symmetry

MDPI

Article

An Unsupervised Multi-Dimensional Representation Learning Model for Short-Term Electrical Load Forecasting

Weihua Bai [1,*], Jiaxian Zhu [1,*], Jialing Zhao [1], Wenwei Cai [1] and Keqin Li [2,*]

[1] School of Computer Science, Zhaoqing University, Zhaoqing 526061, China
[2] Department of Computer Science, State University of New York, New Paltz, NY 12561, USA
[*] Correspondence: baiweihua@zqu.edu.cn (W.B.); zhujiaxian@zqu.edu.cn (J.Z.); lik@newpaltz.edu (K.L.); Tel.: +86-139-2988-1184 (W.B.); +86-138-2261-6184 (J.Z.)

Abstract: The intelligent electrical power system is a comprehensive symmetrical system that controls the power supply and power consumption. As a basis for intelligent power supply control, load demand forecasting in power system operation management has attracted considerable research attention in energy management. In this study, we proposed a novel unsupervised multi-dimensional feature learning forecasting model, named MultiDBN-T, based on a deep belief network and transformer encoder to accurately forecast short-term power load demand and implement power generation planning and scheduling. In the model, the first layer (pre-DBN), based on a deep belief network, was designed to perform unsupervised multi-feature extraction feature learning on the data, and strongly coupled features between multiple independent observable variables were obtained. Next, the encoder layer (D-TEncoder), based on multi-head self-attention, was used to learn the coupled features between various locations, times, or time periods in historical data. The feature embedding of the original multivariate data was performed after the hidden variable relationship was determined. Finally, short-term power load forecasting was conducted. Experimental comparison and analysis of various sequence learning algorithms revealed that the forecasting results of MultiDBN-T were the best, and its mean absolute percentage error and root mean square error were improved by more than 40% on average compared with other algorithms. The effectiveness and accuracy of the model were experimentally verified.

Keywords: feature learning; feature embed encoding; multivariate; short-term power load forecasting; unsupervised

Citation: Bai, W.; Zhu, J.; Zhao, J.; Cai, W.; Li, K. An Unsupervised Multi-Dimensional Representation Learning Model for Short-Term Electrical Load Forecasting. *Symmetry* **2022**, *14*, 1999. https://doi.org/10.3390/sym14101999

Academic Editor: Angelo Freni

Received: 23 August 2022
Accepted: 20 September 2022
Published: 23 September 2022

1. Introduction and Motivation

The intelligent electrical power system, as a comprehensive symmetrical system, controls the power supply and power consumption. Power load forecasting is critical for achieving power analysis and intelligent power supply control and plays a crucial role in power system management. Energy management has become a critical research topic in electric power. Inaccurate load forecasting can result in overload or congestion of power dispatch, which can affect the security of power systems and the implementation of power production plans. Short-term power demand forecasting is a crucial aspect of the load forecasting model and is indispensable in modern power supply systems. Demands for electricity or energy load are a result of the interaction and coupling of various observable factors (such as weather conditions, electricity production, electricity consumption, electricity price fluctuations) or hidden factors (such as different time periods, different regions, and different industrial structures). Demand is key for integrating production, control, and dispatch in an energy system. The predictability of power loads can be improved by considering various explicit and implicit causes, the diversity of various demands, and the complexity of various loads in a forecasting model, as well as finding a strong coupling relationship between multi-dimensional relationships of various factors in the observable data, analyzing all coupling relationships, and performing feature analysis and extraction to optimize the accuracy of the load forecasting model.

Conventional load forecasting methods are based on a single factor for independently forecasting various types of loads. These methods include fitting- or regression-based methods, such as Kalman filtering, fuzzy linear regression, and support vector machine (SVM), as well as neural network-based methods, such as LSTM, XGBoost, gradient boosting regression tree (GBRT), and artificial neural networks (ANN) [1–4]. However, for the processing of multi-dimensional power load time-series data with both multi-dimensional features and long-term time feature dependencies, the aforementioned methods exhibit certain limitations. One study demonstrated the importance of relevant dataset structure for creating optimal logic mining. The authors used a Hopfield neural network to learn and obtain the corresponding optimal induced logical rule in the learning datasets. Finally, the validity of the optimal logic rules for mining associated datasets for analyzing data was verified by experiments [5–7].

When analyzing time-series data, a single historical data series cannot truly reflect the multiple influences of external factors on the generation of the series data and the complex spatiotemporal dependencies between related datasets. On the one hand, the existing analytical frameworks usually utilize given spatiotemporal correlations with incomplete information structures for modeling, which would limit the effective learning of predictive models on spatiotemporal dependencies. On the other hand, for nonlinear complex spatiotemporal data, the existing methods cannot effectively integrate the inherent spatiotemporal correlation, local feature correlation, and global multi-dimensional dependence of the dataset. The motivation of this paper is to propose a new unsupervised multi-dimensional representation learning framework based on data-driven deep feature extraction and a multi-scale self-attention network, which are more conducive to studying the dynamics reflected by different granularities in space and time. The application of this framework can learn and mine the hidden logical rules of spatial and temporal correlation in multi-dimensional datasets, encode different time granularities at different levels, and effectively integrate local and global spatial and temporal correlations. Through this process, the spatiotemporal dependences of the fusion of internal and external factors are fully captured. The proposed unsupervised multi-dimensional representation learning framework can be applied to short-term electricity demand forecasting, traffic flow analysis, and service task sequence analysis.

To address multi-dimensional power load time-series data and improve the accuracy and reliability of power load forecasting, a novel unsupervised multi-dimensional feature learning forecasting model is proposed. First, a deep belief network (DBN) was used to perform feature learning of unsupervised multi-feature extraction on the data, and the features obtained after determining strong coupling relationships between independent factors were embedded. Next, the multi-head self-attention mechanism in the transformer was used to learn the locations or times of strongly coupled features and internal hidden features of the sequence to obtain the feature encoding of the multi-related feature information at each time point in the sequence. Finally, the spliced code regression learned through a neural network with a hidden layer. The objective of the power forecasting model was to forecast the power load demand in the next 12–24 h by analyzing a segment of multi-dimensional historical data (such as the historical data for 48 or 72 h before forecasting). The main contributions of this paper are as follows: (1) a novel data processing and forecasting model based on unsupervised multi-feature feature learning is proposed; (2) the model can learn the strongly coupled features among independent multi-factors and can learn the strongly coupled features and hidden factors between locations or time points in the sequence; (3) experimental results on multi-dimensional power load sequence data reveal that the model outperformed the latest forecasting models in processing multi-dimensional time-series data.

2. Related Work

Numerous studies have been conducted on the analysis of time-series data using conventional methods, such as KLF, SVR, ARIMA, XGBoost, and LSTM [8–10]. In these methods, the single-mode data are used as the original sequence as the analysis target, and subsequent time points are forecasted through fitting or regression to the historical data. To perform forecasting on complex sequence data, based on the classical methods, a LSTM forecasting model with principal component correlation analysis (PCA) [11], an SVM forecasting model based on the fruit fly optimization algorithm (FFOA) [12], improved ARIMA models, SARIMA [13,14], and other models have been proposed for short-term natural gas load forecasting. The SARIMA model transforms nonstationary data into stationary data by using lagged mean values of power load time-series data. The analysis of seasonality in the method can be observed by using the autocorrelation function and the partial autocorrelation function. Because of the nonlinearity of the time-series data, single linear regression, and multiple linear regression methods based on statistical regression variables and methods, the time changes and nonlinear power load patterns in the series cannot be captured; these methods are not suitable for processing such time-series data [15,16]. A novel particle swarm optimization (PSO) two-step method is presented in [17] for increasing the performance of short-term load forecasting (STLF).

The development of machine learning and neural network technology provides novel methods for short-term power load forecasting. Compared with forecasting models based on statistical regression and dimensionality reduction, machine learning methods are remarkably accurate in forecasting nonlinear power loads and peaks, and their algorithms improve the performance of power load demand forecasting models [18]. For weather factors in historical data, an ANN model deeply learns the effect of random weather characteristics on power load and forecasts the power load in a certain period of time in the future [19,20]. Various K-nearest neighbor (KNN)-based machine learning algorithms perform forecasting by learning to process the time-varying features of nonlinear time-series data by learning the K nearest instances in historical data. In the algorithms, using the most similar instances of detected K, the target points closest to these K instances were obtained through local interpolation. However, these methods exhibit considerable drawbacks for high-dimensional time-series data—the lack of hyperparameters can lead to overfitting of a forecasting model, whereas when dealing with high-dimensional data, the curse of dimensionality can occur. These methods cannot handle high-dimensional time-series data [21,22]. In [23], a novel forecasting model based on pooling was proposed, in which LSTM was used to train and extract the features encoded with one-hot encoding. This method mitigated the overfitting problem and improved forecasting accuracy. In [24], a combined model based on EOBL-CSSA-LSSVM, which contains the machine learning algorithms, the swarm intelligence optimization algorithms, and data pre-processing in the prediction model, is proposed for power load forecasting.

Because time-series data are becoming increasingly complex, researchers have proposed novel methods for analyzing multi-dimensional power load time-series data. To obtain the global temporal correlation characteristics (such as overall trend) in historical data, a FEDformer model combined with a transformer was proposed, and seasonal trend decomposition methods were used to learn global change trends in time-series to acquire detailed feature structures in the series [25]. Because of coupling characteristics of deep factors in time-series, convolutional neural network (CNN), gated recurrent unit, and GBRT have been integrated to develop a multi-dimensional energy load forecasting model. A DBN-based multi-task regression neural network model was proposed. The model acquired hidden features in the multi-energy load time-series through training and learning, and finally, the energy load demand was forecasted by multi-prosumers [26,27]. By obtaining the internal correlation of the relevant factors in the time-series data in the multiple energy consumption-load joint forecasting, Wang et al. proposed a joint multi-energy load forecasting model composed of a single encoder and multiple decoders, namely multi-decoder transformer (MultiDeT). In this model, a single encoder was used to uniformly encode the input data, and the results were obtained after using the decoders of the multi-prediction task to learn and train the shared data encoding for forecasting various energy consumption loads [28]. To analyze time-series data, a hybrid ensemble deep learning model combining MLP, CNN, LSTM, and CNN-LSTM was proposed to improve the forecasting performance in [29].

The analysis and processing of multi-dimensional sequence data have attracted considerable research interest. These sequence data exhibit nonlinearity, multi-dimensionality, time-series correlation, and hidden coupling. For multi-dimensional data short-term power load forecasting, the various methods proposed exhibit certain limitations and insufficient capabilities in addressing time-series data that have (1) nonlinearity in processing, (2) implicit coupling and correlation characteristics, (3) high-dimensional and mutually independent characteristics, and (4) multiple coupling and correlations (such as global trend and time-series periodic correlation). Thus, forecasting performance should be improved.

3. Unsupervised MultiDBN-T

To investigate the internal correlation characteristics of the data, the strong coupling correlations of various features in space and time can be obtained by learning historical data. The trend of the data for the forecasting target can be accurately determined. Representation learning, also known as feature learning, is used in machine learning for encoding original data. The transformed encoding retains features that can be easily identified, rendering it easy for various machine learning algorithms to learn and apply data processing procedures [30,31]. To extract the coupled features between dimensions, the temporal coupled features within the dimensions, and the overall change trend features in multi-dimensional time-series data, we propose a novel unsupervised MultiDBN-T. This model forecasts power load demand in the next 24 h by learning and training power load historical data with multi-dimensional data features.

3.1. MultiDBN-T Architecture

The architecture of MultiDBN-T model is displayed in Figure 1, and the detailed composition is shown in Figure 2. The forecasting model is primarily composed of three parts. (1) Pre-DBN: A single-decoder deep belief network used for self-supervised/unsupervised pre-training. This layer is a pre-training layer that solves the training problem of deep neural networks by stacking multiple layers of RBMs. It solves the optimization problem of deep neural network training with high-dimensional correlated but different datasets, as it can make the whole network with better initial weights achieve the optimal solution by fine-tuning the layer-by-layer training. (2) DBN-transformer encoder (D-TEncoder): Transformer encoder based on DBN fine-tuning. After pre-training, by combining with the multi-head self-attention mechanism, it further constructs spatiotemporal features with the codes of feature compression to mine the effective logical rules inherent in the dataset for improving the effectiveness of training. (3) L-Decoder: Fully connected hidden layer decoder. It optimizes the variables of the model by computing the loss function.

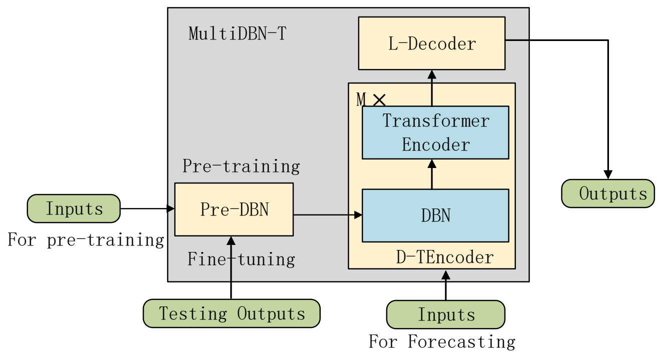


Figure 1. The overall structure of the MultiDBN-T model and its training process.

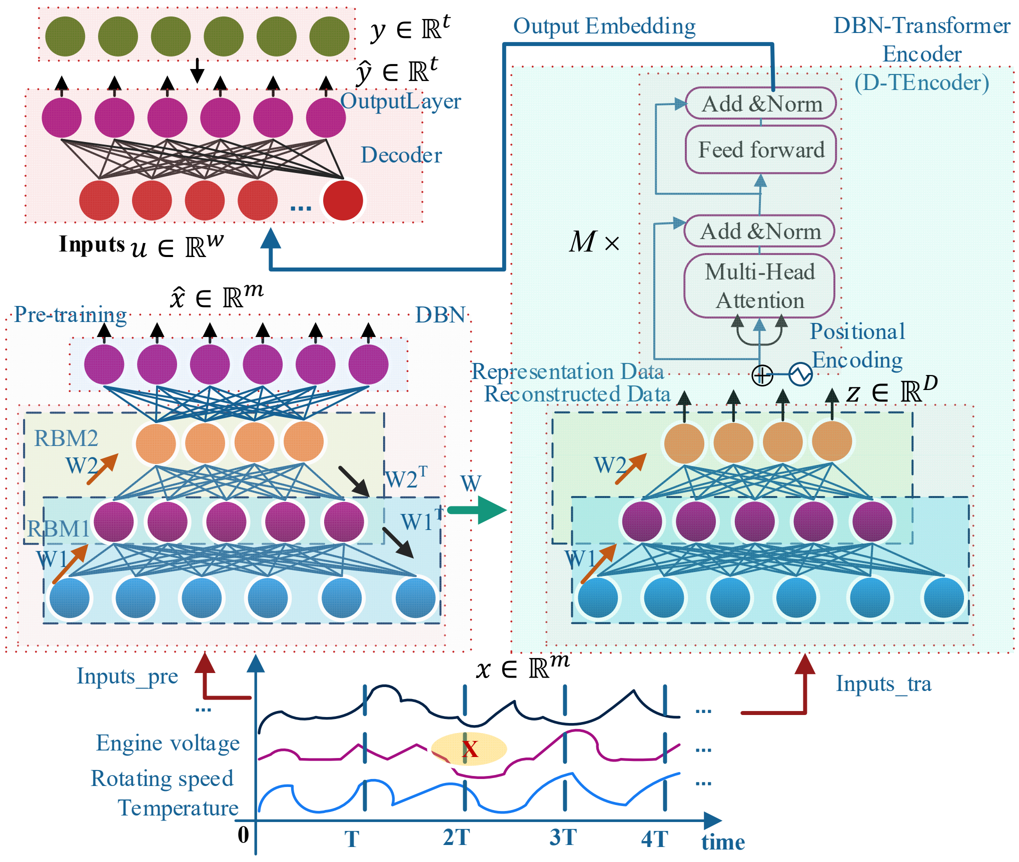


Figure 2. Architecture of the MultiDBN-T model.

The overall structure of the MultiDBN-T model and its training process are shown in Figure 1. The training process steps are as follows. (1) It pre-trains with pre-DBN using the training dataset, and then uses the corresponding data for fine-tuning to obtain the optimized DBN. (2) The training dataset will be directly inputted into the tuned DBN in the D-TEncoder, and be further encoded by the transformer encoder. (3) Finally, the L-Decoder is used to decode and calculate the loss function, and the final prediction model is obtained through the back-propagation gradient calculation and optimization of the neural network.

As displayed in Figure 2, MultiDBN-T uses dual encoders: pre-DBN (based on the DBN encoding layer) and DBN-transformer encoder (based on the transformer-based encoding layer). These two layers of encoders are the core layers of feature learning in the forecasting model. The purpose of pre-DBN is to learn the coupling correlation between dimensions of multi-dimensional data in the self-supervised/unsupervised process, and it can be used for data pre-training. Thus, the initial values of the network weights of the two hidden layers at the bottom of the DBN-transformer encoder were obtained to improve training accuracy. The purpose of the DBN-transformer encoder is to obtain the temporal coupled features in the historical data, such as periodicity, globality, and correlation of external factors, and obtain deeper hidden features in the sequence through the multi-head attention mechanism.

3.2. Pre-DBN

Multi-dimensional power load time-series data present a complex nonlinear data distribution; however, oftentimes, only limited local features, such as weather condition data, productivity consumption data, different regions, and changes in electricity prices, can be observed. The data on these dimensions can considerably affect the future power load demand in a certain area. Therefore, the data are inevitably affected by multiple factors and contain noise. The distribution is modeled to obtain a representation of internal features hidden between the observable variables.

As displayed in Figure 2, pre-DBN consists of three layers of RBM and one fully connected output layer. Because more than three layers of RBM are used to form the encoding part of pre-DBN, as the network deepens, the features extracted by each layer become distinctive; that is, the coupled features of the data between the dimensions become stronger. However, some hidden features may be lost, which is not conducive to the encoding of the transformer.

Figure 3 displays the network structure of a one-layer RBM. Let the random vector observable in the pre-DBN input layer be $\upsilon \in \mathbb{R}^m$. Thus, in the m-dimensional power load time-series data, at the time point t_i, sample vector $\mathrm{X} \in \mathbb{R}^m$, and the first RBM hidden random vector $h^{(0)} \in \mathbb{R}^d$, then $\upsilon^{(1)} = h^{(0)}$. The second is $h^{(1)} \in \mathbb{R}^w$, $\upsilon^{(2)} = h^{(1)}$; the third is $h^{(2)} \in \mathbb{R}^k$; and finally, the output layer is $\hat{\mathrm{X}} \in \mathbb{R}^m$.

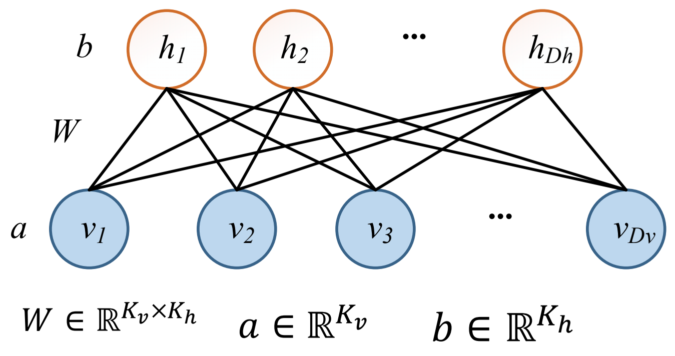


Figure 3. An RBM in pre-DBN.

In each RBM, the dimension of an observable variable is D_v, and the dimension of a hidden variable is D_h; that is, $v \in \mathbb{R}^{D_v}$ and $h \in \mathbb{R}^{\mathrm{D}_h}$. The bias of an observable variable in the visible layer is $a \in \mathbb{R}^{D_v}$, and the bias of a hidden layer variable is $b \in \mathbb{R}^{\mathrm{D}_h}$. The weight matrix between the two layers is $W \in \mathbb{R}^{\mathrm{D}_v \times \mathrm{D}_h}$. a_i is the bias of v_i, b_j is the bias of h_j, and W_{ij} is the weight of the edge between v_i and h_j.

Because the observable variables and hidden variables of the time-series data processed in this study satisfy the Gaussian–Gaussian RBM (GG-RBM), studies [32,33] have revealed that the energy function of GG-RBM $E(v,h)$ can be defined as follows:

$$E(v,h) = \sum_{i \in v} \frac{(v_i - a_i)^2}{2\sigma_i^2} + \sum_{j \in h} \frac{(h_i - b_i)^2}{2\sigma_j^2} - \sum_{i \in v} \sum_{j \in h} \frac{v_i}{\sigma_i} \frac{h_j}{\sigma_j} W_{ij} \quad (1)$$

The observable variable v_i and the hidden variable h_j are calculated by the following:

$$v_i = a_i + \sigma_i \sum_{j \in h} \frac{h_j}{\sigma_j} W_{ij} \tag{2}$$

$$h_j = b_i + \sigma_j \sum_{i \in v} \frac{v_i}{\sigma_i} W_{ij} \tag{3}$$

where σ_j and σ_i represent the standard deviations of the noise for v_i and h_j satisfying Gaussian distribution, respectively. In practice, the data are normalized so that both σ_i and σ_j are equal to 1. When solving the training, σ_i can be obtained from raw input data $\sigma_i = cov(v)_i$; that is, σ_j is based on the two-layer relationship in Figure 3, and can be calculated by $\sigma_j = \sqrt{\sum_{i \in v} \sigma_i W_{ij}^2}$.

As displayed in Figure 4, in the solving process of the CK-*k* algorithm, for the original data $v^{(0)}$ of a given observable variable, the hidden vector $h^{(0)}$ is obtained by sampling the conditional probability $p(h|v^{(0)})$. Next, $h^{(0)}$ is used to calculate the conditional probability $p(v|h^{(0)})$ and to obtain $v^{(1)}$ by sampling. The process is repeated to obtain $p(h|v^{(1)}) \to h^{(1)}$ and $p(v|h^{(1)}) \to v^{(2)}$. According to the aforementioned process, Equations (1)–(3) and the calculation formulas of σ_i and σ_j, the equations for updating the parameters W, a, and b for GG-RBM are follows:

$$W_{ij}^{(l+1)} = W_{ij}^{(l)} + \alpha(\frac{v_i}{\sigma_i}\frac{h_j}{\sigma_j}_{data} - \frac{v_i}{\sigma_i}\frac{h_j}{\sigma_j}_{model}) \tag{4}$$

$$a_i^{(l+1)} = a_i^{(l)} + \alpha(\frac{v_i}{\sigma_i}_{data} - \frac{v_i}{\sigma_i}_{model}) \tag{5}$$

$$a_i^{(l+1)} = a_i^{(l)} + \alpha(\frac{v_i}{\sigma_i}_{data} - \frac{v_i}{\sigma_i}_{model}) \tag{6}$$

where $\alpha > 0$ is the learning rate of the model, and $<\cdot>_{model}$ represents the values of $v^{(l)}$ and $h^{(l)}$ in the first l rounds of sampling. Thus, the parameters of the L-layers of GG-RBM in the pre-DBN are obtained by $\theta_{ggr-k} = \{W_{ggr\_k}, a_{ggr\_k}, b_{ggr\_k}\}, k = 1,2, \ldots, L$.

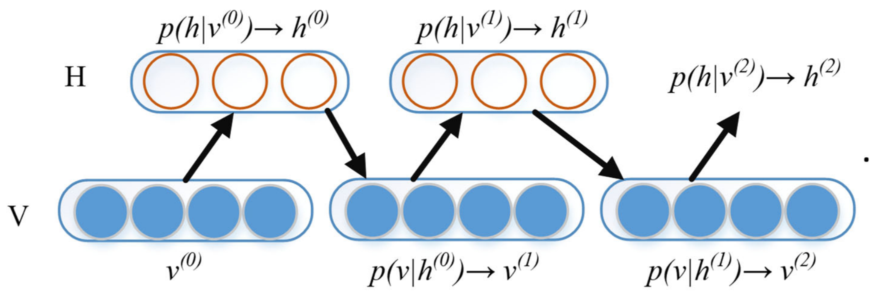

H
V
$p(h|v^{(0)}) \to h^{(0)}$
$p(h|v^{(1)}) \to h^{(1)}$
$p(h|v^{(2)}) \to h^{(2)}$
…
$v^{(0)}$
$p(v|h^{(0)}) \to v^{(1)}$
$p(v|h^{(1)}) \to v^{(2)}$

Figure 4. Solving process of the CK-k algorithm.

Here, pre-DBN is used as a pre-training model. The last layer is a fully connected hidden layer as an output layer. The weights and biases of the corresponding neural network of each layer are obtained by training through the BP algorithm. The training process is as follows:

(1) For each layer of GG-RBM according to Equations (1)–(6), the CD-k algorithm is used to perform layer-by-layer pre-training from the bottom to the top and to obtain the parameters of each layer $\theta_{ggr} = \{W_{ggr}, a_{ggr}, b_{ggr}\}$. The equation for the iterative calculation for the training is as follows:

$$\theta_{ggr-k}^{(e+1)} = \theta_{ggr-k}^{(e)} + \alpha \Delta \theta_{ggr-k}^{(e)}, e = 1,2,\ldots,itera; k = 1,2,\ldots,L \tag{7}$$

where α is the learning rate, e is the number of iterations of GG-RBM for each layer, and k is the first k-th layer of GG-RBM. $\Delta W_{ij} = \frac{v_i h_j}{\sigma_i \sigma_j}_{data} - \frac{v_i h_j}{\sigma_i \sigma_j}_{model}$, $\Delta a_i = \frac{v_i}{\sigma_i}_{data} - \frac{v_i}{\sigma_i}_{model}$, and $\Delta b_i = \frac{h_j}{\sigma_j}_{data} - \frac{h_j}{\sigma_j}_{model}$.

(2) After the pre-training of GG-RBM is completed, the last fully connected hidden layer of pre-DBN is used as the output. The network structure of GG-RBM is converted into the corresponding neural network structure. The network parameters are trained and fine-tuned again with the deep neural network. The initialization parameters of the corresponding network layers are $\theta_{pre-DBN} = \left\{\{W_{ggr}, b_{ggr}\}_l\right\}, l = 1,2,\ldots L$, where z_p is the output of the last corresponding GG-RBM hidden variable layer. The output of the pre-DBN is $\hat{X} \in \mathbb{R}^m$. The output and the loss function of the model are defined as follows:

$$\hat{x}_t = W_p z_p + b_p \tag{8}$$

$$\mathcal{L}_{MSE} = \frac{1}{|P| \times M} \sum_{t \in P} \sum_{i \in M} (\hat{x}(t,i) - x(t,i))^2 \tag{9}$$

where $|P|$ is the number of samples in the pre-training set, $(t,i) \in X$ represents the i-th dimension data of the sample input at time t, and M is the dimension of the time-series data (m observable variables).

(3) The BP algorithm is applied to the pre-DBN and trains with Adam as the optimizer to obtain the network parameters of the pre-DBN: $\theta_{pre-DBN}^{in} = \left\{\{W_{ggr}, b_{ggr}\}_l^{in}\right\}, l = 1,2,\ldots L$. The parameters are the initialization network parameters of the corresponding L-layer neural network in the D-TEncoder encoder.

3.3. D-TEncoder

The multi-head self-attention module of the transformer is applied to re-encode the encoding output from the pre-DBN and capture the interaction information in space and time in multiple different projection spaces. Let the output code of pre-DBN be $z_{1:T} \in \mathbb{R}^{D \times T}$, which is the input sequence of D-TEncoder.

Let D-TEncoder's initial input sequence be $H^{(0)}$, and add the position encoding learnable parameter W_{pos}: $z_t + W_{pos_t}$; then, we have following expression:

$$H^{(0)} = \{z_1 + W_{pos_1}, z_2 + W_{pos_2}, \ldots, z_T + W_{pos_T}\} \tag{10}$$

where $W_{pos_t} \in \mathbb{R}^D$.

Figure 5 displays a multi-head self-attention encoder. During encoding, through multiple learnable matrices W_q^n, W_k^n, W_v^n, and n, the self-attention model is applied in N projection spaces. The query, keys, and values in the projection spaces can be calculated as follows:

$$MultiHead(H) = W_0[head_1; \ldots, head_N]$$

$$head_n = Attention(Q_n, K_n, V_n) = V_n softmax(\frac{{K_n}^T Q_n}{\sqrt{d_k}})$$

$$\begin{cases} Q_n = W_q^{(n)} H \\ K_n = W_k^{(n)} H \quad \forall n \in \{1,2,\ldots,N\} \\ V_n = W_v^{(n)} H \end{cases} \tag{11}$$

where d_k is the dimension of the input matrix Q_n and V_n, d_v is the dimension of the column vector of V_n, $W_0 \in \mathbb{R}^{D\times Nd_v}$ is the output projection matrix, $W_q^{(n)} \in \mathbb{R}^{D\times d_k}$, $W_k^{(n)} \in \mathbb{R}^{D\times d_k}$, and $W_v^{(n)} \in \mathbb{R}^{D\times d_v}$.

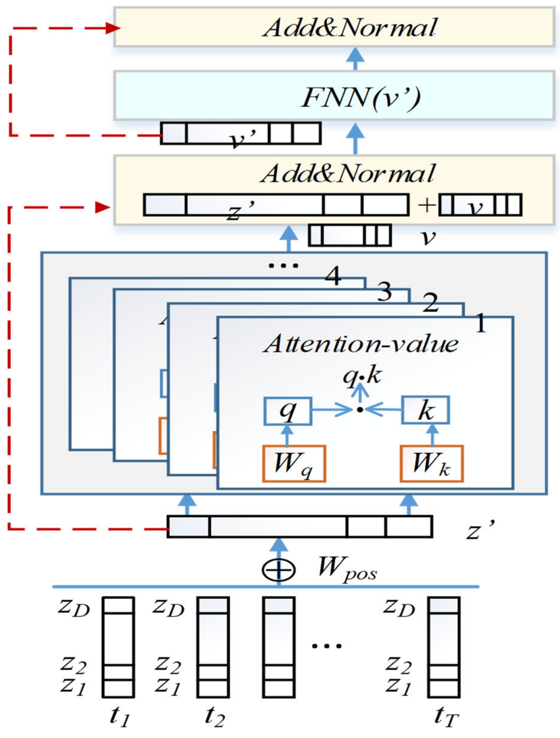


Figure 5. Transformer encoder.

The MultiDBN-T model in Figure 1 reveals that a total of M transformer encoders are set up, and each encoding layer is calculated by a multi-head self-attention module and a nonlinear $\mathrm{FNN}(\cdot)$ for each position feedforward neural network. In Figure 5, in the layer calculation in each encoder, the residuals (time-series data z' and v'' in Figure 5) are connected to the output of the previous layer, and the layer normalization is then performed. Here, $\mathrm{FNN}(\cdot)$ is a fully connected layer with two layers connected with ReLu as the activation function, which is defined as follows:

$$FNN(v') = W_{2FNN}ReLu(W_{1FNN}v' + b_{1FNN}) + b_{2FNN} \tag{12}$$

where $v' \in Z^{(l)}$ is the vector at each position in the input sequence of the previous layer, W_{1FNN} and W_{2FNN} are the weight matrices of the two-layer neural network, and b_{1FNN} and b_{2FNN} are the bias for the two layers of the network, respectively. These are all learnable network parameters. The connection weights of each layer in the encoder D-TEncoder are dynamically calculated by the self-attention mechanism.

3.4. L-Decoder

Through the double encoding of pre-DBN and D-TEncoder, the input multi-dimensional power load time-series data are reconstructed through feature learning. The reconstructed encoding is used as the input of the last fully connected layer for training. Let D-TEncoder represent the reconstructed encoded vector $u_t \in \mathbb{R}^w$ at time t, and the time window width be $\mathcal{T}$. All the codes are then concatenated into vector $\overline{\mathrm{U}} \in \mathbb{R}^{w\mathcal{T}}$ as a linear layer with network parameters $W_O \in \mathbb{R}^{\tau\times w\mathcal{T}}$ and $b_O \in \mathbb{R}^{\tau}$, where τ is the temporal window width of the output forecasting. The output of the L-Decoder is defined as follows:

$$\hat{y} = W_O \bar{\text{U}} + b_O \tag{13}$$

According to the forecasting target of MultiDBN-T, for forecasting the electricity load demand in the future time τ, the loss function of the model is defined as follows:

$$\mathcal{L}_{oss} = \frac{1}{n}\sum \|\hat{y} - y\|^2 \quad \hat{y}, y \in \mathbb{R}^{\tau} \tag{14}$$

3.5. Training Algorithm and Optimal Configuration

The purpose of the forecasting model is to forecast the power load demand in future τ time by inputting an *m*-dimensional historical data with a duration of $\mathcal{T}$ (the *m*-dimension includes a column of power load historical data).

According to the original historical data and forecasting target, the input of the model is $Z^{(0)} \in \mathbb{R}^{m \cdot \mathcal{T}}$, and the output is $\hat{y} \in \mathbb{R}^{\tau}$. The initialization of the forecasting model, the training process, and the list of the output network parameter variable descriptions are presented in Table 1.

Table 1. Description of the major model parameters.

Symbol	Description and Meaning
$\theta^{(e)}_{MultiDBN-T}$	MultiDBN-T network parameter set; *e* is the training parameter epoch
$\{W_O, b_O\}_{L-Decoder}$	MultiDBN-T output layer network parameters
$\{W_i, b_i\}_{ggr}$	Network parameters obtained by the *i*-th layer GG-RBM, $i = 1{,}2{,}3$
W_{pos}	Position encoding information in D-TEncoder
$\left\{W_q^{(i)}, W_k^{(i)}, W_v^{(i)}\right\}_{dt}$	Multi-head self-attention parameters in D-TEncoder, $i \in \{1,2,\ldots,N\}$
$\{W_{iFNN}, b_{iFNN}\}^{(j)}_{FNN}$	Parameters in $FNN(\cdot)$ in D-TEncoder, $j \in \{1,2,\ldots,N\}$

The sample is divided into the 7:2:1 ratio for the training set, validation set, and test set, respectively. Furthermore, the training set is divided into 80–20%, where 20% is used for the fine-tuning and verification of hyperparameters. The process of the forecasting model is as follows. A three-layer RBM (GG-RBM) is built, corresponding to the formation of pre-DBN. The parameters of the pre-DBN network fine-tuned through training and the corresponding three-layer RBM network are used as the corresponding three-layer DNN network configuration and the initial weights of MultiDBN-T. After encoding by D-TEncoder, L-Decoder generates an output. The pseudo code of the algorithm is presented in Algorithm 1 as follows:

Algorithm 1: MultiDBN-T algorithm based on pre-DBN and D-TEncoder feature learning

```
Input: $GG-RBM(v,h), Line(W_{pre}, b_{pre}), \theta^{(0)}_{MultiDBN-T}, training\ batch\ B$
Output: $\theta^{(out)}_{MultiDBN-T} = \{\{W_i, b_i\}_{ggr}\ i = 1{,}2.3, W_{pos}, \left\{W_q^{(i)}, W_k^{(i)}, W_v^{(i)}\right\}_{dt}\ i \in \{1,2,\ldots,N\},$
        $\{W_{iFNN}, b_{iFNN}\}^{(j)}_{FNN}\ j \in \{1,2,\ldots,N\} \& i = 1,2, \{W_O, b_O\}_{L-Decoder}\}$
1.$Initial\ GG-RBM(v,h), \Delta w, \Delta a, \Delta b \leftarrow 0, \theta\_(MultiDBN-T)\^((0))$
2.For i = 1 to 3 Do
3.    $\Delta w, \Delta a, \Delta b \leftarrow CD\_k(GG-RBM(v,h))$;//Use $CD\_k$ to solve GG−RBM according to//Equations (1)−(7)
4.    $\{W_i, b_i\}_{ggr}\ i = 1{,}2.3 \leftarrow pre\_train(Pre-DBN)$;//According to Equations (8)–(9), use the//fine-tuning mode of
          DBN to obtain the initial weights of the network parameters of//the corresponding layer of MultiDBN-T
5. For e = 1 to epoch Do
6.    $\theta^{(e+1)}_{MultiDBN-T} \leftarrow Train(MultiDBN-T(\theta^{(e)}_{MultiDBN-T}))$;//Complete the parameter learning of
//MultiDBN-T according to Equations (10)–(14)
7. Return $\boldsymbol{\theta^{(out)}_{MultiDBN-T}}$
```

Suppose the loss function value obtained according to Equation (14) in the MultiDBN-T training process is $\Delta\theta_{MultiDBN-T}^{(e)}$. We apply BP to its partial derivative and then learn the network parameters and use the Adam optimizer for tuning. The parameter optimization in the model is defined as follows:

$$l_r^{(e+1)} = l_r^{(0)}\sqrt{(1-\beta_2^e)/(1-\beta_1^e)} \tag{15}$$

$$m^{(e+1)} = \beta_1 m^{(e)} + (1-\beta_1)\Delta\theta_{MultiDBN-T}^{(e)} \tag{16}$$

$$\omega^{(e+1)} = \beta_2 \omega^{(e)} + (1-\beta_2)(\Delta\theta_{MultiDBN-T}^{(e)})^2 \tag{17}$$

$$\theta_{MultiDBN-T}^{(e+1)} = \theta_{MultiDBN-T}^{(e)} - l_r^{(e+1)} * m^{(e+1)}/(\sqrt{\omega^{(e+1)}} + \varepsilon \tag{18}$$

where $l_r^{(0)}$ is the initial learning rate with an initial value of 0.01. $l_r^{(e+1)}$ is the new learning rate, calculated from the initial learning rate $l_r^{(0)}$ using Equation (15). $m^{(e+1)}$ is the first-order exponential smoothing value of the gradient during the training process. $\omega^{(e+1)}$ is the first-order exponential smoothing value of gradient squared during training. $\theta_{MultiDBN-T}^{(e+1)}$ represents the new network parameter set of the forecasting model MultiDBN-T. Equation (18) calculates the updated values of the network parameter variables. Furthermore, β_1, β_2, and ε are the parameters of the Adam optimizer, set to 0.9, 0.999, are 10^{-8} in Algorithm 1, respectively.

4. Data Description and Data Formatting

Multi-dimensional time-series data obtained by integrating datasets provided by multiple open source institutions were used to validate the forecasting model.

4.1. Data Description

The data primarily originate from the European Network of Transmission System Operators for Electricity, the Spanish Electricity Network, and the Open Weather API, which integrate regional weather data for five major cities in Spain, namely, Madrid, Barcelona, Barcelona, Valencia, Seville, and Bilbao. These cities are located in the east, south, west, north, and middle of Spain, respectively, and cover the average weather conditions in Spain. The historical series data are the four dimensions of weather average, electricity price, electricity production, and electricity consumption. The data describe the four-dimension historical data in the four years from 1 January 2015 to 31 December 2018. The time interval window for data acquisition was 1 h.

4.2. Description of Observable Variables

According to the data collection time interval, from 1 January 2015 to 31 December 2018, a total of 35,064 rows of data were collected for each dimension. The data dimensions (observable variables) include time label (2), power production capacity (14), power load (1), electricity price (1), and weather conditions (9*5), or 63 columns in total. The historical data are the electricity load in the territory of Spain and cover the electricity consumption history of five cities in the dataset. The electricity load demand is closely coupled with the electricity demand of a city. Therefore, each city name and the corresponding weather indicator are connected to form a label; that is, nine weather indicators and five city names are connected to form a total of 45 labels. All data are normalized as the original data of the model. A summary of the observable variables of the dataset is presented in Table 2.

Table 2. Summary of observable variables.

Term	Description and Number of Observable Variables
Time	2 columns: time and holiday labels
Electricity production	14 columns: various power production capacities
Electrical load	1 column: actual consumption of electric energy per hour
Electricity price	1 column: average price of electricity for a company
Weather	9 columns: data of various weather indicators
City label	5 columns: city names

4.3. Data Format Conversion

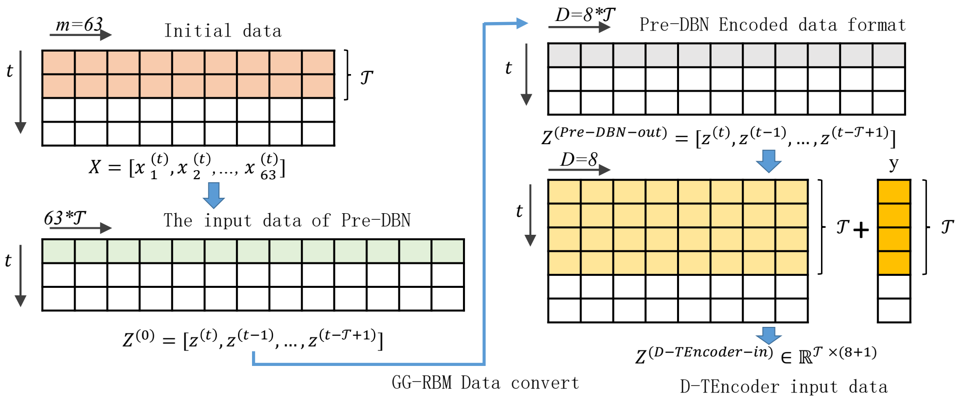
As described in Section 4.2, 63 observable variables were observed in the original data, with a total of 35,064 rows of records. The load forecasting model uses two days (48 h), three days (72 h), and four days (96 h) to forecast the electricity demands for the next day (24 h). Thus, the model input is $Z^{(0)} \in \mathbb{R}^{63\cdot\mathcal{T}}, \mathcal{T} = \{48,72,96\}$.

The data format conversion processes of the MultiDBN-T kernel layers are displayed in Figure 6. The model undergoes the following three format conversion processes from raw data to data output:

(1) Initialization of the original data format to the model input format: As displayed in the left part of Figure 5, after the initial data are cleansed, their format is an array $\mathbb{R}^{35064\times 63}$. According to the format of pre-DBN input in MultiDBN-T, the data of consecutive $\mathcal{T}$ durations ($\mathcal{T} = \{48,72,96\}$, that is, $\mathcal{T}$ hours, $\mathcal{T}$ rows of records, are converted into one-dimensional $\mathbb{R}^{1\times 63\cdot\mathcal{T}}$ array data.

(2) Encoded output after three GG-RBM conversions: A total of three GG-RBMs are configured in pre-DBN. The input format of the first GG-RBM is R$\mathbb{R}^{1\times 63\cdot\mathcal{T}}$, and the output of the last GG-RBM is $\mathbb{R}^{1\times 8\cdot\mathcal{T}}$; that is, after pre-DBN feature learning, the input data are extracted with features close to 63/8 ≈ 8 times compressibility after data reconstruction.

(3) Conversion of pre-DBN output encoding to D-TEncoder encoder input: Because the output of pre-DBN is reconstructed and encoded, the original data are one-dimensional array data $\mathbb{R}^{1\times D\cdot\mathcal{T}}$, and the input of the D-TEncoder encoder is in the $\mathbb{R}^{D\times\mathcal{T}}$ format, the pre-DBN output encoding is converted to $\mathbb{R}^{\mathcal{T}\times D}$ by segment $\mathcal{T}$ for each sample, as displayed in the right part of Figure 6. The original data lose some information through pre-DBN. The power load (y) and the converted data in the format of $\mathbb{R}^{\mathcal{T}\times 8}$ are spliced into $\mathbb{R}^{\mathcal{T}\times(8+1)}$, as the input to the D-TEncoder encoder.



m=63
Initial data
t
$\mathcal{T}$
$X = [x_1^{(t)}, x_2^{(t)}, \ldots, x_{63}^{(t)}]$
63*$\mathcal{T}$
The input data of Pre-DBN
t
$Z^{(0)} = [z^{(t)}, z^{(t-1)}, \ldots, z^{(t-\mathcal{T}+1)}]$
GG-RBM Data convert
D=8*$\mathcal{T}$
Pre-DBN Encoded data format
t
$Z^{(Pre-DBN-out)} = [z^{(t)}, z^{(t-1)}, \ldots, z^{(t-\mathcal{T}+1)}]$
D=8
y
t
$\mathcal{T}$ +
$\mathcal{T}$
$Z^{(D-TEncoder-in)} \in \mathbb{R}^{\mathcal{T}\times(8+1)}$
D-TEncoder input data

Figure 6. MultiDBN-T kernel layer data format conversion.

5. Experiment and Result Analysis

5.1. Evaluation Metric

The electric load demand of the next day (24 h) in the future is predicted by applying the collected electric load time-series data. Taking the mean absolute percentage error (*MAPE*) and root mean square error (*RMSE*) as the comparison indicators, the MultiDBN-T proposed in this paper was analyzed with the algorithms LSTM, Conv3D-LSTM (CNN + LSTM), Adaboost, ARIMA, SVR, and RF. The formulas for calculating *MAPE*, *RMSE*, *MAE*, and *RMSLE* are as follows:

$$MAPE = \frac{1}{n}\sum\nolimits_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \times 100\% \tag{19}$$

$$RMSE = \sqrt{\frac{1}{n}\sum\nolimits_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{20}$$

$$MAE = \frac{1}{n}\sum\nolimits_{i=1}^{n}|y_i - \hat{y}_i| \tag{21}$$

$$RMSLE = \sqrt{\frac{1}{n}\sum\nolimits_{i=1}^{n}(log(1+\hat{y}_i) - log(1+y_i))^2} \tag{22}$$

where $\hat{y}_i$ is the predicted value of the model, y_i represents the actual data, and n is the number of samples.

5.2. Experimental Setup

Hardware environment: 2 × Dell PowerEdge T720 (2×CPUs: Xeon 6-core E5-2630 2.3G, a total of 12 CPU cores, 64 GB RAM).

Dataset division: The dataset has a total of 35,064 samples. The dimension of each sample is 63. They are divided by 7:2:1. A total of 24,545 rows are used as training samples, where 19,636 rows are used for pre-DBN (GG-RBM) pre-training, 4909 rows are used for pre-DBN fine-tuning, 7012 rows are used as the validation set, and 3507 rows are used as the test set for comparing the performance of various models.

According to the input format and forecasting target of the experimental data (considering the computation cost and effectiveness, the model uses the first three days, or 72 h, to forecast the electricity load demand of each time window of 24 h in the next day), the relevant hyperparameters in MultiDBN-T (the network structure and parameter configuration of each module) are presented in Table 3.

Table 3. Summary of observable variables.

Module/Parameter	Description and Number of Observable Variables
Pre-DBN	GG-RBM1 (63 × 72, 32 × 72), GG-RBM2 (32 × 72, 16 × 72) GG-RBM3 (16 × 72, 8 × 72), Linear (63 × 72)
D-TEncoder	Feedforward (256), Multi-Head (8), Output-Dimension (256) Blocks (3), Dropout (0.1)
L-Decoder	Linear (1 × 24)
Learning rate	0.01
Batch size	128

5.3. Experimental Results and Analysis

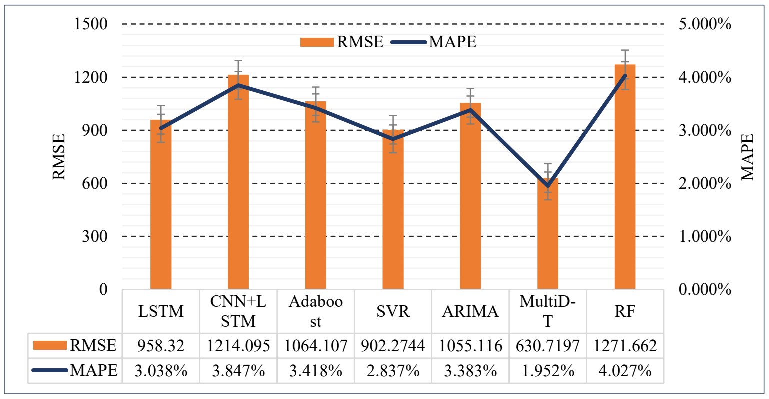
MultiDBN-T was applied and the dataset was tested according to the structure and configuration parameters in Table 3. According to the set evaluation metrics $MAPE$ and

$RMSE$, the new algorithm and the existing algorithms commonly used in time-series analysis, LSTM, Conv 3D-LSTM (CNN+ LSTM), Adaboost, ARIMA, SVR, and RF, are analyzed. Table 4 and Figure 6 are the overall data comparison results and the data plots of the test set (3507 rows of data records) in the dataset.

Table 4. Experimental data.

Algorithm	$MAPE$ (%)	$RMSE$ (KW)	MAE (KW)	RMSLE
LSTM	3.038%	958.320	878.1204	0.032806
CNN+LSTM	3.847%	1214.095	1113.74	0.041546
Adaboost	3.418%	1064.107	987.7954	0.036448
SVR	2.837%	902.274	820.3937	0.030889
ARIMA	3.383%	1055.116	978.5581	0.036015
RF	4.027%	1271.662	1165.123	0.043685
MultiDBN-T	1.952%	630.720	566.3686	0.021454

Table 4 and Figure 7 reveal that MultiDBN-T (abbreviated as MultiD-T in the figure), the proposed forecasting model, outperforms all the compared algorithms in terms of MAPE (1.952%) and RMSE (630.720 KW) of the prediction results in the test set. RF exhibited the worst performance, with MAPE and RMSE of 4.027% and 1271.662 KW, respectively. Among the algorithms involved in the comparison, the comparison between the best, MultiDBN-T, and the worst, RF, reveals that the MAPE and RMSE of the proposed forecasting model were 51.54% and 50.40% higher than RF, respectively.



	LSTM	CNN+LSTM	Adaboost	SVR	ARIMA	MultiD-T	RF
RMSE	958.32	1214.095	1064.107	902.2744	1055.116	630.7197	1271.662
MAPE	3.038%	3.847%	3.418%	2.837%	3.383%	1.952%	4.027%

Figure 7. Results of mean absolute percentage error ($MAPE$) and root mean square error ($RMSE$) from different algorithms.

To analyze the performance of the aforementioned algorithms and MultiDBN-T on the test set, the distribution of the components is calculated according to the MAPE and RMSE of the predicted results and the actual power load demand in each time window (observation interval of one hour). For MAPE, $RE = \left|\frac{\hat{y}_i - y_i}{y_i}\right|$, and for RMSE, $MSE' = (\hat{y}_i - y_i)^2$, where $i = 1,2, \ldots 3507$, $\hat{y}_i$ is the model predicted value, and y_i represents the actual data. The box plots corresponding to the distribution of MAPE and RMSE components are displayed in Figures 8 and 9.

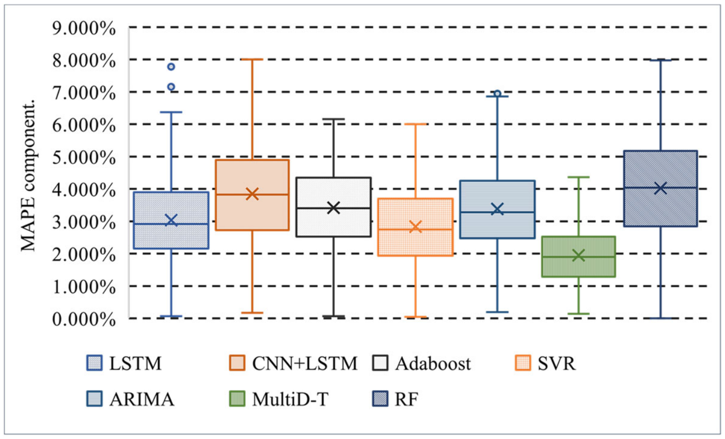


Figure 8. *MAPE* component distribution box plots.

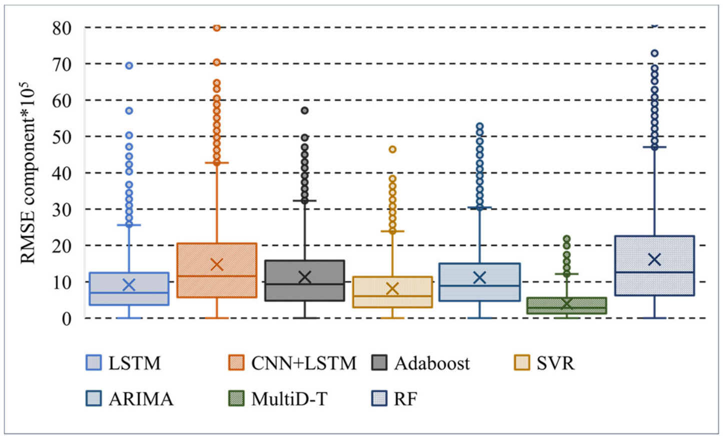


Figure 9. *RMSE* component distribution box plots.

It can be seen from Table 5 and Figure 8 that the values of MAX, Q1, MEDIAN, Q2, and Q3 of the MAPE component of the MultiDBN-T model are the smallest among all the methods. The numerical distribution of its MAPE component is also the most concentrated, which shows that the stability of the model is also the best.

Table 5. The data of MAPE component distribution (%).

Metrics	*LSTM*	*CNN + LSTM*	*Adaboost*	*SVR*	*ARIMA*	*RF*	*MultiDBN-T*
Max	5.369%	8.005%	6.157%	5.999%	6.861%	7.970%	4.361%
Q1	3.899%	4.894%	4.346%	3.701%	4.249%	5.177%	2.524%
Median	3.038%	3.847%	3.418%	2.837%	3.383%	4.027%	1.952%
Q2	2.918%	3.825%	3.401%	2.745%	3.275%	4.038%	1.898%
Q3	2.157%	2.724%	2.523%	1.931%	2.477%	2.846%	1.288%
Min	0.067%	0.172%	0.065%	0.047%	0.192%	0.0%	0.138%

It can be seen from Table 6 and Figure 9 that the values of MAX, Q1, MEDIAN, Q2, and Q3 of the RMSE component of the MultiDBN-T model are still the lowest among all methods.

Table 6. The data of RMSE component distribution (*10³).

Metrics	*LSTM*	*CNN + LSTM*	*Adaboost*	*SVR*	*ARIMA*	*RF*	*MultiDBN-T*
Max	2562.184	4274.847	3226.730	2387.311	3042.323	4706.446	1212.348
Q1	1248.420	2052.355	1581.224	1133.856	1501.843	2255.543	560.560
Median	918.115	1476.606	1132.002	813.867	1112.953	1616.663	397.694
Q2	693.523	1154.171	931.083	602.291	884.029	1258.970	284.569
Q3	360.841	570.103	482.006	296.404	474.368	621.096	125.185
Min	0.247	1.214	0.525	0.240	1.055	0.000004	0.630719

Figures 8 and 9 reveal that the distributions of the $MAPE$ component (RE) and the RMSE component (MSE') in the load demand results forecasted by MultiDBN-T are concentrated in the upper and lower quartiles. The distributions exhibit small whisker lengths and are less spread for MSE'. The data distributions from RF and CNN-LSTM (Conv3D-LSTM) are more spread, which indicates that their performance is unstable. Their whisker lengths and outliers are worse in the distribution of MSE'.

To perform a comparative analysis of MultiDBN-T and several other algorithms, the RE and MSE' of each algorithm's prediction results at each observation point were subtracted from the difference of RE and MSE' obtained by MultiD-T; that is, $RE_a - RE_{MultiDBN-T}$ and $MSE'_a - MSE'_{MultiDBN-T}$, and then the local performance improvement is analyzed. Here, $a = \{\text{LSTM}, \text{CNN} - \text{LSTM}(\text{Conv3D} - \text{LSTM}), \text{Adaboost}, \text{ARIMA}, \text{SVR}, \text{RF}\}$. The box plots corresponding to the distributions of performance improvement are displayed in Figures 10 and 11.

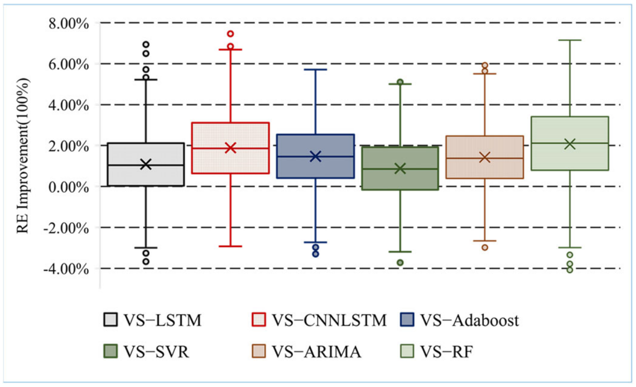


Figure 10. Box plots for MAPE component improvement.

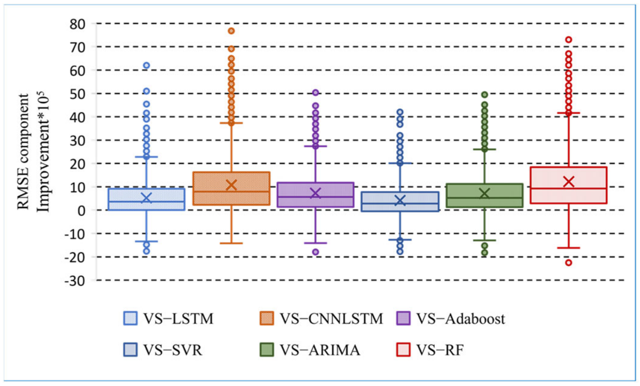


Figure 11. Box plots for distribution of the RMSE component improvement, in comparison with the improvement in MSE'.

It can be seen from Table 7 and Figure 10 that the MultiDBN-T model has the most obvious improvement in the MAPE component compared with the RF method; the improvement is not obvious compared with the SVR method, but its component improvement reached an average of 0.89%. From Table 8 and Figure 11, it can be seen that the situation of the RMSE component improvement is similar to that of the MAPE component improvement.

Table 7. The data of the MAPE component improvement (%).

Metrics	*VS-LSTM*	*VS-CNN + LSTM*	*VS-Adaboost*	*VS-SVR*	*VS-ARIMA*	*VS-RF*
Max	5.22%	6.69%	5.72%	5.00%	5.50%	7.15%
Q1	2.12%	3.11%	2.54%	1.93%	2.46%	3.41%
Median	1.09%	1.90%	1.47%	0.89%	1.43%	2.08%
Q2	1.04%	1.86%	1.46%	0.85%	1.38%	2.12%
Q3	0.03%	0.63%	0.41%	−0.16%	0.39%	0.79%
Min	−3.00%	−2.93%	−2.73%	−3.19%	−2.66%	−2.99%

Table 8. The data of the RMSE component (MSE') improvement (*10^3).

Metrics	*VS-LSTM*	*VS-CNN + LSTM*	*VS-Adaboost*	*VS-SVR*	*VS-ARIMA*	*VS-RF*
Max	2281.035	3730.698	2739.107	2012.757	2599.928	4161.327
Q1	918.079	1628.529	1178.128	778.153	1120.156	1842.060
Median	520.569	1076.218	734.517	416.291	715.463	1219.317
Q2	362.080	788.596	567.339	279.695	524.671	923.383
Q3	6.568	226.248	136.731	−47.682	126.794	290.765
Min	−1338.101	−1422.029	−1412.882	−1265.45	−1297.318	−1618.46

Figures 10 and 11 reveal that except for the RE versus SVR, the upper and lower quartiles of RE and MSE' improved by MultiDBN-T and the several other algorithms were all above 0. Thus, the performance of MultiDBN-T at each observation point of the test set was the best. In particular, the lower whiskers of each box plot on MSE' are short and mainly distributed above the lower quartile line, and the lower quartile line is above 0. Therefore, the performance improvement of the forecasting model in this paper on MSE' is remarkable and stable.

For the test set, we selected the three-day power load forecast and actual demand from 8 to 10 August 2018 and 24 and 26 December 2018, respectively. The plots for the relative errors are displayed in Figures 12 and 13.

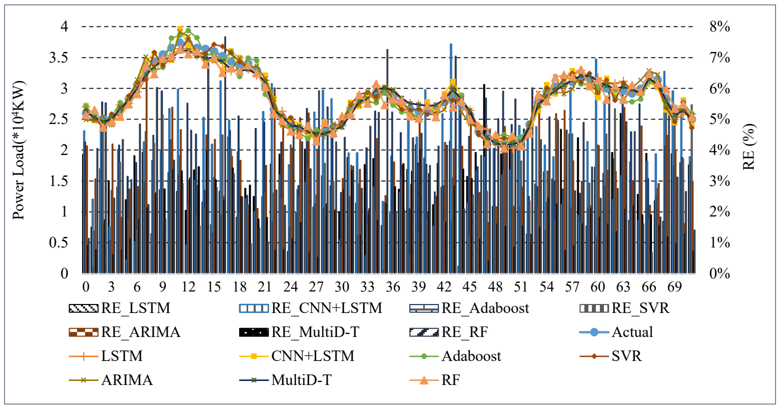


Figure 12. Plots of forecasted values and actual results (8 August 2018 to 10 August 2018).

Figures 12 and 13 reveal that because of the difference in the weather in Spain during these two periods, the electricity demand differed considerably. The data representation code obtained by feature learning can represent the coupling and correlation characteristics of power load demand, weather, and power price. Thus, the power load demand can be predicted accurately with small elative errors (RE).

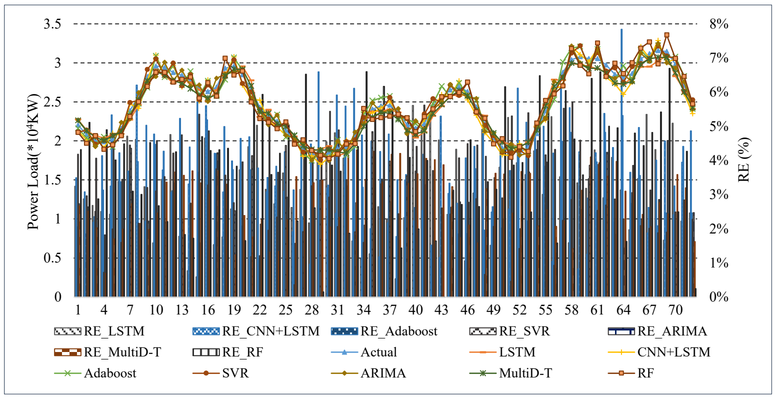


Figure 13. Plots of forecasted values and actual results (24 December 2018 to 26 December 2018).

According to the electricity load demand results predicted by the different methods for these two time periods, the average value of the relative error of each method is calculated as shown in Table 9.

Table 9. The average relative errors of the two periods (%).

Period	*LSTM*	*CNN + LSTM*	*Adaboost*	*SVR*	*ARIMA*	*RF*	*MultiDBN-T*
2018-08-08 to 2018-08-10	3.026%	3.897%	3.430%	2.907%	3.472%	3.803%	1.993%
2018-12-24 to 2018-12-26	3.231%	3.696%	3.517%	2.900%	3.293%	3.903%	2.882%

A lower average relative error (RE) indicates a better prediction. As we can see from Table 9, MultiDBN-T obtains the lowest values in average relative error.

According to the analysis, the overall situation of the experiment in Table 10 and Figure 14 reveals that the proposed MultiDBN-T outperforms other algorithms, whether locally or as a whole. It performs better than SVR (the next best) in terms of MAPE and RMSE by 31.21% and 30.10%, respectively. The average increases in the evaluation metrics of MAPE, RMSE, MAE, and RMSLE were 42.16%, 40.61%, 41.96%, and 40.99% respectively.

Table 10. The improvement of MultiDBN-T (%).

Algorithm	*MAPE Imp*	*RMSE Imp*	*MAE Imp*	*RMSLE Imp*
LSTM	35.76%	34.18%	35.50%	34.60%
CNN+LSTM	49.27%	48.05%	49.15%	48.36%
Adaboost	42.91%	40.73%	42.66%	41.14%
SVR	31.21%	30.10%	30.96%	30.55%
ARIMA	42.30%	40.22%	42.12%	40.43%
RF	51.54%	50.40%	51.39%	50.89%
Average Improvement	42.16%	40.61%	41.96%	40.99%

The analysis of the experimental data revealed that MultiDBN-T achieved satisfactory results for the dataset. Through feature learning, MultiDBN-T performed better in obtaining strong coupling and correlation features between the forecasted target data and multiple observable variables or between hidden variables, time, and time periods. The feature learning improves the prediction accuracy. Although MultiDBN-T improves prediction accuracy, as the dimension of sample data (observable variables) increases, its network structure expands and become too large and too fast, and the network parameters that should be learned become large. The resources required for its training also increase, and the time required for training becomes longer.

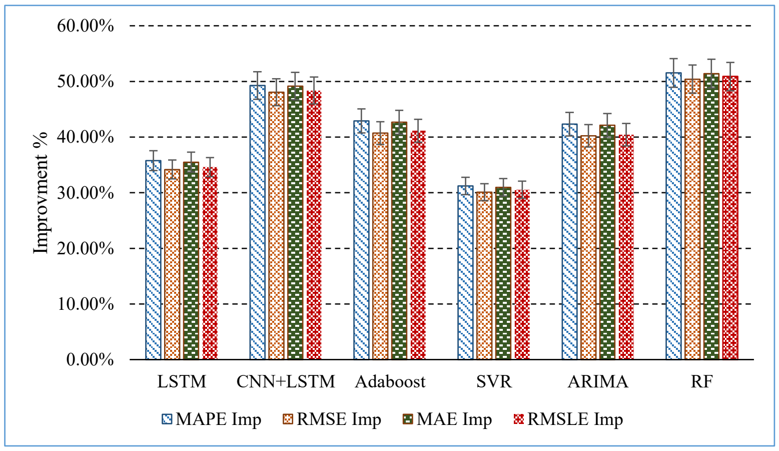


Figure 14. Plots of the improvement of MultiDBN-T.

6. Conclusions

Power load demand forecasting is the basis for power analysis and intelligent power supply control and has attracted considerable research interest attention in energy management. This method is critical in operating and managing power systems. To accurately forecast power load demand and implement power generation planning and scheduling, we proposed a novel unsupervised MultiDBN-T. The two encoders, pre-DBN and D-TEncoder, are used in the model to perform unsupervised multi-dimensional feature learning on historical data of power loads with multiple observable variables. The data representation encoding is obtained through feature learning. Thus, the strongly coupled features between independent multi-factors, the strongly coupled features between positions or time points in the sequence, and the correlation features between unobservable hidden variables are obtained, which improves the predictive ability and accuracy of the model. On integrated multi-dimensional power load data, compared with the classical and new algorithms for time-series analysis, such as LSTM, Conv3D-LSTM (CNN+LSTM), Adaboost, ARIMA, SVR, and RF, MultiDBN-T exhibited the best performance in both local predictive and overall predictive ability, with an average improvement of more than 40% in MAPE and RMSE, verifying the effectiveness and predictive ability of the model.

However, the MultiDBN-T model requires a large network on high-dimensional datasets, too many network parameters to learn, and a long training time. In the future, we will optimize the networks of pre-DBN and D-TEncoder in the forecasting model to reduce its consumption of resources, further improve its efficiency and accuracy, and improve its applicability.

Author Contributions: Conceptualization, W.B. and K.L.; methodology, W.B. and K.L.; software, J.Z. (Jiaxian Zhu) and J.Z. (Jialing Zhao); validation, W.B., W.C., and K.L.; formal analysis, W.B.; investigation, W.B.; resources, J.Z. (Jiaxian Zhu); data curation, J.Z. (Jialing Zhao); writing—original draft preparation, W.B.; writing—review and editing, W.B.; visualization, W.B.; supervision, W.B. and K.L.; project administration, W.B. and K.L.; funding acquisition, W.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: The work in this study was supported by the Guangdong Province Educational Science Planning Project (no. 2019KTSCX199), the Zhaoqing Science and Technology Special Fund Project (no. 2020G1004), Zhaoqing University Science and Technology Projects (no. zlgc201933), and the Teaching Reform Project of University Public Computer Course of Guangdong Province (no. 2021-GGJGJ-012).

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

References

1. Shahid, F.; Zameer, A.; Muneeb, M. A novel genetic LSTM model for wind power forecast. *Energy* **2021**, *223*, 120069.
2. Phan, Q.T.; Wu, Y.K.; Phan, Q.D. A hybrid wind power forecasting model with XGBoost, data preprocessing considering different NWPs. *Appl. Sci.* **2021**, *11*, 1100.
3. Lai, X.; Huang, Y.; Han, X.; Gu, H.; Zheng, Y. A novel method for state of energy estimation of lithium-ion batteries using particle filter and extended Kalman filter. *J. Energy Storage* **2021**, *43*, 103269.
4. Ti, Z.; Deng, X.W.; Zhang, M. Artificial Neural Networks based wake model for power prediction of wind farm. *Renew. Energy* **2021**, *172*, 618–631.
5. Kasihmuddin, M.S.M.; Jamaludin, S.Z.M.; Mansor, M.A.; Wahab, H.A.; Ghadzi, S.M.S. Supervised Learning Perspective in Logic Mining. *Mathematics* **2022**, *10*, 915.
6. Mohd Jamaludin, S.Z.; Mohd Kasihmuddin, M.S.; Md Ismail, A.I.; Mansor, M.A.; Md Basir, M.F. Energy based logic mining analysis with hopfield neural network for recruitment evaluation. *Entropy* **2020**, *23*, 40.
7. Jamaludin, S.Z.M.; Romli, N.A.; Kasihmuddin, M.S.M.; Baharum, A.; Mansor, M.A.; Marsani, M.F. Novel Logic Mining Incorporating Log Linear Approach. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *in press*.
8. Kong, W.; Dong, Z.Y.; Jia, Y.; Hill, D.J.; Xu, Y.; Zhang, Y. Short-term residential load forecasting based on LSTM recurrent neural network. *IEEE Trans. Smart Grid* **2019**, *10*, 841–851.
9. Luo, H.; Zhou, P.; Shu, L.; Mou, J.; Zheng, H.; Jiang, C.; Wang, Y. Energy performance curves prediction of centrifugal pumps based on constrained PSO-SVR model. *Energies* **2022**, *15*, 3309.
10. Liu, M.D.; Ding, L.; Bai, Y.L. Application of hybrid model based on empirical mode decomposition, novel recurrent neural networks and the ARIMA to wind speed prediction. *Energy Convers. Manag.* **2021**, *233*, 113917.
11. Wei, N.; Li, C.; Duan, J.; Liu, J.; Zeng, F. Daily natural gas load forecasting based on a hybrid deep learning model. *Energies* **2019**, *12*, 218.
12. Lu, H.; Azimi, M.; Iseley, T. Short-term load forecasting of urban gas using a hybrid model based on improved fruit fly optimization algorithm and support vector machine. *Energy Rep.* **2019**, *5*, 666–677.
13. Wenlong, H.; Yahui, W. Load forecast of gas region based on ARIMA algorithm. In Proceedings of the 2020 Chinese Control And Decision Conference (CCDC), Hefei, China, 22–24 August 2020; pp. 1960–1965.
14. Musbah, H.; El-Hawary, M. SARIMA model forecasting of short-term electrical load data augmented by fast fourier transform seasonality detection. In Proceedings of the IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), Edmonton, AB, Canada, 5–8 May 2019; pp. 1–4.
15. Yildiz, B.; Bilbao, J.; Sproul, A. A review and analysis of regression and machine learning models on commercial building electricity load forecasting. *Renew. Sustain. Energy Rev.* **2017**, *73*, 1104–1122.
16. Tayaba, U.B.; Zia, A.; Yanga, F.; Lu, J.; Kashif, M. Short-term load forecasting for microgrid energy management system using hybrid HHO-FNN model with best-basis stationary wavelet packet transform. *Energy* **2020**, *203*, 117857.
17. Panapakidis, I.; Katsivelakis, M.; Bargiotas, D. A Metaheuristics-Based Inputs Selection and Training Set Formation Method for Load Forecasting. *Symmetry* **2022**, *14*, 1733.
18. Ahmed, W.; Ansari, H.; Khan, B.; Ullah, Z.; Ali, S.M.; Mehmood, C.A.A. Machine learning based energy management model for smart grid and renewable energy districts. *IEEE Access* **2020**, *8*, 185059–185078.
19. Shabbir, N.; Kutt, L.; Jawad, M.; Iqbal, M.N.; Ghahfarokhi, P.S. Forecasting of energy consumption and production using recurrent neural networks. *Adv. Electr. Electron. Eng.* **2020**, *18*, 190–197.
20. Shirzadi, N.; Nizami, A.; Khazen, M.; Nik-Bakht, M. Medium-term regional electricity load forecasting through machine learning and deep learning. *Designs* **2021**, *5*, 27.
21. Fan, G.-F.; Guo, Y.-H.; Zheng, J.-M.; Hong, W.-C. Application of the weighted K-nearest neighbor algorithm for short-term load forecasting. *Energies* **2019**, *12*, 916.
22. Madrid, E.A.; Antonio, N. Short-term electricity load forecasting with machine learning. *Information* **2021**, *12*, 50.
23. Shi, H.; Xu, M.; Li, R. Deep learning for household load forecasting—A novel pooling deep RNN. *IEEE Trans. Smart Grid* **2018**, *9*, 5271–5280.
24. Wang, X.; Gao, X.; Wang, Z.; Ma, C.; Song, Z. A Combined Model Based on EOBL-CSSA-LSSVM for Power Load Forecasting. *Symmetry* **2021**, *13*, 1579.

25. Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; Jin, R. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. *arXiv* **2022**, *preprint*, arXiv:2201.12740.
26. Xuan, W.; Shouxiang, W.; Qianyu, Z.; Shaomin, W.; Liwei, F. A multi-energy load prediction model based on deep multi-task learning and ensemble approach for regional integrated energy systems. *Int. J. Electr. Power Energy Syst.* **2021**, *126*, 106583.
27. Zhou, B.; Meng, Y.; Huang, W.; Wang, H.; Deng, L.; Huang, S.; Wei, J. Multi-energy net load forecasting for integrated local energy systems with heterogeneous prosumers. *Int. J. Electr. Power Energy Syst.* **2021**, *126*, 106542.
28. Wang, C.; Wang, Y.; Ding, Z.; Zheng, T.; Hu, J.; Zhang, K. A Transformer-Based Method of Multi-Energy Load Forecasting in Integrated Energy System. *IEEE Trans. Smart Grid* **2022**, *13*, 2703–2714. https://doi.org/10.1109/TSG.2022.3166600.
29. Phyo, P.P.; Byun, Y.C. Hybrid Ensemble Deep Learning-Based Approach for Time Series Energy Prediction. *Symmetry* **2021**, *13*, 1942.
30. Bengio, Y.; Courville, A.; Vincent, P. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1798–1828.
31. Schölkopf, B.; Locatello, F.; Bauer, S.; Ke, N.R.; Kalchbrenner, N.; Goyal, A.; Bengio, Y. Toward causal representation learning. *Proc. IEEE* **2021**, *109*, 612–634.
32. Hinton, G.E. A practical guide to training restricted Boltzmann machines. In *Neural Networks: Tricks of the Trade*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 599–619.
33. Zhang, Z.; Zhao, J. A deep belief network based fault diagnosis model for complex chemical processes. *Comput. Chem. Eng.* **2017**, *107*, 395–407.