Energy 293 (2024) 130666

Contents lists available at ScienceDirect

Energy

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

CNNs-Transformer based day-ahead probabilistic load forecasting for weekends with limited data availability☆

Zhirui Tian, Weican Liu, Wenqian Jiang, Chenye Wu *

School of Science and Engineering, the Chinese University of Hong Kong, Shenzhen, Guangdong, China

ARTICLE INFO

Keywords:
Load forecast
Deep learning
Data processing
Quantile regression

ABSTRACT

Independent system operators (ISOs) are pursuing day-ahead probabilistic load forecasting as it offers comprehensive load trend and pattern information. Compared with commonly adopted point forecasting, it enables ISOs to better understand the uncertainty of future demand through interval forecasting with varying confidence levels. In practice, this advantage could enable precise day-ahead forecasting for critical days with irregular load patterns (e.g., weekends or holidays), particularly when the data availability is limited. To this end, we customize a day-ahead probabilistic load forecasting framework with an emphasis on weekends based on data processing and probabilistic deep learning. Specifically, data processing combines data denoising and data augmentation techniques, incorporating peak and trend information into the denoised one-dimensional time series data to aid training. This procedure helps extract more information from the restricted training samples. The probabilistic deep learning, ***CNNs-Transformer***, combines multi-layer Convolutional Neural Networks and Transformer, adopting ***QRLoss*** (quantile regression loss function) to achieve probabilistic forecasting. The loss penalty technique enhances the model's attention to weekend data. Numerical studies based on field data suggest that the proposed framework can obtain accurate day-ahead probabilistic forecasting results (48-time points of the whole day) by using only two-week historical data, and the accuracy improvement over its rivals is remarkable on weekends.

1. Introduction

Load forecasting enables independent system operators (ISOs) to comprehend the changing trend and dynamic patterns of the electrical demand [1], thereby facilitating the effective operation of the power grid [2]. Classical load forecasting methods have predominantly focused on point prediction. Although effective in the past decades, their provided information is increasingly inadequate due to the greater diversity of demand patterns. This inadequacy becomes particularly pronounced during critical days (e.g., weekends or holidays) when load patterns dramatically diverge from that on working days [3]. These periods often witness significant shifts in human activity patterns, such as increased family gatherings, travel, and recreational activities. These changes directly influence electricity consumption patterns and volumes. Accurate load forecasting is thus essential for ISOs to effectively align their supply with the altered demand, ensuring that service quality is maintained during these times of fluctuating usage.

Furthermore, while utilizing massive training samples for ultra-short load forecasting might yield relatively accurate results, this approach may not be very practical for day-ahead economic dispatch especially during critical days, where ISOs need to forecast all the demand data for the next day, a multi-step forecasting task. Most existing day-ahead forecasting methods suffer from low data utilization. For example, many of such methods adopt multi-model technology, which requires establishing the same number of models as the number of forecasting time-steps, imposing high requirements on both the computational capability and amount of training data [4]. In this context, limited data availability, defined in this paper as using load data that contains a few weekend or holiday data, poses a unique challenge for achieving accurate day-ahead probability load forecasting results. To address these challenges, we combine multi-layer Convolutional Neural Networks with the Transformer architecture, adopting data denoising and data augmentation technology along with a customized loss function, to enable accurate day-ahead load forecasting for crucial days with limited data availability.

1.1. Literature review

Recent decades have witnessed an increasing interest in enhancing the precision of ultra-short load forecasting. The initial efforts

☆ This work was supported by the National Natural Science Foundation of China under Grant 72271213, and the Shenzhen Science and Technology Program, China under Grant JCYJ20220530143800001 and Grant RCYX20221008092927070.

* Corresponding author.
E-mail address: chenyewu@yeah.net (C. Wu).

https://doi.org/10.1016/j.energy.2024.130666
Received 29 September 2023; Received in revised form 8 February 2024; Accepted 9 February 2024
Available online 10 February 2024

adopt linear regression [5] and time series analysis (Autoregressive Integrated Moving Average (***ARIMA***) [6,7], Seasonal Autoregressive Integrated Moving Average (***SARIMA***) [8]). However, these methods are inadequate in efficiently utilizing data, which often fail to yield satisfactory forecasting outcomes. Since the 1990s, the utilization of neural network-based approaches has grown significantly with the advancements in computer science technology. In the early stages, the first generation artificial neural networks (***ANN***) gained significant popularity, such as Multi-layer Perception (***MLP***) and Back-propagation neural network (***BPNN***). Xiao et al. employed ***BPNN*** in conjunction with a rough set to forecast demand, resulting in a modest enhancement in the forecasting accuracy [9]. In the early 2000s, there was a gradual shift from ***ANN*** to the second generation neural networks, including Convolutional Neural Network (***CNN***) [10], Long Short Term Memory (***LSTM***) [11] and Gate Recurrent Unit (***GRU***) [12], which have been customized for different prediction tasks. ***CNN*** is specialized at extracting the features of data in depth, while ***LSTM*** and ***GRU*** are more sensitive to the relationship between time series data. Niu et al. proposed a framework based on ***CNN***, Bi-directional ***GRU***, and attention mechanism for load forecasting, aimed to enhance the deep learning model's capacity for nonlinear fitting, which exceeded the forecasting accuracy of benchmark models such as ***LSTM*** [13]. More recently, there is a growing trend in the utilization of combined models. Cao et al. integrated fuzzy information granulation technology with forecasting technology, which focused on examining the impact of the granulation method on the accuracy of load forecast [14]. Hu et al. employed a modified grasshopper optimization algorithm to optimize ***LSTM***, which can find the optimal hyper-parameters through the meta-heuristic optimization algorithm, and validated the superiority of this approach in multiple load data sets [15] . Although the above point prediction methods continue to enhance the accuracy of load forecasting, they can only provide a single prediction pattern, making it challenging to provide effective reference for ISOs. Therefore, probabilistic forecasting has become more and more popular in recent years, it can calculate the interval forecasting results with varying confidence levels, which can better quantify the uncertainty in future demand and serve as the basis for decision-making of the ISOs. Huang et al. adopted a universal load range discretization (LRD) method to construct the probabilistic training samples for ***CNN***, and obtained excellent interval forecasting results significantly better than other typical methods [16]. Zhang et al. developed a deep hybrid model containing deep learning and an improved optimizer, and utilized a selection strategy to obtain more accurate probabilistic forecasting results [17].

Although ultra-short point forecasting has made remarkable progress in improving load forecasting accuracy, day-ahead (multi-step) forecasting has recently attracted increasing interest. Just to name a few, Yuan et al. developed an ensemble multi-step ***M-RMLSSVR*** model using ***VMD*** and a two-group strategy for day-ahead load forecasting, which outperformed ***ANN*** and ***SVR*** models significantly [18]. Xu et al. introduced a new predictive approach for day-ahead electricity load forecasting with probability interpretation using curve-to-curve regression, and demonstrating its robustness and applicability in different scenarios [19]. Wang et al. incorporated data fuzzy granulation and a high-performance optimizer, which can get accuracy point and interval forecasting results for both ultra-short as well as multi-step forecasting tasks [20]. Li et al. proposed a system of load influencing factors with carbon-neutral elements, and achieved dynamic long-term load forecasting results using a modified feedback mechanism [21]. Li et al. utilized manifold learning to extract the underlying factors of load variations and then employed ***LSTM*** to build forecasting models in the low-dimensional space for improving mid-term electrical load forecasting outcomes [22]. Wen et al. integrated weather station data with load data and employed a novel activation mechanism and fuzzy rules to boost the robustness of the neural network [23]. Although the aforementioned methods offer viable approaches for day-ahead load forecasting, they often overlook the issue of sample quantity and the influence of data availability on forecasting results. In fact, due to privacy protection and the difficulty in obtaining relevant data access, economic or weather-related data are not readily available in all cases to assist training. Furthermore, they fail to modify the model to accommodate the critical days, such as weekends or holidays, which is the focus of our work.

1.2. Our contributions

In this paper, we propose to adopt data processing and deep learning framework to enable effective day-ahead probabilistic load forecasting with restricted data availability, especially during critical days. Since the weekends are the most important part of the critical days, this paper mainly focuses on how to enhance the day-ahead prediction accuracy on the weekend load data with verification across three data sets.

The contributions of this paper can be summarized as follows:

- ***Improving the data utilization via data processing***: To boost the deep learning model's ability to analyze trends, peaks, and other relevant information, we process the data through data denoising and data augmentation technique: data denoising adopt data decomposition technology to remove the noise present in time series data, while data augmentation incorporates daily peak variables and trend variables into the denoised data, enabling deep learning models to achieve effective training outcomes even with limited data availability.
- ***Effectively enhance the forecasting framework's attention on weekends***: In this paper, a customized deep learning architecture ***CNNs-Transformer*** is adopted. The three-layer ***CNN*** architecture can fully extract various features of time series data, while ***Transformer*** can better identify the relationship between the leading and lagging terms of time series. The probabilistic forecasting results for the following 48 time steps (the entire day) are output directly adopting the multi-output approach. The interval forecasting results under different confidence levels are obtained through the ***QRLoss***, and the loss penalty technology ensures that the model pays more attention to the weekend load data.
- ***Enabling accurate day-ahead probabilistic load forecasting with limited data availability***: The forecasting framework exhibits excellent performance across all indicators (e.g., interval coverage probability (***PICP***), mean interval width (***MPIW***), etc.) on load data sets compared with the rival forecasting models, which can guarantee the interval coverage while minimizing the interval width. The enhancement of the forecasting ability on weekend data is particularly remarkable.

The rest of the paper proceeds as follows: Section 2 mainly introduces the principles of ***VMD*** and the customized probabilistic deep learning. Section 3 covers data selection, parameter design, and evaluation indexes. Section 4 explains in detail the construction process of the load forecasting framework and shows the forecasting results on three data sets. Section 5 demonstrates the improvement of the proposed framework compared with its rivals through four groups of experiments. Section 6 examines the framework's practical application capabilities. Finally, Section 7 delivers the concluding remarks.

2. Methodology

In this section, we first introduce the whole proposed framework and then all the relevant algorithms involved, namely the variational mode decomposition and the customized deep learning. We also highlight the roles of these components by illustrating examples.

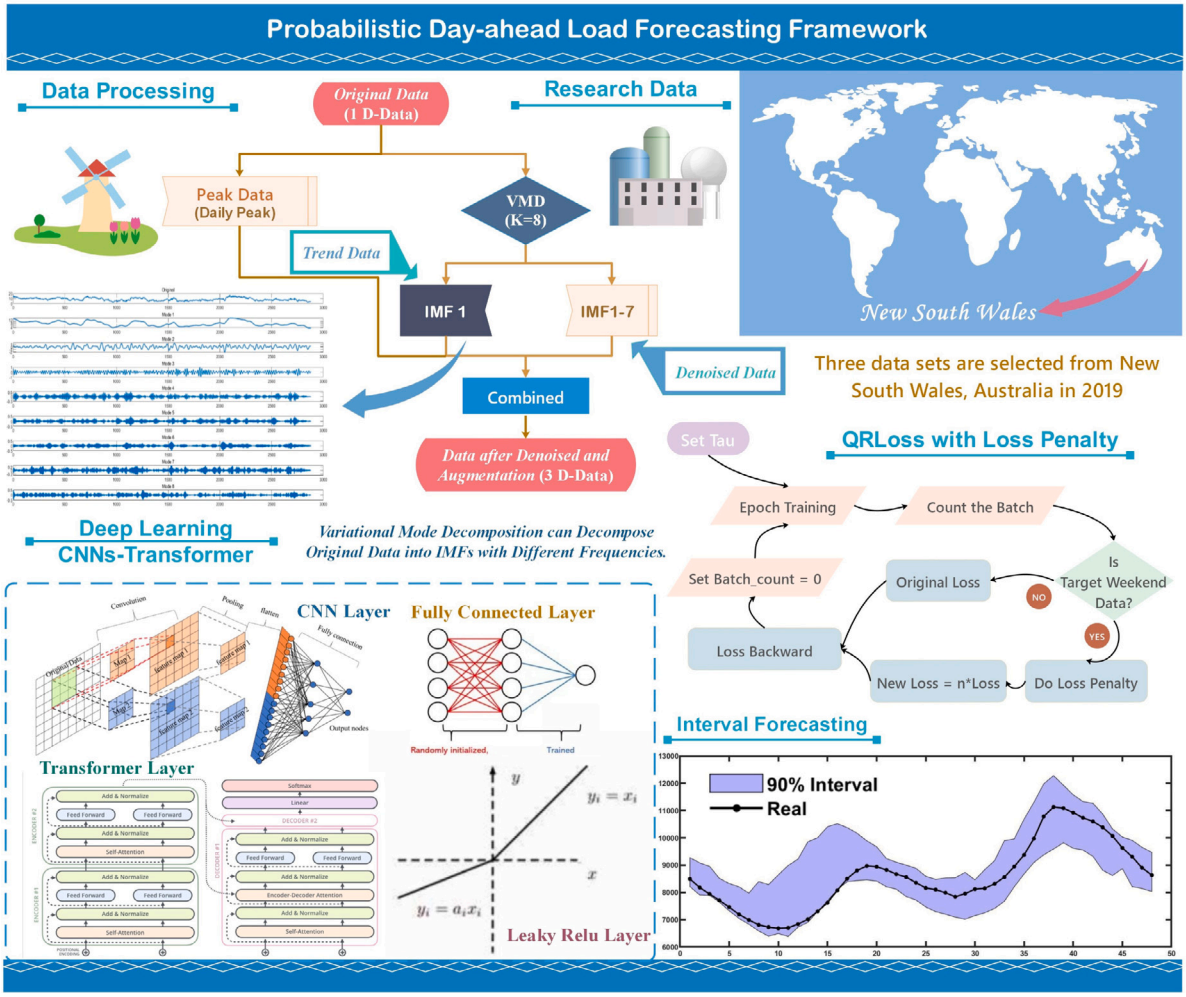

Fig. 1. Flow chart of the whole framework.

2.1. The whole framework

In this paper, a day-ahead probabilistic load forecasting framework is proposed, which can obtain high-quality interval forecasting results for weekends, the framework is composed of two modules, namely, the data processing module and the probabilistic deep learning module. The data processing module enhances the amount of information in the data by data denoising and data augmentation using ***VMD*** method. Even when the data set is limited, the framework can still learn sufficient information from the data. In the probabilistic deep learning module, the multi-layer ***CNN*** and ***Transformer*** structure can deeply extract the trend and pattern and enable day-ahead forecasting (48-time steps of a whole day at a time). Furthermore, the ***QRLoss*** and loss penalty technology are adopted, which further improves the forecasting accuracy on weekends. Fig. 1 shows the flow chart of the proposed load forecasting framework. In the next two sub-sections, the principles of ***VMD*** method and customized deep learning are introduced in detail.

2.2. Variational mode decomposition

Variational mode decomposition (***VMD***) decomposes the original signal by formulating and solving constrained variational problems. This approach effectively mitigates the problems of modal aliasing and boundary effects when decomposing load data [24], while also offering the benefits of customizable decomposition layers and noise resistance. In this study, we employ the ***VMD*** method to decompose the original data, and then extract the trend and the noise component, enabling the subsequent probabilistic load forecasting. The underlying principle of the ***VMD*** method is outlined below:

$$\min_{\{u_k\},\{w_k\}} \left\{ \sum_{k=1}^{K} \left\| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) \times u_k(t) \right] e^{-jw_k t} \right\|_2^2 \right\} \quad s.t. \sum_{k=1}^{K} u_k = f \tag{1}$$

where $\{u_k\}$ and $\{w_k\}$ are the sets of all modes and their center frequencies; f is the original signal; $\delta(t)$ and ∂_t are the Dirac function and the partial derivative regarding time, respectively; K is the number of customized decomposition layers. Augmented Lagrangian function transfers Eq. (1) into an unconstrained optimization problem, which can be solved by alternating direction multiplier method (*ADMM*):

$$L(\{u_k\},\{w_k\},\lambda) = \alpha \sum_{k=1}^{K} \left\| \partial_t [(\delta(t) + j/\pi t) \times u_k(t)] e^{-jw_k t} \right\|_2^2 + \left\| f(t) - \sum_{k=1}^{K} u_k(t) \right\|_2^2 + \left\langle \lambda(t), f(t) - \sum_{k=1}^{K} u(t) \right\rangle \tag{2}$$

Example. In order to highlight the effect of trend information on multi-step forecasting, especially when the data availability is limited. We design a time series signal ($y(t)$, as specified by Eq.(3)), characterized by a gradual upward trend, and use ***LSTM*** to conduct the multi-step forecasting (48-step) with and without trend information respectively.

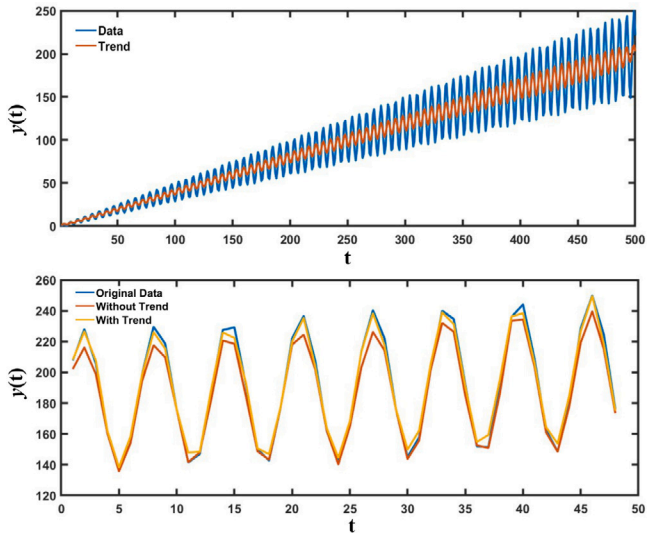

Fig. 2. Comparing the multi-step forecasting results with and without trend information.

Fig. 2 shows the forecasting results.

$$y(t) = \sin(t) \times (1 + 0.1t) + 0.4t, \quad t \in \{0, 1, \ldots, 500\} \tag{3}$$

The results indicate that incorporating trend information enables the model to better capture the upward trend in the data. In contrast, without the trend information, the forecasting results tended to be lower than the actual values, yielding lower accuracy. This demonstrates the benefits of incorporating trend information in multi-step forecasting.

2.3. Customized deep learning framework

The customized deep learning framework consists of a multi-tier architecture, including Convolution Layer, MaxPooling Layer, Leaky-ReLU Layer, Transformer-Encoder Layer, Transformer-Decoder Layer, Fully Connected Layer and Dropoutput Layer. A well-designed combination of these layers can effectively improve the nonlinear expression ability of deep learning.

2.3.1. Convolutional neural network

The convolutional neural network (***CNN***) [25] leverages deeper layers to extract and assemble features from initial and lower layers to create more complex features. In our deep learning framework, we connect three convolution neural networks in series. We then gradually boost the feature expression and nonlinear mapping ability of the deep learning by increasing the number of output channels [26].

Convolution Layer: The aim of the convolution layer is to extract the various characteristics of the input. Eq. (4) can be used to describe the convolution procedure [27]:

$$y_j^{l,\text{conv}} = \sum_{i=1}^{k} \omega_{j,i}^{l} \times y_i^{l-1} + b_j^{l} \tag{4}$$

where $y_j^{l,\text{conv}}$ is the output of channel j in convolution layer l, k is the neurons number in layer $l-1$, $\omega_{i,j}^{l}$ is the weight parameter, and b_j^{l} is a bias value which can aid the model in adapting to the data pattern.

Leaky-ReLU Layer: The derivative of the traditional *ReLU* function is zero when the input value is negative. If a neuron is given a large negative weight during training, the neuron will never be activated, potentially hindering its ability to learn and contribute to the model. *Leaky-ReLU* can produce a non-zero value by introducing a small negative slope, so that more neurons can be activated to achieve more powerful mapping ability. The propagating forward expression of *Leaky-ReLU* [28] is:

$$LeakyReLU(x) = \begin{cases} x, & x > 0 \\ Leak \times x, & x \le 0 \end{cases} \tag{5}$$

In the process of backward propagation, the partial derivatives of the loss function L with respect to the layer l output x^l are denoted by δ^l and its dynamics can be characterized as follows:

$$\delta^l \text{ in LeakyReLU} = \begin{cases} \delta^{l+1}, & \text{if } x^i > 0 \\ Leak \times \delta^{l+1}, & \text{if } x^i \le 0 \end{cases} \tag{6}$$

Max-Pooling Layer: In order to reduce the characteristic dimension of data while retaining the main features, we need to divide the features into multiple regions and perform the pooling operation:

$$y_{kij} = \max_{(p,q) \in R_{ij}} x_{kpq} \tag{7}$$

According to Eq. (8), we take the maximum pooling approach, where y_{kij} represents the maximum pooling output value on the rectangular region R_{ij}, and x_{kpq} represents the element at position (p, q) [29].

2.3.2. Transformer

Transformer adopts *Encoder-Decoder* structure [30], which can better capture long-distance dependencies in time series data, making it suitable for forecasting load data with certain periodicity [31].

Encoder Layer: In the encoder, the data will first be weighted by the self-attention mechanism to obtain a new feature vector $\boldsymbol{H}$, and $\boldsymbol{H}$ can be expressed as [32]:

$$H = \text{Attention}(Q, K, V), \tag{8}$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{9}$$

The value of $\boldsymbol{K}$, $\boldsymbol{Q}$, $\boldsymbol{V}$ can be obtained as follows:

$$Q = W^Q \times X, \qquad K = W^K \times X, \qquad V = W^V \times X \tag{10}$$

where $\boldsymbol{X}$ represents the input vector and $\boldsymbol{W}$ is the corresponding weight matrix.

After obtaining $\boldsymbol{H}$, it will be sent into the feed forward neural network (*FFnn*), which is a two-layer fully connected network, the activation functions are separately ***ReLU*** and linear activation functions:

$$\text{FFnn}(W) = \max(0, HW_1 + b_1)W_2 + b_2 \tag{11}$$

Decoder Layer: Compared with the encoder, a layer named Encoder-Decoder Attention is included [33]. The formula of this mechanism is $Encoder - Decoder\ Attention(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$, which is used to calculate the weights of input and output respectively [34]. In order to better analyze the absolute position and relative position of data, the idea of location coding is introduced into transformer architecture, and we adopt positional encoding (***PE***), whose coding formula is as follows [35]:

$$\boldsymbol{PE}(p, 2i) = \sin\left(\frac{p}{1000^{\frac{2i}{d}}}\right), \qquad \boldsymbol{PE}(p, 2i+1) = \cos\left(\frac{p}{1000^{\frac{2i}{d}}}\right) \tag{12}$$

where p is the abbreviation of position, indicating the location of the data, i represents the index of the vector, and d represents the dimension of the embeddings.

2.3.3. Loss penalty technology based quantile regression loss

In the deep learning framework proposed in this paper, we use the loss function based on quantile regression (***QRLoss***). Quantile regression can set quantile to fit the distribution of data under varying confidence levels to obtain the results of interval prediction [36], which can provide more effective information compared with point prediction.

At the same time, in order to enhance the model's attention to the data which is more difficult to forecast (weekend load data), a penalty mechanism is added to the loss function and a penalty operator is set according to the proportion of unbalanced data.

QRLoss: In deep learning, models typically aim to minimize the mean squared error by bringing the predicted value as close as possible to the actual value [37]. However, for probabilistic forecasting, we are often interested in the complete distribution of the target variable Y. Quantile regression integrates a concept of quantiles into the traditional loss function, which allows us to capture different percentiles of the distribution.

$$MSE = \frac{\left(\sum_{i=1}^{n}(Y_i - \hat{f}(x_i))\right)^2}{N} \tag{13}$$

The ***QRLoss*** is designed to meet specific quantile constraints. For example, when the quantile is 0.9 ($\tau = 0.9$), the model is expected to contain 90% of the true values under the predicted curve. Conversely, for a quantile of 0.1 ($\tau = 0.1$), the model aims to cover 10% of the true values. The interval between these two predictions represents the probabilistic forecasting results when the Prediction Interval Nominal Confidence (PINC) is 0.8. The basic form of quantile regression (denoted by $\hat{Q}_Y(\cdot)$) and the ***QRLoss*** expression used in this paper are as follows:

$$\hat{Q}_Y(\tau) = \arg\min_{\xi_\tau \in R}\left(\sum_{i:Y_i \geq \xi_\tau} \tau \times |Y_i - \xi_r| + \sum_{i:Y_i \geq \xi_\tau} (1-\tau) \times |Y_i - \xi_r|\right) \tag{14}$$

$$QRLoss = \max\left\{\begin{matrix}\tau \times errors \\ (\tau - 1) \times errors\end{matrix}\right\}, \quad errors = Target - Predicts \tag{15}$$

Loss Penalty Technology: In order to make the customized deep learning model pay more attention to the forecasting accuracy on weekends, the loss penalty technique is adopted on the loss function. The specific operation mode is as follows:

$$\begin{aligned} \boldsymbol{Loss} &= \begin{cases} 1 \times \boldsymbol{Loss} & \text{if Target} \notin Weekend\ Data \\ \Delta \times \boldsymbol{Loss} & \text{if Target} \in Weekend\ Data \end{cases} \\ \Delta &= \frac{\text{num}(Weekday) \text{ in Dataset}}{\text{num}(Weekend) \text{ in Dataset}} \end{aligned} \tag{16}$$

In order to realize this idea, we introduce a ***Batch-count*** operator in the training process. At the start of each epoch, the count is initialized and reset to zero at the start of the next epoch. The value of the ***Batch-count*** operator is set as a parameter and sent to ***QRLoss***. The loss penalty is activated only when the target belongs to weekend data. Fig. 3 shows the flow chart of the deep learning framework.

3. Data selection, parameter design and evaluation indexes

In this section, we mainly introduce the selection of load data sets, the super and internal parameters of the load forecasting framework, and then introduce the evaluation indicators.

3.1. Data selection

In terms of data selection, we selected three sets of load data from April, August and December in 2019, New South Wales, Australia to cover all the seasonal patterns [38]. Upon checking, we found that the original data does not contain any missing values. To evaluate the framework's forecasting performance for the case when the data availability is limited, we only use two weeks of historical data (1D load data only) to train the model (the frequency is 30 min, 48 time points per day), and verify the day-ahead forecasting results of the framework on weekends (output the forecasting results of the next 48 time points at once).

As we aim to evaluate the day-ahead load forecasting framework's performance on Saturday and Sunday, we divide the data into separate groups to accommodate different tasks. The first partition model is to train with two full-week data (14 days) and forecast the load for the whole Saturday after 23:30 on Friday. The second is to train with two full-week data plus Saturday data (15 days) and forecast the load for the whole Sunday after 23:30 on Saturday. The division pattern and the characteristics of the data are shown in Table 1 and Fig. 4.

3.2. Parameter design

This paper employs data processing technology and probabilistic deep learning. The processing procedure (i.e., data denoising and data augmentation) is mainly realized by ***VMD***, and the customized deep learning adopts the ***CNNs-Transformer*** architecture based on the Pytorch framework. In this paper, we adopt grid search to optimize the super-parameters of ***VMD*** (α and K) and deep learning (Time-step, Batch-size, and Max-epoch) and show the results in Table 2. Table 3 shows the structure and internal parameters of the ***CNNs-Transformer*** based deep learning framework.[1]

3.3. Evaluation indexes

In the task of day-ahead load forecasting with limited data set availability, the interval forecasting method based on quantile regression is adopted. The interval forecasting results at various confidence levels can be obtained by preset quantiles. For each group of data, we forecast the interval results under ***85%***, ***90%*** and ***95%*** confidence levels, respectively. In order to effectively assess the ability of the electrical load forecasting framework to provide effective information under different confidence levels. In this paper, we adopt three evaluation indicators: ***Prediction Interval Coverage Probability (PICP)*** [39], ***Mean Prediction Interval Width (MPIW)*** [40] and ***Average Interval Score (AIS)*** [41]. Their calculation formulas are as follows:

$$PICP = \frac{1}{n}\sum_{i=1}^{n} I\left(T_i \in [L(Pi), U(P_i)]\right) \tag{17}$$

$$MPIW = \frac{1}{n}\sum_{i=1}^{n}(U(P_i) - L(P_i)) \tag{18}$$

$$AIS = \frac{1}{n}\sum_{i=1}^{n} S(P_i)$$

$$S(P_i) = \begin{cases} -0.02 \times (U(P_i) - L(P_i)) - 4 \times (L(P_i) - T_i) & \text{if } T_i < L(P_i) \\ -0.02 \times (U(P_i) - L(P_i)) & \text{if } L(P_i) \leq T_i \leq U(P_i) \\ -0.02 \times (U(P_i) - L(\mathrm{P}_i)) - 4 \times (T_i - U(P_i)) & \text{if } T_i > U(P_i) \end{cases} \tag{19}$$

where $U(P_i)$ represents the forecasting result at the upper bound of the interval of i, $L(P_i)$ represents the forecasting result at the lower bound of the interval of position i, T_i represents the true value of position i, and $I(\cdot)$ is the indicator function.

The ***PICP*** index measures the coverage of the interval. If the value of ***PICP*** is greater than the preset confidence (***PINC***), we consider the interval prediction result to be valid. The ***MPIW*** index measures the width of the interval. When the interval coverage rate (***PICP***) satisfies the condition, a narrower interval can provide more abundant information. ***AIS*** is the evaluation of interval quality, which comprehensively considers the coverage and width of the interval, offering the interval a relevant score. The higher the score of the interval, the higher the quality of the interval. Through the three indicators, we can effectively evaluate the performance of each model on the day-ahead probabilistic load forecasting with limited data availability in the experiment.

[1] Due to space limits, ***Leaky-ReLU Layers***,***Dropoutput Layers*** and ***Fully Connected Layers*** are not showed in Table 3.

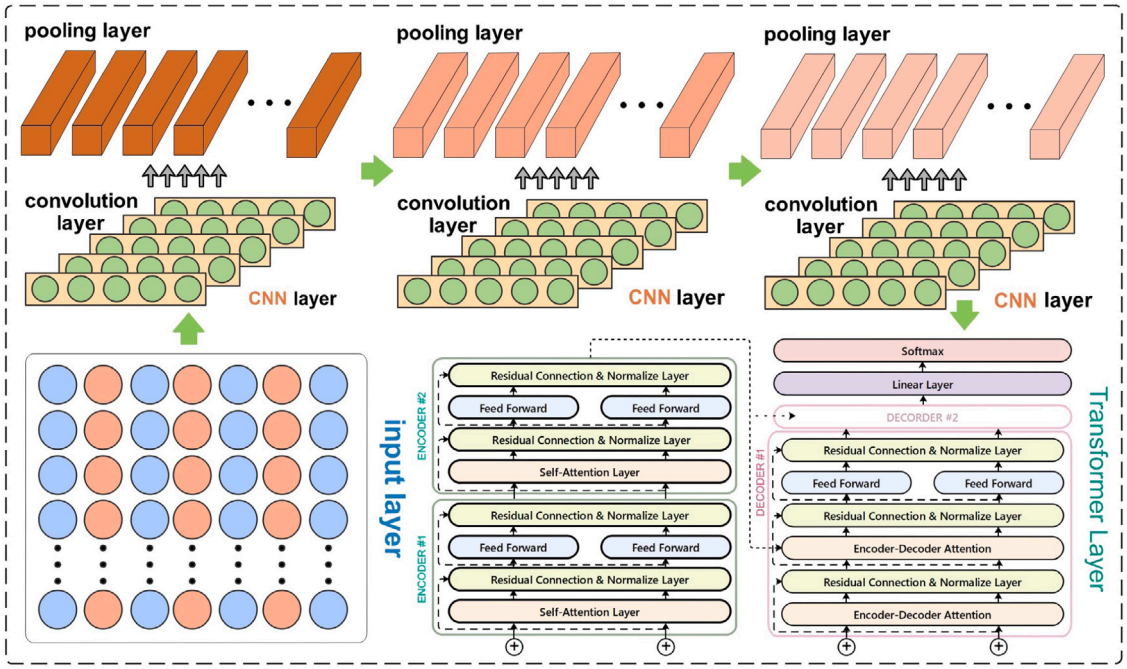

Fig. 3. Flow chart of the deep learning framework.

Table 1
Characteristics of three load data sets (MW).

	Target Day	Training Set	Testing Set	Max	Min	Mean	Var	Std
Data 1	Saturday	672	48	9703.26	5701.45	7299.65	662 886.04	814.18
	Sunday	720	48	9703.26	5701.45	7299.65	662 886.04	814.18
Data 2	Saturday	672	48	11 128.43	6633.020	8507.30	1208256.41	1099.21
	Sunday	720	48	11 128.43	6633.020	8507.30	1208256.41	1099.21
Data 3	Saturday	672	48	11 454.80	5634.88	7594.57	1334214.93	1155.08
	Sunday	720	48	11 454.80	5634.88	7594.57	1334214.93	1155.08

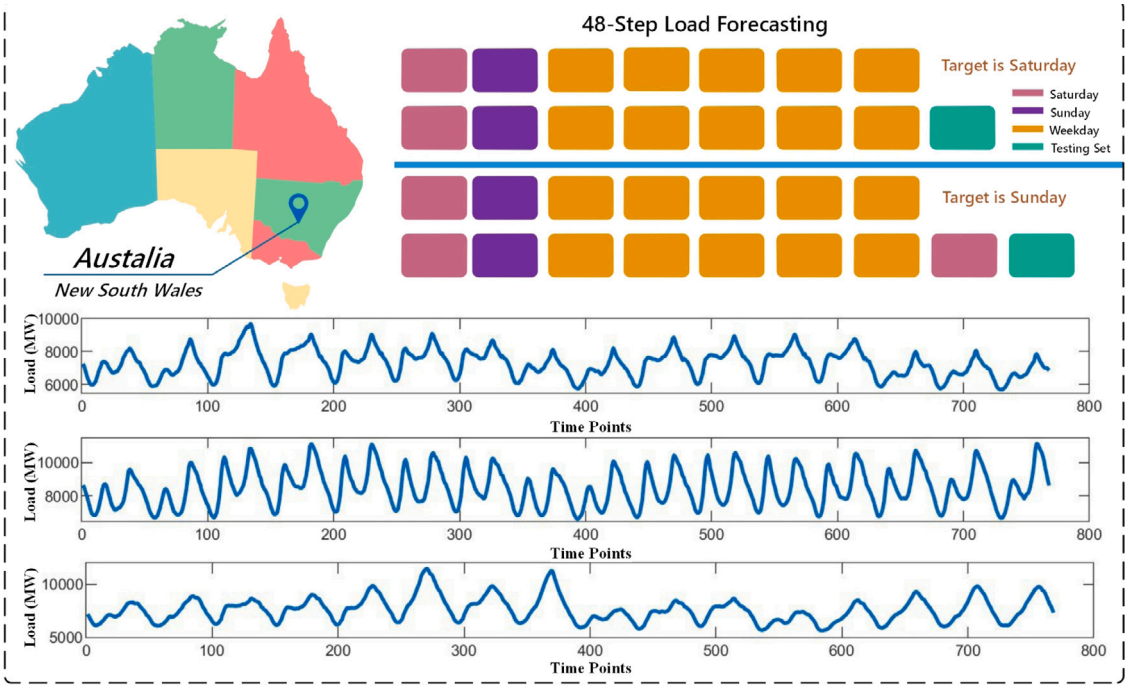

Fig. 4. Load Data in New South Wales, Australia.

Table 2
Super-parameters of the load forecasting framework.

Model	Parameters	Value
VMD	Signal	Data
	α	1,000
	τ	0
	K (Number of decomposition layers)	8
	DC	0
	Init	1
	Tol	1e−6
Deep Learning	Time step	336 (48 × 7)
	Time step feature	3
	Batch size	48
	Max epoch	200
	Output Size	48

Table 3
Internal-parameters of deep learning.

	Parameters	Value
Cov1d 1	Input Channels	Time step feature
	Output Channels	32
	Kernel Size	3
MaxPool 1	Kernel Size	3
	Stride	2
Cov1d 2	Input Channels	32
	Output Channels	64
	Kernel Size	3
MaxPool 2	Kernel Size	3
	Stride	2
Cov1d 3	Input Channels	64
	Output Channels	128
	Kernel Size	3
MaxPool 3	Kernel Size	3
	Stride	2
Transformer Encoder	Model Dimension	96
	Head	12
Transformer Decoder	Model Dimension	96
	Head	12

3.4. Operation environment

The basic operating environment of the proposed load forecasting framework is ***Intel Core i9-13980HX*** and ***RTX4070Ti***. In this paper, ***Matlab 2023a*** is used for data processing, including trend extraction and data de-noising, and we adopt ***PyTorch*** by ***PyCharm*** to build the deep learning framework [42].

4. Module design and performance evaluation

In this section, we will introduce in detail the two modules of the whole load forecasting framework and present the numerical results on three data sets.

4.1. Data processing module design

4.1.1. Design principle

In the context of limited datasets, extracting meaningful information from the original data can be particularly challenging. This is particularly true for weekend load forecasts, where, although there is a close correlation with weekday loads, there are inherent differences in energy consumption patterns and peak times. These variances do not conform to the regular patterns observed in weekday data. If these nuances are not accurately captured, particularly the discrepancies in energy consumption patterns between weekends and weekdays, the predictive accuracy of the model on weekends may suffer significantly. To investigate these discrepancies in-depth, ***VMD*** is employed as the primary technical strategy in this research. By decomposing load data into components with distinct frequencies, we can discern and analyze the disparities in these components across weekend and weekday intervals. Concurrently, the Manhattan distance is calculated to quantify the similarity between different Intrinsic Mode Functions (IMFs). We have selected a dataset from ***Data 1*** for Thursday and Friday, as well as Saturday and Sunday, for visual inspection. Fig. 5 presents the visualized results of the IMFs, with the similarity of each IMF on weekdays versus weekends detailed in Table 4. The formula for calculating the percentage of similarity among different IMFs is as follows:

Table 4
Similarity of each IMF on weekdays and weekends.

	IMF 1	IMF 2	IMF 3	IMF 4	IMF 5	IMF 6	IMF 7
Data 1	49.56%	89.51%	89.65%	86.93%	83.92%	82.51%	85.13%
Data 2	52.17%	89.62%	84.65%	82.81%	86.39%	81.92%	84.94%
Data 3	59.45%	88.83%	85.96%	86.49%	89.43%	90.23%	84.64%

$$\mathrm{PS} = \left(1 - \frac{\sum_{i=1}^{n} |x_i - y_i|}{n \times (\max(\max(x), \max(y)) - \min(\min(x), \min(y)))}\right) \times 100\% \quad (20)$$

where PS is the percentage of similarity of different IMFs, x and y present different IMFs.

Experimental outcomes indicate that among the eight IMFs, the divergence between weekends and weekdays is predominantly manifested in the first IMF, while the similarity in other components exceeds 80%. Additionally, the last component is nearly random and can often be regarded as a white noise component. This analysis lays the groundwork for our data processing approach, which includes data augmentation and denoising techniques. The specific procedures are as follows:

Data augmentation: The daily peak load value is extracted and expanded based on the daily timestamp to create a novel data dimension. This method enhances the model's focus on daily peak characteristics, enabling more precise day-ahead predictions of peak information.

VMD is used to decompose the input data into eight components with varying frequencies. These components are arranged in ascending order of frequency, each representing a unique characteristic of the data. The component with the lowest frequency, serving as the trend component, is integrated into the original data as an additional dimension, aiding the model in effectively capturing trend information. For instance, there is a notable downward trend in electricity consumption during weekends compared to weekdays.

Data denoising: Following the ***VMD*** decomposition, the component with the highest frequency is assumed to be Gaussian white noise and is discarded. The remaining seven components are recombined to form the denoised data. Compared to the original data, the denoised data mitigates outliers and abnormal fluctuations, thereby bolstering the robustness and generalization capacity of the deep learning model during the training process under the constraints of limited datasets.

Through these processes, the input data format for the custom deep learning model evolves from [Original Data], a unidimensional ***(1D)*** format, to [Peak Data, Trend Data, Denoised Data], a multidimensional ***(3D)*** format.

4.1.2. Performance evaluation

After processing the three groups of data according to the above method, we draw the figure and analyze the features of the processed data. Fig. 6 compares data after processing with the original data. In Table 5, we calculate the characteristics of each data dimension through statistical indicators. Due to the size limitation of the figure, it is difficult to directly observe the difference between before and after denoising in Fig. 6. Hence, we include a small window on the right side of Fig. 6 to enlarge the local size of the time series to highlight the de-noising effect.

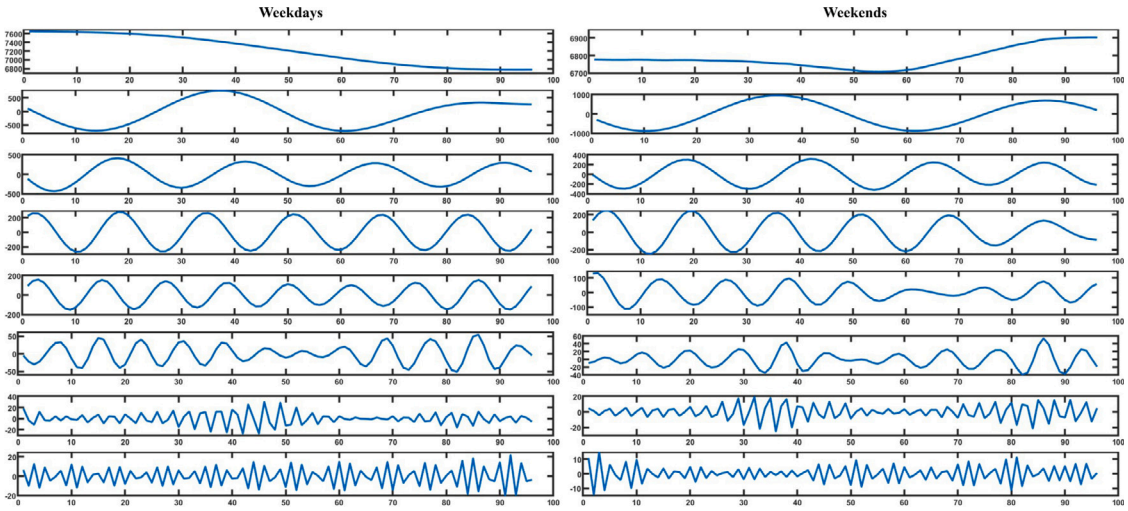

Fig. 5. IMFs of weekdays and weekends of Data 1 (*x*-axis: Time points, *y*-axis: MW).

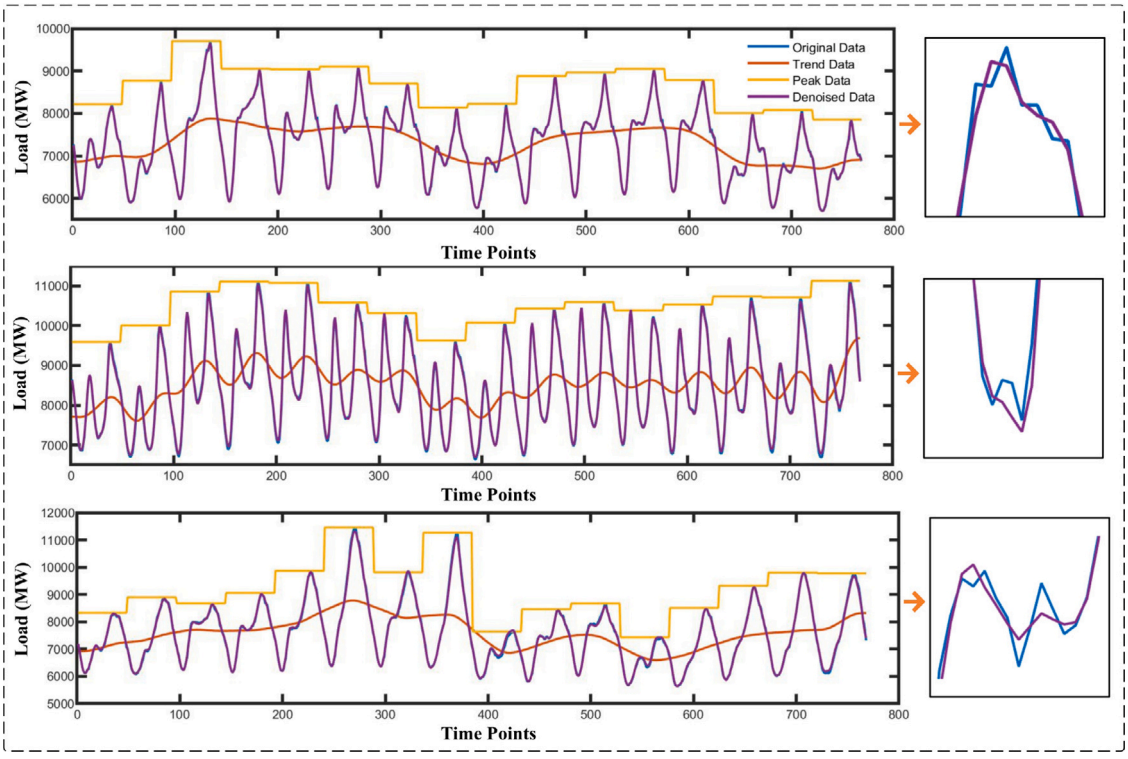

Fig. 6. Data processing based on data denoising and data augmentation.

4.2. Day-ahead deep learning module design

4.2.1. Design principle

After data processing, it is crucial to construct a deep learning framework that can effectively learn and extract the internal information of the data. In this paper, a deep learning framework is developed using the ***CNNs-Transformer*** approach. The reasons for constructing this framework are as follows:

1) It is assumed that the time steps of each time series are related to all time points of the previous ***7*** days, and the granularity of the data is ***30min***, so the time-step is ***336*** (7×48). A large time-step may make it difficult for the deep learning framework to extract local features of the data. Therefore, we choose ***CNN*** which can effectively capture the local characteristics of time series, such as short-term dependence, periodicity or trend, through a series of convolution layers and max-pooling layers. At the same time, it can also effectively reduce the computational burden.

2) Although the ***CNNs*** can effectively extract the features of the data, the time series data are dependent on each other, and the ***CNN*** can hardly capture the long-term dependency in the series, motivating the employment of the ***Transformer*** structure. ***Transformer*** can not only capture the dependency relationship between the data, but also learn different attention weights, thus capturing the feature representation of different concerns, which is very helpful for multiple feature sources in time series prediction.

3) Since our framework needs to forecast the load data for the entire next day at one time on the night before, which is a 48-step prediction, we employ a direct multi-output strategy. This strategy sets the output

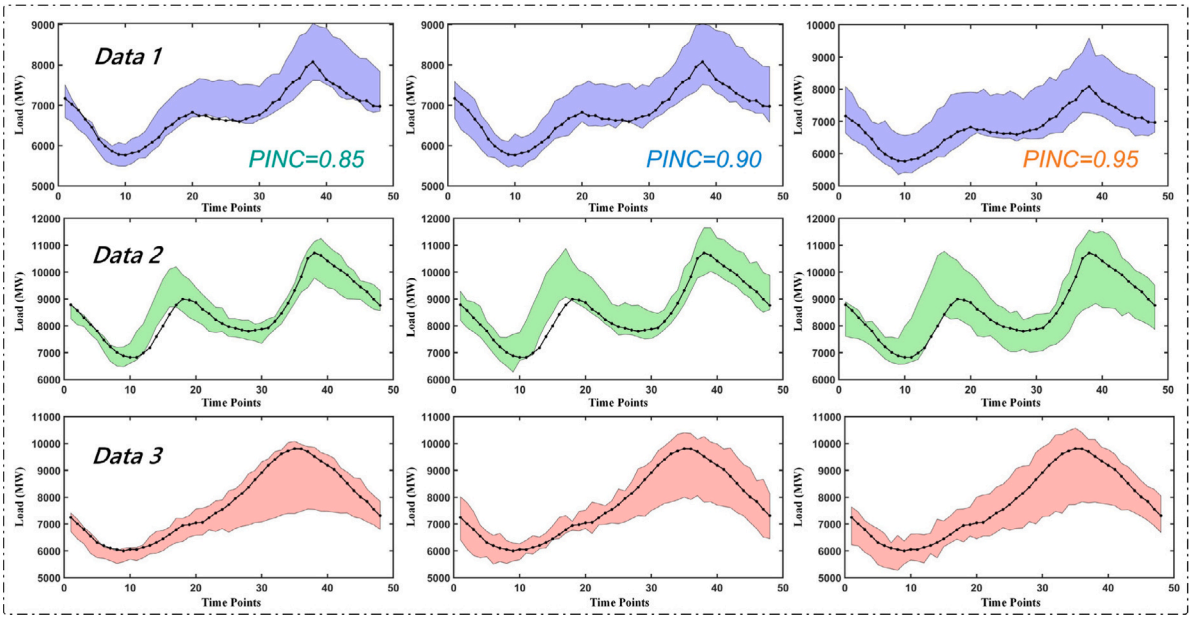

Fig. 7. Day-ahead load forecasting results on Saturday.

Table 5
Characteristics of each data dimension.

	Dimension	Max	Min	Mean	Var	Std
Data 1	Original Data	9703.26	5701.45	7299.65	662 886.04	814.18
	Trend	7878.22	6702.54	7299.65	136 225.96	369.09
	Peak	9703.26	7854.60	8660.28	249 202.32	499.20
	Denoised Data	9657.54	5700.79	7299.74	658 781.26	811.65
Data 2	Original Data	11 128.43	6633.02	8507.30	1208256.41	1099.21
	Trend	9691.03	7611	8507.30	167 040.37	408.71
	Peak	11 128.43	9594.36	10 485.15	212 262.44	460.72
	Denoised Data	11 052.01	6690.82	8507.30	1159766.51	1076.92
Data 3	Original Data	11 454.80	5634.88	7594.57	1334214.93	1155.08
	Trend	8783.88	6594.13	7594.57	285 640.50	534.45
	Peak	11 454.80	7430.68	9186.71	1182765.92	1087.55
	Denoised Data	11 310.92	5634.04	7594.95	1297795.27	1139.21

size as ***48***, allowing the deep learning model to output the results for multiple time steps in the future simultaneously.

4.2.2. Performance evaluation

For each set of data, we carried out two sets of experiments, namely, on Friday night, using 14 days of historical data to output the day-ahead forecasting results of Saturday all-day load data at one time, and on Saturday night, using 15 days of historical data (adding Saturday data) to output day-ahead forecasting results of Sunday all-day load data at one time. The forecasting results are shown in Figs. 7, 8 and Table 6.

Based on the interval forecasting results, it is clear that no matter on Saturday or Sunday, the proposed day-ahead forecasting framework can achieve high-quality probabilistic forecasting results with a high coverage rate and appropriate interval width. These experiment results demonstrate the practical applicability of the framework, particularly in situations with limited data availability.

5. Numerical validation

In this section, we design four sets of experiments and conduct a comparative analysis over three data sets to validate the necessity of data processing techniques, the improvement degree of customized deep learning compared with the baseline models, the effectiveness of loss penalty techniques and sensitivity of the forecasting framework to hyper-parameters.

5.1. The necessity of data processing

The experiment starts by verifying the necessity for data processing. We compare the ***3D-Data*** processed by data denoising and data augmentation with the removal of ***Peak*** features, removal of ***Trend*** features, and the ***Original data*** respectively, while we keep the deep learning structure ***CNNs-Transformer*** unchanged. The experiment results show that the quality of the forecasting interval is enhanced after data processing. Table 7 shows the results of the experiment.

In all three datasets, the forecasting interval quality is significantly enhanced on both Saturdays and Sundays after applying the data denoising and data augmentation technique. In Data 1, when ***PINC*** is ***85%***, the ***PICP*** value of 3D-Data on Saturday is ***92%***, which is ***25%*** higher compared with the original data-based forecasting results. In addition, the mean interval width (***MPIW***) is ***53*** lower than the original data-based forecasting results. When the ***PINC*** is ***90%***, although there is little difference in the ***PICP*** values among the four data processing methods, it is evident that 3D-Data outperforms the other methods when evaluating using the ***AIS*** metric. On Sunday, with a ***PINC*** of ***95%***, the ***PICP*** indicator reaches ***100%***, achieving full coverage of the probability interval. Furthermore, the ***AIS*** value significantly improves to ***-18.32***, surpassing other data processing methods.

In Data 2, when ***PINC*** is ***0.85***, 3D data achieves an interval coverage of ***85%*** on Saturday. Although the ***PICP*** remains unchanged in comparison to the prediction result without trend information, the interval becomes narrower when trend information is included. More specifically, the inclusion of trend information reduces the ***MPIW*** index by nearly ***100***, thereby enhancing the interval's quality. Furthermore, when we do not denoise the data, the ***PICP*** value is ***77%***, when the peak information is not added, the ***PICP*** value is ***83%***, which are both lower than the confidence level of ***85%***. In contrast, the data processing makes the interval forecasting results of the proposed framework have higher coverage and narrower interval width. When the ***PINC*** is ***90%***, although 3D-Data's ***MPIW*** is a bit higher than the ***MPIW*** without peaks

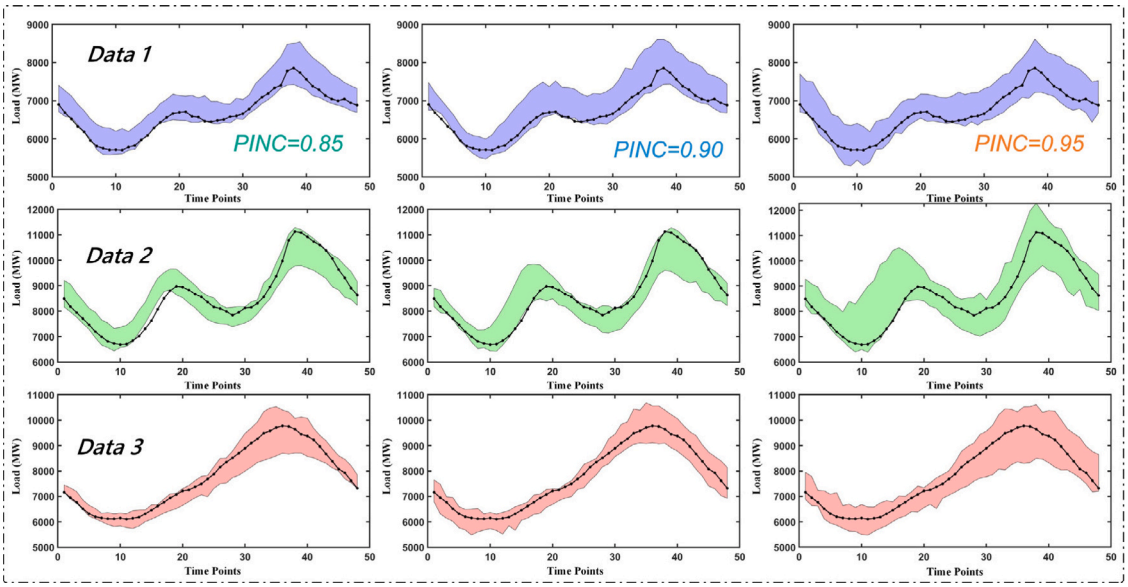

Fig. 8. Day-ahead load forecasting results on Sunday.

Table 6
Day-ahead load forecasting results.

		PINC = 0.85			PINC = 0.90			PINC = 0.95		
		PICP	MPIW	AIS	PICP	MPIW	AIS	PICP	MPIW	AIS
Data 1	Saturday	0.92	917.42	−36.42	1.00	1056.37	−21.13	1.00	1509.27	−30.19
	Sunday	0.85	677.51	−27.84	0.92	824.97	−49.83	1.00	915.81	−18.32
Data 2	Saturday	0.85	943.80	−123.30	0.90	1278.59	−205.17	0.98	1769.48	−36.12
	Sunday	0.88	964.91	−129.30	0.92	1077.81	−48.47	0.96	1715.77	−42.51
Data 3	Saturday	0.94	1323.43	−36.50	1.00	1434.05	−28.68	1.00	1682.00	−33.64
	Sunday	1.00	887.89	−17.76	1.00	963.69	−19.27	1.00	1401.64	−28.03

on Sunday, according to the interval quality index (***AIS***), it achieves better interval quality. Based on the experimental results, it can be observed that data processing has a greater impact on ***PICP*** index when the confidence level is lower. As the confidence level increases, data processing becomes more effective in producing a narrower interval while maintaining coverage, thereby increasing the amount of information provided by the interval.

Remark 1. Experiment 1 validates the importance of the data processing in comparison to scenarios without denoising, trends, and peaks. It is evident that the quality of interval prediction results is significantly better after processing the original data through data denoising and data augmentation techniques.

5.2. The necessity of customized deep learning

In this experiment, the necessity of the customized deep learning structure is verified. We compare the interval prediction results of probabilistic deep learning with basic prediction models such as ***LSTM***, ***CNN*** and ***Transformer***, deep learning models such as ***LSTM-Transformer*** and ***BiLSTM-Transformer*** and evaluate the enhancement of prediction performance achieved by customized deep learning compared with basic models on limited data sets. We also compared the customized deep learning structure with some latest models from the literature review, namely Model 1: Ensemble Strategy and Tuna swarm optimization (TSO) [20], Model 2: RMLLSSVR [18], and Model 3: Curve-to-curve Regression [19]. It is worth noting that since there is still little literature for day-ahead probabilistic load forecasting, we have changed Model 2 from the original point forecasting to probabilistic forecasting to complete this experiment. In order to ensure the control of variables in the experiment, we input the same processed 3D data for all models. Table 8[2] shows the inter-parameters of some basic models. The super-parameters of these models are optimized by grid search. Table 9 and Table 10 presents the experimental results.

Evidently, in all the data sets, no matter in terms of interval coverage or interval width, the customized deep learning (***CNNs-Transformer***) achieves better probabilistic prediction results compared with all rival models. In Data 1, when ***PINC*** is set as ***85%***, the ***PICP*** of the load forecasting framework is ***92%*** on Saturday, which is much higher than the other basic models (***CNN:23%***, ***LSTM:54%***, ***Transformer:77%***, ***LSTM-Transformer:83%*** and ***BiLSTM-Transformer:81%***). Compared with day-ahead load forecasting models from the latest literature, the proposed deep learning structure still maintains the best, which is ***13%*** higher than Model 1, ***27%*** higher than Model 2, and ***25%*** higher than Model 3. In terms of ***MPIW***, although the interval width of ***CNN*** is very narrow, its interval coverage is much lower than the confidence level, so its prediction results are invalid. Among the effective prediction results, the confidence interval obtained by ***CNNs-Transformer*** structure is the narrowest.

In Data 2, the ***CNNs-Transformer*** structure also outperforms not only baseline models but also the latest models, and achieves a significant improvement. The interval coverage probability value ***PICP*** is ***98%*** when ***PINC*** is ***95%*** on Saturday, which is the highest among all the models. On Sunday, the ***PICP*** of the forecasting framework is ***92%*** when the ***PINC*** is ***90%***, which is ***71%*** higher than both ***LSTM*** and ***BiLSTM-Transformer***, ***84%*** higher than ***CNN***, and ***4%*** higher than ***Transformer***.

[2] Due to space limits, ***Leaky-ReLU Layers***,***Dropoutput Layers*** and ***Fully Connected Layers*** are not showed in Table 7.

Table 7
Prediction results of three sets of data in Experiment 1.

			PINC = 0.85			PINC = 0.90			PINC = 0.95		
			PICP	MPIW	AIS	PICP	MPIW	AIS	PICP	MPIW	AIS
Data 1	Saturday	Without Data-denoising	0.67	970.13	−267.08	0.75	633.94	−115.15	1.00	1588.53	−31.77
		Without Trend	0.92	1243.44	−38.82	1.00	1275.96	−25.52	1.00	1543.56	−30.87
		Without Peak	0.90	1045.30	−44.11	1.00	1281.79	−25.64	1.00	1522.65	−30.45
		3D-Data	**0.92**	**917.42**	**−36.42**	**1.00**	**1056.37**	**−21.13**	**1.00**	**1509.27**	**−30.19**
	Sunday	Without Data-denoising	0.83	550.84	−29.10	0.88	836.64	−57.08	1.00	988.99	−19.78
		Without Trend	0.81	801.01	−62.98	0.90	944.08	−58.11	0.98	1022.72	−23.86
		Without Peak	0.65	464.74	−56.81	0.81	701.17	−63.10	0.94	842.96	−22.77
		3D-Data	**0.85**	**677.51**	**−27.84**	**0.92**	**824.97**	**−49.83**	**1.00**	**915.81**	**−18.32**
Data 2	Saturday	Without Data-denoising	0.77	858.53	−368.83	0.88	1415.85	−261.85	0.90	1949.69	−195.78
		Without Trend	0.85	1053.89	−345.16	0.83	1303.18	−308.25	0.96	2178.48	−46.13
		Without Peak	0.83	887.07	−260.59	0.85	1284.75	−224.11	0.85	1893.88	−266.76
		3D-Data	**0.85**	**943.80**	**−123.30**	**0.90**	**1278.59**	**−205.17**	**0.98**	**1769.48**	**−36.12**
	Sunday	Without Data-denoising	0.85	1046.51	−159.60	0.90	1240.21	−72.59	0.94	1725.90	−52.33
		Without Trend	0.69	1061.85	−256.51	0.88	1206.79	−56.64	0.94	1864.83	−52.71
		Without Peak	0.75	975.61	−186.73	0.85	996.36	−57.77	0.90	1535.28	−99.67
		3D-Data	**0.88**	**964.91**	**−129.30**	**0.92**	**1077.81**	**−48.47**	**0.96**	**1715.77**	**−42.51**
Data 3	Saturday	Without Data-denoising	0.90	1417.98	−43.19	1.00	1612.34	−32.25	1.00	2195.68	−43.91
		Without Trend	0.92	1954.64	−54.69	1.00	2383.18	−47.66	1.00	2595.59	−51.91
		Without Peak	0.69	731.06	−160.59	0.81	1079.87	−62.57	1.00	2324.72	−46.49
		3D-Data	**0.94**	**1323.43**	**−36.50**	**1.00**	**1434.05**	**−28.68**	**1.00**	**1682.00**	**−33.64**
	Sunday	Without Data-denoising	0.96	1054.80	−29.00	1.00	1700.97	−34.02	1.00	1886.46	−37.73
		Without Trend	1.00	1398.82	−27.98	1.00	1311.38	−26.23	1.00	2895.65	−57.91
		Without Peak	0.96	787.52	−24.10	1.00	1581.58	−31.63	1.00	1869.86	−37.40
		3D-Data	**1.00**	**887.89**	**−17.76**	**1.00**	**963.69**	**−19.27**	**1.00**	**1401.64**	**−28.03**

Table 8
Parameters of the rival models in Experiment 2.

Model	Layer	Parameters	Value
LSTM	LSTM	Layer Number	1
		Hidden Size	256
CNN	Conv	Out Channels	64
		Kernel Size	3
		Padding	1
	MaxPooling	Kernel Size	3
		Stride	2
Transformer	Encoder	Dimension	96
		Head	12
	Decoder	Dimension	96
		Head	12
LSTM-Transformer	LSTM	Hidden Size	256
	Encoder	Dimension	96
		Head	12
	Decoder	Dimension	96
		Head	12
BiLSTM-Transformer	BiLSTM	Layer Number	2
		Bidirectional	True
		Hidden Size	128 × 2
	Encoder	Dimension	96
		Head	12
	Decoder	Dimension	96
		Head	12

Although the ***MPIW*** values of ***CNN***, ***LSTM*** and ***LSTM-Transformer*** are lower than that of the customized deep learning, the ***AIS*** composite evaluation metrics show that the quality of the forecasting result by ***CNNs-Transformer*** is significantly better than that of the other baseline models, which demonstrates the effectiveness of our proposed deep learning structure. Compared with the models from the latest literature, although the ***PICP*** of ***Curve-to-curve Regression*** is better, its interval width is significantly higher, causing a lower interval score. With increasing confidence levels, most models will gradually increase their interval coverage and width. However, due to the different principles of each model, there will be two kinds of anomalies. On the one hand, for some prediction models (e.g., ***Transformer*** and ***Curve-to-curve Regression***), to guarantee the interval coverage, their intervals may be too wide to provide enough effective information; On the other hand, some models (e.g., ***CNN*** and ***LSTM-Transformer***) may have narrow intervals, but their coverage is too low to be useful. Only the customized ***CNNs-Transformer*** architecture can provide high-quality probability prediction results.

In Data 3, it can be found that the model has reached ***100%*** coverage of the intervals for Sunday when the ***PINC*** is ***85%***, which is the best prediction result compared with the other five methods. Although the basic model ***Transformer*** also shows good interval coverage in predicting the day-ahead electrical load for Sunday: ***PICP*** value is ***94%*** when ***PINC*** is ***85%***; and when ***PINC*** is above ***90%***, the ***PICP*** value is already ***100%***. When we compare the interval width of ***Transformer*** and ***CNNs-Transformer***, we found that the ***MPIW*** value of the proposed deep learning is much lower than that of the ***Transformer*** model, which proved that the ***CNNs-Transformer*** is again the best. When comparing with models from the latest literature, it is obvious that although the ***PICP*** of most methods is satisfied, the interval of the proposed deep learning structure is the narrowest, which means the proposed structure can obtain the best day-ahead load forecasting results.

Remark 2. Experiment 2 demonstrates the indispensability of customized ***CNNs-Transformer*** deep learning. This architecture can effectively capture data features and derive high-quality prediction results even with limited data sets.

5.3. Effectiveness of loss penalty technology

In this experiment, by comparing interval forecasting results with and without the loss penalty technology, we confirmed that including a loss penalty in the loss function can significantly improve the coverage of prediction intervals on weekends. This improvement is particularly important for data sets with limited samples. Table 11 shows the detailed experimental results. In order to observe the effect of loss penalty more intuitively, we plotted the intervals obtained with the loss penalty technique and without the loss penalty technique on the same graph. Fig. 9 shows the difference between them.

It is evident that in all three data sets, the load forecasting framework incorporating the loss penalty exhibits a significant improvement

Table 9
Prediction results of three sets of data in Experiment 2 (Basic Models).

			PINC = 0.85			PINC = 0.90			PINC = 0.95		
			PICP	MPIW	AIS	PICP	MPIW	AIS	PICP	MPIW	AIS
Data 1	Saturday	LSTM	0.54	437.37	−252.75	0.71	477.99	−127.57	0.77	640.09	−105.54
		CNN	0.23	92.67	−698.40	0.38	352.84	−457.7	0.60	857.54	−322.44
		Transformer	0.77	1417.36	−100.54	0.94	1995.72	−70.11	0.98	2131.13	−57.28
		LSTM-Transformer	0.83	606.18	−62.14	0.85	626.31	−29.66	0.88	926.80	−48.29
		BiLSTM-Transformer	0.81	612.40	−63.72	0.85	672.76	−31.83	0.92	1012.31	−42.13
		CNNs-Transformer	**0.92**	**917.42**	**−36.42**	**1.00**	**1056.37**	**−21.13**	**1.00**	**1509.27**	**−30.19**
	Sunday	LSTM	0.21	421.33	−328.10	0.23	485.35	−346.97	0.50	602.96	−172.17
		CNN	0.08	152.28	−454.60	0.08	225.98	−374.5	0.17	438.06	−278.67
		Transformer	0.94	1121.01	−33.88	0.94	1366.2	−53.68	1.00	1888.63	−37.77
		LSTM-Transformer	0.19	420.19	−424.51	0.46	356.31	−154.04	0.48	590.28	−168.91
		BiLSTM-Transformer	0.21	442.17	−420.72	0.48	326.93	−148.92	0.52	614.97	−152.97
		CNNs-Transformer	**0.85**	**677.51**	**−27.84**	**0.92**	**824.97**	**−49.83**	**1.00**	**915.81**	**−18.32**
Data 2	Saturday	LSTM	0.56	347.50	−616.74	0.73	619.56	−606.56	0.58	785.12	−598.24
		CNN	0.52	283.45	−751.62	0.54	368.22	−686.38	0.69	635.46	−543.65
		Transformer	0.75	2300.27	−225.52	0.94	2852.24	−72.40	0.98	3351.44	−69.99
		LSTM-Transformer	0.46	270.45	−498.56	0.58	357.75	−446.41	0.73	641.58	−427.87
		BiLSTM-Transformer	0.56	324.68	−517.40	0.58	394.98	−543.13	0.75	650.49	−444.21
		CNNs-Transformer	**0.85**	**943.80**	**−123.30**	**0.90**	**1278.59**	**−205.17**	**0.98**	**1769.48**	**−36.12**
	Sunday	LSTM	0.25	387.35	−633.22	0.21	464.09	−620.57	0.33	649.13	−461.71
		CNN	0.31	297.47	−814.35	0.08	364.84	−855.10	0.21	553.53	−929.37
		Transformer	0.85	2444.58	−182.99	0.88	2678.36	−111.11	0.88	3485.2	−110.28
		LSTM-Transformer	0.12	85.14	−1012.33	0.25	430.15	−594.03	0.44	622.85	−412.51
		BiLSTM-Transformer	0.19	166.75	−709.5	0.21	348.98	−601.68	0.40	638.36	−428.58
		CNNs-Transformer	**0.88**	**964.91**	**−129.3**	**0.92**	**1077.81**	**−48.47**	**0.96**	**1715.77**	**−42.51**
Data 3	Saturday	LSTM	0.77	1146.59	−122.03	0.77	1152.77	−118.09	0.81	1371.43	−65.77
		CNN	0.25	619.01	−712.61	0.56	848.70	−608.45	0.62	967.26	−304.38
		Transformer	0.54	2057.48	−393.11	0.83	2918.31	−102.03	1.00	3066.99	−61.34
		LSTM-Transformer	0.19	200.81	−398.81	0.46	821.39	−332.24	0.60	729.78	−283.29
		BiLSTM-Transformer	0.38	468.63	−376.74	0.63	890.72	−210.29	0.69	773.02	−231.54
		CNNs-Transformer	**0.94**	**1323.43**	**−36.5**	**1.00**	**1434.05**	**−28.68**	**1.00**	**1682.00**	**−33.64**
	Sunday	LSTM	0.48	437.30	−387.00	0.65	692.17	−88.82	0.94	1245.51	−33.16
		CNN	0.29	225.59	−276.32	0.25	308.11	−306.70	0.29	493.45	−330.06
		Transformer	0.94	1787.00	−61.20	1.00	2162.54	−43.25	1.00	2938.41	−58.77
		LSTM-Transformer	0.17	104.57	−650.35	0.56	326.89	−273.28	0.52	481.5	−325.19
		BiLSTM-Transformer	0.25	253.26	−365.42	0.48	400.67	−372.20	0.54	477.31	−211.40-
		CNNs-Transformer	**1.00**	**887.89**	**−17.76**	**1.00**	**963.69**	**−19.27**	**1.00**	**1401.64**	**−28.03**

Table 10
Prediction results of three sets of data in Experiment 2 (Models from Latest Literature).

			PINC = 0.85			PINC = 0.90			PINC = 0.95		
			PICP	MPIW	AIS	PICP	MPIW	AIS	PICP	MPIW	AIS
Data 1	Saturday	Ensemble Strategy + TSO	0.79	1001.74	−85.79	0.90	1437.4	−52.6	1.00	1565.84	−31.32
		RMLSSVR	0.65	999.96	−237.70	0.75	1207.34	−97.77	0.98	1855.78	−41.78
		Curve-to-curve Regression	0.67	735.59	−127.91	0.92	1134.04	−42.68	1.00	1635.82	−32.72
		CNNs-Transformer	**0.92**	**917.42**	**−36.42**	**1.00**	**1056.37**	**−21.13**	**1.00**	**1509.27**	**−30.19**
	Sunday	Ensemble Strategy + TSO	0.85	1010.16	−51.08	0.90	992.81	−62.12	1.00	1774.21	−35.48
		RMLSSVR	0.75	601.4	−128.81	0.79	614.49	−89.85	0.92	1003.51	−40.83
		Curve-to-curve Regression	0.83	784.19	−74.8	0.92	1095.11	−73.20	1.00	2004.39	−40.09
		CNNs-Transformer	**0.85**	**677.51**	**−27.84**	**0.92**	**824.97**	**−49.83**	**1.00**	**915.81**	**−18.32**
Data 2	Saturday	Ensemble Strategy + TSO	0.81	1226.24	−285.53	0.88	1243.06	−258.62	0.9	2050.7	−179.35
		RMLSSVR	0.77	845.22	−224.96	0.83	1088.92	−238.19	0.88	2037.41	−188.5
		Curve-to-curve Regression	0.79	1393.24	−347.59	0.85	1583.41	−233.5	1.0	2503.92	−50.08
		CNNs-Transformer	**0.85**	**943.8**	**−123.3**	**0.90**	**1278.59**	**−205.17**	**0.98**	**1769.48**	**−36.12**
	Sunday	Ensemble Strategy + TSO	0.81	1787.35	−172.03	0.90	1354.74	−53.99	0.94	3785.87	−98.05
		RMLSSVR	0.75	1237.72	−220.74	0.77	2003.12	−169.72	0.90	1801.5	−51.46
		Curve-to-curve Regression	0.85	1287.16	−135.73	0.96	1775.98	−73.21	0.98	2662.59	−57.31
		CNNs-Transformer	**0.88**	**964.91**	**−129.3**	**0.92**	**1077.81**	**−48.47**	**0.96**	**1715.77**	**−42.51**
Data 3	Saturday	Ensemble Strategy + TSO	0.79	1748.78	−321.5	0.96	1762.08	−43.04	0.98	2818.87	−61.58
		RMLSSVR	0.67	1142.98	−368.56	0.81	1892.28	−222.7	0.88	2305.08	−172.76
		Curve-to-curve Regression	1.0	2291.27	−45.83	1.0	3466.83	−69.34	1.0	4148.15	−82.96
		CNNs-Transformer	**0.94**	**1323.43**	**−36.5**	**1.00**	**1434.05**	**−28.68**	**1.00**	**1682.0**	**−33.64**
	Sunday	Ensemble Strategy + TSO	0.94	1217.65	−60.86	1.0	1312.74	−26.25	1.0	2276.14	−45.52
		RMLSSVR	0.67	1236.36	−187.19	0.96	1674.14	−61.28	0.98	2301.98	−49.48
		Curve-to-curve Regression	1.0	2434.46	−48.69	1.00	2931.02	−58.62	1.0	4604.56	−92.09
		CNNs-Transformer	**1.00**	**887.89**	**−17.76**	**1.00**	**963.69**	**−19.27**	**1.00**	**1401.64**	**−28.03**

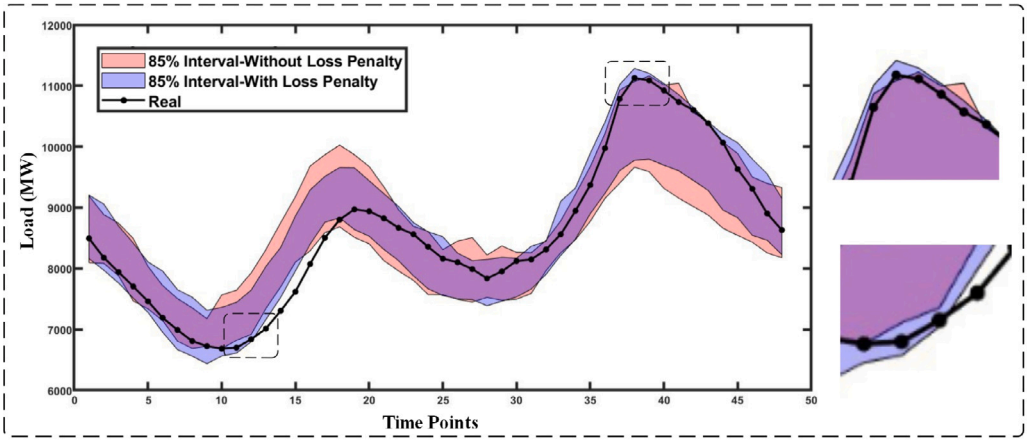

Fig. 9. 85% forecasting interval obtained by loss penalty and without loss penalty.

Table 11
Prediction results of three sets of data in Experiment 3.

			PINC = 0.85			PINC = 0.90			PINC = 0.95		
			PICP	MPIW	AIS	PICP	MPIW	AIS	PICP	MPIW	AIS
Data 1	Saturday	Without loss penalty	0.83	1037.96	−60.57	0.90	1143.16	−31.53	1.00	1740.44	−34.81
		With loss penalty	**0.92**	**917.42**	**−36.42**	**1.00**	**1056.37**	**−21.13**	**1.00**	**1509.27**	**−30.19**
	Sunday	Without loss penalty	0.79	807.87	−88.74	0.83	943.57	−43.74	0.98	1257.58	−32.53
		With loss penalty	**0.85**	**677.51**	**−27.84**	**0.92**	**824.97**	**−49.83**	**1.00**	**915.81**	**−18.32**
Data 2	Saturday	Without loss penalty	0.85	1050.98	−195.98	0.88	1341.80	−293.75	0.90	1941.27	−156.4
		With loss penalty	**0.85**	**943.80**	**−123.30**	**0.90**	**1278.59**	**−205.17**	**0.98**	**1769.48**	**−36.12**
	Sunday	Without loss penalty	0.81	1047.35	−171.65	0.85	1204.68	−76.14	0.90	1933.94	−91.64
		With loss penalty	**0.88**	**964.91**	**−129.30**	**0.92**	**1077.81**	**−48.47**	**0.96**	**1715.77**	**−42.51**
Data 3	Saturday	Without loss penalty	0.90	1379.96	−51.91	0.98	1549.95	−35.60	1.00	1914.43	−38.29
		With loss penalty	**0.94**	**1323.43**	**−36.50**	**1.00**	**1434.05**	**−28.68**	**1.00**	**1682.00**	**−33.64**
	Sunday	Without loss penalty	1.00	1137.21	−22.74	1.00	1240.09	−24.8	1.00	1907.39	−38.15
		With loss penalty	**1.00**	**887.89**	**−17.76**	**1.00**	**963.69**	**−19.27**	**1.00**	**1401.64**	**−28.03**

in the ***PICP*** values compared with those without such a technique. On average, the inclusion of the loss penalty mechanism increased the ***PICP*** value by ***5%***, which demonstrates the effectiveness of the loss penalty on the improvements of the interval coverage of the forecasting results. Specifically, in Data 1, with a confidence level of ***90%***, the load forecasting framework with the loss penalty technology has a ***PICP*** value of ***100%*** on Saturday, compared with the framework without loss penalty, which has a ***PICP*** value of only ***90%***, there is a significantly ***10%*** increase in ***PICP***.

In Data 2, when ***PINC*** is ***85%***, although the two load forecasting frameworks yield the same results with the ***85% PICP*** value on Saturday. The addition of the loss penalty makes the predicted interval width become only ***943***, while the width of no loss penalty is ***1050***. The utilization of loss penalty technology in the load forecasting framework can effectively address the issue of data set imbalance between weekday data and weekend data. By assigning a higher penalty to the weekend, the framework can prioritize and allocate more attention to this specific subset. Consequently, this approach has the potential to enhance the accuracy and reliability of interval prediction outcomes.

Remark 3. In Experiment 3, the effectiveness of the loss penalties is demonstrated, which can greatly improve the framework's ability to forecast weekend loads with irregular patterns while the data availability is limited, leading to a significant enhancement in the quality of the interval as well.

5.4. Sensitive analysis

In this experiment, we analyze the sensitivity of the hyper-parameters for the proposed framework. Many machine learning based algorithms are criticized for extensive hyper-parameter tuning. Therefore, it is crucial to obtain accurate forecasting results even when the hyper-parameters are not precisely tuned. We carried out a one-factor sensitivity analysis of the proposed framework (one factor at a time) [43]. The hyper-parameters we focused on are time-step, batch-size, and max-epoch [44].

We use ***TS*** to represent the abbreviation for time step, ***BS*** to represent the abbreviation for batch size, and ***ME*** to represent the abbreviation for max epoch. Their ranges are: $\boldsymbol{TS} \in [240, 288, 336^*, 384]$, $\boldsymbol{BS} \in [32, 48^*, 64, 128]$, $\boldsymbol{ME} \in [50, 100, 200^*, 300]$. Among them, labeling with an asterisk in the upper right corner indicates the optimal choice. $\boldsymbol{BS\Phi(TS, ME)}$ indicates changing batch size without changing max epoch and time step. We use the std index $\boldsymbol{STD} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$ to measure the sensitivity of the framework to hyper-parameters. Table 12 shows the experiment results:

From Table 12, it is obvious that the maximum number of iterations has a greater impact on the forecasting results, while other super-parameters have no significant impact. That is, our proposed framework is insensitive to most hyper-parameters, which makes it highly suitable for practical applications, because the impact of the maximum number of iterations on the forecasting results is inevitable.

Therefore, it is only necessary to ensure that the maximum number of iterations is set above a certain threshold (e.g., 100). By doing so, satisfactory day-ahead load forecasting results can be achieved regardless of minor changes in the other parameters.

Remark 4. In Experiment 4, we verify the sensitivity of the load forecasting framework. Through the experiment result, it is obvious that the proposed framework is robust across varying hyper-parameters.

6. Discussions

6.1. Do we need this complex framework on weekdays?

In the numerical validation section, we have demonstrated that the framework is advanced in day-ahead forecasting, especially in the case of predicting weekend electrical load on limited data sets. Compared with weekdays, the weekend electrical load is irregular and obviously more difficult to forecast. Therefore, considering the regularity of load patterns on working days, is it truly essential to employ such a complex framework for day-ahead prediction tasks?

In this discussion, we did not pre-process the data and directly used ***DeepBP***, ***GRU***, ***LSTM***, ***CNN***, ***Transformer*** and our proposed deep learning framework (***CNNs-Transformer***) to conduct day-ahead forecasting on the data set. We conducted such numerical studies twice on the test set on Wednesday and Friday. Table 13 shows the load forecasting results.

Based on Table 13, it is evident that the interval coverage of most algorithms meets the standard, except for ***DeepBP*** model. Particularly, on Wednesday's data set, when the confidence level is ***0.85***, the interval coverage of ***LSTM*** [45], ***GRU*** [46], ***CNN*** [47], and ***Transformer*** [48] is all above ***90%***, and the ***PICP*** value of ***CNNs-Transformer*** is ***100%***. Although the interval coverage of most single models is satisfactory, the customized ***CNNs-Transformer*** framework achieves the best interval quality on ***AIS*** index. On Friday's dataset, most algorithms, except for the ***CNNs-Transformer*** model, show a decline in load forecasting performance. Especially when ***PICN*** is ***0.95***, most models fail to achieve the preset interval coverage value. Overall, apart from ***DeepBP***, most models produce relatively satisfactory forecasting results on weekdays [49]. On the interval width, except that the forecasting result obtained by ***DeepBP*** is excessively wide, the other methods show no significant difference. In terms of ***AIS*** index, most of the intervals meet the requirements of practical application.

Based on the results, it appears that for regular working days, many models can grasp the trend of data well without any data processing, and the results of forecasting performance are satisfactory. Although our deep learning framework achieved the best results among all data sets, exceeding the preset value by an average of ***10%*** in the interval coverage index, other deep learning algorithms also produced results of higher quality than the preset confidence level, which has practical application value. Thus, single forecasting models can be used for weekdays if there is limited computational power. However, if we want to obtain accurate day-ahead forecasting results on weekends, our proposed forecasting framework becomes essential.

Conclusion 1. When performing day-ahead forecasting on weekdays, satisfactory results can be achieved without the need for data processing. Although the proposed deep learning framework remains the optimal choice, alternative basic models can still obtain accurate forecasting intervals. Therefore, in situations where computational resources are limited, it is viable to consider using basic models for prediction on working days. However, for weekends, it is essential to utilize our proposed framework.

6.2. Why is the interval forecasting result obtained by CNN very narrow?

CNN processes each input sequence segment independently, making it insensitive to the relationship between the leading and lagging terms of time series, which is very different from ***LSTM*** and ***Transformer*** from this perspective. In some certain predictive scenarios, the interpretation of recent data points may diverge from that of the earlier ones, rendering the ***CNN*** struggle to yield meaningful results. In this discussion, we designed an experiment to verify this claim. We selected a wave of length 2000 and added Gaussian noise to simulate the real-time series data. The formula of the wave $y(t)$ is as follows:

$$y(t)=100+10\times(\sin(6t)\ +\cos(10t))+\xi \tag{21}$$

where ξ follows $N(0,1)$. Then, we used ***CNN*** and ***Transformer*** respectively to predict the data, selected the quantile as 0.8, and observed the forecasting results as shown in Fig. 10.

Obviously, we can see that while ***CNN*** has successfully extracted data features and filtered out most of the noise, it has not effectively captured the data distribution. The results obtained by ***MSELoss*** are not significantly different from the results obtained by ***QRLoss***. This is the reason why the use of ***CNN*** results in narrower intervals.

In order to combine the feature extraction ability of ***CNN*** with the sequential sensitivity of ***Transformer***, an effective method is to add ***CNN*** in front of ***Transformer*** for feature extraction, especially for those long sequences that are challenging for ***Transformer*** to process (e.g., hundreds of time steps). The ***CNN*** can transform the long input sequence into a shorter sequence composed of advanced features (downsampling), and then these sequences composed of the extracted features will become the input of ***Transformer*** in the forecasting framework.

Conclusion 2. ***CNN*** may not be able to analyze time series data very well. Hence the combination of multi-layer convolution and ***Transformer/RNN*** is a better choice.

Table 12
Sensitive analysis result.

		Pattern		
		$TS\Phi(BS,ME)$	$BS\Phi(TS,ME)$	$ME\Phi(TS,BS)$
Data 1	PICP	0.01	0.02	0.09
	MPIW	27.89	32.95	200.17
	AIS	7.13	9.61	50.42
Data 2	PICP	0.012	0.02	0.10
	MPIW	26.81	33.92	150.86
	AIS	7.97	12.06	70.31
Data 3	PICP	0.00	0.00	0.24
	MPIW	14.34	21.69	96.72
	AIS	1.29	3.31	30.17

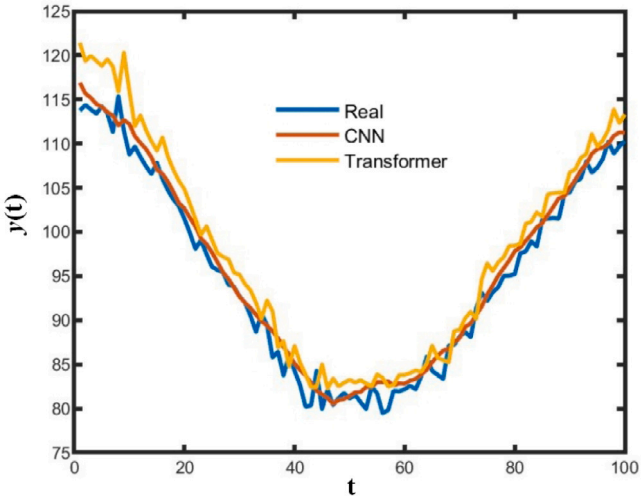

Fig. 10. Forecasting results of CNN and Transformer on 0.8 quantile.

Table 13
Characteristics of each data dimension.

		PINC = 0.85			PINC = 0.90			PINC = 0.95		
		PICP	MPIW	AIS	PICP	MPIW	AIS	PICP	MPIW	AIS
Wednesday	DeepBP	0.77	3158.74	−375.55	0.79	3541.38	−211.29	0.90	3976.32	−166.58
	GRU	0.92	1896.78	−68.91	0.96	1937.07	−50.41	1.00	2584.40	−51.69
	LSTM	0.90	1522.58	−51.91	0.98	1726.50	−40.93	0.94	2271.03	−63.51
	CNN	0.90	1232.40	−43.29	0.94	1628.70	−54.56	0.96	2182.41	−60.94
	Transformer	0.94	1388.33	−66.79	0.94	1778.29	−59.59	0.96	2392.91	−61.04
	CNNs-Transformer	1.00	1843.40	−36.87	1.00	2190.08	−43.80	1.00	2888.77	−57.78
Friday	DeepBP	0.90	3137.31	−148.99	0.90	3576.18	−98.02	0.96	3791.58	−95.65
	GRU	0.81	1678.32	−291.43	0.88	2248.62	−122.07	0.90	2385.69	−149.96
	LSTM	0.90	1910.02	−121.74	0.94	1919.72	−101.17	0.94	2096.60	−66.26
	CNN	0.90	1666.23	−250.25	0.90	1829.76	−149.60	0.94	1957.15	−65.79
	Transformer	0.88	2363.13	−314.18	0.90	2532.71	−198.85	0.92	3045.82	−204.81
	CNNs-Transformer	0.94	1876.4	−57.41	0.98	1926.47	−47.43	0.98	2627.09	−61.99

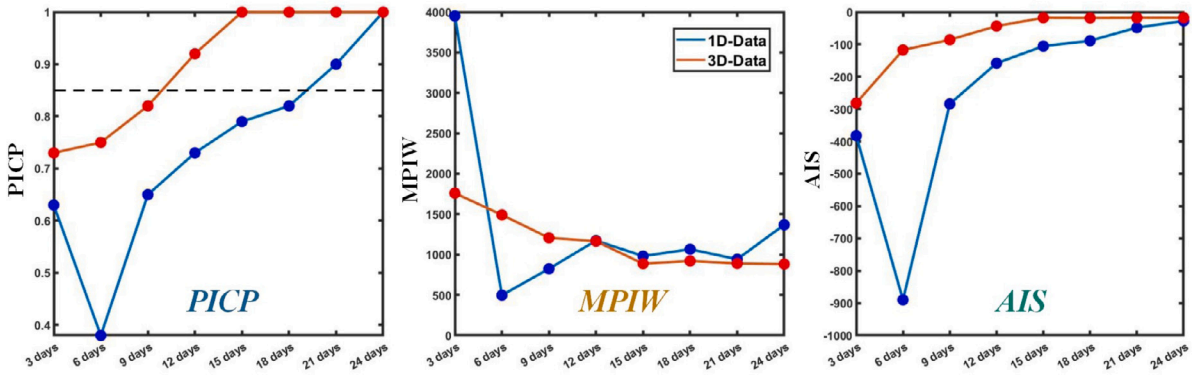

Fig. 11. The change in forecasting performance while the input data quantity changes.

6.3. What is the minimum necessary data input for the framework?

In this paper, we have focused on utilizing load data that includes a limited number of weekend data points to achieve precise day-ahead probability load forecasting results for weekends. This discussion aims to elucidate the minimum necessary data input and to clarify our definition of limited data availability. To this end, we designed an experiment to assess how the prediction accuracy is affected by varying the amount of input data. We selected the scenario of load forecasting for Sundays in ***Data 3***, with a confidence level of 0.85, and incremented the data input in increments of 3 days (144 time points). Additionally, we compared the performance using data that had undergone denoising and augmentation (3D) to that of the original data (1D), to analyze the benefits of data processing in the context of limited data availability. Fig. 11 illustrates the changes in forecasting performance as the input data quantity is varied.

For ***PICP***, a value higher than the confidence level (0.85) indicates that the model is suitable for use. The optimal scenario is for the ***PICP*** to be as close to 1 as possible, signifying that the prediction interval encompasses the actual values. The experimental results reveal that as the data volume increases, the ***PICP*** for both data input modes improves. Notably, the ***PICP*** for the 3D-Data consistently exceeds that of the 1D-Data. The 3D-Data reaches a usable level at 12 days of input data (including three days of weekend data) and achieves full coverage of the true values with 15 days of data input (including five days of weekend data). In contrast, the 1D-Data reaches a usable level only at 21 days of input data (including six days of weekend data) and achieves full coverage with 24 days of data input (including five days of weekend data). The ***AIS*** metric, which reflects the quality score of the interval, clearly demonstrates that the 3D-Data provides significantly better interval quality than the 1D-Data for the same input data length. This highlights the effectiveness of the 3D-Data approach when dealing with limited data availability, indicating that higher quality prediction results can be attained with fewer data points. In summary, to ensure stable and reliable forecasting performance with our proposed framework, a minimum of 12 days of data (576 time points) is required (including at least three days of weekend data), which we consider sufficient for a day-ahead load forecasting task (48-step) for weekends.

Conclusion 3: The forecasting framework we propose is capable of delivering high-quality forecasting results with as little as 12 days of data, offering an effective solution for scenarios with limited data availability.

Table 14
Running time of each model.

Model	Computation time(s)
VMD	3.071
LSTM	71.328
GRU	65.482
CNN	13.975
Transformer	97.736
CNNs-Transformer	35.601
Proposed framework	38.672

6.4. Is the running time of the framework suitable for practical application?

The framework's running time is crucial for practical implementation. In this discussion, we record the running time of the proposed day-ahead load forecasting framework and compare it with the benchmark algorithms. Table 14 shows the running time of each model.

In power grid operation, running time requirements often depend on the granularity of data. For instance, this paper utilizes data with a 30-minute granularity, which implies that after obtaining all the data for the day by 23:30 on the first night, the framework must provide the electrical load forecasting result for the next day before 24:00, that is, within 30 min. Based on experimental results, the framework's

running time is less than a minute, which clearly satisfies the practical application requirements.

Conclusion 3. The proposed framework can obtain high-quality forecasting results in a very short time, which is ideal for practical application.

7. Conclusion

In this paper, a day-ahead load forecasting framework is designed that achieves high-quality probabilistic forecasting results even when the data set is limited. The framework is mainly composed of two modules: the data processing module and the probabilistic deep learning forecasting module. The data processing module extracts the peak and trend values from the original time series. By combining them with the de-noised data, the deep learning model training can be assisted and the robustness and generalization ability can be improved. In the customized deep learning forecasting module based on ***CNNs-Transformer***, we build a deep learning framework composed of three-layer ***CNN*** and ***Transformer*** and employ ***QRLoss*** to achieve probability interval forecasting outcomes under various confidence levels. Additionally, the loss penalty technique is utilized to enhance the model's attention on weekends. Whether it involves day-ahead forecasting for electrical load on weekdays or weekends, the proposed forecasting framework obtains the best performance, and the improvement during weekends is more remarkable. In addition, the framework's practical application ability is also evidenced by its short running time.

However, there are still some limitations of the framework that we plan to improve in the future:

1) We plan to incorporate additional features, such as temperature and wind speed, to further improve forecasting accuracy.

2) We are committed to developing lighter frameworks to enhance the practical application's capabilities.

CRediT authorship contribution statement

Zhirui Tian: Formal analysis, Investigation, Visualization, Writing – original draft. **Weican Liu:** Software, Validation, Visualization. **Wenqian Jiang:** Conceptualization, Investigation, Validation, Writing – review & editing. **Chenye Wu:** Conceptualization, Funding acquisition, Supervision, Validation, Writing – review & editing.

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.

Data availability

We have provided the links to our utilized public datasets.

References

[1] Lai C, Wang Y, Fan K, Cai Q, Ye Q, Pang H, et al. An improved forecasting model of short-term electric load of papermaking enterprises for production line optimization. Energy 2022;245:123225. http://dx.doi.org/10.1016/j.energy.2022.123225.

[2] Zhao Z, Zhang Y, Yang Y, Yuan S. Load forecasting via grey model-least squares support vector machine model and spatial-temporal distribution of electric consumption intensity. Energy 2022;255:124468. http://dx.doi.org/10.1016/j.energy.2022.124468.

[3] Morais LBS, Aquila G, de Faria VAD, Lima LMM, Lima JWM, de Queiroz AR. Short-term load forecasting using neural networks and global climate models: An application to a large-scale electrical power system. Appl Energy 2023;348:121439. http://dx.doi.org/10.1016/j.apenergy.2023.121439.

[4] Rubasinghe O, Zhang T, Zhang X, Choi SS, Chau TK, Chow Y, et al. Highly accurate peak and valley prediction short-term net load forecasting approach based on decomposition for power systems with high PV penetration. Appl Energy 2023;333:120641. http://dx.doi.org/10.1016/j.apenergy.2023.120641.

[5] Dudek G. Pattern-based local linear regression models for short-term load forecasting. Electr Power Syst Res 2016;130:139–47. http://dx.doi.org/10.1016/j.epsr.2015.09.001.

[6] Luo J, Gong Y. Air pollutant prediction based on ARIMA-WOA-LSTM model. Atmos Pollut Res 2023;14(6):101761. http://dx.doi.org/10.1016/j.apr.2023.101761.

[7] Lee C-M, Ko C-N. Short-term load forecasting using lifting scheme and ARIMA models. Expert Syst Appl 2011;38(5):5902–11. http://dx.doi.org/10.1016/j.eswa.2010.11.033.

[8] Chaturvedi S, Rajasekar E, Natarajan S, McCullen N. A comparative assessment of SARIMA, LSTM RNN and Fb prophet models to forecast total and peak monthly energy demand for India. Energy Policy 2022;168:113097. http://dx.doi.org/10.1016/j.enpol.2022.113097.

[9] Xiao Z, Ye S-J, Zhong B, Sun C-X. BP neural network with rough set for short term load forecasting. Expert Syst Appl 2009;36(1):273–9. http://dx.doi.org/10.1016/j.eswa.2007.09.031.

[10] Tian Z, Gai M. New PM2.5 forecasting system based on combined neural network and an improved multi-objective optimization algorithm: Taking the economic belt surrounding the Bohai Sea as an example. J Clean Prod 2022;375:134048. http://dx.doi.org/10.1016/j.jclepro.2022.134048.

[11] Tian Z, Wang J. A wind speed prediction system based on new data preprocessing strategy and improved multi-objective optimizer. Renew Energy 2023;215:118932. http://dx.doi.org/10.1016/j.renene.2023.118932.

[12] Lv Z, Wang N, Lou R, Tian Y, Guizani M. Towards carbon neutrality: Prediction of wave energy based on improved GRU in maritime transportation. Appl Energy 2023;331:120394. http://dx.doi.org/10.1016/j.apenergy.2022.120394.

[13] Niu D, Yu M, Sun L, Gao T, Wang K. Short-term multi-energy load forecasting for integrated energy systems based on CNN-BiGRU optimized by attention mechanism. Appl Energy 2022;313:118801. http://dx.doi.org/10.1016/j.apenergy.2022.118801.

[14] Cao Z, Wang J, Yin L, Wei D, Xiao Y. A hybrid electricity load prediction system based on weighted fuzzy time series and multi-objective differential evolution. Appl Soft Comput 2023;149:111007. http://dx.doi.org/10.1016/j.asoc.2023.111007.

[15] Hu H, Xia X, Luo Y, Zhang C, Nazir MS, Peng T. Development and application of an evolutionary deep learning framework of LSTM based on improved grasshopper optimization algorithm for short-term load forecasting. J Build Eng 2022;57:104975. http://dx.doi.org/10.1016/j.jobe.2022.104975.

[16] Huang Q, Li J, Zhu M. An improved convolutional neural network with load range discretization for probabilistic load forecasting. Energy 2020;203:117902. http://dx.doi.org/10.1016/j.energy.2020.117902.

[17] Zhang D, Wang S, Liang Y, Du Z. A novel combined model for probabilistic load forecasting based on deep learning and improved optimizer. Energy 2023;264:126172. http://dx.doi.org/10.1016/j.energy.2022.126172.

[18] Yuan F, Che J. An ensemble multi-step M-RMLSSVR model based on VMD and two-group strategy for day-ahead short-term load forecasting. Knowl-Based Syst 2022;252:109440.

[19] Xu X, Chen Y, Goude Y, Yao Q. Day-ahead probabilistic forecasting for French half-hourly electricity loads and quantiles for curve-to-curve regression. Appl Energy 2021;301:117465. http://dx.doi.org/10.1016/j.apenergy.2021.117465.

[20] Wang J, Xing Q, Zeng B, Zhao W. An ensemble forecasting system for short-term power load based on multi-objective optimizer and fuzzy granulation. Appl Energy 2022;327:120042. http://dx.doi.org/10.1016/j.apenergy.2022.120042.

[21] Li J, Luo Y, Wei S. Long-term electricity consumption forecasting method based on system dynamics under the carbon-neutral target. Energy 2022;244:122572. http://dx.doi.org/10.1016/j.energy.2021.122572.

[22] Li J, Wei S, Dai W. Combination of manifold learning and deep learning algorithms for mid-term electrical load forecasting. IEEE Trans Neural Netw Learn Syst 2023;34(5):2584–93. http://dx.doi.org/10.1109/TNNLS.2021.3106968.

[23] Wen Z, Xie L, Fan Q, Feng H. Long term electric load forecasting based on TS-type recurrent fuzzy neural network model. Electr Power Syst Res 2020;179:106106. http://dx.doi.org/10.1016/j.epsr.2019.106106.

[24] Fu W, Fu Y, Li B, Zhang H, Zhang X, Liu J. A compound framework incorporating improved outlier detection and correction, VMD, weight-based stacked generalization with enhanced DESMA for multi-step short-term wind speed forecasting. Appl Energy 2023;348:121587. http://dx.doi.org/10.1016/j.apenergy.2023.121587.

[25] Pan S, Yang B, Wang S, Guo Z, Wang L, Liu J, et al. Oil well production prediction based on CNN-LSTM model with self-attention mechanism. Energy 2023;284:128701. http://dx.doi.org/10.1016/j.energy.2023.128701.

[26] Abou Houran M, Salman Bukhari SM, Zafar MH, Mansoor M, Chen W. COA-CNN-LSTM: Coati optimization algorithm-based hybrid deep learning model for PV/wind power forecasting in smart grid applications. Appl Energy 2023;349:121638. http://dx.doi.org/10.1016/j.apenergy.2023.121638.

[27] Yue J, Yang H, Feng H, Han S, Zhou C, Fu Y, et al. Hyperspectral-to-image transform and CNN transfer learning enhancing soybean LCC estimation. Comput Electron Agric 2023;211:108011. http://dx.doi.org/10.1016/j.compag.2023.108011.

[28] Song L, Liu Y, Fan J, Zhou D-X. Approximation of smooth functionals using deep ReLU networks. Neural Netw 2023;166:424–36. http://dx.doi.org/10.1016/j.neunet.2023.07.012.

[29] You H, Yu L, Tian S, Ma X, Xing Y, Xin N, Cai W. MC-Net: Multiple max-pooling integration module and cross multi-scale deconvolution network. Knowl-Based Syst 2021;231:107456. http://dx.doi.org/10.1016/j.knosys.2021.107456.

[30] Tang M, Zhuang W, Li B, Liu H, Song Z, Yin G. Energy-optimal routing for electric vehicles using deep reinforcement learning with transformer. Appl Energy 2023;350:121711. http://dx.doi.org/10.1016/j.apenergy.2023.121711.

[31] Wang Y, Xu H, Song M, Zhang F, Li Y, Zhou S, et al. A convolutional transformer-based truncated Gaussian density network with data denoising for wind speed forecasting. Appl Energy 2023;333:120601. http://dx.doi.org/10.1016/j.apenergy.2022.120601.

[32] Aizpurua JI, Peña-Alzola R, Olano J, Ramirez I, Lasa I, del Rio L, et al. Probabilistic machine learning aided transformer lifetime prediction framework for wind energy systems. Int J Electr Power Energy Syst 2023;153:109352. http://dx.doi.org/10.1016/j.ijepes.2023.109352.

[33] Cui B, Liu M, Li S, Jin Z, Zeng Y, Lin X. Deep learning methods for atmospheric PM2.5 prediction: A comparative study of transformer and CNN-LSTM-attention. Atmos Pollut Res 2023;14(9):101833. http://dx.doi.org/10.1016/j.apr.2023.101833.

[34] Dai J, Li H, Zeng R, Bai J, Zhou F, Pan J. KD-Former: Kinematic and dynamic coupled transformer network for 3D human motion prediction. Pattern Recognit 2023;143:109806. http://dx.doi.org/10.1016/j.patcog.2023.109806.

[35] Geng M, Li J, Xia Y, Chen XM. A physics-informed transformer model for vehicle trajectory prediction on highways. Transp Res C 2023;154:104272. http://dx.doi.org/10.1016/j.trc.2023.104272.

[36] Wang J, Zhou Y, Jiang H. A novel interval forecasting system based on multi-objective optimization and hybrid data reconstruct strategy. Expert Syst Appl 2023;217:119539. http://dx.doi.org/10.1016/j.eswa.2023.119539.

[37] Wang J, Li Z. Wind speed interval prediction based on multidimensional time series of convolutional neural networks. Eng Appl Artif Intell 2023;121:105987. http://dx.doi.org/10.1016/j.engappai.2023.105987.

[38] Luo H, Zhang H, Wang J. Ensemble power load forecasting based on competitive-inhibition selection strategy and deep learning. Sustain Energy Technol Assess 2022;51:101940. http://dx.doi.org/10.1016/j.seta.2021.101940.

[39] Bayram F, Aupke P, Ahmed BS, Kassler A, Theocharis A, Forsman J. DA-LSTM: A dynamic drift-adaptive learning framework for interval load forecasting with LSTM networks. Eng Appl Artif Intell 2023;123:106480. http://dx.doi.org/10.1016/j.engappai.2023.106480.

[40] Li H, Yu Y, Huang Z, Sun S, Jia X. A multi-step ahead point-interval forecasting system for hourly PM2.5 concentrations based on multivariate decomposition and kernel density estimation. Expert Syst Appl 2023;226:120140. http://dx.doi.org/10.1016/j.eswa.2023.120140.

[41] Zhu B, Wan C, Wang P. Interval forecasting of carbon price: A novel multiscale ensemble forecasting approach. Energy Econ 2022;115:106361. http://dx.doi.org/10.1016/j.eneco.2022.106361.

[42] Huang C, Wang J, Wang S, Zhang Y. A review of deep learning in dentistry. Neurocomputing 2023;554:126629. http://dx.doi.org/10.1016/j.neucom.2023.126629.

[43] Tian Z, Gai M. A novel hybrid wind speed prediction framework based on multi-strategy improved optimizer and new data pre-processing system with feedback mechanism. Energy 2023;281:128225. http://dx.doi.org/10.1016/j.energy.2023.128225.

[44] Wang J, Han L, Zhang X, Wang Y, Zhang S. Electrical load forecasting based on variable T-distribution and dual attention mechanism. Energy 2023;283:128569. http://dx.doi.org/10.1016/j.energy.2023.128569.

[45] Jahani A, Zare K, Mohammad Khanli L. Short-term load forecasting for microgrid energy management system using hybrid SPM-LSTM. Sustainable Cities Soc 2023;98:104775. http://dx.doi.org/10.1016/j.scs.2023.104775.

[46] Li C, Li G, Wang K, Han B. A multi-energy load forecasting method based on parallel architecture CNN-GRU and transfer learning for data deficient integrated energy systems. Energy 2022;259:124967. http://dx.doi.org/10.1016/j.energy.2022.124967.

[47] Wan A, Chang Q, AL-Bukhaiti K, He J. Short-term power load forecasting for combined heat and power using CNN-LSTM enhanced by attention mechanism. Energy 2023;282:128274. http://dx.doi.org/10.1016/j.energy.2023.128274.

[48] Yan Q, Lu Z, Liu H, He X, Zhang X, Guo J. An improved feature-time transformer encoder-Bi-LSTM for short-term forecasting of user-level integrated energy loads. Energy Build 2023;113396. http://dx.doi.org/10.1016/j.enbuild.2023.113396.

[49] Wu J, Wang J, Lu H, Dong Y, Lu X. Short term load forecasting technique based on the seasonal exponential adjustment method and the regression model. Energy Convers Manage 2013;70:1–9. http://dx.doi.org/10.1016/j.enconman.2013.02.010.