual optimization problem as follows:

$$L_{\mathrm{DELM}} = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{C}{2}\sum_{i=1}^{N}\|\boldsymbol{\xi}_i\|^2 - \sum_{i=1}^{N}\sum_{j=1}^{m}\alpha_{ij}\left(\mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta}_j - t_{ij} + \xi_{ij}\right), \tag{25}$$

where α_{ij} is the Lagrange multiplier, and KKT conditions of equation (25) are

$$\frac{\partial L_{\mathrm{DELM}}}{\partial \boldsymbol{\beta}_j} = 0 \longrightarrow \boldsymbol{\beta}_j = \sum_{i=1}^{N}\alpha_{ij}\mathbf{h}(\mathbf{x}_i)^T \longrightarrow \boldsymbol{\beta} = \mathbf{H}^T\boldsymbol{\alpha}, \tag{26}$$

$$\frac{\partial L_{\mathrm{DELM}}}{\partial \boldsymbol{\xi}_i} = 0 \longrightarrow \boldsymbol{\alpha}_i = C\boldsymbol{\xi}_i, \quad i = 1, 2, \ldots, N, \tag{27}$$

$$\frac{\partial L_{\mathrm{DELM}}}{\partial \boldsymbol{\alpha}_i} = 0 \longrightarrow \mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta} - \mathbf{t}_i^T + \boldsymbol{\xi}_i^T = 0, \quad i = 1, 2, \ldots, N. \tag{28}$$

Equation (29) can be derived by substituting equations (26) and (27) into equation (28):

$$\left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T\right)\boldsymbol{\alpha} = \mathbf{T}, \tag{29}$$

where $\mathbf{I}$ is an identity matrix, $\mathbf{H}\mathbf{H}^T$ is generated by mapping the input samples through a kernel function, $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_1, \ldots, \mathbf{t}_N]^T$, $\boldsymbol{\alpha}_i = [\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{im}]^T$, and $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_N]^T$.

From equations (26) and (29), we can get

$$\boldsymbol{\beta} = \mathbf{H}^T\left(\frac{\mathbf{I}}{\mathbf{C}} + \mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{T}. \tag{30}$$

The output function is

$$\mathbf{f}(\mathbf{x}) = \mathbf{h}(\mathbf{x})\mathbf{H}^T\left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{T}. \tag{31}$$

We can define the kernel matrix as

$$\begin{cases} \boldsymbol{\Omega}_{\mathrm{KELM}} = \mathbf{H}\mathbf{H}^T, \\ \boldsymbol{\Omega}_{i,j} = \mathbf{h}(\mathbf{x}_i)\cdot\mathbf{h}(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j), \end{cases} \tag{32}$$

where $\mathbf{h}(\mathbf{x})$ is the hidden-layer node output function; K is a kernel function, and its forms include RBF kernel function, linear kernel function, and polynomial kernel function.

The kernel matrix $\boldsymbol{\Omega}_{\mathrm{KELM}}$ replaces the random matrix $\mathbf{H}\mathbf{H}^T$ in equation (31) and uses the kernel function to map all input of the n-dimensional space to a high-dimensional space. After the kernel parameter setting is completed, the mapped value of the kernel matrix $\boldsymbol{\Omega}_{\mathrm{KELM}}$ is a fixed value.

Now, the output function can be rewritten as [40]

$$\mathbf{f}(\mathbf{x}) = \mathbf{h}(\mathbf{x})\mathbf{H}^T\left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{T} = \begin{bmatrix} K(\mathbf{x}, \mathbf{x}_1) \\ \cdots \\ K(\mathbf{x}, \mathbf{x}_N) \end{bmatrix}^T\left(\frac{\mathbf{I}}{C} + \boldsymbol{\Omega}_{\mathrm{KELM}}\right)^{-1}\mathbf{T}. \tag{33}$$

In the kernel-based ELM algorithm, as long as a function satisfies Mercer's condition, it can be used as the kernel function, such as radial basis kernel function (RBF) and polynomial function (polynomial). Each kernel function usually has its own application fields, and a single kernel function often cannot maximize the representation ability. Therefore, this paper presents a new kernel ELM framework that combines different types of kernels called multiple kernel ELM (MKELM). MKELM replaces the single kernel function in KELM with weighted combination of different kernel functions. The popular single kernels used in machine learning are as follows.

RBF kernel function:

$$K_{\mathrm{rbf}}(x, x_i) = \exp\left(-\frac{\|x - x_i\|^2}{a}\right). \tag{34}$$

Polynomial kernel function:

$$K_{\mathrm{poly}}(x, x_i) = (xx_i + a)^q. \tag{35}$$

Linear kernel function:

$$K_{\mathrm{lin}}(x, x_i) = xx_i. \tag{36}$$

Wave kernel function:

$$K_{\mathrm{wav}}(x, x_i) = \prod_{i=1}^{d}\left[\cos\left(\frac{a(x - x_i)}{b}\right)\exp\left(-\frac{\|x - x_i\|^2}{c}\right)\right]. \tag{37}$$

So, the combined kernel function can be formulated as

$$K_{\mathrm{comb}}(x, x_i) = \sum_{i=1}^{n_1}a_iK_{\mathrm{rbf}}(x, x_i) + \sum_{i=1}^{n_2}b_iK_{\mathrm{poly}}(x, x_i)k(x, x_i) + \sum_{i=1}^{n_3}c_iK_{\mathrm{lin}}(x, x_i) + \sum_{i=1}^{n_4}d_iK_{\mathrm{wav}}(x, x_i), \tag{38}$$

where n_1, n_2, n_3, and n_4 represent the number of four kernel functions in order, a_i represents the weight of corresponding single RBF kernel function, and the same for b_i, c_i, and d_i. The coefficients satisfy $\sum_{i=1}^{n_1}a_i + \sum_{i=1}^{n_2}b_i + \sum_{i=1}^{n_3}c_i + \sum_{i=1}^{n_4}d_i = 1$. There are 1, 2, 0, and 3 parameters to be optimized for a single RBF kernel, a single polynomial kernel, a single linear kernel, and a single wave kernel, respectively. So, in the combined kernel, all the number of parameters for optimization are the sum of the parameters for single kernels $n_1 + 2n_2 + 3n_4$ and the weights of all the kernels $n_1 + n_2 + n_3 + n_4$, which is $2n_1 + 3n_2 + n_3 + 4n_4$.

3. The Proposed ICEEMDAN-GWO-MKELM Approach

3.1. GWO-Based Multiple Kernel Extreme Learning Machine. This paper uses GWO to optimize the weights and parameters for MKELM, namely, GWO-MKELM, which can be described as follows:

Step 1: define the fitness function according to root mean square error (RMSE):

$$f(p_i) = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_i - \phi(x_i, p_i))^2}, \quad (39)$$

where N denotes the number of training samples, y_i denotes the actual value of input x_i, $p(i)$ is the vector of the parameters and weights in the proposed model which need to be optimized, and $\phi(x_i, p_i)$ is the prediction value of the model with x_i and p_i.

Step 2: set parameters for running GWO, including maximum number of iterations, population size, and upper and lower boundaries for parameters of four different types of kernel functions and the regularization parameter. Randomly initialize the position for each wolf between the upper and lower boundaries, and set the fitness value of the α, β, and δ wolves to be infinite. Set the times of iteration $t = 1$. Initialize a, $\vec{A}$, and $\vec{C}$.

Step 3: for each wolf, if the fitness of current wolf is less than that of the α wolf, replace the α wolf with it; if the fitness is between β wolf and α wolf, replace the β wolf with it; if the fitness value is between δ wolf and β wolf, replace the δ wolf with it.

Step 4: update a, $\vec{A}$, and $\vec{C}$ by equations (7) and (9). Update all the search agents by equation (16). Update $t = t + 1$.

Step 5: judge whether t is larger than the maximum number of iterations; if not, go to Step 3; if so, break the iteration and output the position of α wolf p_α as the optimized parameters and weights for MKELM.

3.2. The Proposed ICEEMDAN-GWO-MKELM Approach. Because the electricity load time series is hard to predict for its nonlinear and nonstationary features, this paper adopts a forecasting framework called "decomposition and ensemble." The framework decomposes the original time series into several relatively simple components and predicts these components separately, and the results are aggregated as the final predicted value. It is worth noting that the components obtained by the decomposition can effectively preserve the characteristics of the original time series at different levels and can be directly processed by relatively simple methods, so it is possible to forecast load from the decomposed components.

The proposed ICEEMDAN-GWO-MKELM consists of three stages, namely, decomposition, individual forecasting, and ensemble, following the very popular framework of "decomposition and ensemble." The details of the approach are illustrated in Figure 1.

In the first stage, i.e., decomposition, the scheme uses ICEEMDAN to decompose the raw load data into a set of components ($n - 1$ IMFs and one residue). The decomposed components will show simpler characteristics than the raw data, making it easier to forecast the fluctuation of the components than to forecast the raw data directly. After that, an individual MKELM model is built on each component. To improve the forecasting accuracy, GWO is applied to optimizing both the parameters and weights adaptively. In the last stage, the forecasting results of all the components are simply summated as the final result.

From the figure, it can also be seen that the proposed ICEEMDAN-GWO-MKELM is a typical strategy of "divide and conquer." That is to say, the tough and challenging task of STLF is now divided into some subtasks of forecasting the decomposed components individually. Because the decomposed components show relatively simple features compared with the raw load data, the forecasting accuracy with such components might be significantly improved.

4. Experimental Results

4.1. Data Description. The data of electricity load demand can be accessed from Australian Energy Market Operator (AEMO) [55]. Especially, we use the hal-fhour demand data of 2014 from New South Wales (NSW), Tasmania (TAS), Queensland (QLD), Victoria (VIC), and South Australia (SA) to test and verify the proposed method. For each region, to reflect the difference of season, four months including three solar months of 31 days (January, July, and October) and one lunar month of 30 days (April) were chosen. The statistics of the used datasets are shown in Table 1. We use the first 80% observations as training data and the rest as testing data for each month. The size of datasets from a solar month is 1488, so the numbers of training and testing samples are 1190 and 298, respectively. Likewise, the size for a lunar month is 1440, so the numbers of the two parts are 1152 and 288.

We use the previous 48 data to predict the next data, so the horizon is 1 and the lag is 48, which can be formulated as

$$\hat{x}_{t+h} = f\left(x_{t-(l-1)}, x_{t-(l-2)}, \ldots, x_{t-1}, x_t\right), \quad (40)$$

where $\hat{x}_{t+h}$ is the prediction of horizon h at time t, x_t is the actual data at time t, and l is the lag.

4.2. Evaluation Criteria. We use several widely used indices to evaluate the proposed approach: the mean absolute error (MAE), the mean absolute percent error (MAPE), and the root mean squared error (RMSE), as defined in equations (41)–(43), respectively:

$$\text{MAE} = \frac{1}{N}\sum_{n=1}^{N}\left|x_t - \hat{x}_t\right|, \quad (41)$$

$$\text{MAPE} = \frac{1}{N}\sum_{n=1}^{N}\left|\frac{x_t - \hat{x}_t}{x_t}\right|, \quad (42)$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(x_t - \hat{x}_t)^2}, \quad (43)$$

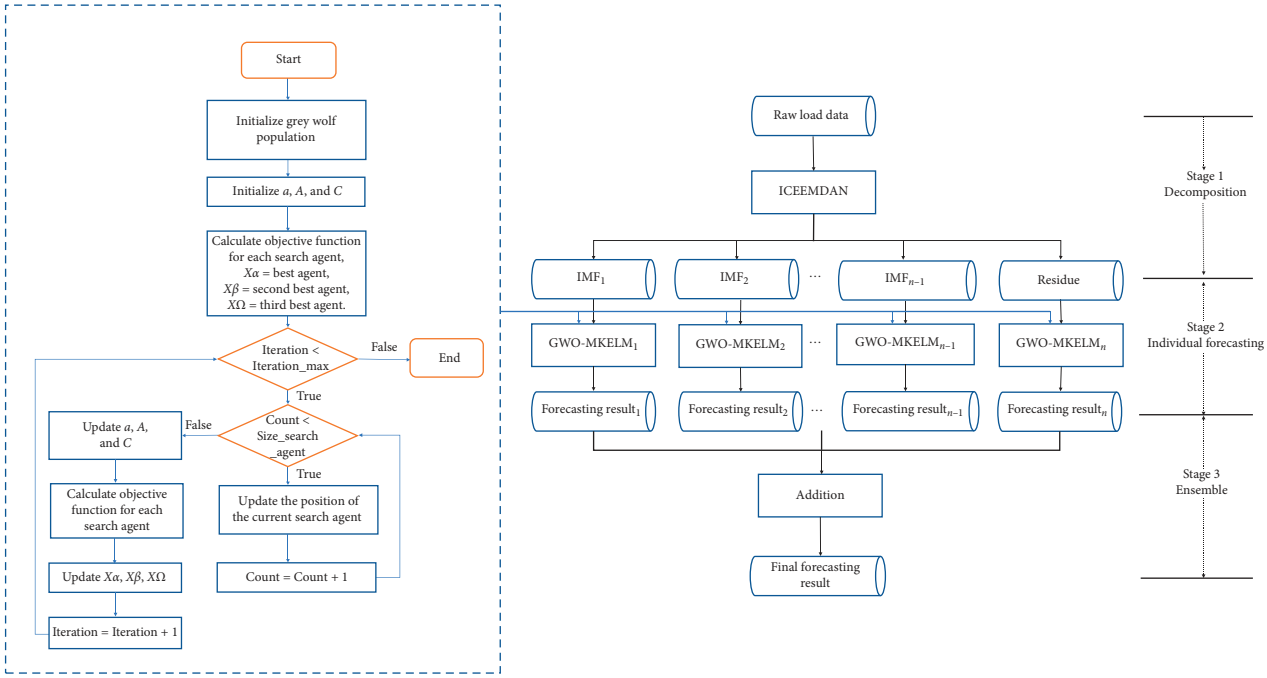

Figure 1: The flowchart of the proposed ICEEMDAN-GWO-MKELM.

Table 1: Statistics of the used datasets.

Dataset	Month	Length	Min	Median	Mean	Max	Std
NSW	Jan	1488	5484.19	8020.36	8072.02	11845.77	1399.20
	Apr	1440	5507.79	7490.73	7473.87	9563.73	999.90
	Jul	1488	6157.55	8754.69	8668.21	11282.96	1162.95
	Oct	1488	5489.52	7588.19	7433.56	9930.88	892.49
TAS	Jan	1488	770.96	1038.53	1037.21	1282.24	90.31
	Apr	1440	844.41	1110.49	1111.48	1437.70	118.15
	Jul	1488	902.13	1245.24	1244.60	1624.12	148.61
	Oct	1488	809.92	1049.90	1054.96	1407.43	102.11
QLD	Jan	1488	4458.42	6146.61	6067.62	8364.63	872.03
	Apr	1440	4279.21	5699.76	5596.11	6925.55	688.97
	Jul	1488	4073.00	5613.99	5678.53	7288.08	706.08
	Oct	1488	4312.69	5550.25	5520.67	7376.08	703.10
VIC	Jan	1488	3576.27	5262.46	5657.32	10240.22	1451.69
	Apr	1440	3616.14	5152.89	5177.05	7697.72	772.45
	Jul	1488	3923.27	5831.98	5762.46	7573.03	828.18
	Oct	1488	3470.86	4984.60	4951.61	6517.63	662.85
SA	Jan	1488	844.01	1430.33	1578.44	3245.92	502.79
	Apr	1440	868.00	1314.88	1313.63	2103.43	214.83
	Jul	1488	922.86	1504.01	1509.43	2344.03	285.94
	Oct	1488	801.79	1284.20	1299.10	2162.02	204.07

where N is the number of testing data and x_t and $\hat{x}_t$ are the true value and the prediction value at time t. Obviously, the lower the three criteria, the better the predict method.

Meanwhile, the Nemenyi test is also conducted to show that the proposed method is significantly superior to the other methods [56]. The basic idea of the Nemenyi test is to determine whether the algorithm is similar to each other according to the average performance of the algorithm on different datasets. First, it sorts different algorithms from good to bad on each dataset according to the evaluation criteria so that the ordinal value is the position of the algorithm on the dataset, and 1 is the best. If the position is the same, the ordinal value is the same, and it can be calculated by $k + ((n-1)/2)$, where k is the position and n is the number of algorithms in the same position. Then, it averages all the ordinal values of each algorithm to get the average ordinal value. Under a certain level of significance, the critical difference of the average ordinal value difference can be calculated by

$$\mathrm{CD} = q_\alpha \sqrt{\frac{k(k+1)}{6N}}, \tag{44}$$

where α is the level of significance, k is the number of algorithms, and N is the number of datasets.

If the average ordinal value difference between the two algorithms exceeds CD, then the assumption that the two algorithms have the same performance is rejected at the level of significance of $1 - \alpha$. Here, we take a significant level of 0.05 and divide all prediction methods into two parts: one is without decomposition and the other one is with decomposition. The two parts of the algorithms take RMSE, MAPE, and MAE as the criteria for measuring performance to conduct the Nemenyi test, respectively, so a total of 6 tests were performed finally.

4.3. Experimental Settings. First of all, some state-of-the-art prediction models will be introduced as comparative approaches to verify the validity of the proposed method ICEEMDAN-GWO-MKELM. According to whether the original data are decomposed and how they are decomposed, benchmark methods can be divided into three parts. The first part is methods without decomposition: SVR, ANN, random forest (RF), deep belief network (DBN) [15], ELM, kernel extreme learning machine with single RBF kernel (KELM), and GWO-MKELM. The second part is methods with EMD decomposition: EMD-SVR, EMD-ANN, EMD-RF, EMD-DBN, EMD-ELM, EMD-KELM, and EMD-GWO-MKELM. The third part is methods with ICEEMDAN decomposition: ICEEMDAN-SVR, ICEEMDAN-ANN, ICEEMDAN-RF, ICEEMDAN-DBN, ICEEMDAN-ELM, ICEEMDAN-KELM, and ICEEMDAN-GWO-MKELM. Comparison between each part can reflect the impact of decomposition on the prediction effect of the model, and the comparison within each group can reflect the impact of different prediction methods on the effectiveness.

The parameters for the proposed approach and the compare methods are as follows. For ANN with back propagation, we set 10 and 5,000 as the number of neurons in the hidden layer and the iteration epochs, respectively. We set 100 as the number of trees in RF. For DBN, the batch size is 6 and the epoch number is 100. For ELM, the number of hidden neurons is 20 and we use the sigmoid function as the activation function. For KELM and MKELM, the regularization coefficient is 1. For the ICEEMDAN, the standard deviation of the white noise and the number of ensembles are 0.2 and 100, respectively. The decomposition results of the electricity load demand of NSW in January of 2014 by ICEEMDAN are shown in Figure 2. Also, for GWO, the parameters are shown in Table 2.

We apply the min-max normalization to preprocessing the data before being fed into the models, as formulated by using the following equation:

$$x_{\text{norm}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}, \tag{45}$$

where x_{norm} is the normalized data, x is the original data, and $x_{\min}$ and $x_{\max}$ are corresponding minimal and maximal of each data dimension. With this normalization, every data are mapped into a value in the range of $[0, 1]$.

We conduct all the experiments by MATLAB R2016b on 64-bit Windows 10, and the main hardware includes a 3.6 GHz CPU as well as 32 GB RAM.

4.4. Results and Analysis. To validate the proposed ICEEMDAN-GWO-MKELM, we conduct two groups of experiments: single models, which mean performing STLF with the raw load data, and ensemble models, which mean performing STLF with the decomposed components.

4.4.1. Single Models. The results of MAPE, RMSE, and MAE by single models on the different datasets are listed in Table 3, where we use bold and italics to mark the best and the worst results, respectively.

From Table 3, we can find that ELM and KELM obtain the worst results in 43 and 17 out of 60 cases, respectively. It reveals that the kernel trick can improve the forecasting results for ELM to some extent. We can also see that none of the other models obtain the worst results, indicating that the nonkernel ELM and ELM with a single RBF kernel (KELM) underperform the other models. It may owe to both the forecasting ability of ELM and the representation ability of a single RBF which are still poor for STLF. However, when we use multiple kernel ELM optimized with GWO (GWO-MKELM) for STLF, it achieves the best results in 29 out of 60 cases, which indicates that GWO-MKELM outperforms other single models significantly. The possible reason is that GWO-MKELM can optimize the weights and parameters of the multiple kernels for ELM to improve the representation ability of the kernels. SVR and DBN achieve the best results 22 and 10 times, ranked second and third, respectively. ANN and RF obtain similar results in most cases.

We further use the Nemenyi test with RMSE, MAPE, and MAE to compare the models, as shown in Figures 3–5, respectively. Regarding Figure 3, the proposed GWO-MKELM is ranked first, with a value of 2, followed by DBN and SVR with 2.2 and 2.4, respectively. The ELM is ranked last, with a value of 6.6. Therefore, the results show that GWO-MKELM is the best single model in terms of the Nemenyi test of RMSE. When we look at the Nemenyi tests of MAPE and MAE, we can see that both tests are almost the same. Specifically, SVR is ranked first with a value of 2, followed by GWO-MKELM and DBN with values of 2.1 and 2.15 in both tests, respectively. Once again, the ELM is ranked last in both tests.

All the experimental results with single models show that none of the single models can always outperform others, although GWO-MKELM achieves the best results the most times. Therefore, to improve the accuracy of STLF, a possible way is to adopt the "decomposition and ensemble" framework.

4.4.2. Ensemble Models. To demonstrate the performance of ICEEMDAN, we utilize EMD as a compared decomposition method. We still applied the seven forecasting approaches in single models to conduct individual forecasting for both decomposition methods, so there are a total of 14 ensemble

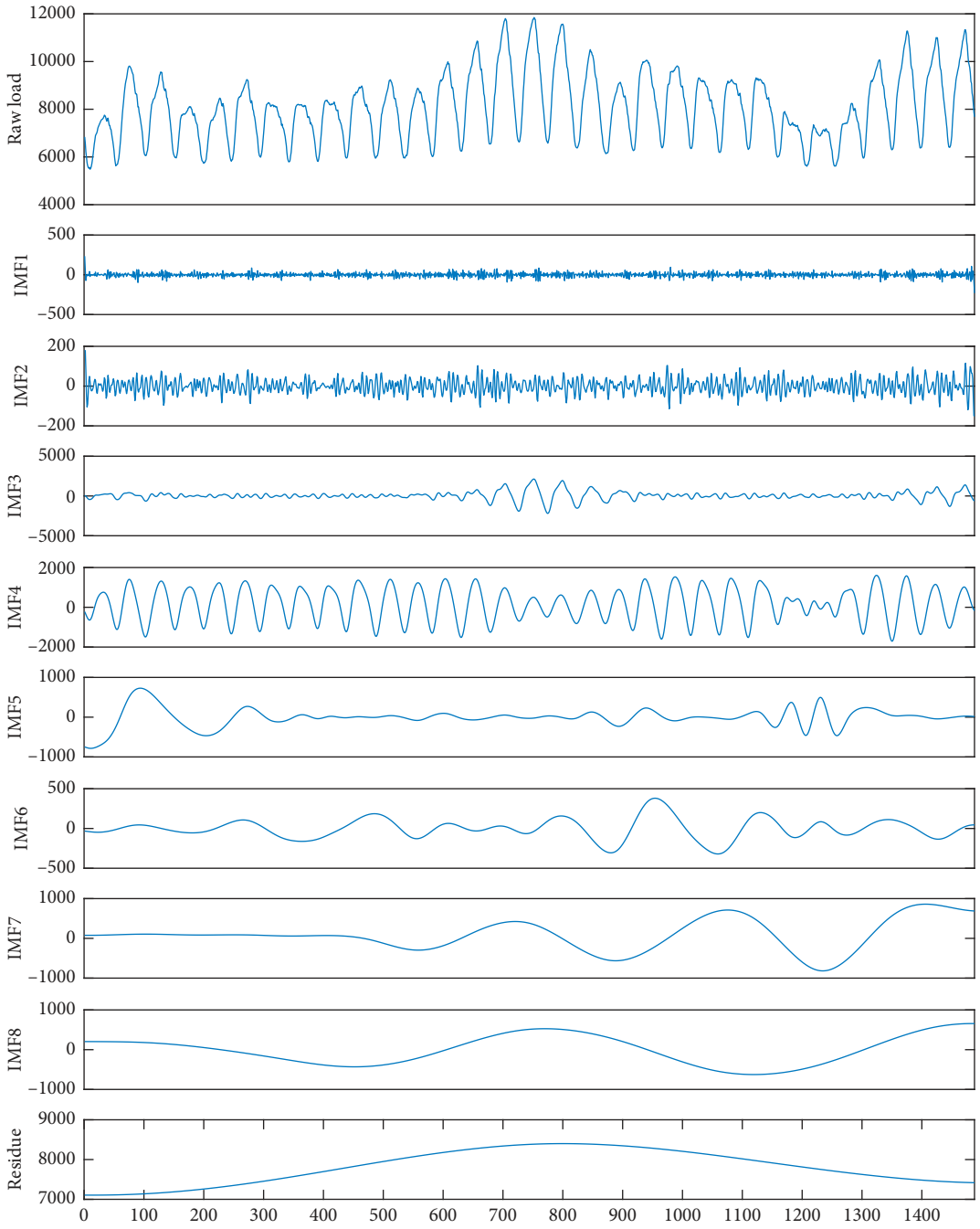

Figure 2: An example of load data and their decomposed components by ICEEMDAN.

forecasting models. The results of ensemble models are listed in Table 4.

From the table, it can be seen that the ensemble models with EMD and ICEEMDAN obtain the worst results, i.e., 51 and 9 out of 60 times, respectively, while all the best results of the 60 cases are achieved by the models associated with ICEEMDAN. Among the ensemble models with EMD, EMD-ELM and EMD-KELM obtain the worst results 23 and 28 times, respectively. The possible reason is that nonkernel ELM and ELM with a single RBF kernel lack good forecasting ability, so it is necessary to use multiple kernels to improve such ability. If we further investigate each individual forecasting method, we can find that the results with ICEEMDAN usually outperform those with EMD, confirming that the components decomposed by ICEEMDAN are more powerful than those by EMD for STLF.

The best results of all cases are associated with ICEEMDAN, among which GWO-MKELM achieves the best results in 57 out of 60 cases while the remaining three best results are achieved by SVR. Regarding the individual forecasting models, DBN and RF obtain neither the best nor the worst results and their performance is at an intermediate level. ANN obtains the worst results 3 times. Note that although KELM with a single RBF cannot improve the

Table 2: Parameters for GWO.

Description	Symbol	Range/Value
Population size	P	50
Maximal iterations	T	20
RBF kernel number	n_1	40
Polynomial kernel number	n_2	1
Linear kernel number	n_3	1
Wave kernel number	n_4	1
Particle dimension	D	88
Kernel weight	a, b, c, d	$[0, 1]$
Parameter in K_{rbf}	p_1	$[2^{-4}, 2^{12}]$
Constant in K_{poly}	p_2	$[0, 10]$
Exponent in K_{poly}	p_3	$[1, 4]$
Parameters in K_{wav}	p_4, p_5, p_6	$[1, 10]$

Table 3: Results of single models.

Dataset	Month	Metric	ANN	DBN	ELM	KELM	RF	SVR	GWO-MKELM
NSW	JAN	MAE	154.6764	64.4121	182.5331	*182.6954*	133.8832	70.5883	**64.1313**
		MAPE	0.0195	0.0084	*0.0231*	0.0230	0.0179	0.0088	**0.0082**
		RMSE	187.7779	81.9792	*224.1725*	220.6236	169.3942	87.3505	**79.0744**
	APR	MAE	137.1602	61.7357	*191.6156*	126.8218	85.8328	72.4596	**52.0814**
		MAPE	0.0188	0.0083	*0.0262*	0.0170	0.0115	0.0097	**0.0071**
		RMSE	178.4644	78.0035	*239.2961*	165.7563	119.9159	94.7145	**68.1083**
	JUL	MAE	171.4667	97.4067	*248.7272*	214.9098	162.6317	83.8706	**82.8769**
		MAPE	0.0206	0.0115	*0.0305*	0.0255	0.0203	0.0103	**0.0101**
		RMSE	216.4860	134.9842	*317.5506*	286.6340	216.4821	110.6867	**109.9804**
	OCT	MAE	115.5595	75.2309	*201.0932*	184.3484	131.0651	61.1197	**55.3547**
		MAPE	0.0149	0.0096	*0.0261*	0.0232	0.0163	0.0080	**0.0072**
		RMSE	153.4577	107.2169	250.8892	*252.1916*	206.4041	77.8372	**72.6525**
TAS	JAN	MAE	18.3073	14.1093	*23.6164*	20.4768	16.4694	**13.3090**	17.9694
		MAPE	0.0173	0.0133	*0.0221*	0.0190	0.0156	**0.0124**	0.0171
		RMSE	24.4586	18.7513	*31.4773*	26.8768	21.3632	**18.3332**	24.5034
	APR	MAE	24.3973	20.9139	38.0553	*46.1968*	23.8966	**15.8858**	27.3674
		MAPE	0.0206	0.0178	0.0319	*0.0372*	0.0196	**0.0135**	0.0230
		RMSE	31.6557	27.7336	47.5372	*59.9119*	33.1396	**22.0547**	35.3487
	JUL	MAE	26.3055	21.9166	*39.0484*	34.6745	26.3530	**19.0973**	22.1679
		MAPE	0.0217	0.0183	*0.0320*	0.0281	0.0225	**0.0156**	0.0185
		RMSE	32.6556	27.7648	*49.5600*	45.3531	34.5904	**25.2990**	28.5182
	OCT	MAE	21.4702	16.4177	*27.8931*	20.7424	19.1465	**15.3503**	21.0459
		MAPE	0.0203	0.0153	*0.0261*	0.0194	0.0179	**0.0142**	0.0196
		RMSE	27.5290	22.4137	*35.8661*	28.5251	25.1375	**21.0999**	28.1134
QLD	JAN	MAE	99.5483	55.2755	*121.3485*	107.8171	87.4767	52.7251	**43.4113**
		MAPE	0.0177	0.0100	*0.0221*	0.0195	0.0162	0.0094	**0.0076**
		RMSE	87.4767	68.5140	*152.3092*	141.7001	116.6662	65.7924	**56.2672**
	APR	MAE	73.4552	39.0620	*118.0295*	79.0556	65.4980	51.2859	**38.8761**
		MAPE	0.0133	**0.0071**	*0.0220*	0.0142	0.0120	0.0094	**0.0071**
		RMSE	97.5407	50.8372	*147.9867*	115.9868	93.4801	64.4567	**49.7598**
	JUL	MAE	101.6366	52.7022	*133.5809*	103.8697	70.2883	53.6579	**44.8970**
		MAPE	0.0183	0.0092	*0.0238*	0.0181	0.0126	0.0096	**0.0079**
		RMSE	134.0305	67.8439	*175.2375*	142.6298	100.6117	69.3967	**57.3941**
	OCT	MAE	171.5361	97.9332	102.6511	*374.9998*	180.8467	**47.4042**	88.3029
		MAPE	0.0289	0.0158	0.0172	*0.0584*	0.0278	**0.0079**	0.0150
		RMSE	211.1436	127.1105	133.9282	*470.6390*	260.8355	**62.8133**	120.4800

Table 3: Continued.

Dataset	Month	Metric	ANN	DBN	ELM	KELM	RF	SVR	GWO-MKELM
VIC	JAN	MAE	173.2485	89.5098	223.1598	*250.0737*	162.1459	**75.5767**	89.5373
		MAPE	0.0298	0.0154	0.0369	*0.0390*	0.0261	**0.0131**	0.0147
		RMSE	228.9064	118.8186	284.7641	*362.5156*	214.6654	**100.3084**	127.9768
	APR	MAE	114.6478	63.5387	*156.0478*	131.2278	100.3078	63.7165	**53.1056**
		MAPE	0.0226	0.0125	*0.0307*	0.0252	0.0198	0.0126	**0.0106**
		RMSE	142.7216	85.6728	*196.3560*	171.3722	139.6513	89.2660	**71.1012**
	JUL	MAE	142.8986	75.2688	*208.8160*	151.0656	151.2386	**64.0587**	65.3535
		MAPE	0.0271	0.0146	*0.0388*	0.0279	0.0299	**0.0122**	0.0123
		RMSE	178.0544	99.4085	*285.9721*	200.6208	198.9308	88.7265	**83.1036**
	OCT	MAE	91.9358	51.8505	*137.4309*	110.5691	63.8513	62.1709	**45.7309**
		MAPE	0.0189	0.0105	*0.0279*	0.0229	0.0130	0.0125	**0.0092**
		RMSE	117.3665	67.4336	*174.8739*	150.3012	87.4146	87.5464	**59.0015**
SA	JAN	MAE	66.1579	53.5970	76.1489	*94.2302*	55.8282	**33.1162**	38.1895
		MAPE	0.0389	0.0311	0.0436	*0.0505*	0.0318	**0.0195**	0.0219
		RMSE	83.6863	77.3360	94.3229	*125.3239*	71.4316	53.5724	**53.3662**
	APR	MAE	40.1770	**25.0730**	*55.5536*	43.7650	33.8890	29.4414	29.6150
		MAPE	0.0312	**0.0196**	*0.0437*	0.0337	0.0259	0.0223	0.0230
		RMSE	58.4774	**33.1353**	*73.5395*	57.3714	47.3599	59.0514	38.8852
	JUL	MAE	52.9771	**28.4243**	*66.5991*	57.4608	49.0961	34.8949	30.5323
		MAPE	0.0370	**0.0196**	*0.0464*	0.0394	0.0344	0.0238	0.0215
		RMSE	71.5559	**36.2251**	*86.2606*	73.8378	64.8605	62.2525	40.5406
	OCT	MAE	44.4122	**22.1417**	49.4206	*54.4296*	32.4702	28.6027	23.1714
		MAPE	0.0334	**0.0166**	0.0376	*0.0417*	0.0244	0.0211	0.0174
		RMSE	59.6492	**29.3357**	64.9982	*68.7959*	44.5576	51.2978	30.7468

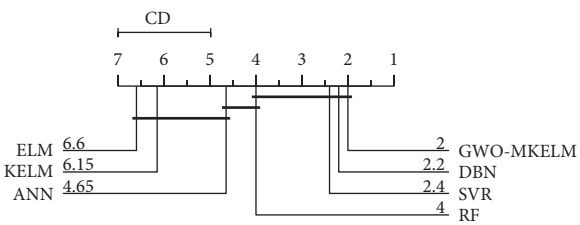

Figure 3: The Nemenyi test results of RMSE for STLF with single models.

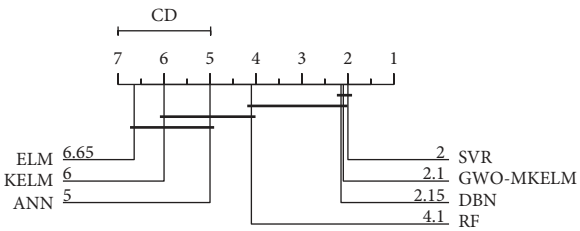

Figure 4: The Nemenyi test results of MAPE for STLF with single models.

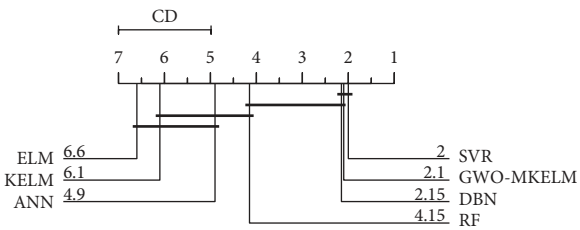

Figure 5: The Nemenyi test results of MAE for STLF with single models.

forecasting ability of ELM, the proposed GWO-MKELM can significantly outperform both ELM and KELM, showing that the proposed multiple kernel learning framework and the GWO-MKELM are very effective for STLF.

When we look at the best result of each evaluation criterion, we can find that the proposed ICEEMDAN-GWO-MKELM is obviously advantageous over the compared methods in most cases. We take NSW data in JAN as an example, the ICEEMDAN-GWO-MKELM achieves the best MAPE 0.0037, which is far below the second best result 0.0076 achieved by the EMD-SVR and the ICEEMDAN-SVR. Likewise, the best MAE by the ICEEMDAN-GWO-MKELM is 30.2777, and it is less than half of the second best MAE by the ICEEMDAN-SVR. The results of MAE by some other methods are an order of magnitude larger than the MAE by the ICEEMDAN-GWO-MKELM. The results on RMSE show a similar trend.

Again, we use the Nemenyi test to compare all the involved ensemble models. The results regarding RMSE, MAPE, and MAE can be found in Figures 6–8, respectively. In all the figures, the proposed ICEEMDAN-GWO-MKELM is ranked first with a fixed value of 1.05, followed by ICEEMDAN-SVR. In addition, EMD-KELM is still ranked last in all cases, showing that it has the worst ability for STLF when compared with the other ensemble models.

From the above analysis, we can see that (1) ensemble models outperform single models for STLF; (2) regarding the decomposition in ensemble models, ICEEMDAN is superior to EMD; (3) GWO is very useful for optimizing weights and parameters in MKELM simultaneously; and (4)

Table 4: Results of ensemble models.

Market	Month	Metric	Ensemble model													
			EMD							ICEEMDAN						
			ANN	DBN	ELM	KELM	RF	SVR	GWO-MKELM	ANN	DBN	ELM	KELM	RF	SVR	GWO-MKELM
NSW	JAN	MAE	532.1479	64.0923	251.3688	*899.4359*	293.7202	63.2038	203.9169	387.4338	66.4144	99.7673	278.0649	159.3435	62.5048	**30.2777**
		MAPE	0.0640	0.0078	0.0301	*0.0940*	0.0371	0.0076	0.0229	0.0516	0.0080	0.0130	0.0331	0.0207	0.0076	**0.0037**
		RMSE	714.4395	83.5211	359.3834	*1554.8546*	444.2917	78.1247	366.5190	478.8873	86.9761	127.4645	364.4434	197.7704	78.0538	**44.7637**
	APR	MAE	92.0776	70.0957	135.8802	*192.3355*	130.5212	65.4841	58.6348	73.9317	63.7078	90.4835	188.3402	120.9785	52.9669	**20.8687**
		MAPE	0.0127	0.0093	0.0183	*0.0256*	0.0177	0.0087	0.0077	0.0099	0.0085	0.0124	0.0245	0.0159	0.0072	**0.0029**
		RMSE	116.2282	88.3786	169.8043	*262.4362*	199.6548	83.6755	76.6728	91.0963	82.3428	117.4523	248.4441	158.0092	69.0897	**27.8428**
	JUL	MAE	288.5418	114.3288	*449.8745*	368.8029	272.5387	119.3253	97.9137	129.1449	84.9656	194.5966	143.6068	103.3341	61.7211	**26.4664**
		MAPE	0.0361	0.0136	*0.0559*	0.0446	0.0337	0.0145	0.0120	0.0161	0.0102	0.0240	0.0173	0.0128	0.0077	**0.0032**
		RMSE	383.9581	154.5431	*550.5212*	507.1267	327.1870	152.8752	125.4308	170.3315	118.0816	252.7243	181.7253	135.6029	76.6042	**34.0012**
	OCT	MAE	165.2688	89.3787	223.0494	*278.7917*	128.7734	58.6089	46.1860	69.9928	88.8514	85.9756	164.7142	97.2712	47.6529	**26.2783**
		MAPE	0.0210	0.0112	0.0290	*0.0352*	0.0168	0.0076	0.0061	0.0093	0.0112	0.0116	0.0205	0.0125	0.0061	**0.0035**
		RMSE	221.1114	130.1587	291.6073	*426.7584*	181.7774	73.0523	61.0572	86.4628	123.4277	106.6205	224.3288	141.5616	58.9118	**33.5002**
TAS	JAN	MAE	12.0319	14.8445	15.1883	*17.5295*	12.8384	8.5656	7.7325	9.3085	15.7932	10.9207	16.5583	12.0946	**4.1673**	5.4762
		MAPE	0.0113	0.0141	0.0143	*0.0163*	0.0122	0.0080	0.0072	0.0089	0.0151	0.0102	0.0153	0.0113	**0.0039**	0.0051
		RMSE	15.9108	18.8116	20.0542	*25.7161*	17.7013	11.1029	10.0454	11.9391	20.4012	14.5234	24.6161	16.9451	**5.6004**	8.1104
	APR	MAE	16.5227	22.5023	21.1901	17.3010	14.2613	11.0523	9.0950	13.8248	21.0588	18.4397	*24.3540*	13.7477	7.4024	**6.2314**
		MAPE	0.0139	0.0191	0.0180	0.0144	0.0119	0.0093	0.0077	0.0116	0.0180	0.0161	*0.0200*	0.0115	0.0062	**0.0053**
		RMSE	21.3402	28.7879	27.9040	22.8104	19.9158	15.1266	12.6033	18.6720	26.5731	24.0637	*31.4381*	18.6492	10.3011	**9.1335**
	JUL	MAE	20.5425	19.7755	*24.3450*	24.0461	21.1641	12.6718	10.9381	13.7216	22.9221	22.2249	22.4648	22.4684	8.5574	**7.8087**
		MAPE	0.0170	0.0165	*0.0204*	0.0196	0.0179	0.0105	0.0091	0.0112	0.0189	0.0187	0.0184	0.0192	0.0072	**0.0065**
		RMSE	25.7853	25.2186	30.3598	30.4113	26.8088	15.6586	13.9382	18.5313	29.5136	28.3545	*32.0117*	27.9190	10.8462	**10.5979**
	OCT	MAE	17.0689	16.0147	*23.8229*	19.6132	16.0841	10.8386	11.5317	9.9318	16.1262	20.2521	12.7372	12.4946	5.6426	**4.7009**
		MAPE	0.0161	0.0149	*0.0225*	0.0182	0.0151	0.0101	0.0108	0.0093	0.0151	0.0191	0.0119	0.0118	0.0052	**0.0044**
		RMSE	22.1784	21.7199	*30.5709*	26.1356	21.7475	14.8730	15.9903	13.2728	21.7826	25.2093	16.2744	15.7333	7.1928	**7.0755**
QLD	JAN	MAE	97.0179	59.8173	98.9209	*148.3699*	77.6649	51.0916	43.1395	49.9364	62.9715	61.0149	97.3860	87.7026	44.6055	**16.5420**
		MAPE	0.0169	0.0108	0.0178	*0.0270*	0.0137	0.0093	0.0077	0.0089	0.0112	0.0108	0.0172	0.0157	0.0082	**0.0030**
		RMSE	137.2054	73.9378	127.4975	*196.3960*	109.2073	62.5900	64.1784	63.5004	78.2413	85.2092	125.7110	102.3245	59.2733	**21.6150**
	APR	MAE	83.6535	43.7720	105.1157	*178.2371*	88.4275	45.9305	37.5891	39.8414	45.7536	53.7099	83.7228	58.7209	45.9206	**15.0487**
		MAPE	0.0153	0.0080	0.0193	*0.0313*	0.0164	0.0082	0.0067	0.0072	0.0083	0.0098	0.0149	0.0106	0.0088	**0.0028**
		RMSE	113.9442	55.0538	140.0627	*269.4808*	124.6052	59.9737	49.6899	51.5596	58.4156	69.0369	117.8969	77.9772	54.1398	**19.8375**
	JUL	MAE	99.9916	46.3983	*138.4338*	116.4045	102.8028	72.9487	65.2888	58.1534	45.6668	97.6248	61.9103	63.6282	33.8343	**19.3205**
		MAPE	0.0180	0.0082	*0.0253*	0.0212	0.0187	0.0131	0.0116	0.0108	0.0080	0.0177	0.0111	0.0114	0.0061	**0.0035**
		RMSE	132.8591	58.9545	*184.3857*	150.9853	132.4211	102.1267	98.0093	77.0966	57.9448	128.6205	80.1042	82.7943	45.4939	**25.2896**
	OCT	MAE	166.5655	96.6660	240.8430	*529.4592*	216.7251	55.1873	58.6462	99.2401	91.2599	154.4574	407.5372	178.1564	56.4071	**26.3191**
		MAPE	0.0291	0.0158	0.0414	*0.0829*	0.0357	0.0093	0.0099	0.0164	0.0148	0.0262	0.0644	0.0289	0.0091	**0.0045**
		RMSE	199.2270	130.1168	294.1400	*640.2580*	259.1100	68.8884	78.6975	129.1497	121.8645	218.4007	514.2186	212.3671	68.6110	**35.6525**

Table 4: Continued.

Market	Month	Metric	Ensemble model													
			EMD							ICEEMDAN						
			ANN	DBN	ELM	KELM	RF	SVR	GWO-MKELM	ANN	DBN	ELM	KELM	RF	SVR	GWO-MKELM
VIC	JAN	MAE	293.9192	127.3526	135.7483	252.7043	188.9913	72.1121	65.0419	*299.9859*	115.2471	98.8877	235.3073	128.2519	55.3838	**29.9182**
		MAPE	0.0479	0.0212	0.0236	0.0410	0.0314	0.0126	0.0116	*0.0514*	0.0207	0.0166	0.0378	0.0211	0.0096	**0.0049**
		RMSE	373.5188	165.6043	181.4578	324.7539	238.5033	95.6134	86.0050	*380.7922*	149.7854	126.7161	302.2185	170.5561	70.8372	**39.8705**
	APR	MAE	114.6927	71.9436	*165.6342*	147.0590	106.0477	95.1020	85.4564	62.3934	59.7394	59.7588	84.2827	67.6939	43.6030	**18.5937**
		MAPE	0.0229	0.0143	*0.0334*	0.0295	0.0212	0.0189	0.0170	0.0126	0.0119	0.0120	0.0167	0.0135	0.0086	**0.0038**
		RMSE	148.7746	95.7464	*206.8913*	180.4697	138.6782	126.3458	123.4141	78.8045	80.9129	73.8266	108.1936	88.9409	52.9345	**25.5223**
	JUL	MAE	144.9525	96.1076	*226.5710*	201.4074	118.8257	82.4412	77.9388	69.0872	89.4013	108.4300	97.5361	102.9053	48.5709	**16.4306**
		MAPE	0.0269	0.0186	*0.0437*	0.0394	0.0229	0.0162	0.0150	0.0131	0.0174	0.0205	0.0184	0.0196	0.0092	**0.0031**
		RMSE	184.6855	119.5930	*294.7883*	253.8873	157.9688	112.9990	114.7735	84.9291	113.7514	138.9305	117.4428	127.0492	60.0419	**21.4260**
	OCT	MAE	149.6622	84.1755	146.1449	*160.3790*	115.2219	64.0668	64.3320	49.3394	50.0670	71.4507	67.3644	41.9993	35.2742	**17.8874**
		MAPE	0.0304	0.0176	0.0296	*0.0326*	0.0241	0.0132	0.0132	0.0100	0.0100	0.0146	0.0138	0.0086	0.0071	**0.0036**
		RMSE	189.3913	111.1213	184.0786	*211.3786*	146.8726	82.1106	86.5842	61.6734	65.3774	94.1915	86.1123	54.9688	42.2119	**23.3598**
SA	JAN	MAE	58.0563	58.7852	65.1545	84.8856	62.1518	26.3860	31.0183	41.2435	59.9898	39.3240	*89.4063*	60.3000	18.1465	**9.7534**
		MAPE	0.0359	0.0347	0.0381	*0.0515*	0.0376	0.0164	0.0191	0.0254	0.0268	0.0228	0.0460	0.0341	0.0103	**0.0058**
		RMSE	77.4661	85.2414	87.2945	103.2162	78.6570	37.7597	42.1278	50.0789	45.1721	49.3220	*123.3603*	74.1613	24.0220	**12.4997**
	APR	MAE	61.2293	30.0954	63.3756	*81.1355*	57.5993	51.9504	51.8011	19.8194	26.7758	28.8929	39.1257	30.8584	14.5780	**8.3278**
		MAPE	0.0495	0.0232	0.0511	*0.0627*	0.0455	0.0429	0.0426	0.0155	0.0207	0.0229	0.0299	0.0244	0.0111	**0.0065**
		RMSE	87.3231	37.9836	83.2580	*108.3714*	78.8301	79.4451	81.5557	26.1784	34.6466	37.3583	52.6758	38.5325	18.8276	**11.1451**
	JUL	MAE	64.8026	32.2184	*99.8736*	70.9772	60.0825	53.5036	51.2457	28.2712	32.9979	44.8921	34.5995	36.7410	22.2268	**10.0439**
		MAPE	0.0459	0.0224	*0.0712*	0.0495	0.0440	0.0381	0.0371	0.0204	0.0228	0.0318	0.0238	0.0250	0.0157	**0.0070**
		RMSE	95.9762	41.0355	*145.3466*	94.5524	87.5885	83.8094	91.7469	35.2667	41.7827	57.0407	44.1829	52.7963	27.9484	**13.5639**
	OCT	MAE	32.2637	25.9550	*46.7875*	40.1246	33.9269	26.4662	24.7309	19.2839	24.8155	28.4471	27.4123	17.8795	14.0603	**8.0023**
		MAPE	0.0244	0.0194	*0.0352*	0.0304	0.0260	0.0201	0.0187	0.0147	0.0184	0.0217	0.0209	0.0135	0.0107	**0.0061**
		RMSE	42.4686	32.7050	*60.2892*	53.6864	43.4875	36.6935	34.1090	24.6109	31.8924	37.8393	35.2461	22.9129	17.0981	**10.0953**

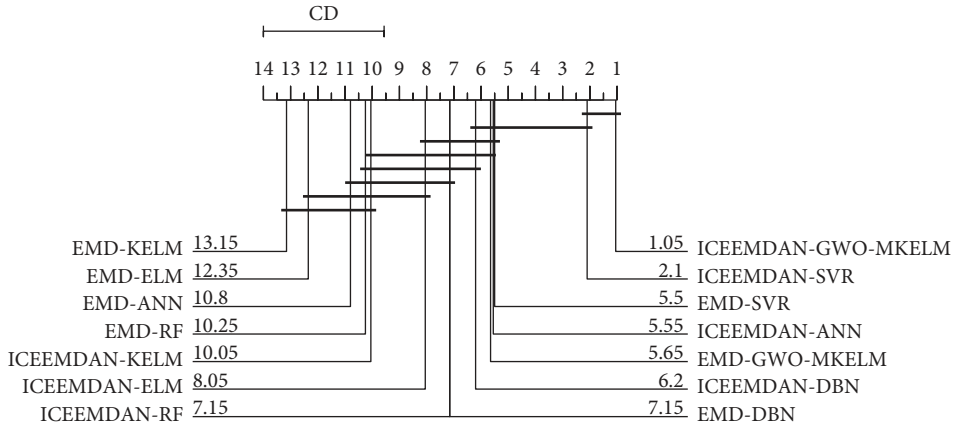

Figure 6: The Nemenyi test results of RMSE for STLF with ensemble models.

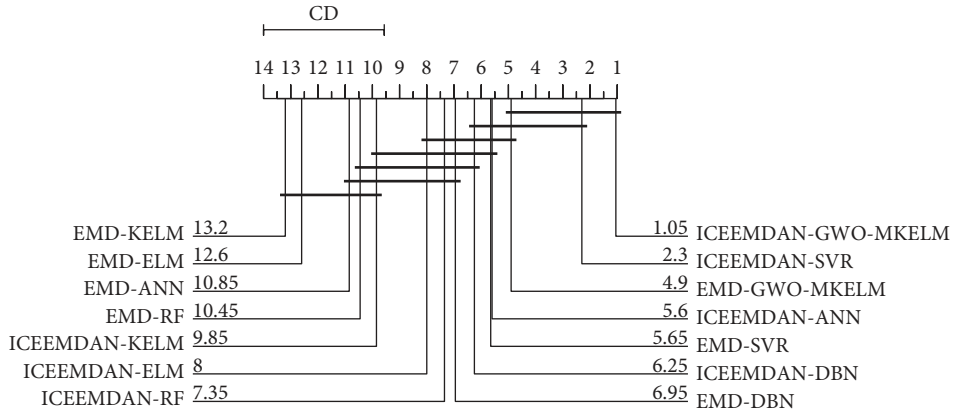

Figure 7: The Nemenyi test results of MAPE for STLF with ensemble models.

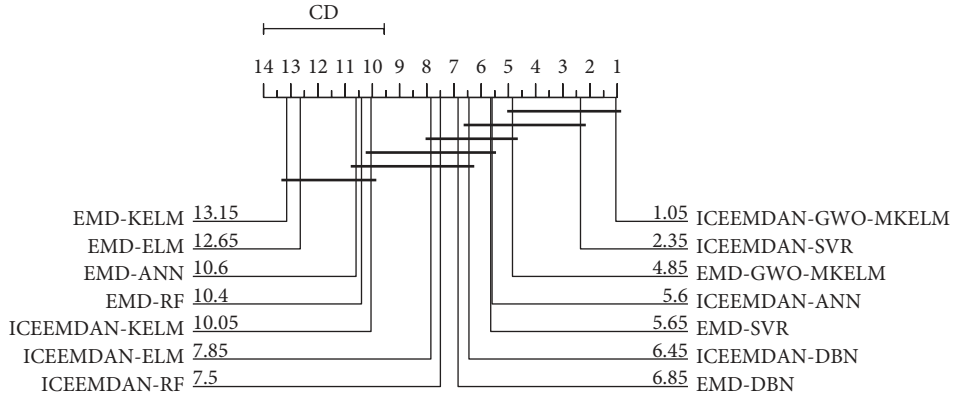

Figure 8: The Nemenyi test results of MAE for STLF with ensemble models.

the proposed ICEEMDAN-GWO-MKELM is very effective for STLF.

4.5. Discussion. In this subsection, we will study the influence of parameter settings of ICEEMDAN, lag order, GWO, and KELM. We will also discuss the running time of the forecasting models associated with ICEEMDAN. All the discussion is with the NSW dataset in JAN.

4.5.1. The Influence of Parameters for ICEEMDAN. The realization number and the noise strength are two key parameters for the ICEEMDAN. Here, we study their influence on STLF. First, we investigate the influence of the realization number in the set of $\{10, 25, 50, 75, 100, 300, 500, 1000, 1500, 2000\}$ while fixing the other parameters as set in Section 4.3. The results are shown in Figure 9.

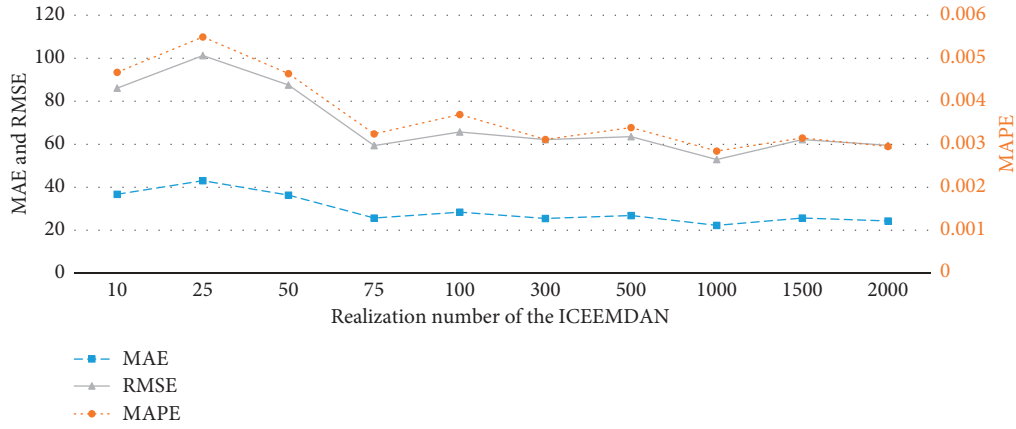

Figure 9: The influence of the realization number of the ICEEMDAN.

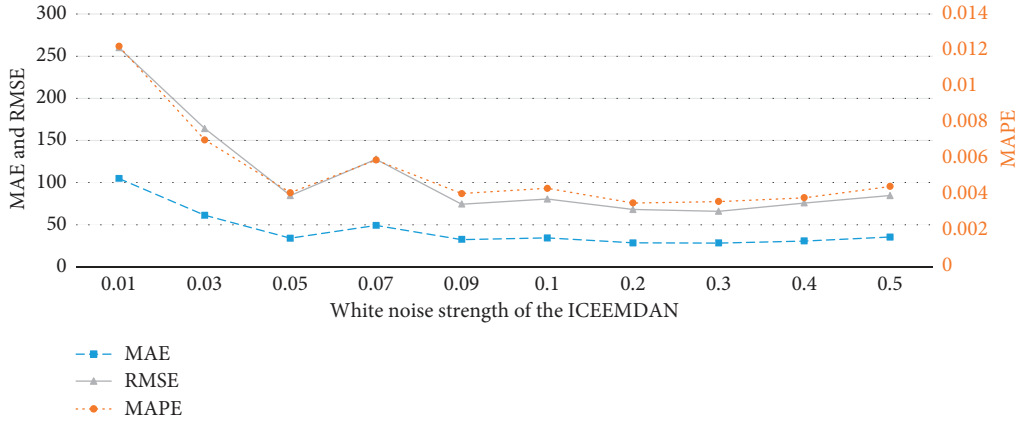

Figure 10: The influence of the white noise strength of the ICEEMDAN.

From this table, it can be seen that when the realization number is less than 75, all the evaluation criteria are poor, while when it is equal to or greater than 75, all the results are rather stable. Especially when it equals 1000, all the evaluation criteria obtain the best values. Therefore, to balance the computation cost and effectiveness, a value equal to or greater than 75 is reasonable for the realization number of the ICEEMDAN.

For the white noise strength, when the value is less than 0.09, the results are rather bad. However, when the value falls in the range of $[0.09, 0.4]$, the results show obvious stability, as shown in Figure 10. More specifically, the proposed approach achieves the best results when the white noise strength equals 0.2.

4.5.2. The Influence of the Lag Order. The lag order determines the length of the input data. A large lag order means more computation resources, while a small one may result in bad forecasting results. We test the lag orders in the range of $\{12, 24, 36, 48, 72, 96, 120, 144\}$, and the results are shown in Figure 11.

We can see that when the lag orders are equal to or less than 48, the results are very close, while after that, the results become worse and worse with the increase of the lag order. Because the experiments use half-hour load data and it just has 48 data points for one day, 48 is a reasonable lag order to balance the computation resource and forecasting results.

4.5.3. The Influence of GWO-MKELM's Parameter Settings. In the proposed approach, the combined kernel is built by four types of kernels, among which the number of RBF kernels is usually large. Here, we explore the influence of different numbers of RBF kernels, say $\{5, 10, 20, 30, 40, 50, 60, 80, 100, 150, 200\}$, as shown in Figure 12. It is shown that the proposed approach achieves similar results when the number of RBF kernels is less than 60, and after that, the results become worse with the increase in the number of RBF kernels. The ideal value for such parameter is in the range of $[5, 50]$.

The iteration number and search agent number (population size) are two key parameters for GWO, whose impacts are shown in Figures 13 and 14, respectively.

From Figure 13, we can see that when the number of iteration varies from 5 to 15, the experimental results become better and better. When the iteration number is equal to 20, the results become slightly worse, and then the results remain stable with the increase of the iteration number. Likewise, the number of search agents has a relatively stable

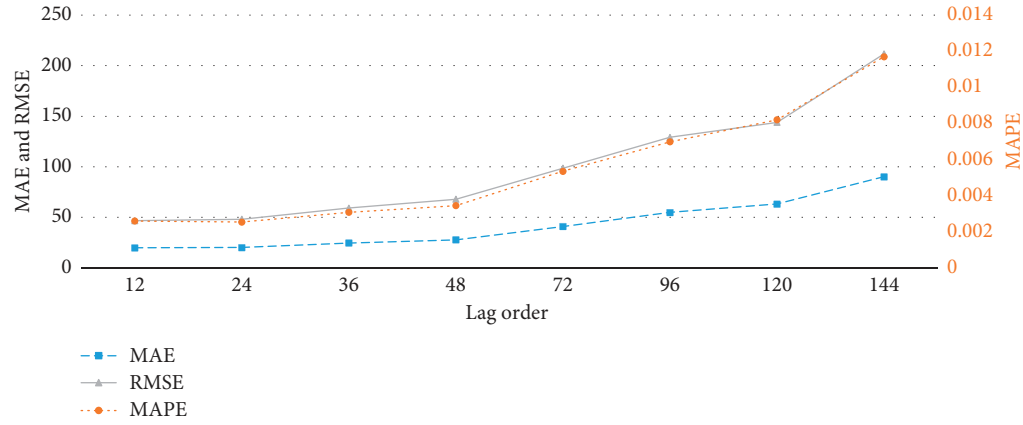

Figure 11: The influence of the lag order.

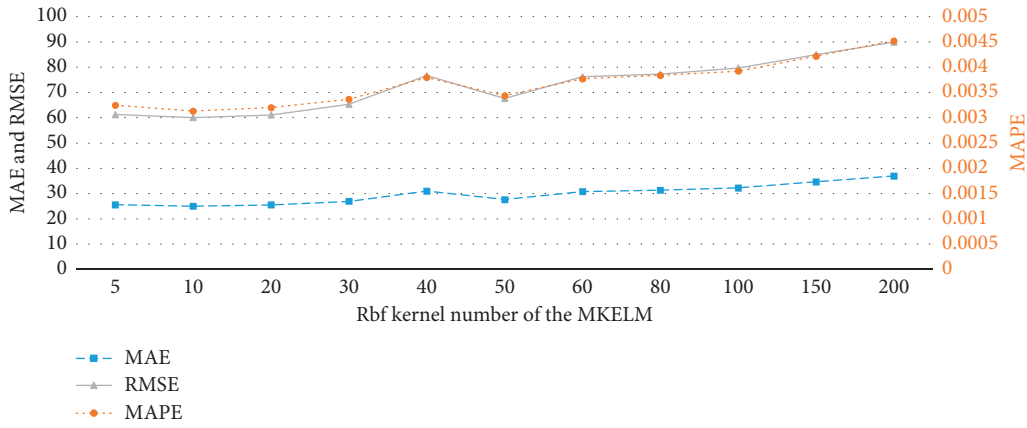

Figure 12: The influence of the number of RBF kernels.

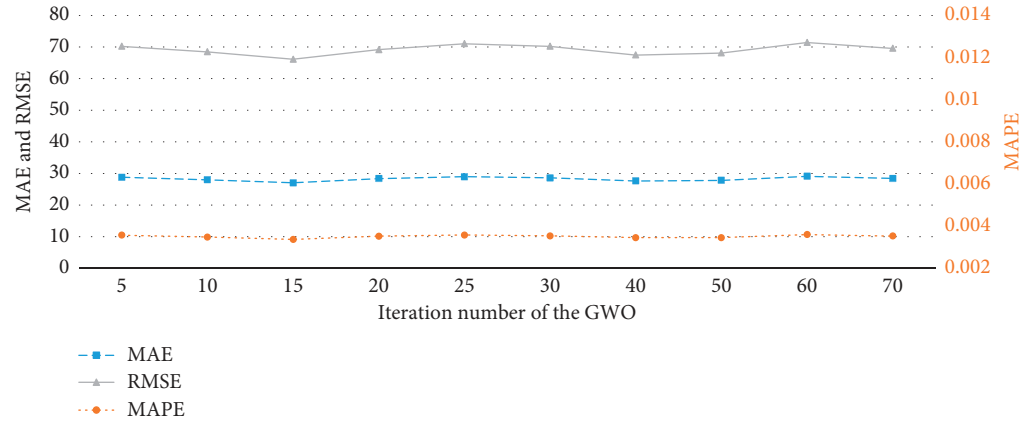

Figure 13: The influence of the iteration number of GWO.

impact on the forecasting results. The optimal number of search agents is 50, and for other numbers of search agents, the forecasting results are very close, as shown in Figure 14.

4.5.4. Running Time. We report the running time of the methods associated with ICEEMDAN in Table 5. It can be seen that ELM and DBN take the least time for training and testing, respectively. The GWO-MKELM takes the most time to train the forecasting model and test the samples because it has to perform matrix operations many times when evaluating each individual in the wolf pack. Fortunately, the training process can be conducted offline, and it needs to be performed only once. With the optimal weights and

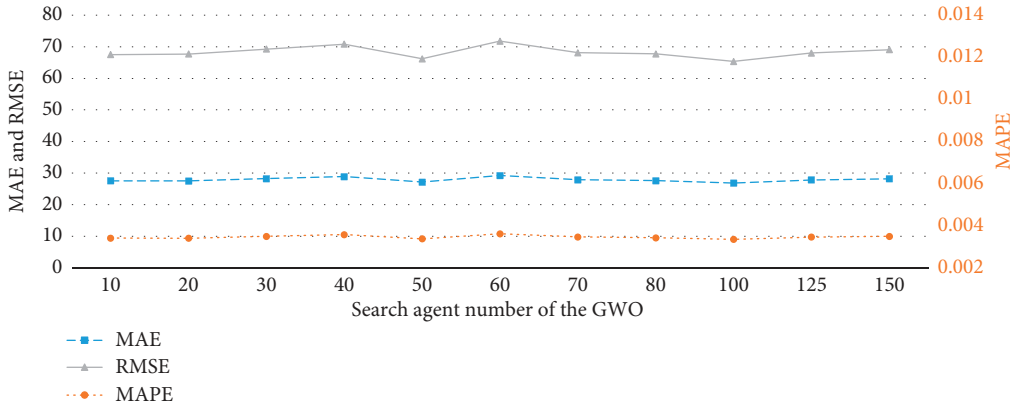

Figure 14: The influence of the search agent number of GWO.

Table 5: The running time of the methods associated with ICEEMDAN.

Models (ICEEMDAN)	Training time (s)	Testing time (s)
ANN	73.2102	0.2937
DBN	335.2610	0.0028
ELM	0.0450	0.0079
KELM	0.4614	0.0514
RF	0.4673	0.0722
SVR	10.2546	0.3431
GWO-MKELM	1264.3547	0.5156

parameters optimized by GWO, the running time for testing the 298 samples is 0.5156 seconds. In other words, it takes about 0.0017 seconds to test one sample, which is acceptable in real-world applications.

5. Conclusions

STLF plays an important role in smart grids. Because of the nonstationary and nonlinearity, it is a challenging task of STLF from the raw load data. To address this issue, we propose a novel approach for STLF based on the well-known "decomposition and ensemble" framework. We firstly present a novel multiple kernel learning approach that uses GWO to optimize the weights and the parameters of every single kernel in the combined kernel for ELM simultaneously. After that, a new approach integrating ICEEMDAN, GWO, and Multiple Kernel ELM, namely, ICEEMDAN-GWO-MKELM, is proposed for STLF. We also compare the proposed approach with some state-of-the-art forecasting methods on publicly accessible datasets containing half-hour load data of 4 months from 5 regions in Australia. From the experimental results, we can draw the following conclusions:

(1) Ensemble models significantly outperform the corresponding single models because the decomposed components can better express the intrinsic nature of the load data than the raw signal

(2) ICEEMDAN is superior to EMD for the decomposition of the load data

(3) The GWO-MKELM outperforms ELM with a single kernel (KELM) as well as nonkernel ELM for load forecasting

(4) The proposed ICEEMDAN-GWO-MKELM approach is advantageous over the state-of-the-art competitive methods in terms of the evaluation criteria, showing that the ICEEMDAN-GWO-MKELM is very promising for STLF.

One limitation of the proposed ICEEMDAN-GWO-MKELM is that it has to evaluate each wolf and many kernel matrices must be computed. Hence, more time is needed during the training phase. Fortunately, the training process can be performed offline. Therefore, once the forecasting model is built, the running time for testing is acceptable.

In the future, we will apply the ICEEMDAN-GWO-MKELM to forecasting other energy time series, such as electricity prices, natural gas prices, and wind speed.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Ministry of Education of Humanities and Social Science Project (Grant no. 19YJAZH047), the Fundamental Research Funds for the Central Universities (Grant no. JBK2003001), and the Scientific Research Fund of Sichuan Provincial Education Department (Grant no. 17ZB0433).

Supplementary Materials

All the experimental data used in this manuscript are stored in an EXCEL file and are included in the "Optional Supplementary Materials." All the data were accessed and downloaded via the Australian Energy Market Operator 2014

("https://www.aemo.com.au/Electricity/National-Electricity-Market-NEM/Data-dashboard#aggregated-data") on March 14, 2019. The EXCEL file contains 21 sheets. Please refer to README sheet for detailed information. Each of the other sheets contains half-hour load data of one region in one month. For example, the sheet "NSW-Jan" contains half-hour load data of New South Wales in January 2014. (*Supplementary Materials*)

References

[1] H. Jiang and Y. Dong, "Global horizontal radiation forecast using forward regression on a quadratic kernel support vector machine: case study of the Tibet autonomous region in China," *Energy*, vol. 133, pp. 270–283, 2017.

[2] Y. He, Y. Qin, S. Wang, X. Wang, and C. Wang, "Electricity consumption probability density forecasting method based on LASSO-quantile regression neural network," *Applied Energy*, vol. 233-234, pp. 565–575, 2019.

[3] C.-M. Lee and C.-N. Ko, "Short-term load forecasting using lifting scheme and ARIMA models," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5902–5911, 2011.

[4] H. Cui and X. Peng, "Short-term city electric load forecasting with considering temperature effects: an improved ARIMAX model," *Mathematical Problems in Engineering*, vol. 2015, Article ID 589374, 10 pages, 2015.

[5] Y. Yang, J. Che, Y. Li, Y. Zhao, and S. Zhu, "An incremental electric load forecasting model based on support vector regression," *Energy*, vol. 113, pp. 796–808, 2016.

[6] Y. Chen, P. Xu, Y. Chu et al., "Short-term electrical load forecasting using the support vector regression (SVR) model to calculate the demand response baseline for office buildings," *Applied Energy*, vol. 195, pp. 659–670, 2017.

[7] K. Nose-Filho, A. D. P. Lotufo, and C. R. Minussi, "Short-term multinodal load forecasting using a modified general regression neural network," *IEEE Transactions on Power Delivery*, vol. 26, no. 4, pp. 2862–2869, 2011.

[8] S. Kulkarni, S. P. Simon, and K. Sundareswaran, "A spiking neural network (SNN) forecast engine for short-term electrical load forecasting," *Applied Soft Computing*, vol. 13, no. 8, pp. 3628–3635, 2013.

[9] F. Yu and X. Xu, "A short-term load forecasting model of natural gas based on optimized genetic algorithm and improved BP neural network," *Applied Energy*, vol. 134, pp. 102–113, 2014.

[10] C.-U. Yeom and K.-C. Kwak, "Short-term electricity-load forecasting using a TSK-based extreme learning machine with knowledge representation," *Energies*, vol. 10, no. 10, p. 1613, 2017.

[11] T. Li, Z. Hu, Y. Jia, J. Wu, and Y. Zhou, "Forecasting crude oil prices using ensemble empirical mode decomposition and sparse Bayesian learning," *Energies*, vol. 11, no. 7, p. 1882, 2018.

[12] J. Wu, Y. Chen, T. Zhou, and T. Li, "An adaptive hybrid learning paradigm integrating CEEMD, ARIMA and SBL for crude oil price forecasting," *Energies*, vol. 12, no. 7, p. 1239, 2019.

[13] L. Han, Y. Peng, Y. Li, B. Yong, Q. Zhou, and L. Shu, "Enhanced deep networks for short-term and medium-term load forecasting," *IEEE Access*, vol. 7, pp. 4045–4055, 2019.

[14] C. Tong, J. Li, C. Lang, F. Kong, J. Niu, and J. J. P. C. Rodrigues, "An efficient deep model for day-ahead electricity load forecasting with stacked denoising auto-encoders," *Journal of Parallel and Distributed Computing*, vol. 117, pp. 267–273, 2018.

[15] X. Qiu, Y. Ren, P. N. Suganthan, and G. A. J. Amaratunga, "Empirical mode decomposition based ensemble deep learning for load demand time series forecasting," *Applied Soft Computing*, vol. 54, pp. 246–255, 2017.

[16] H. J. Sadaei, P. C. de Lima e Silva, F. G. Guimarães, and M. H. Lee, "Short-term load forecasting by using a combined method of convolutional neural networks and fuzzy time series," *Energy*, vol. 175, pp. 365–377, 2019.

[17] S. Wang, X. Wang, S. Wang, and D. Wang, "Bi-directional long short-term memory method based on attention mechanism and rolling update for short-term load forecasting," *International Journal of Electrical Power & Energy Systems*, vol. 109, pp. 470–479, 2019.

[18] W.-C. Hong, Y. Dong, C.-Y. Lai, L.-Y. Chen, and S.-Y. Wei, "SVR with hybrid chaotic immune algorithm for seasonal load demand forecasting," *Energies*, vol. 4, no. 6, pp. 960–977, 2011.

[19] B. Deng, D. Peng, H. Zhang, and Y. Qian, "An intelligent hybrid short-term load forecasting model optimized by switching delayed PSO of micro-grids," *Journal of Renewable and Sustainable Energy*, vol. 10, Article ID 024901, 2018.

[20] Y. Li, Z. Wen, Y. Cao, Y. Tan, D. Sidorov, and D. Panasetsky, "A combined forecasting approach with model self-adjustment for renewable generations and energy loads in smart community," *Energy*, vol. 129, pp. 216–227, 2017.

[21] Y. Hu, J. Li, M. Hong et al., "Short term electric load forecasting model and its verification for process industrial enterprises based on hybrid GA-PSO-BPNN algorithm—a case study of papermaking process," *Energy*, vol. 170, pp. 1215–1227, 2019.

[22] L. Tang, Y. Wu, and L. Yu, "A non-iterative decomposition-ensemble learning paradigm using RVFL network for crude oil price forecasting," *Applied Soft Computing*, vol. 70, pp. 1097–1108, 2018.

[23] Y. Zhou, T. Li, J. Shi, and Z. Qian, "A CEEMDAN and XGBOOST-based approach to forecast crude oil prices," *Complexity*, vol. 2019, Article ID 4392785, 15 pages, 2019.

[24] T. Li, Y. Zhou, X. Li, J. Wu, and T. He, "Forecasting daily crude oil prices using improved CEEMDAN and ridge regression-based predictors," *Energies*, vol. 12, no. 19, p. 3063, 2019.

[25] H. Liu, H.-q. Tian, D.-f. Pan, and Y.-f. Li, "Forecasting models for wind speed using wavelet, wavelet packet, time series and artificial neural networks," *Applied Energy*, vol. 107, pp. 191–208, 2013.

[26] Y. Ren, P. N. Suganthan, and N. Srikanth, "A novel empirical mode decomposition with support vector regression for wind speed forecasting," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 8, pp. 1793–1798, 2016.

[27] W.-C. Hong and G.-F. Fan, "Hybrid empirical mode decomposition with support vector regression model for short term load forecasting," *Energies*, vol. 12, no. 6, p. 1093, 2019.

[28] X. Gao, X. Li, B. Zhao, W. Ji, X. Jing, and Y. He, "Short-term electricity load forecasting model based on EMD-GRU with feature selection," *Energies*, vol. 12, no. 6, p. 1140, 2019.

[29] M. A. Kabir and C. Shahnaz, "Denoising of ECG signals based on noise reduction algorithms in EMD and wavelet domains," *Biomedical Signal Processing and Control*, vol. 7, no. 5, pp. 481–489, 2012.

[30] T. Li and M. Zhou, "ECG classification using wavelet packet entropy and random forests," *Entropy*, vol. 18, no. 8, p. 285, 2016.

[31] H. Zhao, J. Zheng, J. Xu, and W. Deng, "Fault diagnosis method based on principal component analysis and broad learning system," *IEEE Access*, vol. 7, pp. 99263–99272, 2019.

[32] T. Li, M. Yang, J. Wu, and X. Jing, "A novel image encryption algorithm based on a fractional-order hyperchaotic system and DNA computing," *Complexity*, vol. 2017, Article ID 9010251, 13 pages, 2017.

[33] X. Li, Z. Xie, J. Wu, and T. Li, "Image encryption based on dynamic filtering and bit cuboid operations," *Complexity*, vol. 2019, Article ID 7485621, 16 pages, 2019.

[34] T. Li, J. Shi, X. Li, J. Wu, and F. Pan, "Image encryption based on pixel-level diffusion with dynamic filtering and DNA-level permutation with 3D Latin cubes," *Entropy*, vol. 21, no. 3, p. 319, 2019.

[35] J. Wu, J. Shi, and T. Li, "A novel image encryption approach based on a hyperchaotic system, pixel-level filtering with variable kernels, and DNA-level diffusion," *Entropy*, vol. 22, no. 1, p. 5, 2019.

[36] N. E. Huang, Z. Shen, S. R. Long et al., "The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[37] Z. Wu and 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.

[38] M. E. Torres, M. A. Colominas, G. Schlotthauer, and P. Flandrin, "A complete ensemble empirical mode decomposition with adaptive noise," *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, pp. 4144–4147, IEEE, Prague, Czech Republic, May 2011.

[39] M. A. Colominas, G. Schlotthauer, and M. E. Torres, "Improved complete ensemble EMD: a suitable tool for biomedical signal processing," *Biomedical Signal Processing and Control*, vol. 14, pp. 19–29, 2014.

[40] G.-B. Huang, "An insight into extreme learning machines: random neurons, random features and kernels," *Cognitive Computation*, vol. 6, no. 3, pp. 376–390, 2014.

[41] G. Song and Q. Dai, "A novel double deep ELMs ensemble system for time series forecasting," *Knowledge-Based Systems*, vol. 134, pp. 31–49, 2017.

[42] A. A. Abdoos, "A new intelligent method based on combination of VMD and ELM for short term wind power forecasting," *Neurocomputing*, vol. 203, pp. 111–120, 2016.

[43] Ö. F. Ertugrul, "Forecasting electricity load by a novel recurrent extreme learning machines approach," *International Journal of Electrical Power & Energy Systems*, vol. 78, pp. 429–435, 2016.

[44] N. Liu, Q. Tang, J. Zhang, W. Fan, and J. Liu, "A hybrid forecasting model with parameter optimization for short-term load forecasting of micro-grids," *Applied Energy*, vol. 129, pp. 336–345, 2014.

[45] Y. Chen, M. Kloft, Y. Yang, C. Li, and L. Li, "Mixed kernel based extreme learning machine for electric load forecasting," *Neurocomputing*, vol. 312, pp. 90–106, 2018.

[46] M. Dorigo and C. Blum, "Ant colony optimization theory: a survey," *Theoretical Computer Science*, vol. 344, no. 2-3, pp. 243–278, 2005.

[47] W. Deng, H. Zhao, L. Zou, G. Li, X. Yang, and D. Wu, "A novel collaborative optimization algorithm in solving complex optimization problems," *Soft Computing*, vol. 21, no. 15, pp. 4387–4398, 2017.

[48] J. Kennedy, "Particle swarm optimization," *Encyclopedia of Machine Learning*, pp. 760–766, Springer, Berlin, Germany, 2010.

[49] W. Deng, H. Zhao, X. Yang, J. Xiong, M. Sun, and B. Li, "Study on an improved adaptive PSO algorithm for solving multi-objective gate assignment," *Applied Soft Computing*, vol. 59, pp. 288–302, 2017.

[50] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[51] M. Pradhan, P. K. Roy, and T. Pal, "Grey wolf optimization applied to economic load dispatch problems," *International Journal of Electrical Power & Energy Systems*, vol. 83, pp. 325–334, 2016.

[52] Q. Li, H. Chen, H. Huang et al., "An enhanced grey wolf optimization based feature selection wrapped kernel extreme learning machine for medical diagnosis," *Computational and Mathematical Methods in Medicine*, vol. 2017, Article ID 9512741, 15 pages, 2017.

[53] M. Wang, H. Chen, H. Li et al., "Grey wolf optimization evolving kernel extreme learning machine: application to bankruptcy prediction," *Engineering Applications of Artificial Intelligence*, vol. 63, pp. 54–68, 2017.

[54] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006.

[55] AEMO, "Australian energy market operator 2014," 2019, http://www.aemo.com.au/.

[56] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.