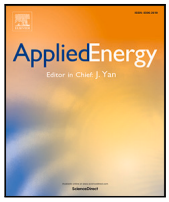
Applied Energy 357 (2024) 122544

Contents lists available at ScienceDirect

Applied Energy

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



Customer baseline load estimation for virtual power plants in demand response: An attention mechanism-based generative adversarial networks approach

Zhenyi Wang, Hongcai Zhang *

State Key Laboratory of Internet of Things for Smart City and Department of Electrical and Computer Engineering, University of Macau, Macao Special Administrative Region of China

ARTICLE INFO

Keywords:
Attention mechanism
Baseline load estimation
Demand response
Generative adversarial networks
Virtual power plant

ABSTRACT

The virtual power plant (VPP) that aggregates demand-side resources, is a new type of entity to participate in the electricity market and demand response (DR) program. Accurate customer baseline load (CBL) estimation is critical for DR implementation, especially the financial settlement in incentive-based DR. However, this is a challenging task as CBLs cannot be measured and are not equal to actual loads when DR events occur. Moreover, VPPs with different aggregation scales form heterogeneous electricity customers, which increases the difficulty of CBL estimation. In order to address this challenge, this paper proposes a novel deep learning-based CBL estimation method for varied types of electricity customers with different load levels. Specifically, we first transform the CBL estimation problem into a time-series missing data imputation issue, by regarding actual load sequences as CBL sequences with missing data, during DR periods. Then, we propose an attention mechanism-based neural network model to learn load patterns and characteristics of various CBLs, and also create the DR mask to avoid the disturbance of actual loads of DR periods on CBL's normal pattern. Further, we develop the generative adversarial networks (GAN)-based data imputation framework to produce the corresponding complete CBL sequence according to the actual load sequence, and then recover the missing values accordingly. Finally, comprehensive case studies are conducted based on public datasets, and our proposed method outperforms all benchmarks, where the mean and standard deviation of its estimation percentage error are 5.85% and 1.74%, respectively. This validates the effectiveness and superiority of the proposed method.

1. Introduction

1.1. Background and motivation

In response to the energy crisis and global warming issues, renewable generation has been vigorously developed across the world [1]. However, owing to the stochastic and intermittent nature of renewable generation, the real-time balance between supply and demand is becoming more challenging, which threatens the stable operation of power systems [2]. Demand response (DR) is an effective technology to tackle this challenge by dispatching demand-side flexible resources in coordination with renewable generation [3].

Generally, there are two main categories of DR: price-based and incentive-based [4]. The price-based DR relieves grid pressure through time-varying tariffs, while the incentive-based DR encourages customers to reduce their loads during peak periods using financial compensation [5]. Considering the limited capacity of an individual demand-side resource, it is difficult to meet the access requirement of DR programs [6]. To facilitate demand-side customer participation, the virtual power plant (VPP) is considered as a promising solution, which aggregates multiple demand-side resources to act as a virtual entity (aggregated customer), and then participates in electricity markets and provides grid services [7]. In order to fairly compensate customers in the incentive-based DR program, it is necessary for power system (or electricity market) operators to accurately estimate the actual contributions of customers in DR programs, i.e., load reductions following regulation signals [8–10]. The customer's load reduction refers to the gap between the actual power consumption and normal power demand, which is called the customer baseline load (CBL) [11], as shown in Fig. 1. Furthermore, since the customer needs to change the normal load behavior when DR events occur, its CBLs during DR periods cannot be measured in any way [12]. It should be mentioned that both the overestimated and underestimated CBL will lead to a bad effect on the normal operation of incentive-based demand response [13].

* Corresponding author.
E-mail address: hczhang@um.edu.mo (H. Zhang).

https://doi.org/10.1016/j.apenergy.2023.122544
Received 6 July 2023; Received in revised form 12 November 2023; Accepted 20 December 2023
Available online 27 December 2023
0306-2619/© 2023 Elsevier Ltd. All rights reserved.

Table 1
Summary of existing studies for CBL estimation.

Ref.	Method	Year	Description
[14]	Averaging	2011	Use the average of the top 4 days in the prior 5 non-DR days.
[15]	Averaging	2014	Use the average of the top 5 days in the prior 10 non-DR days.
[16]	Averaging	2014	Use exponential moving average-based method.
[17]	Regression	2017	Adopt the SVR model considering ambient temperature.
[18]	Regression	2019	Propose a probabilistic method with quantile regression model.
[19]	Regression	2020	Raise a mixed robust method with three regression models.
[20]	Control Group	2018	Propose a synchronous pattern matching-based method.
[21]	Control Group	2019	Design a CBL estimation method with virtual control group.
[22]	Control Group	2022	Use LASSO regression with spatial and temporal information.
[23]	Deep Learning	2021	Develop a graph neural network-based CBL estimation method.
[24]	Deep Learning	2021	Propose a method using contextual bandit with policy gradient.
[25]	Deep Learning	2019	Design a SAE-based method with pseudo-load selection.
[26]	Deep Learning	2021	Propose a cascaded SAE-based method with privacy framework.

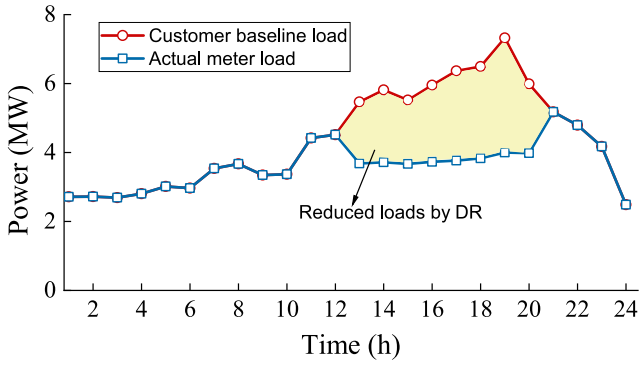


Fig. 1. Illustration of DR event and CBL.

1.2. Literature review

In recent years, CBL estimation is a research hot spot. The published papers can be primarily split into three categories, i.e., the *averaging*, *regression*, and *control group* methods. Moreover, there are also some CBL estimation studies based on deep learning. We summarize the existing related work according to the methodology, as shown in Table 1.

The *averaging* methods use the average of historical loads of non-DR days with the same periods as the DR event as the estimated CBLs. The averaging methods have been extensively adopted by operators in existing electricity markets [27], such as the High4of5 method in PJM [14], the High5of10 method in New York ISO [15], and the exponential moving average-based method ISO New England [16]. However, these methods may cause significant estimation errors because customer load patterns may be not identical on adjacent days and susceptible to changing environmental factors.

The *regression* methods aim to build functions or models to describe the relationship between CBLs and input features, e.g., time, weather, and historical load, etc. Chen et al. [17] adopted a support vector regression model for CBL estimation of office buildings, which considers the ambient temperature of two hours prior to the DR event. Sun et al. [18] proposed a probabilistic CBL estimation framework using the quantile regression forest model and deep learning-based clustering method, where they also build the daily load profile pool to catch CBL uncertainty. Zhou et al. [19] raised a mixed robust method to estimate CBLs in Southern California, USA, which combines the segmented regression model, the least trimmed squares estimation method, and the random effect regression model. However, the regression-based methods cannot guarantee the model's robustness and generalization, because they rely on valuable features but it is hard to quantify the effectiveness of these manually selected features.

In the *control group* methods, all customers are divided into two groups according to whether they participate in DR programs, i.e., the DR group and the control group. The CBL estimation for DR participants is performed based on the load data of customers in the control group, especially those with similar load patterns to the DR group customers. Wang et al. [20] propose a residential CBL estimation method based on the synchronous pattern matching principle, and estimate the CBL of DR participants using the optimized weight combination. Lee et al. [21] designed a CBL estimation method based on the virtual control group, which is able to apply to each DR event by collecting the DR participation information in advance, thus improving the method's adaptability. Ge et al. [22] combine the spatial and temporal information of load data in control group to estimate the CBLs of DR group, where the LASSO regression is also used to acquire more efficient feature selection. However, these methods require the assumption that there are enough customers in the control group, which have similar load patterns as DR participants, but this may be difficult to meet in practice due to the heterogeneity of customer load.

In addition, with the rapid development of artificial intelligence technology in recent years, some researchers intend to learn the load pattern of customers to perform CBL estimation, with the high capability of emerging deep learning models [28], such as graph neural network, reinforcement learning, autoencoder. For example, Lin et al. [23] developed a graph neural network-based framework for CBL estimation considering customers' spatial information, which extracts the implicit relationship of customers' houses, even without obtaining the specific geography information. Zhang et al. [24] proposed a closed-loop CBL estimation method using the contextual bandit with policy gradient, where they also designed a pre-event and post-event adjustment for estimation accuracy improvement. Additionally, the training efficiency of network weights is improved by collaboratively optimizing CBL estimation and customer segmentation. However, the above two methods either require non-load information (e.g., house location) or load data from other customers. This may cause the training data of customers to be limited by factors other than their own load data, and then reduce the applicability of these methods.

To overcome the aforementioned data limitation issue, some researchers propose CBL estimation methods based on the generative deep learning model by using the customers' own historical actual loads, rather than non-load features or other customers' load data. Wang et al. [25] utilize the reconstruction capability of a stacked autoencoder (SAE) to estimate residential CBLs, where the support vector machine classifier is adopted to select the pseudo-load to improve model accuracy. On this basis, Chen et al. [26] propose a cascaded SAE-based method that eliminates the pseudo-load selection to improve computational efficiency and model accuracy. Moreover, the federated learning framework is also used to protect the customer's data privacy. However, these two methods consider all actual loads during possible DR periods as missing data and replace them with pseudo-loads, regardless of whether DR events actually occur or not. This leads to the loss of valuable customer load information, since the generative model need to learn the normal pattern and distribution of customer load through as much historical load data as possible.

1.3. Contributions

In order to address the aforementioned problems, especially the information loss caused by discarding load data, we propose a novel deep learning-based method based on the attention mechanism [29] and generative adversarial networks (GAN) [30], to estimate CBL for varied types of customers with different load levels in DR programs. As shown in Fig. 1, actual loads are not equal to CBLs during DR periods. Moreover, since CBLs during DR periods cannot be measured, we regard the customer's actual loads as its CBLs with possible missing data, where missing values appear when DR events occur. Specifically, we first transform the CBL estimation problem into a time-series missing data imputation issue, which refrains from directly abandoning load data of DR periods and results in information loss. Then, we propose a novel Transformer model based on the attention mechanism to learn the load pattern and characteristic of CBLs. In addition, we develop a data imputation framework based on GAN to generate complete CBL sequences according to actual load sequences and then recover the corresponding missing values. In summary, compared with existing studies, the main contributions of this paper are threefold, as follows:

(1) We propose a novel deep learning-based method for CBL estimation, which is applicable to varied types of customers with different load levels. We transform the CBL estimation problem into a time-series missing data imputation issue to exploit the load data of non-DR periods, thus improving the estimation performance of the proposed method.
(2) We design a Transformer neural network model based on the attention mechanism to effectively extract the complex temporal dependency of load data and learn the load characteristics of different customers. The DR mask is created to conceal actual loads of DR periods, which enables the designed model to adapt to different DR periods.
(3) We develop a data imputation framework based on GANs to estimate CBLs of DR periods for various types of customers and load levels. We adopt a new estimation loss based on the masked reconstruction for model training, which improves the robustness of the proposed method.

The rest of this paper is organized as follows. Section 2 introduces the CBL estimation problem. The proposed method is elaborated in Section 3 and validated by numerical experiments in Section 4. Section 5 concludes this paper and proposes some future works.

2. Problem statement

This paper mainly focuses on the CBL estimation used for the financial settlement in the incentive-based DR. Different from load forecasting, the CBL estimation is usually performed after the DR event has finished, where CBL estimation is a posterior task while load forecasting is a prior task. Hence, the system operator can utilize the customers' historical load records and the actual load profile in the current DR day for CBL estimation.

For the ith customer, the CBL during period t on day d, denoted by $p_{i,d}^{\mathrm{CBL}}$, can be expressed as:

$$p_{i,d}^{\mathrm{CBL}}(t)=\begin{cases} p_{i,d}(t), & \forall t\in\mathcal{T}_d^{\mathrm{NDR}} \\ p_{i,d}(t)+p_{i,d}^{\mathrm{DR}}(t), & \forall t\in\mathcal{T}_d^{\mathrm{DR}}, \end{cases} \forall i\in\mathcal{I},\forall d\in\mathcal{D}, \tag{1}$$

where $p_{i,d}$ is the actual load during the same period; $p_{i,d}^{\mathrm{DR}}$ is the reduced load during DR period; $\mathcal{I}$ and $\mathcal{D}$ denote the sets of costumers and days, respectively; $\mathcal{T}_d^{\mathrm{NDR}}$ and $\mathcal{T}_d^{\mathrm{DR}}$ denote the sets of non-DR periods and DR periods on day d, respectively.

To preserve the temporal information of load data, we divide CBLs by day and arrange the data according to time order, which is represented as

$$\boldsymbol{p}_{i,d}^{\mathrm{CBL}}=\left[\, p_{i,d}^{\mathrm{CBL}}(1)\ ,\ p_{i,d}^{\mathrm{CBL}}(2)\ ,\ldots,\ p_{i,d}^{\mathrm{CBL}}(T)\,\right], \tag{2}$$

where T is the number of time intervals in one day, i.e., $T=|\mathcal{T}_d|$ and $\mathcal{T}_d=\mathcal{T}_d^{\mathrm{NDR}}\cup\mathcal{T}_d^{\mathrm{DR}},\forall d\in\mathcal{D}$. Here, we set $T=48$ owing to the half-hour time granularity so that $\boldsymbol{p}_{i,d}^{\mathrm{CBL}}\in\mathbb{R}^{1\times 48}$.

In this way, the CBL matrix representation of the ith customer can be expressed as:

$$\boldsymbol{P}_i^{\mathrm{CBL}}=\left[\,\boldsymbol{p}_{i,1}^{\mathrm{CBL}}\ ,\ \boldsymbol{p}_{i,2}^{\mathrm{CBL}}\ ,\ldots,\ \boldsymbol{p}_{i,D}^{\mathrm{CBL}}\,\right]^{\mathrm{T}}, \tag{3}$$

where D is the number of days in the dataset, i.e., $D=|D|$, and $\boldsymbol{P}_i^{\mathrm{CBL}}\in\mathbb{R}^{D\times 48}$.

Because the CBL estimation is a posterior event estimation issue, we utilize the DR mask to indicate whether the DR event occurs, as follows:

$$m_{i,d}(t)=\begin{cases} 1, & \forall t\in\mathcal{T}_{\mathrm{NDR}} \\ 0, & \forall t\in\mathcal{T}_{\mathrm{DR}}, \end{cases} \forall i\in\mathcal{I},\forall d\in\mathcal{D}, \tag{4}$$

where $m_{i,d}(t)$ denotes the DR mask of the ith customer during period t on day d.

Since the DR mask is in one-to-one correspondence with CBL, the DR mask matrix can be written similarly to the CBL representation as:

$$\boldsymbol{M}_i=\left[\,\boldsymbol{m}_{i,1}\ ,\ \boldsymbol{m}_{i,2}\ ,\ldots,\ \boldsymbol{m}_{i,D}\,\right]^{\mathrm{T}}, \tag{5}$$

where $\boldsymbol{m}_{i,d}$ denotes the DR mask vector of the ith customer on day d. Thus, $\boldsymbol{m}_{i,d}\in\mathbb{R}^{1\times 48}$ and $\boldsymbol{M}_i\in\mathbb{R}^{D\times 48}$.

According to Eq. (1), the CBL is equal to the actual load during non-DR periods but not during DR periods. Moreover, the CBL in DR period cannot be measured and is counterfactual, because the customer's load pattern is changed. Thus, we can regard the actual load sequence as the CBL sequence with missing data. To avoid information loss caused by directly discarding load data during DR periods, we throughout use actual loads for CBL estimation, whether or not the DR event occurred. In addition, since the actual load is not real CBL during DR periods, we take advantage of DR masks to shield the interference of these load data. Therefore, we convert the CBL estimation problem into a time-series missing data imputation issue [31]. It uses customer's actual loads and corresponding DR masks as inputs, as follows:

$$\boldsymbol{p}_{i,d}^{\mathrm{CBL}}=f(\boldsymbol{p}_{i,d}\,,\boldsymbol{m}_{i,d};\boldsymbol{\theta}), \tag{6}$$

where $\boldsymbol{p}_{i,d}$ is the actual load vector of the ith customer on day d and has the identical shape as $\boldsymbol{p}_{i,d}^{\mathrm{CBL}}$, i.e., $\boldsymbol{p}_{i,d}\in\mathbb{R}^{1\times 48}$; and f denotes the estimation model with parameters $\boldsymbol{\theta}$.

From the perspective of VPPs, we also take into account the case of aggregated customers, where the CBL estimation processes are almost the same as individual customers, except for the load level. For example, the CBL of the jth aggregation customer during period t on day d is expressed as:

$$p_{a_j,d}^{\mathrm{CBL}}(t)=\sum_{i\in\mathcal{I}_j}p_{i,d}^{\mathrm{CBL}}(t),\ j\in\mathcal{J}, \tag{7}$$

where $\mathcal{I}_j$ is the set of individual customers for the jth aggregation customer and $\mathcal{I}_j\subseteq\mathcal{I}$; $\mathcal{J}$ denotes the set of aggregation customers, thus, $\bigcup_{j\in\mathcal{J}}\mathcal{I}_j=\mathcal{I}$ and $\sum_{j\in\mathcal{J}}|\mathcal{I}_j|=|\mathcal{I}|$.

3. Proposed methodology

In this section, we elaborate on the proposed CBL estimation method based on the attention mechanism and GAN. First, the methodology architecture is outlined, and then we expound the designed transformer-based model, which is followed by the GAN-based data imputation framework.

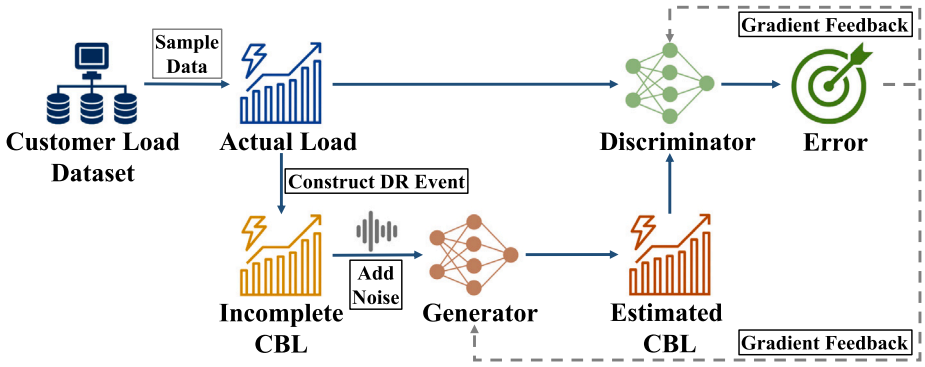


Fig. 2. The general framework of the proposed method.

3.1. Methodology architecture

The proposed CBL estimation method is mainly divided into a transformer-based model and a WGAN-based data imputation framework, where the overall architecture is shown in Fig. 2. Specifically, the WGAN-based framework adopts the proposed transformer-based model as its generator, which is used to learn the CBL's normal pattern and produce the complete CBL sequence. The framework trains the generator and discriminator through a two-player zero-sum game. After the adversarial training converges, this framework can generate the complete CBL sequence to fill up the original incomplete CBLs, thus accomplishing the CBL estimation.

In this paper, we suppose the power system operator is responsible for implementing the proposed CBL estimation method to perform the financial settlement. Considering that the system operator usually possesses actual load data of customers under its jurisdiction, which is recorded and uploaded by smart meters. In this way, the system operator can implement the proposed method for CBL estimation by using abundant customers' load data. Specifically, the sample data in Fig. 2 are customers' daily load profiles.

3.2. Transformer-based neural network model

Because the input and output are both load sequences, the transformer-based model adopts the encoder–decoder architecture [29], which is shown in Fig. 3. Unlike the RNN-class models, the proposed model only exploits the attention mechanism to process time-series data (i.e., load). The Transformer model has recently been gradually applied in power systems. For example, Wang et al. [32] propose a multi-task model based on Transformer to perform joint prediction of multi-energy load. Li et al. [33] build a Transformer-based model for false data injection attacks detection in smart grid.

3.2.1. Encoder

Encoder aims to map the input sequence (i.e., actual loads) into a high-dimensional representation sequence for information extraction. The encoder is stacked by multiple identical encoder layers, and each encoder layer mainly consists of two sub-layers and two add&norm modules. Furthermore, actual load sequences are transformed by the embedding module before entering the encoder. The abovementioned modules and sub-layer are introduced as follows:

(a) Embedding Module: To enhance the representation of input data, we use the embedding module to successively convert each value of the input sequence into a vector. More detailed, we apply the linear layer of neural networks for data transformation, where each actual load is converted into a vector of dimension d_{model}:

$$\boldsymbol{X}_t = Embedding(x_t), \tag{8}$$

where $x_t \in \mathbb{R}^{1\times 1}$ denotes the tth element of input sequence $\boldsymbol{x}$, that is, $\boldsymbol{p}_{i,d}(t)$; $\boldsymbol{X}_t \in \mathbb{R}^{1\times d_{\text{model}}}$ represents the corresponding embedding vector; and $Embedding(\cdot)$ is the linear layer.

Because the proposed model does not involve any recurrent or convolutional operation, we add the positional encoding [29] into embedding vectors to exploit the temporal-order information of input sequence. The positional encoding is expressed as:

$$PE_{(t,2i)} = \sin\left(\frac{t}{10000^{2i/d_{\text{model}}}}\right), \tag{9}$$

$$PE_{(t,2i+1)} = \cos\left(\frac{t}{10000^{2i/d_{\text{model}}}}\right), \tag{10}$$

where $PE(\cdot)$ represents the positional encoding function; $t \in [0, T-1]$ denotes the element's position of input sequence $\boldsymbol{x}$; and $i \in [0, d_{\text{model}}/2]$ is the index of vector dimension. Thus, the output of this embedding module is represented as:

$$\boldsymbol{X} \leftarrow Embedding(\boldsymbol{x}) + PE(Embedding(\boldsymbol{x})), \tag{11}$$

where $\boldsymbol{X} \in \mathbb{R}^{T\times d_{\text{model}}}$ is the ultimate embedding vector.

(b) DR-Masked Multi-Head Self-Attention Sub-Layer: Inspired by the human behavior and psychology, the proposed model extracts the temporal information by using the attention mechanism, which focuses on relevant parts and ignores useless contents. The attention mechanism [29] is implemented by the attention function that contains three elements, namely the query and key–value pair, which are all vectors. According to the sources of the query and key–value pair, there are two types of attention functions. If the query and key–value pair come from the same source, it is called the self-attention function, otherwise it is called the attention function. The output of this function is a weighted sum of values, where the weight of each value is calculated based on the similarity of the corresponding key and the query. Formally, the matrix calculation of the attention function is expressed as:

$$Attention(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \underbrace{Softmax\left(\frac{\overbrace{\boldsymbol{Q}\cdot\boldsymbol{K}^{\mathrm{T}}}^{\text{vector similarity}}}{\sqrt{d_k}}\right)}_{\text{value weight}}\cdot\boldsymbol{V}, \tag{12}$$

where $Attention(\cdot)$ is the attention function; $\boldsymbol{Q} \in \mathbb{R}^{T\times d_{\text{query}}}$, $\boldsymbol{K} \in \mathbb{R}^{T\times d_{\text{key}}}$ and $\boldsymbol{V} \in \mathbb{R}^{T\times d_{\text{value}}}$ are matrices of query, key and value, respectively, where $d_{\text{query}} = d_{\text{key}}$; $Softmax(\cdot)$ represents the softmax function that maps the similarity to $[0, 1]$. We also scale the vector similarity by $1/\sqrt{d_k}$ to avoid vanishing gradients and guarantee the training stability [29].

Considering that only actual loads are used for CBL estimation, we adopt the self-attention function, which is also calculated in the form of Eq. (12), but its elements all come from $\boldsymbol{X}$:

$$\begin{cases} \boldsymbol{Q} = \boldsymbol{X}\cdot\boldsymbol{W}_q, \\ \boldsymbol{K} = \boldsymbol{X}\cdot\boldsymbol{W}_k, \\ \boldsymbol{V} = \boldsymbol{X}\cdot\boldsymbol{W}_v, \end{cases} \tag{13}$$

where $\boldsymbol{W}_q$, $\boldsymbol{W}_k$, and $\boldsymbol{W}_v$ are three different transformation matrices,whose dimensions are all $\mathbb{R}^{d_{\text{model}}\times d_{\text{model}}}$.

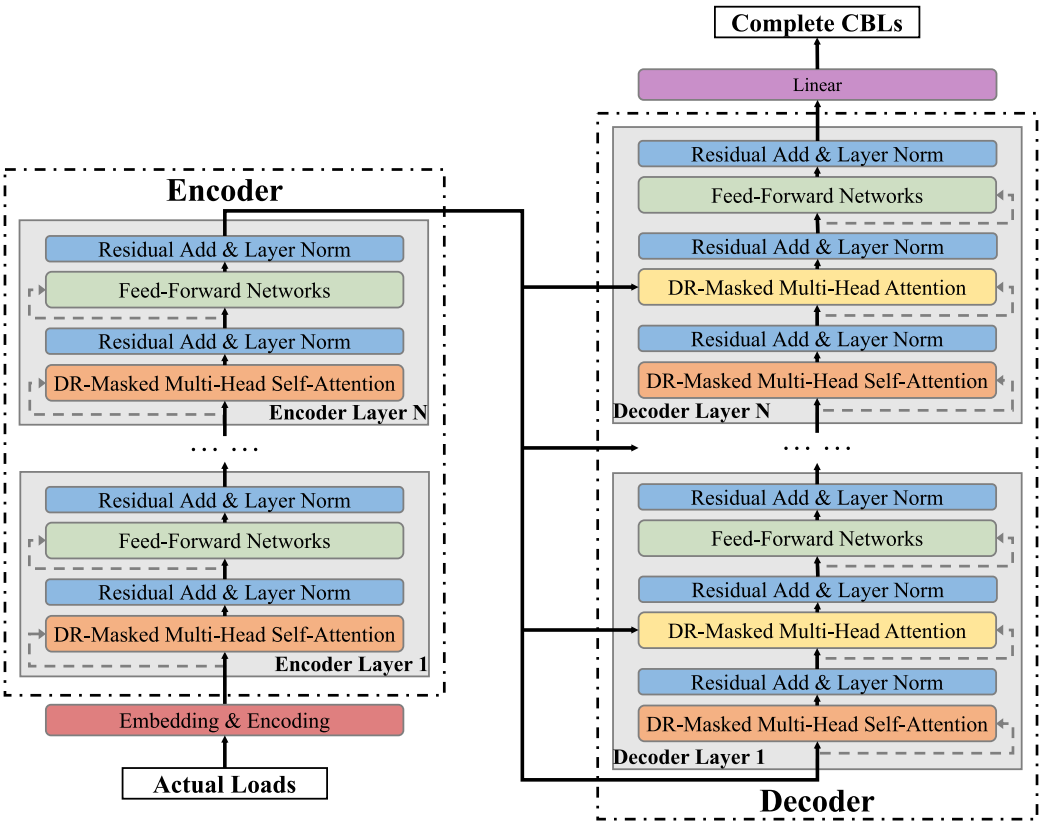


Fig. 3. The architecture of transformer-based CBL estimation model.

In order to capture multiple temporal relationships of input sequence, we employ the multi-head self-attention function [29]. Different from the single-head self-attention function, this function first projects three elements into multiple different sub-spaces, then performs the Eq. (12) and concatenates the results, and finally projects back to the original space, which is represented as:

$$Multhead(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = Concat(\text{head}_1,\dots,\text{head}_h)\boldsymbol{W}^O, \tag{14}$$

$$\text{head}_i = Attention(\boldsymbol{Q}\boldsymbol{W}_i^Q, \boldsymbol{K}\boldsymbol{W}_i^K, \boldsymbol{V}\boldsymbol{W}_i^V), \tag{15}$$

where $Multhead(\cdot)$ is the multi-head self-attention function; $Concat(\cdot)$ represents matrix concatenation operation with the matrix $\boldsymbol{W}^O \in \mathbb{R}^{(h\cdot d_v)\times d_{\text{model}}}$, and h denotes the total number of heads; $\boldsymbol{W}_i^Q \in \mathbb{R}^{d_{\text{model}}\times d_k}$, $\boldsymbol{W}_i^K \in \mathbb{R}^{d_{\text{model}}\times d_k}$ and $\boldsymbol{W}_i^V \in \mathbb{R}^{d_{\text{model}}\times d_v}$ are the ith projection matrices of query, key and value, respectively, where $d_k = d_v = d_{\text{model}}/h$.

In this paper, we propose and incorporate the DR mask $\boldsymbol{M}$ (see details in Section 2) into the attention mechanism to remove the interference of actual loads during DR periods, because they are not equal to CBLs. Specifically, after calculating the vector similarity in Eq. (12), we manually set the similarity values to $-\infty$ for all positions where DR events occur according to whether the DR mask is 0 or not. Based on the softmax function's property, the weights of actual loads during DR periods all become 0. Therefore, these positions' actual loads will not affect the subsequent calculation.

(c) Feed-Forward Networks Sub-Layer: In addition to the attention sub-layer, we also utilize a feed-forward neural network [34] that is applied to each position of vectors separately and identically, to improve the non-linear fitting ability of the model. There are two linear layers and a ReLU activation function [35] in this sub-layer, which can be formulated as:

$$FFN(\boldsymbol{X}) = \max(\boldsymbol{0}, \boldsymbol{X}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2, \tag{16}$$

where $FFN(\cdot)$ represents the feed-forward networks sub-layer of the proposed model; $\boldsymbol{W}_1 \in \mathbb{R}^{d_{\text{model}}\times d}$ and $\boldsymbol{W}_2 \in \mathbb{R}^{d\times d_{\text{model}}}$ are both weight matrices, where d is the hidden layer dimension; $\boldsymbol{b}_1 \in \mathbb{R}^{T\times d}$ and $\boldsymbol{b}_2 \in \mathbb{R}^{T\times d_{\text{model}}}$ denote bias vectors. Moreover, we apply the dropout layer [36] for each sub-layer to reduce model overfitting by randomly ignoring some neurons.

(d) Add&Norm Module: For each of the above two sub-layers, there is a residual connection around it and followed by the layer normalization. The residual connection [37] helps alleviate the vanishing gradient problem. For example, there are approximately half of the cases that the gradient is zero in ReLU activation function. Moreover, the contextual information of time-series data is also preserved by the residual connection [37]. The layer normalization [38] is a regularization method, which avoids the model overfitting and reduces the training time. Considering the temporal correlation of actual loads within a day and the parallelization of model training, we choose layer normalization instead of batch normalization [38]. Therefore, the output of this module is represented as:

$$\boldsymbol{X} \leftarrow LN(\boldsymbol{X} + Sublayer(\boldsymbol{X})), \tag{17}$$

where $LN(\cdot)$ and $Sublayer(\boldsymbol{X})$ represent layer normalization and the output of the sub-layer, respectively.

The encoder's output is the high-dimensional representation of input sequence, i.e., the extracted information of actual load data. As shown in Fig. 3, the output will be used as an input to each decoder layer to generate the estimated CBL sequence.

3.2.2. Decoder

Decoder produces the generated CBL sequence based on extraction information, and is also stacked by multiple identical decoder layers. As shown in Fig. 3, the decoder layer involves the abovementioned sub-layers and modules in encoder layers. The difference is that the decoder layer has an additional sub-layer (i.e., *DR-masked multi-head attention sub-layer*), where the key and value are both from encoder's output and the query comes from the input of decoder layers. Moreover, there

is also a distinction in the *DR-masked multi-head self-attention sub-layer* between the decoder and encoder layer. We introduce the differences of the decoder layer in detail, as follows:

(a) DR-Masked Multi-Head Attention Sub-Layer: To make full use of the encoder's output for guiding generation, we adopt the attention function in the decoder layer rather than the self-attention function, since the source of the key and value is different from the query. In particular, the key and value come from the high-dimensional representation of actual loads. According to Eq. (12), this sub-layer will focus on importance positions in the input sequence, and then estimate the CBL for each period based on the entire actual load sequence.

(b) DR-Masked Multi-Head Self-Attention Sub-Layer: For the period t, the decoder should only rely on actual loads not exceeding period t for CBL estimation, because the model cannot acquire future data in practice [29]. To prevent the decoder from snooping on future actual loads, we utilize the sequence mask to shield load data after period t. Given a vector $\boldsymbol{X}$, the sequence mask is expressed as:

$$\boldsymbol{M}_t^{seq} = [\underbrace{1, \dots, 1}_{\text{t}}, \underbrace{0, \dots, 0}_{\text{T-t}}], \quad \forall t \in [1, T], \tag{18}$$

where $\boldsymbol{M}_t^{seq} \in \mathbb{R}^{1*T}$ is the sequence mask during period t.

Similar to the DR mask, we replaces the similarity of future positions with $-\infty$ according to the sequence mask, so that the weight of the corresponding position becomes 0 (see details in). Moreover, the DR mask is also utilized in the decoder to reduce interference of non-CBL data.

Unlike the auto-regressive way in traditional Transformer [29], we directly output the entire sequence to improve the computational efficiency of the decoder. The decoder's output subsequently passes through a linear layer consisting of a fully connected neural network, to form model's final output, i.e., the complete estimated CBL sequence.

3.3. GAN-based data imputation framework

GAN [30] is composed of a *generator* and a *discriminator*, which is used to address the missing data problem in power systems. For example, Ren et al. [39] and Zhang et al. [40] exploit the GAN for power system dynamic security assessment with missing data and solar data imputation, respectively. Further, Li et al. [41] build a cyber–physical model based on the GAN method to detect and defend against false data injection attacks in the load frequency control system.

The *generator* learns to produce the target data based on the input data, and its objective is to make the generated data distribution as close to the real data distribution as possible. The *discriminator* distinguishes the generated data from real data, and its objective is to assign the corresponding correct labels to both the generated data and the real data. Through the adversarial game, when the Nash equilibrium is reached, the generator can generate new data that obey the real data distribution and cannot be distinguished by the discriminator. According to [30], the generator and discriminator play the two-player min–max game with value function $V(D,G)$, as:

$$\begin{aligned}\min_G \max_D V(D,G) = {} & \mathbb{E}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[\log(1-D(\tilde{\boldsymbol{x}}))] \\ & + \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_r}[\log(D(\boldsymbol{x}))], \quad \tilde{\boldsymbol{x}} = G(\boldsymbol{z}),\end{aligned} \tag{19}$$

where D denotes the discrimination function, and $D(\boldsymbol{x})$ represents the probability that $\boldsymbol{x}$ comes from the real data distribution; G and $G(\boldsymbol{z})$ are the generation function and generated sample, and $\boldsymbol{z}$ denotes the input data of G; $\boldsymbol{x}$ and $\mathbb{P}_r$ denote the real data and its distribution; $\tilde{\boldsymbol{x}}$ and $\mathbb{P}_g$ are the generated data and its distribution. We train D to maximize the probability of assigning the correct labels to both the real data and generated data from G. Meanwhile, we also train G to minimize the probability that $D(G(\boldsymbol{z}))$ is discriminated as generated data.

According to [30], given any generator G, the optimal discrimination function $D^*(\boldsymbol{x})$ can be expressed as follows:

$$D^*(\boldsymbol{x}) = \frac{\mathbb{P}_r(\boldsymbol{x})}{\mathbb{P}_r(\boldsymbol{x}) + \mathbb{P}_g(\boldsymbol{x})}. \tag{20}$$

Under the optimal discrimination function, the generation function G is formulated according to Eqs. (19) and (20) as:

$$\begin{aligned}G(\boldsymbol{x}) = {} & \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_r}\left[\log\left(\frac{\mathbb{P}_r(\boldsymbol{x})}{\mathbb{P}_r(\boldsymbol{x}) + \mathbb{P}_g(\boldsymbol{x})}\right)\right] \\ & + \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_g}\left[\log\left(\frac{\mathbb{P}_g(\boldsymbol{x})}{\mathbb{P}_r(\boldsymbol{x}) + \mathbb{P}_g(\boldsymbol{x})}\right)\right] \\ = {} & 2\cdot JSD(\mathbb{P}_r\|\mathbb{P}_g) - 2\log 2,\end{aligned} \tag{21}$$

where $JSD(\mathbb{P}_r\|\mathbb{P}_g)$ denotes the Jensen–Shannon divergence between $\mathbb{P}_r$ and $\mathbb{P}_g$. The minimum generation function $G^*(\boldsymbol{x}) = -2\log 2$ if and only if $\mathbb{P}_r = \mathbb{P}_g$.

According to Eq. (21), the objective function of the generation function is equivalent to minimizing the discrepancy between $\mathbb{P}_r$ and $\mathbb{P}_g$, which is measured by the Jensen–Shannon divergence. However, this may cause the generator's gradient vanishing or low diversity of generated data, also known as mode collapse problem [42]. To improve the training stability and get rid of the mode collapse, we utilize Wasserstein GAN (WGAN) in the proposed framework [43]. WGAN applies the Wasserstein distance to reflect the proximity between $\mathbb{P}_r$ and $\mathbb{P}_g$, as:

$$W(\mathbb{P}_r,\mathbb{P}_g) = \inf_{\gamma\in\prod(\mathbb{P}_r,\mathbb{P}_g)} \mathbb{E}_{(\boldsymbol{x},\tilde{\boldsymbol{x}})\sim\gamma}[\|\boldsymbol{x}-\tilde{\boldsymbol{x}}\|], \tag{22}$$

where $W(\mathbb{P}_r,\mathbb{P}_g)$ is Wasserstein distance; $\prod(\mathbb{P}_r,\mathbb{P}_g)$ denotes the set of all joint distribution γ whose marginal distributions are $\mathbb{P}_r$ and $\mathbb{P}_g$; $\mathbb{E}_{(x,\tilde{x})\sim\gamma}[\|x-\tilde{x}\|]$ represents the expected distance between $\mathbb{P}_r$ and $\mathbb{P}_g$ under joint distribution γ.

However, the infimum in Eq. (22) is highly intractable [43]. For this issue, WGAN applies Kantorovich–Rubinstein duality [44] to reformulate the Wasserstein distance in Eq. (22), as:

$$W(\mathbb{P}_r,\mathbb{P}_g) = \sup_{\|f\|_L\le 1} \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_r}[f(\boldsymbol{x})] - \mathbb{E}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[f(\tilde{\boldsymbol{x}})], \tag{23}$$

where $\|f\|_L \le 1$ represents that the function f satisfies Lipschitz continuity, and its Lipschitz constant is 1[45]; that is, $\forall x_1, x_2 \in \mathcal{X}, |f(x_1) - f(x_2)| \le 1\cdot|x_1 - x_2|$. In addition, WGAN takes advantage of neural networks that have the powerful fitting ability, to replace the function f, as follows:

$$W(\mathbb{P}_r,\mathbb{P}_g) \approx \max_{\theta:\|f_\theta\|_L\le 1} \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_r}[f_\theta(\boldsymbol{x})] - \mathbb{E}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[f_\theta(\tilde{\boldsymbol{x}})], \tag{24}$$

where f_θ is the neural network model with parameters θ.

According to Eq. (24), we utilize the discrimination function of WGAN to serve as f_θ, thereby calculating $W(\mathbb{P}_r,\mathbb{P}_g)$. Furthermore, the generation function of WGAN aims to minimize $W(\mathbb{P}_r,\mathbb{P}_g)$ to generate data samples that follow the real data distribution as much as possible. Therefore, the value function of WGAN can be written as:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_r}[D(\boldsymbol{x})] - \mathbb{E}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[D(\tilde{\boldsymbol{x}})], \tag{25}$$

where D is responsible for calculating the Wasserstein distance based on Eq. (24), rather than distinguishing data in Eq. (19).

Because we use D to replace f_θ in Eq. (24), the discrimination function D needs to satisfy the Lipschitz continuity in Eq. (23). Since the function is 1-Lipschitz if and only if the norm of its gradient does not exceed 1, we perform a constraint on the gradient norm of the output of D. In order to circumvent tractability issues, we penalize the gradient norm for random data $\hat{\boldsymbol{x}}$, rather than $\boldsymbol{x}$ or $\tilde{\boldsymbol{x}}$, according to [46]. Thus, the loss function of D is formulated as:

$$\begin{aligned}L_D = {} & \underbrace{\mathbb{E}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[D(\tilde{\boldsymbol{x}})] - \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_r}[D(\boldsymbol{x})]}_{\text{original loss}} \\ & + \underbrace{\mathbb{E}_{\hat{\boldsymbol{x}}\sim\mathbb{P}_{\hat{\boldsymbol{x}}}}[(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2]}_{\text{gradient penalty}},\end{aligned} \tag{26}$$

where $\|\cdot\|_2$ is the 2-norm; and $\mathbb{P}_{\hat{\boldsymbol{x}}}$ denotes the random data distribution. The random data is interpolated between the real data and generated data, as follows:

$$\hat{\boldsymbol{x}} = \epsilon\cdot\boldsymbol{x} + (1-\epsilon)\cdot\tilde{\boldsymbol{x}}, \tag{27}$$

where $\epsilon \sim U[0,1]$ is the random interpolation number.

Moreover, according to the value function in Eq. (25), given an arbitrary D, the loss function of G is written as:

$$L_G = -\mathbb{E}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g}[D(\tilde{\boldsymbol{x}})]. \tag{28}$$

3.4. Time-series missing data imputation for CBL estimation

Algorithm 1: Training Process with WGAN

Input : The initial parameters of generator ω_0 and discriminator θ_0, Adam parameters α, β_1, β_2, the epoch number E, the batch size B, the discriminator iteration number n_d, the coefficient of squared error loss λ_1 and gradient penalty λ_2.

Output: The well-trained parameters of generator ω and discriminator θ.

Procedure:
for $e = 1, \ldots, E$ **do**
  for $n = 1, \cdots, n_d$ **do**
    for $b = 1, \cdots, B$ **do**
      Sample random noise $\eta \in \mathcal{N}(0,1)$, a random number $\epsilon \sim U[0,1]$ and real data $\boldsymbol{x} \sim \mathbb{P}_r$ with corresponding DR masks $\boldsymbol{m}$;
      Generate new data $\tilde{\boldsymbol{x}} \leftarrow G(\boldsymbol{z} + \boldsymbol{\eta}, \boldsymbol{m})$;
      Produce random data $\hat{\boldsymbol{x}} \leftarrow \epsilon \boldsymbol{x} + (1-\epsilon)\tilde{\boldsymbol{x}}$;
      Calculate loss of the discriminator $L_D^{(i)} = D(\tilde{\boldsymbol{x}}) - D(\boldsymbol{x}) + \lambda_2(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2$;
    end
    Update model parameters of the discriminator $\theta \leftarrow Adam(\frac{1}{B}\nabla_\theta \sum_{b=1}^{B} L_D^{(i)}, \theta, \alpha, \beta_1, \beta_2)$;
  end
  for $b = 1, \cdots, B$ **do**
    Sample real data $\boldsymbol{x} \sim \mathbb{P}_r$ with corresponding DR masks $\boldsymbol{m}$;
    Generate new data $\tilde{\boldsymbol{x}} \leftarrow G(\boldsymbol{z} + \boldsymbol{\eta}, \boldsymbol{m})$;
    Calculate loss of the generator $L_G^{(i)} = \lambda_1 \cdot \|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|_2 - D(\tilde{\boldsymbol{x}})$;
  end
  Update model parameters of the generator $\omega \leftarrow Adam(\frac{1}{B}\nabla_\omega \sum_{b=1}^{B} L_G^{(i)}, \omega, \alpha, \beta_1, \beta_2)$.
end

In this paper, since the CBL estimation is converted to a time-series missing data imputation issue, we train the generator to produce complete estimated CBL sequences through the adversarial game, and fill the missing CBLs with generated data. To learn CBL normal patterns accurately and efficiently, the generator is implemented by the proposed Transformer-based model in Section 3.2. Moreover, the discriminator is also composed of an encoder model that has the same structure as the generator's encoder, to make sure consistent information extraction for both real data and generated data.

Considering that we regard actual loads as CBLs with missing data, so real data in GAN-based framework are actual loads. Because actual loads are equivalent to real CBLs except for DR periods, we exploit actual load sequences as input data to make the generated data as close as possible to real CBLs. Furthermore, we add random noise to input data to prevent the generator from degenerating into a naive linear transformation. Hence, the generation process is expressed as:

$$\tilde{\boldsymbol{x}} = G(\boldsymbol{x} + \boldsymbol{\eta}, \boldsymbol{m}), \tag{29}$$

where $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ denote actual loads and the corresponding estimated CBLs, respectively; G is the generation function of WGAN; $\boldsymbol{m}$ denotes the DR mask vector (see details in Eq. (5)); and $\boldsymbol{\eta} \sim \mathcal{N}(0,1)$ is the random noise.

According to [47], although the generated sequences obey the distribution of real sequences, the difference between the generated sequence and real sequence may also be large. Therefore, we adopt the masked reconstruction loss to gauge the generation effect of generator's output, and then the loss functions of WGAN are defined based on Eqs. (26)–(28), as:

$$L_G = \lambda_1 \cdot \|\boldsymbol{m} \odot \boldsymbol{x} - \boldsymbol{m} \odot \tilde{\boldsymbol{x}}\|_2 - D(\tilde{\boldsymbol{x}}), \tag{30}$$

$$L_D = D(\tilde{\boldsymbol{x}}) - D(\boldsymbol{x}) + \lambda_2 \cdot (\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2, \tag{31}$$

where $\odot$ is the element-wise multiplication; λ_1 and λ_2 denote two loss weights, respectively. Besides, we employ the Adam algorithm [48] for model optimization to improve computational efficiency (see details in Appendix), and the training processes are summarized in Algorithm 1.

Finally, we exploit the estimated CBL sequence to fill in the missing CBLs during DR periods, which is formulated as:

$$\boldsymbol{x}_{\text{complete}} = \boldsymbol{m} \odot \boldsymbol{x} + (1 - \boldsymbol{m}) \odot \tilde{\boldsymbol{x}}. \tag{32}$$

After the comprehensive training, the system operator can estimate participants' CBLs by only using their daily load profiles of demand response days, without additional training. In this way, the required time duration for the input Sample Data is one day (i.e., 24 h) in the practical application stage. Furthermore, with the high capability of neural network models, the proposed method can be applicable to different customers or even new customers for CBL estimation.

It is worth noting that the proposed method is suitable for different types of customers, which can estimate the CBL of each customer individually. This improves the efficiency and adaptability of our proposed method in real-world deployments. Moreover, since the deep learning-based method can be used directly after training once, this further reduces the required time of CBL estimation using the proposed method in practice. In addition, because we display the framework and architecture of the proposed method, it is feasible to build an identical CBL estimation model accordingly. We also detail implementation parameters for model training in real scenarios. Therefore, our proposed method is replicable and computationally efficient when deploying DR programs in practice.

4. Case studies

4.1. Experiment settings

4.1.1. Environmental setup

The proposed transformer-based model and WGAN-based framework are implemented by the machine learning framework PyTorch, which is based on the Torch library and Python. All experiments are performed on a Ubuntu 18.04 LTS platform with the Intel Core i9-10980XE CPU and NVIDIA GeForce RTX 3090 GPU (64 GB RAM). Moreover, the details of model hyper-parameters and training settings are summarized in Table 2.

4.1.2. Dataset description

All numerical experiments are conducted using the smart meter data collected from the Commission for Energy Regulation (i.e., CER dataset) [49]. The CER dataset contains load data of more than 6400 customers in three types, covering 536 days from July 14, 2009 to December 31, 2010, with 30 min of time granularity. Considering that there are some customers with missing load data in the CER dataset, we cleaned the dataset and finally filtered out 3600 residential customers, 400 small and medium-sized enterprise customers, and 550 other customers with full load records for 536 days. Moreover, all data are divided into the training and test sets, accounting for 80% and 20% of all customers, respectively.

We assume that all filtered customers are DR participants and consider all customers' loads in the dataset as their baseline loads,

Table 2
Implementation details in case studies.

Parameter	Definition	Value
N	the encoder/decoder layers number	6
h	the head number	4
d_{model}	the embedding vector dimension	16
d_k	the key vector dimension	4
d_v	the value vector dimension	4
E	the epoch number	300
B	the batch size	16
n_d	the discriminator iteration number	3
λ_1	the squared loss weight	2.0
λ_2	the gradient penalty weight	10.0
α	the learning rate of Adam	0.0001
$(\beta_1, \beta 2)$	the decay rate pair of Adam	(0.9, 0.999)

which are used as the ground-truth to verify the performance of CBL estimation. Moreover, we generate the customers' actual loads by artificially constructing DR events. Specifically, when the DR event occurs, we manually reduce the baseline load of this time by a certain degree according to Eq. (1), and consequently produce the actual load during the DR period. It is worth noting that we only consider the DR program with load reduction in this paper. Thus, the DR periods are selected on the basis of the peak period of the time-of-use scheme in the electricity market. In addition, we also form the CBL and actual load for scenarios with aggregated customers based on Eq. (7), and the aggregation set is randomly selected rather than any clustering algorithm to avoid manual intervention.

4.1.3. Performance metrics

There are two metrics used in this paper to calculate the CBL estimation discrepancy and measure the performance of the proposed method, which are the root mean square error (RMSE) and mean absolute percentage error (MAPE), respectively. Moreover, because the non-DR periods' CBLs are not our concern, we only calculate the error of estimated CBLs during the DR periods.

4.1.4. Benchmarks

To validate the effectiveness of the proposed method, we compare the CBL estimation performance of our proposed method with the following benchmarks:

- B1: The Mid4of6 method recommended by PJM [14];
- B2: The High5of10 method adopted by NYISO [15];
- B3: The exponential moving average method used by ISONE [16];
- B4: The support vector regression-based method [17];
- B5: The stacked autoencoder-based method with pseudo-load selection [25];
- B6: The cascaded stacked autoencoder-based method with privacy-preserving [26].

4.2. Performance comparison with state-of-the-art studies

In this part, we compare the performance of our proposed method with the aforementioned six benchmarks for the three types of customers. To comprehensively validate the proposed method, we consider four aggregation numbers (i.e., 1, 10, 50 and 100 customers) to simulate different customer load levels, because the public dataset only contains individual customers. Here, the VPP is regraded as an aggregated customer. Owing to the limited data of small and medium-sized enterprise customers and other customers, we do not perform the 100-customer aggregation for these two types. According to the electricity market, we set the load reduction degree to be 30% and the DR period to be from 4 p.m. to 7 p.m.

Table 3 summarizes the whole performance comparison results, in terms of RMSE and MAPE. Obviously, the proposed method outperforms all the benchmarks in each customer aggregation level for all the three types of customers. Its RMSE and MAPE are kept within 5 kW and 9%, respectively. In contrast, the estimation performances of benchmarks are much inferior, where the worst discrepancy is over 16 kW and the maximum MAPE is close to 18%. Even the most advanced benchmark B6 has larger errors compared with the proposed method, reflecting in a 2 kW increase in RMSE and 2% arise in MAPE. Besides, the proposed method is robust to heterogeneity of customers, the minimum standard deviation of MAPE in our proposed method is only 0.19% among all scenarios, which is also smaller than benchmarks. Therefore, the proposed method is validated to have accurate and stable CBL estimation performance for different customer types.

In addition, as the aggregation number of aggregated customers increases, the model performance of both our proposed method and six benchmarks enhances. This is because the aggregation of customers with different load patterns leads to a decrease in load uncertainty. Specifically, when the aggregation number rises from 1 to 50, the MAPE of the proposed method decreases by 42.42%, 52.56% and 42.84% for the three types of customers, respectively. Especially for the residential customer with the aggregation number of 100, the RMSE of the proposed method is around 3 kW and the MAPE is only 3.43%, which is lower than 4.1% in the best benchmark. Therefore, the proposed method is verified to have accurate and stable CBL estimation performance for different load levels.

To demonstrate the model performance comparison in a more intuitive and clear way, we randomly select one day for each type of customer as examples, where the aggregation number is 50. The estimation results of our proposed method and 6 benchmarks are shown in Fig. 4. Because B1, B2 and B3 are average methods, the estimated CBLs are highly dependent on historical loads and subject to large estimation errors, owing to the fluctuation of daily customer loads. Moreover, the estimation accuracy of B4, B5 and B6 is improved by using data-driven models, where B6's estimation result even approximately fits the real CBL curve. However, there is still a visible gap between the real CBL and estimated results of these data-driven-based benchmarks. This is because these data-driven-based benchmarks are trained using load data that have a similar load level as 300-customer aggregation, but are not applicable for other load levels. In contrast, the estimated curve of our proposed method is closest to the real CBL curve, for each type of customer. Specifically, the average estimation bias of the proposed method is approximately 2, 4 and 3 kW in each example, which is apparently lower than all benchmarks. Therefore, this further demonstrates the effectiveness and superiority of the proposed method.

4.3. Model sensitive analysis

In this part, we validate the robustness of the proposed method through different DR periods and load reduction degrees, because these two parameters influence the model performance. Similarly, we choose residential customers from the test set for demonstration and the aggregation number is 50. According to the DR experience and peak period in the electricity market, we pick six load reduction degrees (i.e., 5%, 10%, 15%, 20%, 25%, and 30%) and two DR periods (i.e., 10 a.m. to 12 noon and 4 p.m. to 7 p.m.).

Fig. 5 presents the estimation errors of the proposed method under six load reduction degrees and two DR periods. It is clear that the estimation discrepancy rises as the load reduction degree increases, reflected in the boost of median RMSE from 1.85 kW to 2.06 kW during the noon period. Similarly, in the period between 4 p.m. to 7 p.m., the estimation error also rises, where the average RMSE increases from 1.89 kW to 2.04 kW. However, despite the boost in the estimation error, the maximum RMSE of our proposed method is still below 2.2 kW. Furthermore, although the model performances in the two DR periods are not exactly identical, the variation trend and statistical result of RMSE are very similar. This demonstrates that our proposed method is robust to the DR period and load reduction degree.

Table 3
Numerical results of the proposed method and benchmarks with varying aggregation numbers for three customer types.

C	A	B1		B2		B3		B4		B5		B6		Proposed	
		RMSE	MAPE	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE
R	**1**	0.1852	16.6769	0.1902	16.9476	0.1781	15.2201	0.14878	13.0762	0.1367	11.9877	0.1152	9.9748	**0.0882**	**7.7693**
	10	1.3890	12.4892	1.4335	12.9055	1.2913	11.6767	1.04748	10.2089	0.8862	8.6442	0.7808	7.6442	**0.6248**	**5.6139**
	50	5.9135	9.0312	5.7108	8.7950	5.2977	8.3546	4.2758	6.8856	3.69119	6.0038	3.1593	5.1741	**2.0541**	**4.4735**
	100	11.7248	8.8268	11.3635	8.43973	10.6607	7.5866	8.8589	6.6203	6.3912	4.9696	5.2681	4.1672	**3.5445**	**3.4329**
S	**1**	0.8040	18.9603	0.7962	18.2878	0.6167	16.4970	0.3994	15.5285	0.3512	13.0844	0.2791	10.6631	**0.2103**	**8.8835**
	10	3.7982	13.3207	3.6556	13.1030	3.3492	12.4465	2.3609	10.1546	1.8213	8.6332	1.2204	7.4013	**0.7349**	**5.3583**
	50	16.2278	10.0473	15.9451	9.8626	14.8366	9.0190	11.3402	7.6118	8.9222	6.3371	6.8862	5.3433	**4.6585**	**4.2139**
O	**1**	0.2216	17.8715	0.2143	17.1312	0.1901	15.9503	0.1708	13.6542	0.1449	11.6352	0.1294	10.1611	**0.0957**	**8.2205**
	10	1.5208	13.0323	1.5926	13.3181	1.3947	11.9999	1.1370	10.4087	0.9143	8.5519	0.7263	7.3002	**0.5592**	**5.9015**
	50	9.0834	9.8956	9.6146	10.1292	7.8839	9.1665	6.1274	7.5462	5.4034	6.8630	4.3798	5.6103	**2.9586**	**4.6985**

*RMSE and MAPE are in kilowatts (kW) and percent (%), respectively.
**R, S, and O stand for residential customers, small and medium-sized enterprise customers, and other customers, respectively.

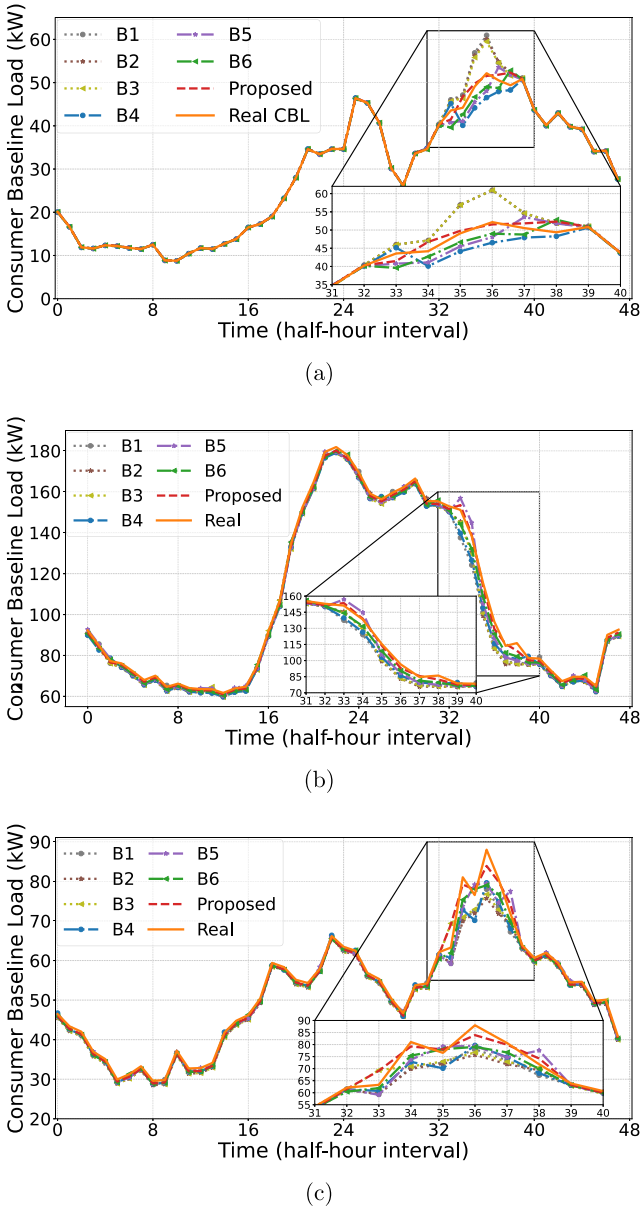


Fig. 4. Performance comparison examples of the proposed method and six benchmarks. (a) Residential customer, (b) Small and medium-sized enterprise customer, (c) Other customer.

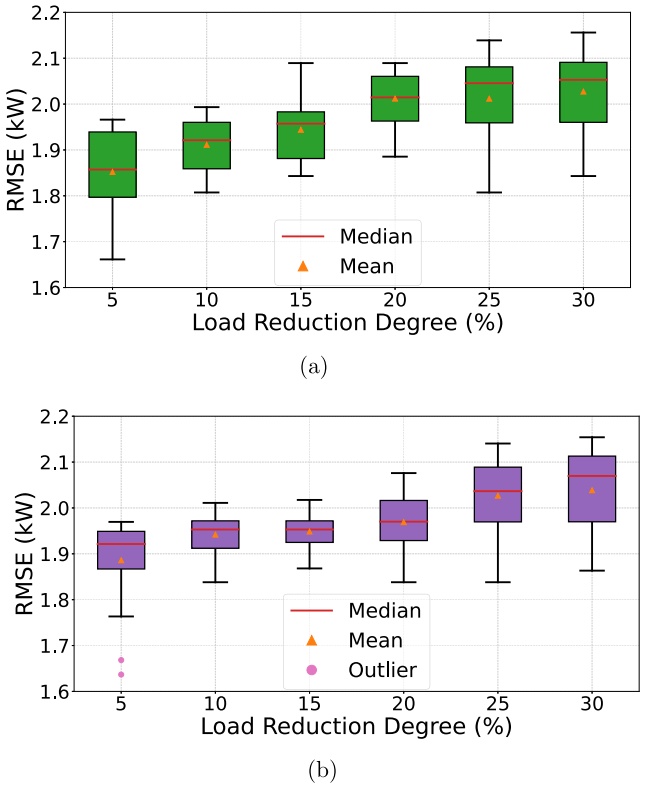


Fig. 5. The robustness testing results of the proposed method on the CER dataset. (a) 10 a.m. to 12 noon, (b) 4 p.m. to 7 p.m.

In order validate the robustness of the proposed method to different customers, we conduct a new set of experiments adopting a different public dataset (i.e., LCL dataset) [50]. Specifically, we choose 200 residential customers with load data for the whole year of 2013 from the LCL dataset. Then, we randomly construct aggregated customers and DR events in the same way as the CER dataset, where the aggregation number is 50. We conduct the same experiment on the LCL dataset as previously by using the CBL estimation model trained on the CER dataset, and the results are shown in Fig. 6. It should be noted that customers in the LCL dataset do not participate in model training, but are only used to evaluate model performance. Compared to the CER dataset, the performance of our proposed method on the LCL dataset is degraded, where the increases of the average RMSE are 0.75 kW and 0.79 kW in two DR periods, respectively. Moreover, the RMSE distribution on the LCL dataset obviously becomes wider, e.g., the

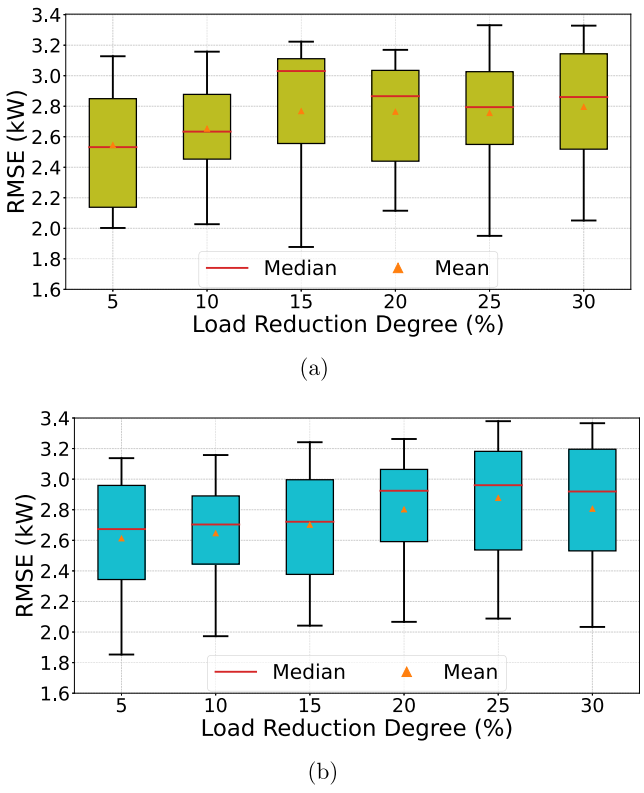


Fig. 6. The robustness testing results of the proposed method on the LCL dataset. (a) 10 a.m. to 12 noon, (b) 4 p.m. to 7 p.m.

maximum inter-quartile raises by 400% and 364% in two DR periods, respectively, thus indicating that the performance of the proposed method is not as stable as on the CER dataset. This is reasonable because the load patterns in the LCL dataset are very likely to be different from the CER dataset, so there is a slight increase in estimation error. However, although the accuracy and stability of the proposed method for these new customers have declined, the maximum RMSE is still within 3.5 kW, while their daily average load of is around 30 kw. This indicates that our proposed method still achieve the good CBL estimation performance on the LCL dataset. Therefore, the robustness of our proposed method is further validated.

4.4. DR managerial insight from CBL estimation

The accurate CBL estimation is critical for DR implementation, especially the financial settlement, which is usually performed by power system (or electricity market) operators to pay compensation to DR participants. The performance of baseline load estimation has a direct impact on the efficiency and effect of financial settlement, which further affects the enthusiasm of customers to participate in DR programs. According to the results of comprehensive case studies in this paper, we provide our management insights for DR programs to system operators, as follows:

(a) Aggregate individual customers: The system operator should encourage and facilitate the aggregation of individual customers. Furthermore, during the actual DR operation, the operators need to give priority to aggregated customers to participate in DR programs. Based on the results of performance comparison, as the aggregation number of customers increases, the performance of CBL estimation improves. In detail, when the aggregation number rises from 1 to 50, the MAPE of the proposed method decreases by 42.42%, 52.56%, and 42.84% for the three types of customers, respectively.

(b) Avoid excessive load variation: The system operator should avoid excessively changing the normal load levels of participants when assigning DR tasks. Specifically, the operators need to consider distributing the overall task among all participants as much as possible, rather than some individual participants. According to the results of model sensitive analysis, the estimation discrepancy of the proposed method rises as the load reduction degree increases. Specifically, when the load reduction degree rises from 5% to 30%, the average RMSE of the proposed method increases by 11.35% and 7.94% in the two DR periods, respectively. The proposed method has better performance for the lower load reduction degrees.

(c) Participate up to once per day: The system operator should try to prevent arranging the same participant to participate in DR programs multiple times in a day. Furthermore, while ensuring the normal operation of DR programs, the operators need to control the participating frequency of individual participants. Considering that the proposed method exploits actual loads of non-DR periods to extract characteristics of daily loads, so participants are advised to engage in DR programs up to once in one day. This is because if participants are heavily involved in DR programs on a single day, there are only a few or even no actual loads available for characteristics extraction.

In summary, we make these above insights to enhance the CBL estimation performance of the proposed model, thus improving the efficiency and effect of financial settlement. Furthermore, a good financial settlement can allow participants to receive reasonable compensation and then promote them to participate in DR programs, which helps power system operators to guarantee the normal operation of DR.

5. Conclusion

In this paper, we focus on the CBL estimation issue for VPPs in incentive-based DR programs, which is nontrivial because a customer's CBL cannot be recorded and is also affected by uncertainty and heterogeneity of customer load. To realize the high-quality CBL estimation, we propose a Transformer neural network model based on the attention mechanism and develop a GAN-based data imputation framework. The proposed method can achieve an accurate CBL estimation for various types of customers through learning CBL's normal pattern and training with an adversarial game. Case studies validate the effectiveness and superiority of the proposed method, compared with existing methods. Its RMSE and MAPE are both kept within 5 kW and 9% among three types of customers, respectively. Furthermore, the robustness of our proposed method is also verified by performance comparison under different load reduction degrees and DR periods.

In this paper, we do not consider the interaction of CBL estimation and price design, so we intend to design an attractive and personalized price for customers based on CBL estimation results, thus improving the efficiency of DR. Moreover, since the proposed method is based on deep learning, there is an interpretability issue, which leads to customers not being able to fully trust estimated CBLs. In the future, we plan to enhance the interpretability of the proposed method to improve its transparency and credibility. In addition, with the continuous penetration of distributed energy resources, it is necessary to make the proposed method adapt to customers with energy storage or renewable generation. In our future work, we will consider using customer-reported renewable generation and charging/discharging strategies to eliminate their impacts on customers' normal load patterns. We can also add some environmental and strategy-related features as inputs to our proposed model to cope with the presence of distributed energy resources.

CRediT authorship contribution statement

Zhenyi Wang: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Hongcai Zhang:** Writing – review & editing,

Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization.

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

The authors do not have permission to share data.

Acknowledgments

This paper is funded in part by The Science and Technology Development Fund, Macau SAR (File No. 0011/2022/AGJ and File No. SKL-IOTSC(UM)-2021-2023).

Appendix. Adam optimization algorithm

The loss function of neural network models is usually a non-convex function, so it is difficult to find the global optimal solution. Moreover, due to the large number of model parameters and large training data, the computational cost of an optimization algorithm is usually high [48]. At present, the parameter learning of deep neural network models mainly uses the gradient descent method to find a set of optimal parameters [51]. In this paper, we choose the Adam algorithm as the optimization algorithm to take into account the optimization speed and stability at the same time, which can be regarded as the combination of the momentum method [52] and the RMSprop algorithm [53]. Specifically, in order to make the optimization process more stable, the Adam algorithm uses momentum as the direction of parameter update. The actual update amount of model parameters depends on the weighted average of past gradients, which can increase the stability by reducing the rate of gradient descent later in the iteration. Furthermore, in order to increase the optimization speed, the gradient estimation correction is adopted to adaptively adjust the learning rate. It avoids the premature decay of the learning rate due to its constant monotonous decrease, thus ensuring the optimization speed.

The Adam algorithm uses the exponentially weighted average of the gradient to update the first moment estimate of gradient on the one hand, and uses the exponentially weighted average of gradient square to update the second moment estimate of gradient on the other hand, as follows:

$$u_t = \gamma_1 \cdot u_{t-1} + (1-\gamma_1)\cdot \nabla_t, \tag{A.1}$$

$$v_t = \gamma_2 \cdot v_{t-1} + (1-\gamma_2)\cdot \nabla_t \odot \nabla_t, \tag{A.2}$$

where ∇_t denotes the gradient at iteration t; u_t and v_t are the first and second moment estimates of the gradient, respectively; γ_1 and γ_2 denote two exponential decay rates, whose values are usually 0.9 and 0.99, respectively; $\odot$ is the element-wise multiplication.

Since the first and second moment estimates are usually initialized to 0, these two estimates have a large bias from the real value of mean and variance at the beginning of the iteration. So the biases need to be corrected, as follows:

$$\hat{u}_t = \frac{u_t}{1-\gamma_1^t}, \tag{A.3}$$

$$\hat{v}_t = \frac{v_t}{1-\gamma_2^t}. \tag{A.4}$$

In conclusion, the model parameters ω are updated as follows:

$$\omega_t \leftarrow \omega_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t}+\epsilon}\cdot \hat{u}_t, \tag{A.5}$$

where η denotes the learning rate, which is usually set to 0.001; ϵ is a small constant to maintain numerical stability and usually set to 10^{-8}.

References

[1] Gurgel A, Mignone BK, Morris J, Kheshgi H, Mowers M, Steinberg D, et al. Variable renewable energy deployment in low-emission scenarios: The role of technology cost and value. Appl Energy 2023;344:121119.
[2] Haider HT, See OH, Elmenreich W. A review of residential demand response of smart grid. Renew Sust Energy Rev 2016;59:166–78.
[3] Wang J, Zhong H, Ma Z, Xia Q, Kang C. Review and prospect of integrated demand response in the multi-energy system. Appl Energy 2017;202:772–82.
[4] Zhong H, Xie L, Xia Q. Coupon incentive-based demand response: Theory and case study. IEEE Trans Power Syst 2012;28(2):1266–76.
[5] Ming H, Meng J, Gao C, Song M, Chen T, Choi D-H. Efficiency improvement of decentralized incentive-based demand response: Social welfare analysis and market mechanism design. Appl Energy 2023;331:120317.
[6] Wang Z, Yu P, Zhang H. Privacy-preserving regulation capacity evaluation for HVAC systems in heterogeneous buildings based on federated learning and transfer learning. IEEE Trans Smart Grid 2023;14(5):3535–49.
[7] Yang Q, Wang H, Wang T, Zhang S, Wu X, Wang H. Blockchain-based decentralized energy management platform for residential distributed energy resources in a virtual power plant. Appl Energy 2021;294:117026.
[8] Kong X, Wang Z, Liu C, Zhang D, Gao H. Refined peak shaving potential assessment and differentiated decision-making method for user load in virtual power plants. Appl Energy 2023;334:120609.
[9] Chen Y, Xu L, Egea-Àlvarez A, Marshall B. Accurate and general small-signal impedance model of LCC-HVDC in sequence frame. IEEE Trans Power Deliv 2023.
[10] Yu P, Zhang H, Song Y, Hui H, Chen G. District cooling system control for providing operating reserve based on safe deep reinforcement learning. IEEE Trans Power Syst 2023. http://dx.doi.org/10.1109/TPWRS.2023.3237888.
[11] Zhang L, Li G, Huang Y, Jiang J, Bie Z, Li X, et al. Distributed baseline load estimation for load aggregators based on joint FCM clustering. IEEE Trans Ind Appl 2022.
[12] Li K, Wang F, Mi Z, Fotuhi-Firuzabad M, Duić N, Wang T. Capacity and output power estimation approach of individual behind-the-meter distributed photovoltaic system for demand response baseline estimation. Appl Energy 2019;253:113595.
[13] Wang Z, Zhang H. Consumer baseline load estimation in demand response: A generative adversarial networks approach. In: 2022 IEEE 6th conference on energy internet and energy system integration. IEEE; 2022, p. 1723–8.
[14] PJM Load Management Task Force KEMA, Inc. PJM empirical analysis of demand response baseline methods. 2011.
[15] DNV KEMA, Inc. NYISO SCR baseline study. 2014.
[16] ISO New England Inc. Measurement and verification of demand reduction value from demand resources. 2014.
[17] Chen Y, Xu P, Chu Y, Li W, Wu Y, Ni L, et al. Short-term electrical load forecasting using the support vector regression (SVR) model to calculate the demand response baseline for office buildings. Appl Energy 2017;195:659–70.
[18] Sun M, Wang Y, Teng F, Ye Y, Strbac G, Kang C. Clustering-based residential baseline estimation: A probabilistic perspective. IEEE Trans Smart Grid 2019;10(6):6014–28.
[19] Zhou X, Gao Y, Yao W, Yu N. A robust segmented mixed effect regression model for baseline electricity consumption forecasting. J Mod Power Syst Clean Energy 2020;10(1):71–80.
[20] Wang F, Li K, Liu C, Mi Z, Shafie-Khah M, Catalão JP. Synchronous pattern matching principle-based residential demand response baseline estimation: Mechanism analysis and approach description. IEEE Trans Smart Grid 2018;9(6):6972–85.
[21] Lee E, Lee K, Lee H, Kim E, Rhee W. Defining virtual control group to improve customer baseline load calculation of residential demand response. Appl Energy 2019;250:946–58.
[22] Ge X, Xu F, Wang Y, Li H, Wang F, Hu J, et al. Spatio-temporal two-dimensions data based customer baseline load estimation approach using LASSO regression. IEEE Trans Ind Appl 2022.
[23] Lin W, Wu D, Boulet B. Spatial-temporal residential short-term load forecasting via graph neural networks. IEEE Trans Smart Grid 2021;12(6):5373–84.
[24] Zhang Y, Wu Q, Ai Q, Catalão JP. Closed-loop aggregated baseline load estimation using contextual bandit with policy gradient. IEEE Trans Smart Grid 2021;13(1):243–54.
[25] Wang X, Wang Y, Wang J, Shi D. Residential customer baseline load estimation using stacked autoencoder with pseudo-load selection. IEEE J Sel Areas Commun 2019;38(1):61–70.
[26] Chen Y, Chen C, Zhang X, Cui M, Li F, Wang X, et al. Privacy-preserving baseline load reconstruction for residential demand response considering distributed energy resources. IEEE Trans Ind Inf 2021;18(5):3541–50.
[27] Wijaya TK, Vasirani M, Aberer K. When bias matters: An economic assessment of demand response baselines for residential customers. IEEE Trans Smart Grid 2014;5(4):1755–63.
[28] Zhang L, Li G, Huang Y, Jiang J, Bie Z, Li X, et al. Distributed baseline load estimation for load aggregators based on joint FCM clustering. IEEE Trans Ind Appl 2022;59(1):567–77.

[29] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In: Advances in neural information processing systems, vol. 30. 2017.
[30] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative adversarial networks. Commun ACM 2020;63(11):139–44.
[31] Luo Y, Zhang Y, Cai X, Yuan X. E2gan: End-to-end generative adversarial network for multivariate time series imputation. In: Proceedings of the 28th international joint conference on artificial intelligence. AAAI Press; 2019, p. 3094–100.
[32] Wang C, Wang Y, Ding Z, Zheng T, Hu J, Zhang K. A transformer-based method of multienergy load forecasting in integrated energy system. IEEE Trans Smart Grid 2022;13(4):2703–14.
[33] Li Y, Wei X, Li Y, Dong Z, Shahidehpour M. Detection of false data injection attacks in smart grid: A secure federated deep learning approach. IEEE Trans Smart Grid 2022.
[34] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings; 2010, p. 249–56.
[35] Nair V, Hinton GE. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning. 2010, p. 807–14.
[36] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 2014;15(1):1929–58.
[37] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016, p. 770–8.
[38] Ba JL, Kiros JR, Hinton GE. Layer normalization. 2016, arXiv preprint arXiv:1607.06450.
[39] Ren C, Xu Y. A fully data-driven method based on generative adversarial networks for power system dynamic security assessment with missing data. IEEE Trans Power Syst 2019;34(6):5044–52.
[40] Zhang W, Luo Y, Zhang Y, Srinivasan D. SolarGAN: Multivariate solar data imputation using generative adversarial network. IEEE Trans Sustain Energy 2020;12(1):743–6.
[41] Li Y, Huang R, Ma L. False data injection attack and defense method on load frequency control. IEEE Internet Things J 2020;8(4):2910–9.
[42] Arjovsky M, Bottou L. Towards principled methods for training generative adversarial networks. In: International conference on learning representations. 2017.
[43] Arjovsky M, Chintala S, Bottou L. Wasserstein generative adversarial networks. In: International conference on machine learning. PMLR; 2017, p. 214–23.
[44] Villani C. Optimal transport: Old and new, vol. 338. Springer; 2009.
[45] Zhou Z, Liang J, Song Y, Yu L, Wang H, Zhang W, et al. Lipschitz generative adversarial nets. In: International conference on machine learning. PMLR; 2019, p. 7584–93.
[46] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville A. Improved training of wasserstein GANs. In: Proceedings of the 31st international conference on neural information processing systems, vol. 30. 2017.
[47] Luo Y, Cai X, Zhang Y, Xu J, xiaojie Y. Multivariate time series imputation with generative adversarial networks. In: Advances in neural information processing systems, vol. 31. 2018.
[48] Kingma DP, Ba J. Adam: A method for stochastic optimization. 2014, arXiv preprint arXiv:1412.6980.
[49] Commission for Energy Regulation (CER). CER smart metering project - Electricity customer behaviour trial. 2012, 2009-2010 [dataset]. 1st Edition. Irish Social Science Data Archive. SN: 0012-00. https://www.ucd.ie/issda/data/commissionforenergyregulationcer/.
[50] Schofield JR, Carmichael R, Tindemans S, Bilton M, Woolf M, Strbac G, et al. Low carbon London project: Data from the dynamic time-of-use electricity pricing trial, 2013. UK Data Serv SN 2015;7857(2015):1–5.
[51] Ruder S. An overview of gradient descent optimization algorithms. 2016, arXiv preprint arXiv:1609.04747.
[52] Sutskever I, Martens J, Dahl G, Hinton G. On the importance of initialization and momentum in deep learning. In: International conference on machine learning. PMLR; 2013, p. 1139–47.
[53] Hinton G, Srivastava N, Swersky K. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent, 14 (8). 2012, p. 2, Cited on.