Expert Systems With Applications 206 (2022) 117784

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Random vector functional link neural network based ensemble deep learning for short-term load forecasting

Ruobin Gao [a], Liang Du [a], Ponnuthurai Nagaratnam Suganthan [b,c,*], Qin Zhou [a], Kum Fai Yuen [a,*]

[a] School of Civil & Environmental Engineering, Nanyang Technological University, Singapore
[b] School of Electrical & Electronic Engineering, Nanyang Technological University, Singapore
[c] KINDI Center for Computing Research, College of Engineering, Qatar University, Doha, Qatar

ARTICLE INFO

Keywords:
Forecasting
Random vector functional link network
Deep learning
Machine learning

ABSTRACT

Electric load forecasting is essential for the planning and maintenance of power systems. However, its unstationary and non-linear properties impose significant difficulties in predicting future demand. This paper proposes a novel ensemble deep Random Vector Functional Link (edRVFL) network for electricity load forecasting. The weights of hidden layers are randomly initialized and fixed during the training process. The hidden layers are stacked to enforce deep representation learning. Then, the model generates the forecasts using the ensemble of the outputs of each layer. Moreover, we also propose to augment the random enhancement features by empirical wavelet transformation (EWT). The raw load data are decomposed by EWT in a walk-forward approach without introducing future data leakage problems in the decomposition process. Finally, all the sub-series generated by the EWT, including raw data, are fed into the edRVFL for forecasting purposes. The proposed model is evaluated on sixteen publicly available time series from the Australian Energy Market Operator of the year 2020. The simulation results demonstrate the proposed model's superior performance over eleven forecasting methods in two error metrics and statistical tests on electricity load forecasting tasks.

1. Introduction

Forecasting electricity load accurately benefits electric power system planning for maintenance and construction. After collecting raw electricity demand, a reliable forecasting model established on raw historical data can estimate how much electricity is expected. Therefore, accurate forecasts can help the supplier to decrease energy generation and expenses and plan the resources efficiently (Heydari et al., 2020). Furthermore, short-term load forecasting models assist electricity organizations in making opportune decisions in a data-driven fashion. As a result, developing novel and accurate forecasting models for short-term load is beneficial.

The electricity load forecasting is a kind of time series forecasting tasks. Forecasting the future using intelligent forecasting models is a well-developed field, where the models established from the historical data are used to extrapolate future values (Makridakis, Wheelwright, & Hyndman, 2008). There are plentiful forecasting models, such as Auto-regressive integrated moving average (ARIMA) (Contreras, Espinola, Nogales, & Conejo, 2003), fuzzy time series (Gao & Duru, 2020), support vector regression (SVR) (Chen et al., 2004; Yang, Li, & Yang, 2019; Zhang & Hong, 2021), randomized neural networks (Ren, Suganthan, Srikanth, & Amaratunga, 2016), hybrid models (Gao, Du, & Yuen, 2020; Gao, Du, Yuen, & Suganthan, 2021; Qiu, Ren, Suganthan, & Amaratunga, 2017; Ren, Suganthan, & Srikanth, 2014a), ensemble learning (Qiu, Zhang, Suganthan, & Amaratunga, 2017; Qiu, Suganthan, & Amaratunga, 2018) and deep learning models (Liu et al., 2022). Accurate and reliable forecasts of electricity load is a challenging and significant problem for the electric power domain. In the field of load forecasting domain, the methods can be classified into three categories (i) statistical models, (ii) computational intelligence models and (ii) hybrid models. The statistical models, such as ARIMA (Contreras et al., 2003) and exponential smoothing (Taylor, 2011), are computationally efficient and theoretically solid, but their performance is not outstanding. The second huge branch is the computational intelligence models including fuzzy system (Ali, Adnan, Tariq, & Poor, 2020; Gao et al., 2020), SVR (Chen et al., 2004), shallow artificial neural networks (ANN) (Ren et al., 2016) and deep learning (Almalaq & Edwards, 2017; Shi, Xu, & Li, 2017; Hafeez, Alimgeer, & Khan, 2020; Qiu, Zhang, Ren, Suganthan, & Amaratunga, 2014). Although deep learning has dominated the intelligent methods for forecasting, it owns some limitations, such as overfitting, hyper-parameter optimization, and a

* Corresponding author.
E-mail addresses: gaor0009@e.ntu.edu.sg (R. Gao), liang011@e.ntu.edu.sg (L. Du), epnsugan@ntu.edu.sg (P.N. Suganthan), ZH0031IN@e.ntu.edu.sg (Q. Zhou), kumfai.yuen@ntu.edu.sg (K.F. Yuen).

https://doi.org/10.1016/j.eswa.2022.117784
Received 9 February 2022; Received in revised form 14 May 2022; Accepted 4 June 2022
Available online 11 June 2022

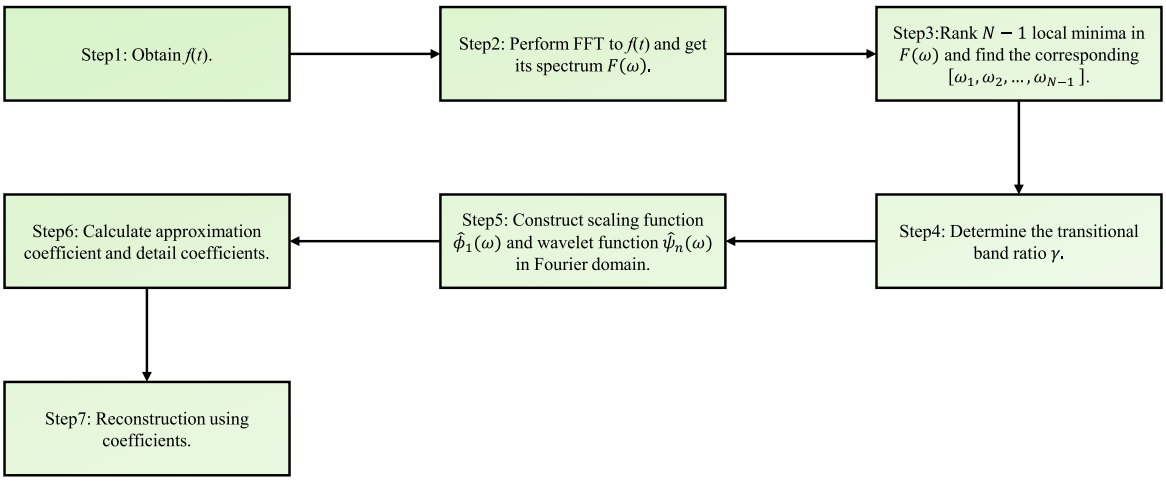

Fig. 1. EWT implementation.

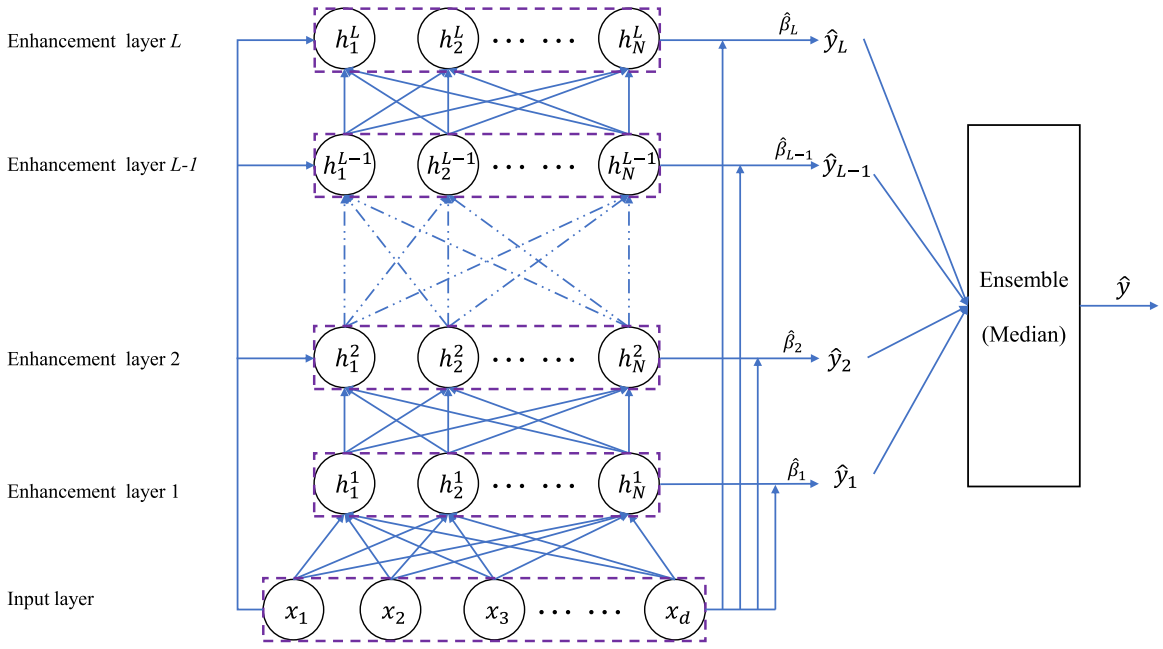

Fig. 2. Architecture of the edRVFL.

huge computation burden. Researchers have dedicated themselves to addressing these limitations and improving forecasting accuracy. For instance, the authors construct a diverse pool of training samples by integrating all neighbors' observations, and such diversity helps the deep recurrent neural network (RNN) address the overfitting problems (Shi et al., 2017). A deep factored conditional restricted Boltzmann machine (FCRBM) whose parameters are optimized via a genetic wind-driven optimization (GWDO) for load forecasting is proposed by Hafeez et al. (2020). The last category, hybrid models, includes the combination of feature extraction blocks and several forecasting models to form a single model. For example, the empirical mode decomposition (EMD) is utilized to extract modes from the load and then deep belief network (DBN) is implemented to forecast each mode in Qiu, Ren, et al. (2017). Empirical wavelet transformation (EWT) is applied to decompose the load data into sub-series in a walk-forward fashion and then the concatenation of raw data and sub-series is fed into a random vector functional link (RVFL) network for forecasting purposes (Gao et al., 2021).

Neural networks are popular models for mining complex knowledge due to their high accuracy and strong ability to handle nonlinearity (Almalaq & Edwards, 2017; Liang, Liu, et al., 2021). The deep learning models succeed in solving problems for transportation (Liang, Liu, et al., 2021), shipping industry (Liang, Zhan, & Liu, 2021) and load forecasting (Shi et al., 2017; Hafeez et al., 2020; Qiu et al., 2014). The deep learning models succeed in forecasting short-term load accurately because of their hierarchical structures which learn a meaningful representation of the input data. However, most fully trained deep learning models suffer from huge computation burdens. The huge computation of deep learning is due to the iterative optimization of the weights and biases. The network needs to do feed-forward and backward computations of the output and gradients during each iteration. To achieve adequate training, the deep learning models must process the whole dataset in multiple iterations. Therefore, this paper proposes a fast ensemble deep learning algorithm for short-term load forecasting. The proposed model inherits the advantages of ensemble learning and deep learning without imposing much computational burden at the same time. This paper investigates the forecasting ability of a special kind of randomized deep neural networks, the deep RVFL network.

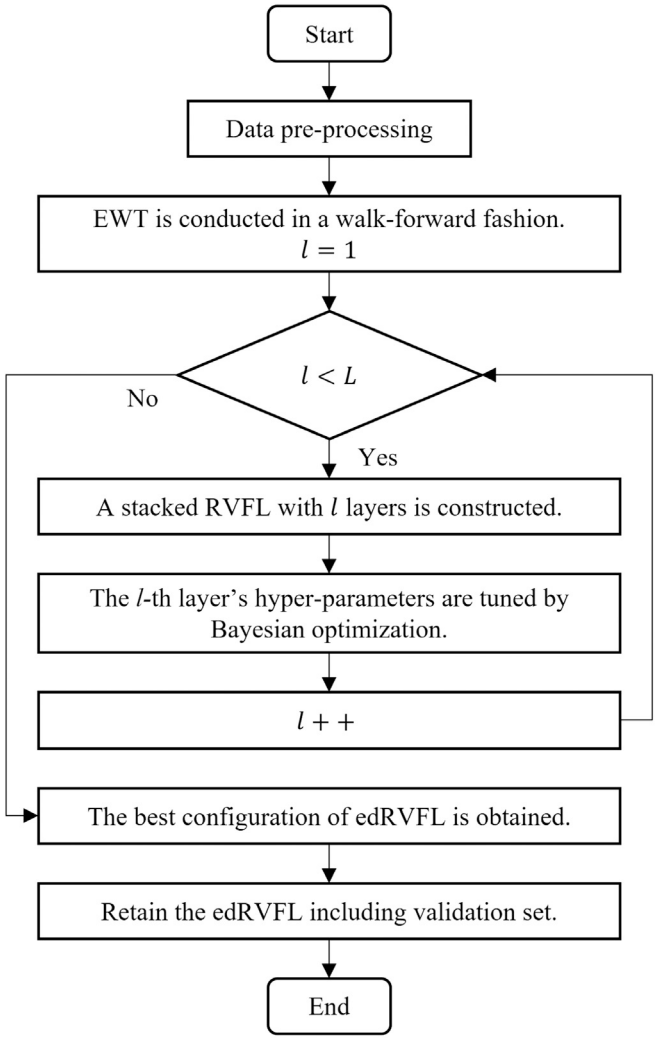

Fig. 3. The proposed model's flowchart.

The randomization of the RVFL network simultaneously achieves strong non-linearity and generalization abilities (Zhang & Suganthan, 2016b). The hidden layers of the RVFL network are randomly initialized and do not need training. The unique part, which requires training, is the linear output layer. The output layer's weights are computed with the closed-form solution without iterative steps. Therefore, the training of the deep RVFL network is faster than the deep learning models with an iterative training process. Ensemble learning techniques are combined with the deep RVFL to reduce the uncertainty caused by a single model. Since the deep RVFL's hidden features are randomly generated and remained fixed during the training process, the EWT is utilized to extract features with different frequencies to augment the deep RVFL's random features. Recently, the universal function approximation ability of the RVFL network is proved in Needell, Nelson, Saab, and Salanevich (2020). This paper uses EWT to decompose the raw data in a walk-forward fashion which is different from decomposing the whole time series altogether (Qiu, Ren, et al., 2017; Ren et al., 2014a; Ren, Suganthan, & Srikanth, 2014b; Qiu et al., 2018). The future data are not involved in the walk-forward decomposition process. Therefore there is no data leakage problem in terms of forecasting.

The novel perspectives of the proposed model are summarized as follows:

1. This paper implements the edRVFL for short-term load forecasting for the first time. The median computation is utilized as the ensemble approach which is different from the edRVFL for classification (Shi, Katuwal, Suganthan, & Tanveer, 2021).
2. The EWT is combined with the edRVFL as a feature engineering block to augment the random features. Furthermore, the EWT is conducted in a walk-forward fashion to avoid future data leakage problems. Finally, a novel hybrid forecasting model based on walk-forward EWT and edRVFL is proposed for short-term load forecasting.
3. The hyper-parameters of the proposed model are optimized in a layer-wise fashion. The succeeding layers are based on the optimized previous layer's features. Therefore, each layer has its suitable hyper-parameters and does not degrade the performance.
4. The proposed model is compared with various benchmark models from statistical ones to state-of-the-art models on sixteen load time series. Two error metrics and two statistical tests are conducted for precise comparisons. The statistical tests demonstrate the proposed model's superiority both in a group-wise and pair-wise fashion.

The remainder of this paper is organized as follows: Section 2 describes the methodologies and the proposed model in detail. We first describe the EWT and the walk-forward decomposition. Then, the ensemble deep RVFL and its combination with the walk-forward EWT is presented. Section 3 presents the experimental step-up and the results. Finally, conclusions are drawn and potential future directions are discussed in Section 4.

2. Methodology

This section describes the methodologies in detail. First, we introduce the EWT and the walk-forward decomposition procedure. Then, we describe the ensemble deep RVFL network and the proposed model.

2.1. Empirical wavelet transformation

The EWT is an automatic signal processing approach with solid theories in decomposing non-stationary time series (Gilles, 2013). Unlike discrete wavelet transform (DWT) and EMD (Flandrin, Rilling, & Goncalves, 2004), EWT precisely investigates the time series in the Fourier domain after fast Fourier transform (FFT). The EWT separates the spectrum using data-driven band-pass filtering. Fig. 1 shows the EWT's regular procedures. In the EWT, limited freedom is provided for selecting wavelets. The algorithm employs Littlewood–Paley and Meyer's wavelets because of the analytic accessibility of the Fourier domain's closed-form formulations (Spencer, 1994). We represent the normalized frequency as $\omega \in [0, \pi]$. We utilize ω_n to represent the limits between the segments that are obtained from the Fourier support $[0, \pi]$. These band-pass filters' formulations are denoted using Eqs. (1) and (2)

$$\hat{\phi}_n(\omega) = \begin{cases} 1 & \text{if } |\omega| \le (1-\gamma)\omega_n \\ \cos\left[\frac{\pi}{2}\beta\left(\frac{1}{2\gamma\omega_n}\left(|\omega| - (1-\gamma)\omega_n|\right)\right)\right] & \\ & \text{if } (1-\gamma)\omega_n \le |\omega| \le (1+\gamma)\omega_n \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

$$\hat{\psi}_n(\omega) = \begin{cases} 1 & \text{if } (1+\gamma)\omega_n \le |\omega| \le (1-\gamma)\omega_{n+1} \\ \cos\left[\frac{\pi}{2}\zeta\left(\frac{1}{2\gamma\omega_{n+1}}\left(|\omega| - (1-\gamma)\omega_{n+1}|\right)\right)\right] & \\ & \text{if } (1-\gamma)\omega_{n+1} \le |\omega| \le (1+\gamma)\omega_{n+1} \\ \sin\left[\frac{\pi}{2}\zeta\left(\frac{1}{2\gamma\omega_n}\left(|\omega| - (1-\gamma)\omega_n|\right)\right)\right] & \\ & \text{if } (1-\gamma)\omega_n \le |\omega| \le (1+\gamma)\omega_n \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

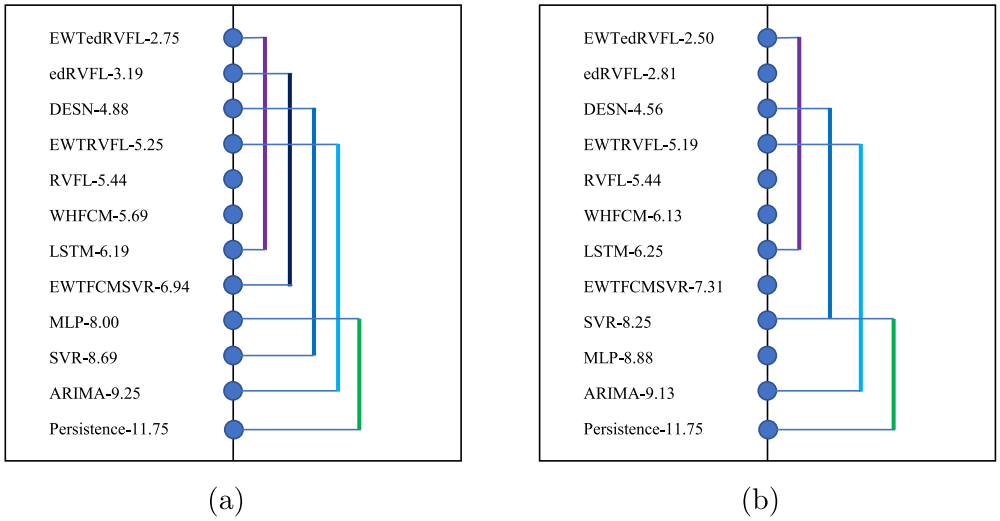

Fig. 4. Nemenyi testing results for load forecasting based on: (a) RMSE and (b) MASE. The critical distance is 4.17. The Friedman p-values are: (a) 7.92e−28 and (b) 4.55e−22.

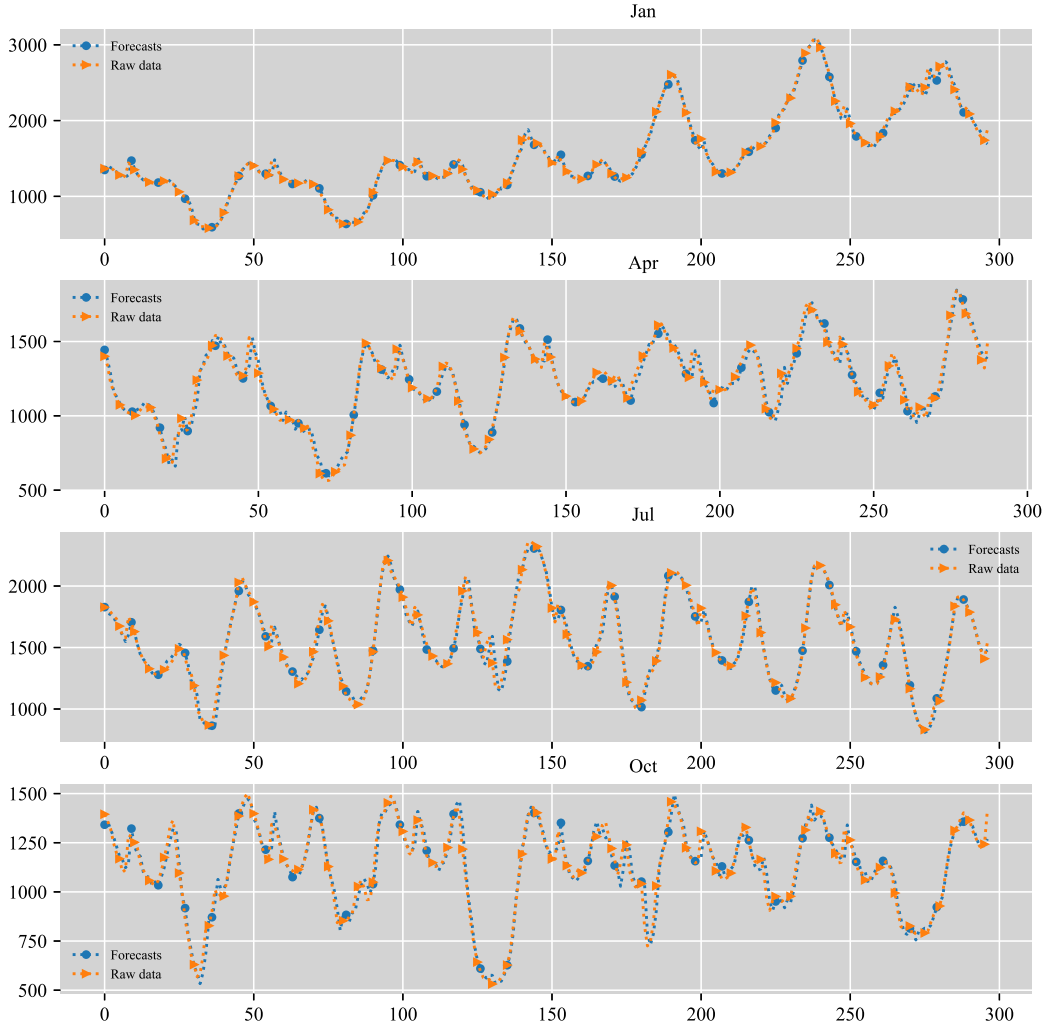

Fig. 5. Comparisons of raw data and forecasts for the SA dataset.

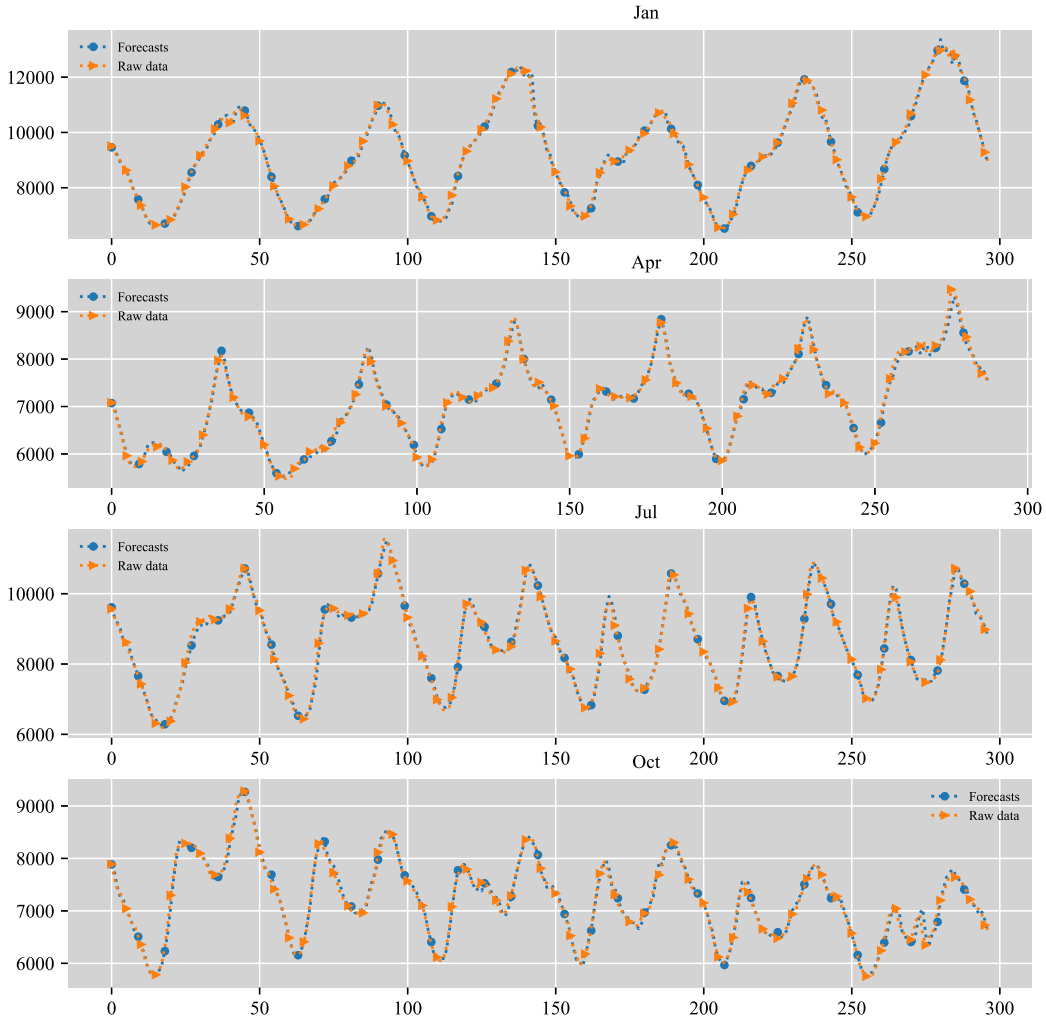

Fig. 6. Comparisons of raw data and forecasts for the NSW dataset.

with a transitional band width parameter γ satisfying $\gamma \leq \min_n \frac{\omega_{n+1}-\omega_n}{\omega_{n+1}+\omega_n}$. The most common function $\zeta(x)$ in Eqs. (1) and (2) are presented in Eq. (3).

$$\beta(x) = x^4(35 - 84x + 70x^2 - 20x^3) \tag{3}$$

This empowers the formulated empirical scaling and wavelet function $\{\hat{\phi}_1(\omega), \{\hat{\psi}_n(\omega)\}_{n=1}^N\}$ to be a tight frame of $L^2(\mathbb{R})$ (Casazza et al., 2000). It can be observed that $\{\hat{\phi}_1(\omega), \{\hat{\psi}_n(\omega)\}_{n=1}^N\}$ are used as band-pass filters centered at assorted center frequencies.

2.2. Walk-forward decomposition

Plentiful works utilize signal decomposition techniques as a feature engineering block for the forecasting algorithms (Qiu, Ren, et al., 2017; Ren et al., 2014a; Ghelardoni, Ghio, & Anguita, 2013; Gao et al., 2020; Yang & Liu, 2018; Gao et al., 2021; Huang & Deng, 2021), however, most do not implement the decomposition in a proper way (Gao et al., 2021; Huang & Deng, 2021). As mentioned in Gao et al. (2020, 2021), Huang and Deng (2021), direct utilization of signal decomposition algorithm to the whole time series causes the data leakage problem in terms of forecasting. The decomposed data are actually the output from convolution operations and the future data definitely are involved during the convolution. Therefore, decomposing the whole time series is incorrect and improper, especially for establishing forecasting models.

Some solutions are proposed to avoid the future data leakage problem for decomposition-based forecasting models, such as the data-driven padding (Gao et al., 2020), moving window strategy (Huang & Deng, 2021) and walk-forward decomposition (Gao et al., 2021). The data-driven padding approach is to train a simple learning algorithm which aims at padding its forecast to the end of the time series (Gao et al., 2020). The moving window strategy only decomposes the data located in the window (order) and then the decomposed series are fed into forecasting models (Huang & Deng, 2021). Different from the moving window strategy, only part of the decomposed sub-series are used as input in the walk-forward decomposition. The moving window strategy is a subset of the walk-forward decomposition. When the order is equal to the window, the moving window and the walk-forward strategies are the same.

This paper adopts the walk-forward decomposition for the EWT. The walk-forward EWT decomposes the data within a rolling window w, which consists of $x(t-1), x(t-2), \ldots \ldots x(t-w)$, into k scales with the aim to forecast $x(t)$. Then only the last order data points are used as input for the forecasting model. The whole process of the walk-forward EWT is presented in Algorithm 1. Therefore, only historical observations are involved both in the decomposition process and the model' training.

2.3. BayesIan optimization algorithm for hyper-parameters tuning

Bayesian optimization (Pelikan, Goldberg, Cantú-Paz, et al., 1999) utilizes a surrogate function to model the conditional probability of the performance on the validation set. Different from grid search method, Bayesian optimization memorizes all the calculations. The

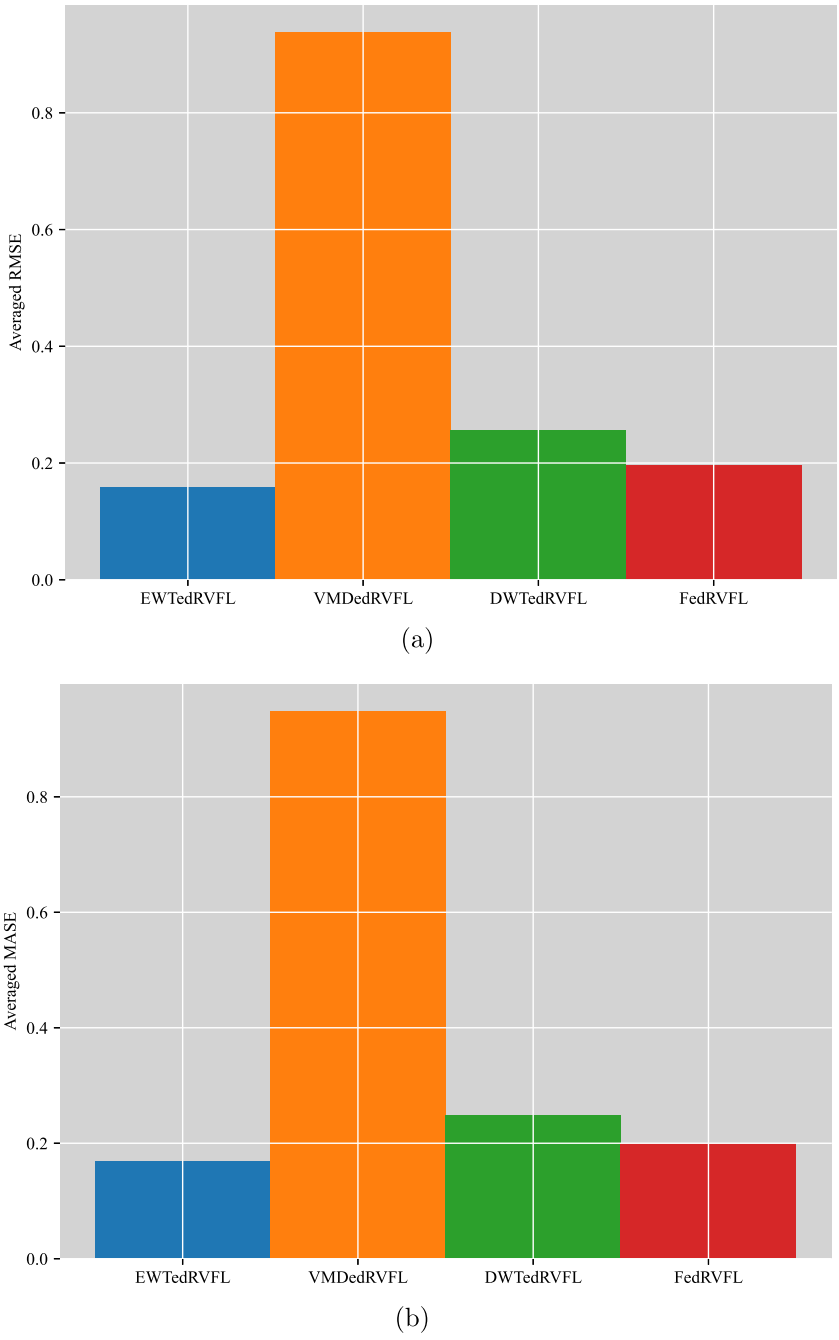

Fig. 7. Comparative results of different signal processing techniques: (a) Average RMSE and (b) Average MASE.

algorithm avoids the redundant computation to evaluate inferior hyper-parameters. Then, an acquisition function is utilized to seek the best configuration to be evaluated in the next iteration.

Bayesian optimization algorithm (BOA) consists of five elements: surrogate function, the searching space of hyper-parameters, acquisition function, objective function, and evaluations' history. In this paper, the objective function is defined as the prediction performance on the validation set. This paper utilizes the popular tree-based Parzen window estimation (TPE) algorithm to model the surrogate function and uses expected improvement as the acquisition function S shown in Eq. (4)

$$S_{f^*}(v) = \int_{-\infty}^{f^*} (f^* - f) P(f|v) df, \tag{4}$$

where f is the objective function and f^* is the threshold of objective function given the hyper-parameters v. Finally, the procedures of TPE are presented in Algorithm 2.

2.4. Ensemble deep RVFL

Inspired by the deep representation learning, the deep RVFL is an extension of the RVFL with a shallow structure (Shi et al., 2021). The deep RVFL is established by stacking multiple enhancement layers to achieve deep representation learning. The clean data are fed into each enhancement layer to guide the random features' generation. In this fashion, the enhancement features of hidden layers are generated based on the information from the clean data and the features from the previous layer. A diverse set of features is generated with the help of

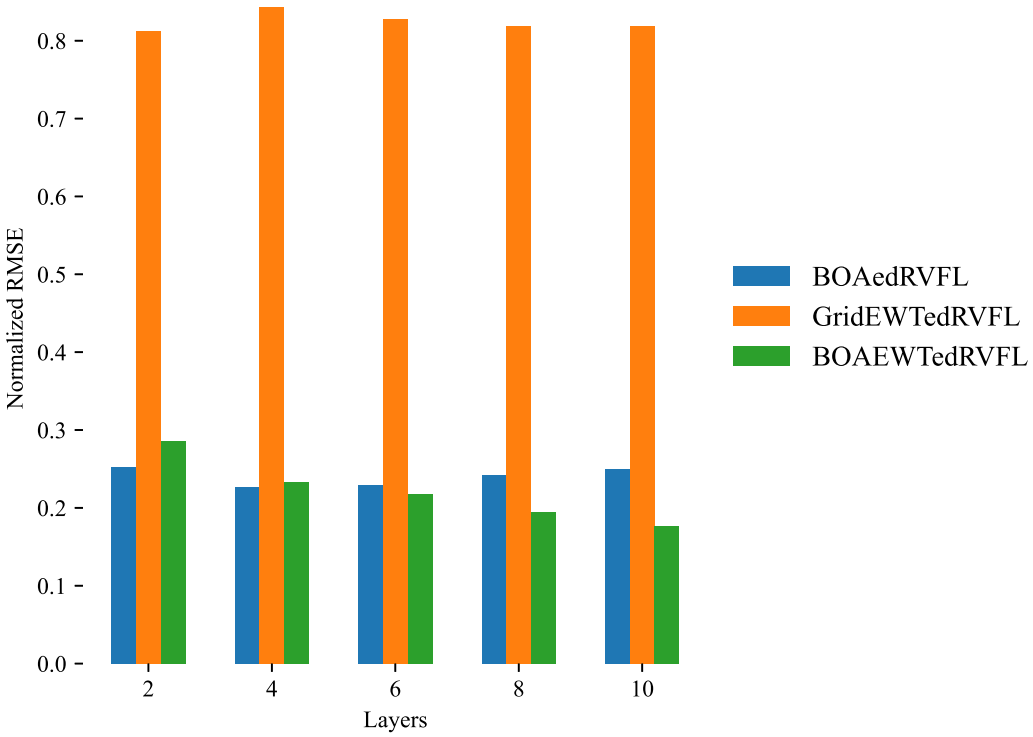

Fig. 8. The barplot of the performance measured in normalized RMSE. The BOAedRVFL represents the edRVFL trained using layer-wise BOA. The GridEWTedRVFL represents the EWTedRVFL optimized by grid search. Finally, the BOAEWTedRVFL represents the proposed model whose input is augmented by EWT, and the layer-wise BOA optimizes architecture.

Algorithm 1: Walk-forward EWT.

Input: Input time series $x(t)$, rolling window size w, decomposition scales k, time lag used for modeling p
Output: Sub-series, $\mathbf{X}$

$\mathbf{X} = zeros(k, p, len(x(t)) - w)$
for $i = w : len(x(t))$ **do**
  Apply EWT to decompose $x(i-w:i)$ to k scales $x_1(i-w:i), x_2(i-w:i), ..., x_k(i-w:i)$
  for $j = 1 : k$ **do**
    $\mathbf{X}[j, :, i-w] = x_j(i-p:i)$
  end
end

Algorithm 2: TPE for hyper-parameters tuning.

Input: Objective function f, TPE method $\mathcal{M}$, hyper-parameters space $\mathbb{H}_v$, acquisition function validation S, initialized memory D
Output: Best hyper-parameters v^* in memory D

for $i = 1 : N$ **do**
  $P(f|v) \leftarrow$ Fit memory D using $\mathcal{M}$
  $v_{i+1} \leftarrow$ Maximize acquisition function S in Equation (4) in search for the next hyper-parameters choice
  $f(v_{i+1}) \leftarrow$ Evaluate the objective function
  $D \leftarrow D \cup (v_{i+1}, f(v_{i+1}))$
end

hierarchical structures. Ensemble learning is introduced into the deep RVFL architecture to formulate the ensemble deep RVFL (edRVFL). Different from the popular deep learning models with a single output layer, the edRVFL trains multiple output layers based on all the hidden features. Finally, the forecasts from all output layers are combined for forecasting.

For the sake of presentation simplicity, we only present the edRVFL with a structure of L enhancement layers and there are N enhancement nodes in each layer. Fig. 2 shows the architecture of the edRVFL network with L hidden layers. The input feature vector with d dimensions is propagated to the enhancement layers with N non-linear nodes. When computing the output, each enhancement layer's features are concatenated with the input data to generate its output $\hat{y}_i$. Finally, the aggregation of all $\hat{y}_i$ is the entire network's output. Suppose that the input data are $\mathbf{X} \in \mathbb{R}^{n \times d}$, where n and d represent the number of samples and feature dimension, respectively. d is the time lag (order) for the time series forecasting model. The features generated by the first enhancement layer are defined as

$$\mathbf{H^1} = g(\mathbf{X}\mathbf{W_1}), \tag{5}$$

where $\mathbf{W_1} \in \mathbb{R}^{d \times N}$ represents the weight vector of the first enhancement layer, $\mathbf{H^1} \in \mathbb{R}^{n \times N}$ denotes the enhancement features and $g()$ is a non-linear activation function. The readers can refer to Zhang and Suganthan (2016a) for a comprehensive evaluation of different activation functions. Then, for the deeper enhancement layer l, the enhancement features can be computed as

$$\mathbf{H}^l = g([\mathbf{H}^{l-1}, \mathbf{X}]\mathbf{W}_l), \tag{6}$$

where $\mathbf{W}_l \in \mathbb{R}^{(d+N) \times N}$ and $\mathbf{H}^l \in \mathbb{R}^{n \times N}$. The enhancement weight vectors $\mathbf{W_1}$ and $\mathbf{W}_l$ are randomly initialized and remained fixed during training.

The edRVFL computes the output weights by splitting the task into l small tasks. The output weights are calculated separately for each layer. There are several differences from using the last layer's features and all layers' features for decisions. Most deep learning models only use the last layer's features for decisions, however, the information from the intermediate features is lost. These intermediate features

from the hidden layers formulate a hierarchical representation of the input data. Simply ignoring them may lose multi-scale information and deteriorate the network's performance. Using all layers' features requires a computation on the feature matrix with a huge dimension. Moreover, both of the above architectures only train one network, but our method benefits from the ensemble approach, which reduces the uncertainty of a single model.

The loss function of lth enhancement layer is defined as

$$Loss_l = \|[\mathbf{H}^l, \mathbf{X}]\beta_l - Y\|^2 + \lambda\|\beta_l\|^2, \tag{7}$$

where β_l denotes the output vector of lth layer and λ is the regularization parameter. The minimization of $Loss_l$ can be solved via a closed-form solution based on ridge regression (Saunders, Gammerman, & Vovk, 1998).

$$\beta_l = (\mathbf{D^T D} + \lambda \mathbf{I})^{-1}\mathbf{D}^T Y, \tag{8}$$

where $\mathbf{D} = [\mathbf{H}^l, \mathbf{X}]$. After computing all β_l, the deep network can output L forecasts. The output of lth layer is calculated by

$$Y = [\mathbf{H}^l, \mathbf{X}]\beta_l \tag{9}$$

The training algorithm of the edRVFL is summarized in Algorithm 3. First, it is necessary to define the network's hyper-parameters, such as the hidden dimension, number of layers, and regularization strength. After initializing the hidden layer's weights, a feed-forward computation of the hidden state $\mathbf{H}^l$ is conducted using Eqs. (5) or (6). Finally, this training algorithm outputs the output layers' weights $\beta = [\beta_1, \dots, \beta_l]$.

The final forecast is an ensemble of all outputs. Any forecast combination approach can be applied to this procedure (Timmermann, 2006). According to the suggestions in Timmermann (2006), the median operation is always likely to improve the forecast combination's performance. Therefore, we use the median as the combination operator and propose the edRVFL. The whole test process for the edRVFL is presented in Algorithm 4. When the training process is finished, the edRVFL computes the output $\hat{y}$ in a feed-forward way.

Algorithm 3: Training algorithm for edRVFL

Input: N, the hidden dimension
L,the number of layers
λ_l,the regularization strength of lth layer
Output: $\beta = [\beta_1, ..., \beta_l]$

Initialize the $\mathbf{W}_1, ..., \mathbf{W}_L$ randomly
$l = 1$
for $l \le L$ **do**
  if $l = 1$ **then**
    Compute $\mathbf{H^1}$ using $\mathbf{W}_1$ as in Equation (5)
    Compute $\beta_\mathbf{1}$ using λ_1 as in Equation (8)
  else
    Compute $\mathbf{H}^l$ using $\mathbf{W}_l$ as in Equation (6)
    Compute β_l using λ_l as in Equation (8)
  end
  $l++$
end

2.5. EWT-edRVFL

The model EWT-edRVFL consists of two blocks, the walk-forward EWT decomposition and the edRVFL. The walk-forward EWT is first applied to the load data to extract some features in a causal fashion. Then the raw data concatenated with the sub-series are fed into the edRVFL with L enhancement layers for learning purposes. The output weights β_l of the lth enhancement layer are computed according to Eq. (8). Finally, we ensemble the L forecasts with median operation to obtain the output $\hat{y}$.

Algorithm 4: Test process for edRVFL

Input: W, the hidden layers' weights
β,the output layers' weights
λ_l,the regularization strength of lth layer
Output: $\hat{y}$
Data: X,the test samples

$l = 1$
for $l \le L$ **do**
  if $l = 1$ **then**
    Compute $\mathbf{H}^1$ using $\mathbf{W}_1$ as in Equation (5)
    Compute the output Y_1 of each output layer according to Equation (9)
  else
    Compute $\mathbf{H}^l$ using $\mathbf{W}_l$ as in Equation (6)
    Compute Y_l using $\mathbf{W}_l$ as in Equation (9)
  end
  $l++$
end
Compute the final output via $median(\hat{y}_1, ..., \hat{y}_L)$

Since the higher enhancement layer's performance depends on the lower ones', the hyper-parameters of the whole model are tuned in a layer-wise fashion. Once the shallow layer's hyper-parameters are determined, they are fixed, and the Bayesian optimization approach is applied to the next layer. The layer-wise Bayesian optimization algorithm is shown in Algorithm 5. The layer-wise Bayesian optimization offers a different set of hyper-parameters for each layer. Therefore each enhancement layer has its own regularization parameter, which helps the overall edRVFL learns a diverse set of output layers. The flowchart of the proposed model is shown in Fig. 3.

Algorithm 5: Layer-wise cross-validation algorithm for edRVFL

Input: Search space for N and λ
L, number of layers
Output: The best configuration of edRVFL

$l = 1$
for $l \le L$ **do**
  if $l = 1$ **then**
    Cross-validation for RVFL
    Obtain the best configuration, N and λ_1
    Compute $\mathbf{H^1}$ using the N and λ_1 based on Equation (5)
    $\mathbf{H}^1$ is fixed and used for the following layers
  else
    Cross-validation for lth layer based on $\mathbf{H}^{l-1}$
    Obtain the best configuration, N and λ_l
    Compute $\mathbf{H}^l$ using the N and λ_l based on Equation (6)
    $\mathbf{H}^l$ is fixed and used for the following layers
  end
  $l++$
end

3. Empirical study

This section presents the empirical study on twenty load time series collected from the Australian Energy Market Operator (AEMO). First, we briefly introduce the dataset and pre-processing steps. Then, the benchmark models and hyper-parameter optimization are described. Finally, the simulation results are shown, and discussions are conducted.

3.1. Data and its nature

Table 1 summarizes the descriptive statistics of the twenty load time series. These load data are collected from the states of South Australia

Table 1
Descriptive statistics.

Location	Month	Max	Min	Median	Mean	Std	Skewness	Kurtosis
SA	Jan	3085.49	440.54	1212.79	1268.80	427.93	1.26	2.60
	Apr	1841.85	503.67	1177.78	1161.61	248.31	−0.33	−0.37
	Jul	2383.18	765.27	1489.76	1514.57	338.45	0.26	−0.59
	Oct	1955.46	288.92	1140.50	1095.25	266.31	−0.55	0.21
NSW	Jan	13330.14	5765.85	8053.13	8264.22	1535.24	0.85	0.42
	Apr	9471.04	5384.58	6983.91	6926.61	792.43	0.20	−0.58
	Jul	11739.02	5678.37	8670.19	8690.30	1247.70	0.17	−0.75
	Oct	9324.77	5221.13	6999.92	6955.32	771.00	0.01	−0.62
VIC	Jan	9507.26	3060.58	4565.41	4765.55	1017.14	1.82	4.39
	Apr	6515.96	3094.45	4453.18	4485.45	632.63	0.29	−0.42
	Jul	7354.11	3816.70	5497.73	5514.65	832.99	0.04	−0.92
	Oct	6142.91	2975.43	4325.26	4379.82	587.84	0.27	−0.53
TAS	Jan	1298.63	794.25	1036.17	1040.35	84.44	0.09	−0.26
	Apr	1379.49	843.31	1087.11	1093.91	113.14	0.22	−0.71
	Jul	1597.64	887.09	1240.32	1246.55	151.24	0.08	−0.86
	Oct	1447.61	842.78	1068.39	1087.26	112.91	0.47	−0.33

(SA), New South Wales (NSW), Victoria (VIC), and Tasmania (TAS) of the year 2020, which is significantly affected by Covid-19. Four months, January, April, July, and October are selected to reflect the four seasons' features as in Gao et al. (2021), Jalali et al. (2021), Qiu, Ren, et al. (2017). The data are recorded every half an hour. Therefore, there are 48 data points per day. The data are publicly available and can be accessed at the official website of Australian Energy Market Operator.[1]

A suitable and correct data pre-processing approach helps the machine learning model generate accurate outputs. We utilize the max–min normalization to pre-process the raw data. We assume that the maximum and minimum of the training set are x_{max} and x_{min}, respectively. The data are transformed into the range [0,1] using the following equation:

$$x_{normalized} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{10}$$

where $x_{normalized}$ and x represent the normalized and original time series, respectively.

All datasets are split into three sets, the training, validation and test set, to adopt the cross-validation (Bergmeir & Benítez, 2012). The validation and test set account for 10% and 20% of the dataset, respectively. The remaining data are used as the training set.

3.2. Results and discussion

Two forecasting error metrics are employed to appraise the accuracy of these models. The first error metric is the regular root mean square error (RMSE) whose definition is

$$RMSE = \sqrt{\frac{1}{L_{test}} \sum_{j=1}^{L_{test}} (\hat{x_j} - x_j)^2}, \tag{11}$$

where L_{test} is the size of the test set, x_j and $\hat{x_j}$ are the raw data and predictions. The second error metric implemented in the paper is the mean absolute scaled error (MASE) (Hyndman & Koehler, 2006). The definition of MASE is

$$MASE = mean(\frac{\hat{x_j} - x_j}{\frac{1}{L_{train}-1} \sum_{t=2}^{L_{train}} |x_t - x_{t-1}|}), \tag{12}$$

where L_{train} represents the size of training set. The denominator of MASE is the mean absolute error of the in-sample naive forecast.

We compare the proposed model with many classical and state-of-the-art models. These models are Persistence model (Makridakis et al., 2008), ARIMA (Contreras et al., 2003), SVR (Chen et al., 2004), MLP (Almalaq & Edwards, 2017), LSTM (Kong et al., 2017), hybrid EWT fuzzy cognitive map (FCM) learned with SVR (EWTFCMSVR) (Gao et al., 2020), Wavelet High-order FCM (WHFCM) (Yang & Liu, 2018), EWTRVFL (Gao et al., 2021) and RVFL (Ren et al., 2016). The previous one day, 48 data points are used as input for all the models as in Gao et al. (2021). To achieve a fair comparison, all models' hyper-parameters are optimized by cross-validation. The hyper-parameter search space is presented in Table 2. Some parameters are not involved in the optimization process and they are set to the same values for all the relevant models, which include the batch size equals to 32, learning rate equals to 0.001 and epochs equal to 200. For the edRVFL, the iterations of Bayesian optimization are set to 50 and there are 500 iterations in total for the edRVFL with 10 layers. For RVFL, EWTRVFL, DESN, edRVFL and EWTedRVFL, the iterations of Bayesian optimization are the same for fair comparison.

Tables 3 and 4 present the comparative results on each time series in terms of RMSE and MASE. The numbers in bold represent the corresponding model's performance is the best on the specific time series. The proposed models achieve outstanding performance on most datasets. Although EWTFCMSVR achieves the best performance on several datasets, there is a significant variance in its performance. In addition, the DESN also achieves outstanding accuracy comparing with the other models. It is challenging to distinguish these forecasting methods only based on the error metrics. Therefore, statistical tests are implemented to investigate the difference among all the models further. We first implement the Friedman test, and the p-value is smaller than 0.05, which represents that these forecasting models are significantly different on these twenty datasets. Therefore, a post-hoc Nemenyi test is utilized to distinguish them (Demšar, 2006). The critical distance of the Nemenyi test is calculated by:

$$CD = q_\alpha \sqrt{\frac{N_m(N_m+1)}{6N_d}} \tag{13}$$

where q_α is the critical value coming from the studentized range statistic divided by $\sqrt{2}$, N_m represents the number of models and N_d is the number of datasets (Demšar, 2006). Fig. 4 represents the Nemenyi test results. The figures show that the models that achieve excellent performance are at the top, whereas the model with the worst performance is at the bottom. The models within the vertical line indicate that their performance is not significantly different, although the upper models outperform the lower ones to some extent. The proposed two methods are the best ones. The Persistence method is the tailender because it learns nothing about the patterns. ARIMA is a penultimate because of its simple linear structure. The DESN outperforms all the other baseline models. According to the Nemenyi test results, we claim that deep architectures usually outperform the shallow ones. MLP and SVR achieve comparable performance. A pair-wise Nemenyi post-hoc

[1] https://aemo.com.au/.

Table 2
Hyper-parameter search space for the benchmark models.

Model	Parameter	Values
ARIMA	p/q	[1,2,3]
	d	0,1
SVR	C	$[2^{-10}, 2^{0}]$
	ϵ	[0.001,0.01,0.1]
	Radius	[0.001,0.01,0.1]
MLP	Hidden nodes	[2,4,8,16,32]
	Layers	[1,2,3]
	Optimizer	Adam
	Activation	*Relu*
LSTM	Hidden nodes	[2,4,8,16,32]
	Layers	[1,2,3]
	Optimizer	Adam
	Activation	*Tanh*
TCN	Filters	[2,4,8,16,32]
	Kernel size	2
	Optimizer	Adam
	Activation	*Relu*
EWTFCMSVR	Concepts	[2,6]
WHFCM	Regularization	$[2^{0}, 2^{-8}]$
DESN	Reservoir size	[20,200]
	Spectral radius	[0.95,1]
	Input scalings	[0,1]
	Leaky rate	[0.95,1]
	Transient	[1,49]
	Regularization parameter	[0,1]
RVFL-based models	Enhancement nodes	[20,200]
	Regularization parameter	$[0, 1]$
	Input scalings	[0,1]
	k	$[2, 3, 4]$
	Window	[48, 96, 144]

statistical comparison is further. Tables 5 and 6 summarize the pair-wise comparison results based on RMSE and MASE, respectively. The pair-wise comparison shows that the edRVFL significantly outperforms Persistence, ARIMA, MLP, SVR and EWTFCMSVR. The EWTedRVFL relatively outperforms LSTM, WHFCM, RVFL, DESN and EWTRVFL. The EWTedRVFL demonstrates a slight superiority over edRVFL. Finally, the comparison of raw data and the forecasts of EWTedRVFL is shown in Figs. 5 and 6. Due to page restrictions, we only present the forecasts of two states, the SA and NSW. It is clear to find that the proposed model predicts future trends, cycles, and fluctuations accurately.

3.3. Comparison with other signal processing techniques

This section compares the EWT with other signal processing techniques to demonstrate the suitability and superiority of EWT. These decomposition techniques are variational mode decomposition (VMD) (Dragomiretskiy & Zosso, 2013), discrete wavelet transform (DWT) (Shensa et al., 1992) and Fourier transform (Yu, Zhang, & Qin, 2018) for denoising. All the decomposition methods are implemented in the walk-forward fashion, which does not utilize future data points. Then these features are fed into the edRVFL with the same hyper-parameter optimization process. Four variants, EWTedRVFL, VMDedRVFL, DWT-edRVFL, and FedRVFL, are constructed based on the decomposition. These four variants are applied to forecast these sixteen time series with the same split of training, validation and test set.

Finally, the average performance on these sixteen datasets based on RMSE and MASE are obtained, normalized and summarized in Fig. 7. Fig. 7 shows that the edRVFL combined with EWT outperforms the other variants. The FedRVFL slightly outperforms the DWTedRVFL. Besides, the VMDedRVFL is the worst.

4. Ablation study

The proposed model has three critical components: the EWT decomposition, BOA hyper-parameter tuning, and edRVFL. An ablation study is conducted to investigate the significance of each element. We generate three variants to trace the performance of different layers. These three variants are summarized below:

- BOAedRVFL: The edRVFL utilizes BOA to optimize hyper-parameters, but does not combine with the EWT.
- GridEWTedRVFL: The edRVFL utilizes grid search for hyper-parameter optimization and is combined with the EWT.
- BOAEWTedRVFL: The edRVFL utilizes both the BOA and EWT, which is the proposed model.

The evaluation results are shown in Fig. 8. The vertical axis represents the errors, which are scaled to highlight the difference among the three variants. Based on the results, we find that the GridEWTedRVFL is the worst, no matter the number of layers. The main reason is that the whole edRVFL owns the same regularization strength for each readout layer, which significantly reduces the diversity. The superiority of BOAEWTedRVFL over GridEWTedRVFL demonstrates the necessity of BOA. The layer-wise BOA substantially boosts the performance due to the different configurations for each layer. The BOAedRVFL achieves the best performance when the number of layers is smaller than four, but the performance is not improved when the number of layers increases. A potential reason is that the randomized hidden layer cannot mine hierarchical features without the augmentation from signal decomposition. Without the EWT, the deep layer's features are only generated from the shallow layer's randomized features and the raw input. Therefore, the information for the deep representations is limited. Another reason is that the fully-connected hidden layers cannot extract features of different frequencies. However, the EWT decomposes the time series into sub-series of different frequencies and amplitudes, which makes up for the hidden layers' shortcomings. Finally, the proposed model outperforms the other variants when the networks become deep. Therefore, both the EWT and BOA are recommended to boost the performance of the edRVFL. This phenomenon demonstrates the superiority of deep networks over shallow ones of mining complex features from the input data.

5. Conclusion

This paper proposes a novel ensemble deep RVFL network combined with walk-forward decomposition for short-term load forecasting. The enhancement layers' weights are randomly initialized and kept fixed as in the shallow RVFL network. Only the output weights of each layer are computed in a closed form. Since the enhancement features are unsupervised and randomly initialized, the walk-forward EWT is implemented to augment the feature extraction. The walk-forward EWT is different from most literature, where the whole time series is decomposed at one time. Therefore, there is no data leakage problem during the decomposition process. Finally, the median of all forecasts is used as the final output. The experiments on twenty electricity loads demonstrate the superiority and efficiency of the proposed model. Moreover, the proposed model does not suffer from a colossal computation burden compared with other deep learning models which are fully trained.

There are several reasons for the superiority of the proposed model:

1. The edRVFL's structure benefits from ensemble learning. The edRVFL treats each enhancement layer as a single forecaster. Therefore, the ensemble multiple forecasters reduce the uncertainty of a single forecaster.
2. The clean raw data are fed into all enhancement layers to calibrate the random features' generation.
3. The output layer learns both the linear patterns from the direct link and nonlinear patterns from the enhancement features.

Table 3
Comparative results in terms of RMSE.

Location	Month	Persistence	ARIMA	MLP	SVR	LSTM	EWTFCMSVR	WHFCM	RVFL	DESN	EWTRVFL	edRVFL	EWTedRVFL
SA	Jan	72.887	55.638	54.087	66.969	59.617	70.211	**45.652**	53.170	52.020	55.812	48.021	47.209
	Apr	67.985	55.639	49.519	52.796	55.789	111.526	50.725	50.886	54.080	50.547	**49.104**	49.344
	Jul	96.180	61.595	56.444	50.478	46.080	**43.705**	45.918	49.303	50.445	49.647	47.553	47.208
	Oct	66.494	51.914	54.993	48.780	44.572	63.008	45.159	44.673	44.934	45.183	44.660	**44.411**
NSW	Jan	241.660	128.522	177.779	176.637	124.134	124.784	125.046	123.006	**119.353**	123.469	123.791	122.186
	Apr	170.539	107.029	82.397	123.231	109.891	262.121	**75.458**	79.778	102.188	83.560	76.256	76.579
	Jul	294.962	131.801	119.467	130.393	100.255	**85.548**	95.036	97.840	94.851	98.747	94.670	93.621
	Oct	179.376	97.906	96.813	147.033	105.038	99.516	85.903	85.131	90.042	84.850	**82.415**	82.716
VIC	Jan	166.027	107.452	105.513	476.514	160.840	**80.530**	96.099	112.781	123.474	105.947	99.655	100.624
	Apr	161.652	95.063	91.879	112.789	90.132	157.680	79.565	81.300	83.322	81.098	75.444	**75.245**
	Jul	202.388	100.130	73.957	76.369	**66.879**	234.387	77.278	70.122	71.584	70.356	68.484	67.820
	Oct	146.720	93.550	84.794	85.882	78.058	**68.096**	77.101	73.834	73.571	75.050	72.778	73.708
TAS	Jan	22.690	18.884	20.378	21.712	**17.709**	18.647	18.790	18.935	18.150	18.522	18.442	18.423
	Apr	29.511	20.464	25.939	**17.250**	18.122	17.538	20.026	17.428	17.526	17.293	17.412	17.294
	Jul	41.606	24.189	22.461	22.544	20.485	**20.126**	24.928	21.194	20.167	20.575	20.297	20.382
	Oct	30.881	21.364	20.147	19.440	19.322	20.087	20.885	20.044	**19.236**	19.858	20.077	19.890

Table 4
Comparative results in terms of MASE.

Location	Month	Persistence	ARIMA	MLP	SVR	LSTM	EWTFCMSVR	WHFCM	RVFL	DESN	EWTRVFL	edRVFL	EWTedRVFL
SA	Jan	1.2552	0.8463	0.8405	0.9083	0.8355	1.0406	0.7138	0.7649	0.7739	0.7984	0.6983	**0.6909**
	Apr	1.1203	0.8195	0.7581	0.7782	0.8278	1.7619	0.8120	0.7768	0.8182	0.7654	**0.7363**	0.7469
	Jul	1.1060	0.5701	0.5598	0.4610	0.4125	**0.4049**	0.4437	0.4861	0.4810	0.4850	0.4550	0.4525
	Oct	1.0056	0.7204	0.8209	0.6215	**0.6088**	0.8455	0.6309	0.6113	0.6190	0.6168	0.6094	0.6089
NSW	Jan	1.4312	0.5671	0.9933	0.8771	0.5749	0.5576	0.5445	0.5580	**0.5364**	0.5584	0.5570	0.5505
	Apr	1.0095	0.6026	0.4514	0.5571	0.5353	1.5863	0.4308	0.4345	0.5042	0.4523	**0.4100**	0.4159
	Jul	0.9287	0.3917	0.3415	0.3625	0.3018	**0.2551**	0.2842	0.2856	0.2859	0.2867	0.2729	0.2686
	Oct	1.0979	0.5425	0.5264	0.7185	0.5644	0.5590	0.4746	0.4657	0.4875	0.4599	**0.4438**	0.4461
VIC	Jan	1.3105	0.7993	0.8405	2.3222	1.0803	**0.6126**	0.7456	0.7800	0.8411	0.7473	0.7241	0.7203
	Apr	1.1833	0.6260	0.6363	0.7401	0.5785	0.9166	0.5284	0.5325	0.5480	0.5361	0.4946	**0.4926**
	Jul	1.0659	0.4864	0.3608	0.3698	0.3264	1.2698	0.3729	0.3366	0.3428	0.3383	0.3257	**0.3209**
	Oct	0.9891	0.5518	0.5154	0.5032	0.4693	**0.4141**	0.4647	0.4600	0.4489	0.4629	0.4504	0.4545
TAS	Jan	1.1101	0.8751	0.9565	1.0609	0.8581	0.8967	0.8819	0.9070	**0.8441**	0.8897	0.8873	0.8884
	Apr	1.0463	0.6983	0.9746	0.6081	0.6298	0.5870	0.6926	0.5890	0.5929	0.5852	0.5872	**0.5831**
	Jul	1.1317	0.6349	0.5926	0.5599	0.5358	0.5169	0.6721	0.5393	**0.5059**	0.5199	0.5084	0.5114
	Oct	1.0218	0.6730	0.6354	0.6162	0.6145	0.6295	0.6598	0.6267	**0.6070**	0.6262	0.6270	0.6219

Table 5
Pairwise comparisons using Nemenyi post-hoc test based on RMSE.

	Persistence	ARIMA	MLP	SVR	LSTM	EWTFCMSVR	WHFCM	RVFL	DESN	EWTRVFL	edRVFL	EWTedRVFL
edRVFL	0.001	0.001	0.009	0.001	0.442	0.126	0.693	0.816	0.900	0.900	1.000	0.900
EWTedRVFL	0.001	0.001	0.002	0.001	0.228	0.047	0.476	0.600	0.877	0.693	0.900	1.000

Table 6
Pairwise comparisons using Nemenyi post-hoc test based on MASE.

	Persistence	ARIMA	MLP	SVR	LSTM	EWTFCMSVR	WHFCM	RVFL	DESN	EWTRVFL	edRVFL	EWTedRVFL
edRVFL	0.001	0.001	0.001	0.001	0.281	0.097	0.569	0.723	0.900	0.754	1.000	0.900
EWTedRVFL	0.001	0.001	0.001	0.001	0.111	0.030	0.309	0.476	0.754	0.508	0.900	1.000

4. The walk-forward EWT is used as a feature engineering block to boost the accuracy further.

Although our model shows its superiority in these twenty datasets, there are still some limitations. For the walk-forward EWT process, whether to discard the highest frequency is an open problem. It is challenging to determine how valuable information is in the highest frequency component. Moreover, other learning techniques can be considered to further boost the performance, like incremental learning and semi-unsupervised learning.

CRediT authorship contribution statement

Ruobin Gao: Theoretical development, Empirical study, Literature review, Finishing the manuscript. **Liang Du:** Development of the proposed model, Revision of the manuscript. **Ponnuthurai Nagaratnam Suganthan:** Empirical study, Revision of the manuscript, Making suggestions on the comparison in the experiments. **Qin Zhou:** Revision of the manuscript. **Kum Fai Yuen:** Revision of the manuscript.

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.

References

Ali, M., Adnan, M., Tariq, M., & Poor, H. V. (2020). Load forecasting through estimated parametrized based fuzzy inference system in smart grids. *IEEE Transactions on Fuzzy Systems*.

Almalaq, A., & Edwards, G. (2017). A review of deep learning methods applied on load forecasting. In *2017 16th IEEE international conference on machine learning and applications* (pp. 511–516). IEEE.

Bergmeir, C., & Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, *191*, 192–213.
Casazza, P. G., et al. (2000). The art of frame theory. *Taiwanese Journal of Mathematics*, *4*(2), 129–201.
Chen, B.-J., Chang, M.-W., et al. (2004). Load forecasting using support vector machines: A study on EUNITE competition 2001. *IEEE Transactions on Power Systems*, *19*(4), 1821–1830.
Contreras, J., Espinola, R., Nogales, F. J., & Conejo, A. J. (2003). ARIMA models to predict next-day electricity prices. *IEEE Transactions on Power Systems*, *18*(3), 1014–1020.
Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*(Jan), 1–30.
Dragomiretskiy, K., & Zosso, D. (2013). Variational mode decomposition. *IEEE Transactions on Signal Processing*, *62*(3), 531–544.
Flandrin, P., Rilling, G., & Goncalves, P. (2004). Empirical mode decomposition as a filter bank. *IEEE Signal Processing Letters*, *11*(2), 112–114.
Gao, R., Du, L., & Yuen, K. F. (2020). Robust empirical wavelet fuzzy cognitive map for time series forecasting. *Engineering Applications of Artificial Intelligence*, *96*, Article 103978.
Gao, R., Du, L., Yuen, K. F., & Suganthan, P. N. (2021). Walk-forward empirical wavelet random vector functional link for time series forecasting. *Applied Soft Computing*, *108*, Article 107450.
Gao, R., & Duru, O. (2020). Parsimonious fuzzy time series modelling. *Expert Systems with Applications*, *156*, Article 113447.
Ghelardoni, L., Ghio, A., & Anguita, D. (2013). Energy load forecasting using empirical mode decomposition and support vector regression. *IEEE Transactions on Smart Grid*, *4*(1), 549–556.
Gilles, J. (2013). Empirical wavelet transform. *IEEE Transactions on Signal Processing*, *61*(16), 3999–4010.
Hafeez, G., Alimgeer, K. S., & Khan, I. (2020). Electric load forecasting based on deep learning and optimized by heuristic algorithm in smart grid. *Applied Energy*, *269*, Article 114915.
Heydari, A., Nezhad, M. M., Pirshayan, E., Garcia, D. A., Keynia, F., & De Santoli, L. (2020). Short-term electricity price and load forecasting in isolated power grids based on composite neural network and gravitational search optimization algorithm. *Applied Energy*, *277*, Article 115503.
Huang, Y., & Deng, Y. (2021). A new crude oil price forecasting model based on variational mode decomposition. *Knowledge-Based Systems*, *213*, Article 106669.
Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688.
Jalali, S. M. J., Ahmadian, S., Khosravi, A., Shafie-khah, M., Nahavandi, S., & Catalao, J. P. (2021). A novel evolutionary-based deep convolutional neural network model for intelligent load forecasting. *IEEE Transactions on Industrial Informatics*.
Kong, W., Dong, Z. Y., Jia, Y., Hill, D. J., Xu, Y., & Zhang, Y. (2017). Short-term residential load forecasting based on LSTM recurrent neural network. *IEEE Transactions on Smart Grid*, *10*(1), 841–851.
Liang, M., Liu, R. W., Li, S., Xiao, Z., Liu, X., & Lu, F. (2021). An unsupervised learning method with convolutional auto-encoder for vessel trajectory similarity computation. *Ocean Engineering*, *225*, Article 108803.
Liang, M., Zhan, Y., & Liu, R. W. (2021). MVFFNet: Multi-view feature fusion network for imbalanced ship classification. *Pattern Recognition Letters*, *151*, 26–32.
Liu, R. W., Liang, M., Nie, J., Yuan, Y., Xiong, Z., Yu, H., et al. (2022). STMGCN: Mobile edge computing-empowered vessel trajectory prediction using spatio-temporal multi-graph convolutional network. *IEEE Transactions on Industrial Informatics*, http://dx.doi.org/10.1109/TII.2022.3165886, (in press).
Makridakis, S., Wheelwright, S. C., & Hyndman, R. J. (2008). *Forecasting methods and applications*. John wiley & sons.
Needell, D., Nelson, A. A., Saab, R., & Salanevich, P. (2020). Random vector functional link networks for function approximation on manifolds. arXiv preprint arXiv:2007.15776.
Pelikan, M., Goldberg, D. E., Cantú-Paz, E., et al. (1999). BOA: The Bayesian optimization algorithm. In *Proceedings of the genetic and evolutionary computation conference, vol. 1* GECCO-99, (pp. 525–532). Citeseer.
Qiu, X., Ren, Y., Suganthan, P. N., & Amaratunga, G. A. (2017). Empirical mode decomposition based ensemble deep learning for load demand time series forecasting. *Applied Soft Computing*, *54*, 246–255.
Qiu, X., Suganthan, P. N., & Amaratunga, G. A. (2018). Ensemble incremental learning random vector functional link network for short-term electric load forecasting. *Knowledge-Based Systems*, *145*, 182–196.
Qiu, X., Zhang, L., Ren, Y., Suganthan, P. N., & Amaratunga, G. (2014). Ensemble deep learning for regression and time series forecasting. In *2014 IEEE symposium on computational intelligence in ensemble learning* (pp. 1–6). IEEE.
Qiu, X., Zhang, L., Suganthan, P. N., & Amaratunga, G. A. (2017). Oblique random forest ensemble via least square estimation for time series forecasting. *Information Sciences*, *420*, 249–262.
Ren, Y., Suganthan, P., & Srikanth, N. (2014a). A comparative study of empirical mode decomposition-based short-term wind speed forecasting methods. *IEEE Transactions on Sustainable Energy*, *6*(1), 236–244.
Ren, Y., Suganthan, P. N., & Srikanth, N. (2014b). A novel empirical mode decomposition with support vector regression for wind speed forecasting. *IEEE Transactions on Neural Networks and Learning Systems*, *27*(8), 1793–1798.
Ren, Y., Suganthan, P. N., Srikanth, N., & Amaratunga, G. (2016). Random vector functional link network for short-term electricity load demand forecasting. *Information Sciences*, *367*, 1078–1093.
Saunders, C., Gammerman, A., & Vovk, V. (1998). Ridge regression learning algorithm in dual variables. In *Proceedings of the 15th international conference on machine learning*.
Shensa, M. J., et al. (1992). The discrete wavelet transform: Wedding the a trous and mallat algorithms. *IEEE Transactions on Signal Processing*, *40*(10), 2464–2482.
Shi, Q., Katuwal, R., Suganthan, P., & Tanveer, M. (2021). Random vector functional link neural network based ensemble deep learning. *Pattern Recognition*, *117*, Article 107978.
Shi, H., Xu, M., & Li, R. (2017). Deep learning for household load forecasting—A novel pooling deep RNN. *IEEE Transactions on Smart Grid*, *9*(5), 5271–5280.
Spencer, J. (1994). *Ten lectures on the probabilistic method, vol. 64*. SIAM.
Taylor, J. W. (2011). Short-term load forecasting with exponentially weighted methods. *IEEE Transactions on Power Systems*, *27*(1), 458–464.
Timmermann, A. (2006). Forecast combinations. In *Handbook of economic forecasting, vol. 1* (pp. 135–196). Elsevier.
Yang, A., Li, W., & Yang, X. (2019). Short-term electricity load forecasting based on feature selection and least squares support vector machines. *Knowledge-Based Systems*, *163*, 159–173.
Yang, S., & Liu, J. (2018). Time-series forecasting based on high-order fuzzy cognitive maps and wavelet transform. *IEEE Transactions on Fuzzy Systems*, *26*(6), 3391–3402.
Yu, X., Zhang, X., & Qin, H. (2018). A data-driven model based on Fourier transform and support vector regression for monthly reservoir inflow forecasting. *Journal of Hydro-Environment Research*, *18*, 12–24.
Zhang, Z., & Hong, W.-C. (2021). Application of variational mode decomposition and chaotic grey wolf optimizer with support vector regression for forecasting electric loads. *Knowledge-Based Systems*, *228*, Article 107297.
Zhang, L., & Suganthan, P. N. (2016a). A comprehensive evaluation of random vector functional link networks. *Information Sciences*, *367*, 1094–1105.
Zhang, L., & Suganthan, P. N. (2016b). A survey of randomized algorithms for training neural networks. *Information Sciences*, *364*, 146–155.