Contents lists available at ScienceDirect

Applied Energy

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

Robust learning-based real-time load estimation using sparsely deployed smart meters with high reporting rates

Md. Zahidul Islam [a], Yuzhang Lin [a,*], Vinod M. Vokkarane [a], Nanpeng Yu [b]

[a] Department of Electrical and Computer Engineering, University of Massachusetts, Lowell, MA 01854, USA
[b] Department of Electrical and Computer Engineering, University of California, Riverside, CA 92501, USA

HIGHLIGHTS

- A novel high-resolution Real-Time Load-Estimation (RTLE) method is proposed to estimate unmeasured feeder or subfeeder load.
- A state-of-the-art deep-learning model is utilized for RTLE.
- A HRRSM placement strategy is proposed for effective monitoring of feeder- or subfeeder-level load.
- The proposed RTLE method is designed to be robust against anomalies in real-time HRRSM data.

ARTICLE INFO

Keywords:
Anomaly detection
Clustering
Distribution system
Load estimation
Load forecasting
Meter placement
Smart meter

ABSTRACT

With increasing renewable generation and demand response, the load profiles of distribution feeders become more fluctuating and uncertain, requiring real-time load estimation (RTLE) with high temporal granularity. Smart meters (SM) provide new data sources that have the potential to enable RTLE. However, it is cost prohibitive to communicate and process real-time high-resolution data from a massive number of SMs. To address the challenge, this paper proposes a novel solution to RTLE using High-Reporting-Rate SMs (HRRSMs) installed at a sparsely selected subset of customers in the feeder. The first step is to select customers for installing HRRSMs based on clustering, such that load profiles can best represent those of the others and the whole feeder. Then, a state-of-the-art Deep Learning (DL) model is trained to capture the relation between the historical load profiles of the selected customers and that of the feeder. Finally, real-time HRRSM data from the selective customers is fed to the trained model to perform RTLE with high resolution. The method is also robustified to address anomalies in real-time HRRSM data streams. The proposed method is validated on a large real-world SM dataset. Simulation results show that even with a small number of HRRSM installation, the proposed method can track feeder loads with much improved accuracy and temporal granularity compared with conventional methods based on historical data of regular SMs, providing a cost-effective solution to the monitoring of distribution feeder loads.

1. Introduction

With the increase of distributed generation (DG), energy storage, electric vehicles, and microgrids, traditional passive distribution systems are evolving towards actively-controlled systems [1–3], where the utility-customer interactions will become more complex and uncertain [4]. Consequently, the power flow profiles will be more volatile and unpredictable, requiring real-time monitoring of feeder- or subfeeder-level loads in high granularity to support various decision making processes, e.g., renewable energy hosting capacity estimation [5], voltage control [6], demand response [7], fault/outage detection [8].

In recent years, the extensive deployment of smart meters (SMs) in millions of households provides a tremendous volume of electricity consumption data at 15-min to 1-h granularity. However, efficient data analytics are necessary for extracting information from this data asset [9] in order to benefit situational awareness, resilience, and reliability of distribution systems [10,11]. Typical applications of SM data analytics include state estimation [6], load modeling [12], electricity price forecasting [13], outage management [14], etc.

Load forecasting (LF) is a critical function for the operation of active distribution systems. Existing feeder-level LF methods in literature forecast feeder load based on the historical data from feeder-head

* Corresponding author.
E-mail address: yuzhang_lin@uml.edu (Y. Lin).

https://doi.org/10.1016/j.apenergy.2023.121964
Received 6 June 2022; Received in revised form 22 August 2023; Accepted 11 September 2023
Available online 22 September 2023
0306-2619/© 2023 Published by Elsevier Ltd.

SCADA measurements [15–18]. However, these kinds of methods can only forecast the load of a whole feeder measured at the feeder head; for the load at a subfeeder level (e.g., the load of a lateral, a section of the feeder, or a micro-grid), load forecasting cannot be performed due to the lack of measurements.

With the extensive deployment of SMs, historical data at both the feeder level [19–21] and the customer level [22] [23] are incorporated into load forecasting. As SMs usually record measurement data at a resolution of 15 min or longer and send the stored data to the control center on a daily basis, their data becomes available for use with a one-day delay [24]. Therefore, most existing methods use the SM data up to the previous day to perform Day-Ahead Load Forecasting (DALF). These methods adopt various algorithms such as Linear Regression (LR), Autoregressive Integrated Moving Average (ARIMA) and machine learning-based algorithms such as Support Vector Machine (SVM), Multilayer Perceptron (MLP), Long Short Term Memory (LSTM) networks, Temporal Convolutional Network (TCN) and so on [25,26]. Their performances are highly dependent on the assumption that the daily load profile follows a very similar pattern as reflected in the historical data. However, when the actual load profile deviates from historical patterns, which will only become more likely in the future due to the increasing uncertainty from renewable generation and customer participation, the performances of these methods may degrade as they do not receive any real-time information to update the inference.

With the advancement in SM technology, high-reporting-rate SMs (HRRSMs), which can provide real-time measurements at 1-s to 1-min resolution, have drawn attention recently [27]. Several projects are currently underway around the world to realize the next generation smart meter (e.g., HRRSM) with increased computation, memory, programmability, and data capture capability [27] [28]. It is also expected that HRRSM will grow from 4% to 35% by 2030, bringing together the research and industry communities to support HRRSM projects both financially and technically. With the help of real-time high-resolution SM data, real-time load estimation (RTLE) with higher accuracy and refined granularity than conventional DALF can be achieved [29]. The real-time estimated load will make the distribution grid more observable and enable the execution of distribution system state estimation at higher resolution, capturing the dynamic operation of the grid with highly integrated renewables and controllable loads [30]. In addition, it will open the door for many other applications such as voltage management [6], real-time demand response [7], fault/outage detection [8], service restoration [31], among others.

Despite great advantages of RTLE, it is considered impractical to equip every customer with HRRSMs in the near future, as it involves substantial investments and will also raise prohibitive data communication, storage, and processing requirements. To date, very few studies have presented strategies to address this challenge. Ref. [29] makes an attempt of estimating the load of a distribution transformer by a linear regression model using real-time SM measurements from a few customers. However, this method deals with a very small group of customers and is not adequate for feeder- or subfeeder-level load estimation where there are hundreds or thousands of customers. Another challenge is that real-time SM data may contain anomalies due to a variety of reasons, including meter malfunction, communication delay or failure, or even false data injection attack [32–34], which can severely degrade the accuracy of RTLE.

In view of the outstanding challenges, this paper proposes a novel method for RTLE of unmeasured feeder or subfeeder load using sparsely but strategically placed HRRSMs with tractable cost and communication requirements. The key idea is to exploit the similarity between load profiles of different customers, and install HRRSMs at a few representative customers to capture the characteristics of a feeder or any feeder section of interest. Then, the real-time HRRSM data of the selected customers will be used for RTLE based on a deep-learning method. The approach is also made robust against anomalies, such that the RTLE result can remain reliable even in the presence of abnormal data from certain HRRSMs. A large real-world SM dataset is used to validate the proposed method. The main contribution of this article is summarized as below.

1) A deep-learning-based RTLE method is proposed, which can agilely track the aggregated load of any unmeasured feeder section of interest containing hundreds to thousands of customers using sparsely but strategically placed HRRSMs with modest capital cost and communication requirement.
2) A HRRSM placement strategy is proposed, which can be used to guide the deployment of HRRSMs for effective monitoring of feeder- or subfeeder-level load.
3) The proposed RTLE method is designed to be robust against anomalies in real-time HRRSM data due to meter malfunction, communication delays or failure, or false data injection.

The rest of the paper is organized as follows: Section 2 summarizes the proposed RTLE. Sections 3 and 4 detail the proposed RTLE, while Sections 5 and 6 describe the case studies and conclusions, respectively.

2. Overview of the proposed RTLE method

With the motivation described in Section 1, the proposed framework constitutes two main stages: the *planning* stage and the *operational* stage. The operational stage further constitutes two substages, namely the *offline model training* substage and the *online load estimation* substage as shown in Fig. 1.

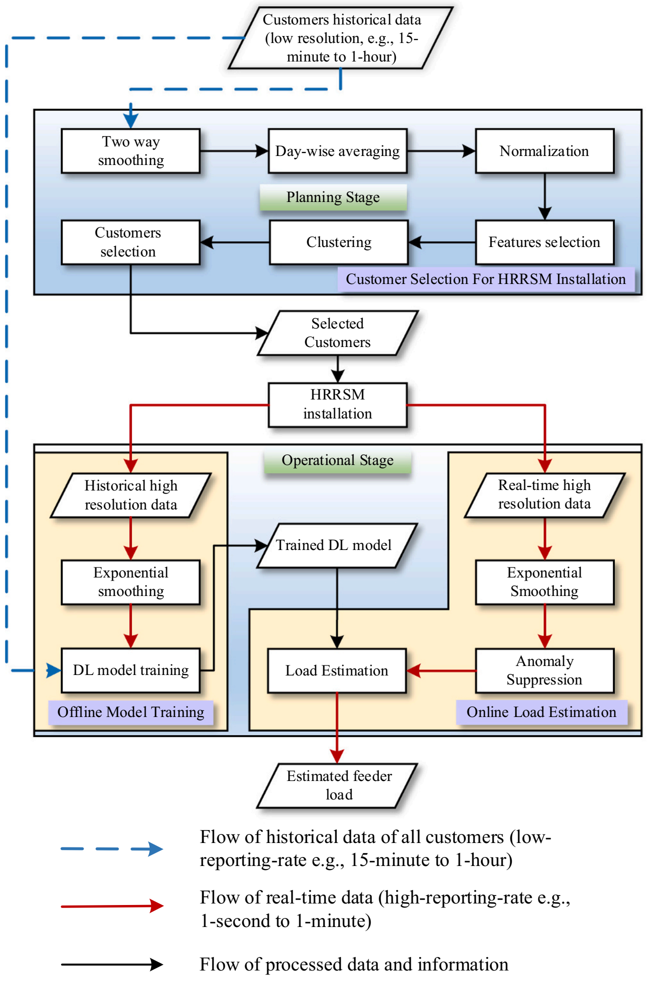
Customers historical data (low resolution, e.g., 15-minute to 1-hour)
Two way smoothing
Day-wise averaging
Normalization
Planning Stage
Customers selection
Clustering
Features selection
Customer Selection For HRRSM Installation
Selected Customers
HRRSM installation
Operational Stage
Historical high resolution data
Real-time high resolution data
Trained DL model
Exponential smoothing
Exponential Smoothing
DL model training
Load Estimation
Anomaly Suppression
Offline Model Training
Online Load Estimation
Estimated feeder load
Flow of historical data of all customers (low-reporting-rate e.g., 15-minute to 1-hour)
Flow of real-time data (high-reporting-rate e.g., 1-second to 1-minute)
Flow of processed data and information

Fig. 1. Overview of the proposed method.

In the *planning* stage, a small subset of customers from all the customers on the same feeder (or feeder section) of interest is strategically selected such that they can fairly represent the load profiles of all the customers. This will guide the placement of HRRSMs to track the feeder (or feeder section) load. For this purpose, features are extracted from the commonly available historical SM data (*low-reporting-rate*, e.g., 15-min to 1-h) of all customers and fed into a clustering algorithm. From the resulting clusters, a few representative customers are selected for the placement of HRRSMs. It is thus cost-effective to communicate and process data (*high-reporting-rate* e.g., 1-min to 1-s) from such a small group of customers in real time.

In the *operational* stage, real-time data from the sparsely but strategically placed HRRSMs will be used to perform RTLE for a feeder or feeder section of interest (e.g., an unmeasured lateral or micro-grid). In the *offline-model-training* substage, a state-of-the-art deep learning (DL) model is trained to capture the relation between the loads of selected customers with HRRSMs and the aggregated load of the feeder or feeder section of interest. Before training, the historical dataset is filtered in several steps for reducing the fluctuations and dimensionality of the dataset to improve the performance of the training process. Some additional important features (e.g., time periodicity, weather data) are also used in the training process. A trained DL model with optimal weights will be obtained in this substage and used in the *online-load-estimation* substage.

In *online-load-estimation* substage, high-resolution real-time data from the HRRSMs of the selected customers is fed into the trained DL model to estimate feeder (or feeder section) load in real time. This substage also contains preprocessing steps in order to smooth the time series and suppress anomalies incorporated into the data. The outcome of this substage is the estimated feeder (or feeder section) load, whose resolution is same as the HRRSMs data (e.g., 1-s to 1-min).

Note that the *planning* stage only uses historical *low-reporting-rate* data from regular SMs with the resolution of 15 min to 1 h, which is widely available in distribution systems today; whereas the *offline-model-training* substage and the *online-load-estimation* substage of the *operational* stage use real-time *high-reporting-rate* data from a small number of HRRSMs with a resolution of 1 s to 1 min.

3. Customer selection for HRRSMs installation

This section aims to address the *planning-stage* problem: to develop a method for selecting a small subset of customers for HRRSM placement. The selected customers should be good representatives of all customers on the same feeder (or feeder section), such that their real-time measurements can provide sufficient information for accurate RTLE of the feeder (or feeder section). This is accomplished by performing customer clustering based on the features of historical load profiles, and selection of representative customers from each resulting cluster.

3.1. Feature selection

Let us consider that all customers on a feeder (or a feeder section) have low-reporting-rate historical data from their regular SMs. Suppose customer c has n available load data points with a certain time resolution for the d days, which can be gathered in set L_c. If there are a total of n_{TC} customers, the dataset of all customers, L, can be written as:

$$L = \{L_c\}, c = 1, 2, \ldots, n_{TC}, \\ L_c = \left\{l^c_{i,j}\right\}, i = 1, 2, \ldots, n, j = 1, 2, \ldots, d, \tag{1}$$

where $l^c_{i,j}$ is the i^{th} load sample for the j^{th} day of c^{th} customer. For instance, when the SM data resolution is 15-min, there will be 96 data samples on a single day ($n = 96$).

As the daily load curve of an individual customer is highly fluctuating, a two-way smoothing technique is applied to pre-processing the daily load curve [35]:

$$\widehat{l}^{\,c}_{i,j} = 0.5\cdot\left(\mathbf{m}_\alpha{}^T l_{\mathrm{prev}} + \mathbf{m}_\alpha{}^T l_{\mathrm{post}}\right), \mathbf{m}_\alpha = \left[1\ (1-\alpha)\ (1-\alpha)^2\ldots(1-\alpha)^{i_d-1}\right]^T, l_{\mathrm{prev}} = \left[l^c_{i-1,j}\ l^c_{i-2,j}\ldots l^c_{i-i_d,j}\right]^T, l_{\mathrm{post}} = \left[l^c_{i+1,j}\ l^c_{i+2,j}\ldots l^c_{i+i_d,j}\right]^T, \tag{2}$$

where $[.]^T$ represents vector transpose operation, α sets the weighting factors in $\mathbf{m}_\alpha$, and i_d determines a window around the current sample for smoothing. For the marginal samples of a day, $\boldsymbol{l}_{prev}$ and $\boldsymbol{l}_{post}$ contain samples from the (j-1)th and (j + 1)th days to complete the smoothing windows, respectively.

In order to perform clustering, features of load profiles should be extracted [36]. The selected features should be able to distinguish different customers based on their load patterns while reducing the dimensionality of the dataset. Using the two-way smoothed data, the feature selection process consists of two steps. First, data samples at the same times of a day are averaged as below:

$$\overline{l}^c_i = \frac{1}{d}\sum_{j=1}^{d}\widehat{l}^{\,c}_{i,j}, \forall i. \tag{3}$$

The average load of each time of a day of each customer $\overline{L}^c = \{\overline{l}^c_i\}$ is then normalized as:

$$\widehat{L}^c = \left\{\widehat{l}^c_i\right\}, \widehat{l}^c_i = \overline{l}^c_i / max(\overline{L}^c), \forall i. \tag{4}$$

The normalization of the dataset ensures that clustering is done based on the shape of the load curve, not the magnitude of the load. At the second step of feature selection, $\widehat{L}^c$ is divided into several time windows, and the average of each window is taken as a feature. A possible option is to consider the daily load pattern shown in Table 1. In Table 1, $\widehat{L}^c$ is divided into *night*, *morning*, *day*, and *evening* time windows (tw_{1-4}). For each time window, the corresponding data samples of $\widehat{L}^c$ for 15-min resolution is also shown in the table, where first sample ($i = 1$) represents midnight, the last sample ($i = 96$) ends at 11:45 pm.

The mean of $\widehat{L}^c$ for each time window, which is selected as a feature, is obtained as below:

Table 1
Time window definition for feature selection.

Time window 1, tw_1 (night)	Time window 2, tw_2 (morning)	Time window 3, tw_3 (day)	Time window 4, tw_4 (evening)
10 pm –11:59 pm and 12 am - 5:59 am $tw_1 \in \{89, 90, \ldots, 96\} \cup \{1, 2, \ldots, 24\}$	6 am - 8:59 am $tw_2 \in \{25, 26, \ldots, 36\}$	9 am - 3:59 pm $tw_3 \in \{37, 38, \ldots, 64\}$	4 pm –9:59 pm $tw_4 \in \{65, 66, \ldots, 88\}$

$$F_f^c = \frac{1}{|tw_f|}\sum_{i\in tw_f}\hat{l}_i^c, f = 1,2,3,4. \quad (5)$$

An additional feature F_5^c can be chosen as the standard deviation of the whole dataset, which conveys the information of load fluctuation of each customer.

3.2. Customer selection

Based on the selected features in the previous subsection, the customers will be divided into K clusters by applying the K-medoid algorithm [15]. This algorithm is similar to the widely applied K-means algorithm. However, the advantage of K-medoid algorithm is that it returns *medoids* of the clusters which are the actual data point in the dataset whereas the *centroids* (average of all data points in a cluster) returned by K-means may not be an actual data point within the data set. In the customer selection problem, it means that the K-medoid algorithm will return an actual customer that best represents the cluster of customers, while the K-means algorithm will return a "virtual" customer that has the average features of the cluster, which is not helpful for customer selection for HRRSM placement. Moreover, unlike the K-means clustering, the performance of K-medoid is more robust against outliers in the dataset. The applied K-medoid algorithm in this article selects the medoids with "K-means++" seeding and then assigns other customers to each cluster based on their distances from the medoids [15]. There are several methods to calculate the distances in clustering algorithms, among which Euclidean distance is the most popular one. To find the Euclidean distance between features $F_f^{c_1}$,$F_f^{c_2}$ of two customers c_1, c_2, the following formulation can be used.

$$d\left(F_f^{c_1}, F_f^{c_2}\right) = \sqrt{\sum_{f=1}^{5}\left(F_f^{c_1} - F_f^{c_2}\right)^2}. \quad (6)$$

After applying the clustering algorithm, K clusters will be obtained. However, if some of the clusters have unevenly large numbers of customers, then the clusters can be sub-clustered such that customers are distributed evenly into the sub-clusters. If there are n_k customers in the k^{th} cluster ($k = 1$ to K), then the customers can be ranked based on their distances from the medoid customer as $[c_{k,0}\ c_{k,1}\dots c_{k,n_k}]$, where $c_{k,0}$ is the medoid customer, $c_{k,1}$ is the closest to the medoid customer, and so on. If n_{SC} customers need to be selected from a total of n_{TC} customers, then the number of selected customers $n_{sc,k}$ from the k^{th} cluster can be obtained as,

$$n_{sc,k} = round\left(n_k\cdot\frac{n_{SC}}{n_{TC}}\right), \quad (7)$$

where, *round*(.) operator rounds a fraction value to the closest integer. Note that the goal of customer selection is to select a small subset of customers whose load patterns can represent those of others well. As the medoid and close to medoid customers are the best representative of other customers, they are prioritized in the selection. By combining the selected customers from each cluster, the set of selected customers, C_s, can be obtained. Now, with this small set of selected customers, it becomes cost-effective to install HRRSMs and communicate/process their data in real time, which will pave the way for RTLE with high accuracy and time granularity in the *operational* stage.

4. RTLE based on sparsely selected smart meters with high reporting rates

The proposed method in Section 3 selects a small subset of customers for HRRSM installation in the *planning* stage. In this section, the *operational-stage* problem will be addressed. A RTLE method based on the real-time measurements of sparsely placed HRRSMs will be proposed. A DL model will be used to capture the relation between the measurements of HRRSMs and the feeder (or feeder section) load to be estimated. Auxiliary data preprocessing and anomaly detection/suppression methods will also be developed to ensure the robustness of the RTLE.

4.1. Data preprocessing for offline model training

This subsection will describe the preprocessing of historical datasets needed for DL model training. As the raw SM data of individual customers are highly volatile, a smoothing algorithm is first implemented to filter out the fluctuation before the data is passed to the DL model for training [37]. The goal of the smoothing is to remove spikes and oscillations (e.g., on/off cycles of AC) but track sustained trends. Using the smoothing, the correlation between the input sequence and the output increases making it easier for the DL model to map the input-output relationship. Exponential smoothing is a weighted moving average technique, where exponentially decreasing weights are assigned to the past data points [35]. Reorganizing the load data of a customer, $l_{i,j}^c$ as below,

$$l_t^c = reshape\left(l_{i,j}^c\right), \quad (8)$$

where l_t^crepresents the customer load at timestamp t, exponential smoothing can be implemented as below,

$$x_t^c = \begin{cases} l_t^c, \text{if } t = 1, \\ \alpha l_t^c + (1-\alpha)x_{t-1}^c, \text{otherwise}, \end{cases} \quad (9)$$

where α is the weighting factor of smoothing ranging from 0 to 1. When α is close to 0, less weight is given to the most recent data points and as a consequence, changes in current data will only be slightly followed, which leads to better filtering performance for spikes and oscillations, but worse (i.e., slower) tracking performance for sustained trends. On the other hand, when α is close to 1, the smoothed data quickly respond to the current changes, leading to better (faster) tracking performance of sustained trends, but worse filtering performance for spikes and oscillations. Therefore, the value of α should be carefully set to achieve a satisfactory tradeoff. Along with the selected customers' data, the features also include the sine and cosine of the timestamp, t, to capture the daily and weekly periodicity in the data. The timestamps representing minutes are converted to *sine* and *cosine* terms, as below:

$$t_{dx} = cos\left(t_{stamp}\cdot 2\pi/(24\cdot 60)\right), \quad (10)$$

$$t_{dy} = sin\left(t_{stamp}\cdot 2\pi/(24\cdot 60)\right), \quad (11)$$

$$t_{wx} = cos\left(t_{stamp}\cdot 2\pi/(7\cdot 24\cdot 60)\right), \quad (12)$$

$$t_{wy} = sin\left(t_{stamp}\cdot 2\pi/(7\cdot 24\cdot 60)\right). \quad (13)$$

In addition, as the weather information (i.e., solar irradiance $\mathscr{S}$, temperature θ) has a significant impact on power consumption and renewable generation, they are adopted as additional features. Therefore, at each time step, the number of spatial features would be the combination of the selected customers' data, the time periodicity, and the weather information, which can be written as below:

$$x_t = \left\{x_t^c, T_t, W_t\right\} \in \mathbb{R}^s, \quad (14)$$

where $T_t = \{t_{dx}, t_{dy}, t_{wx}, t_{wy}\}$, $W_t = \{\mathscr{S}_t, \theta_t\}$, and s is the number of features at time step t. As the input features incorporate different types of data, the data is normalized in the range of $[-1,1]$ in order to increase the robustness of the DL model. The complete input vector of the DL model consists of past t_d time steps as below,

$$X_t = \left\{x_{t-t_d+1}, \dots, x_{t-1}, x_t\right\} \in \mathbb{R}^{s\times t_d}. \quad (15)$$

The target of the DL model is the feeder or feeder section load of our interest. The feeder or feeder section load can be obtained by

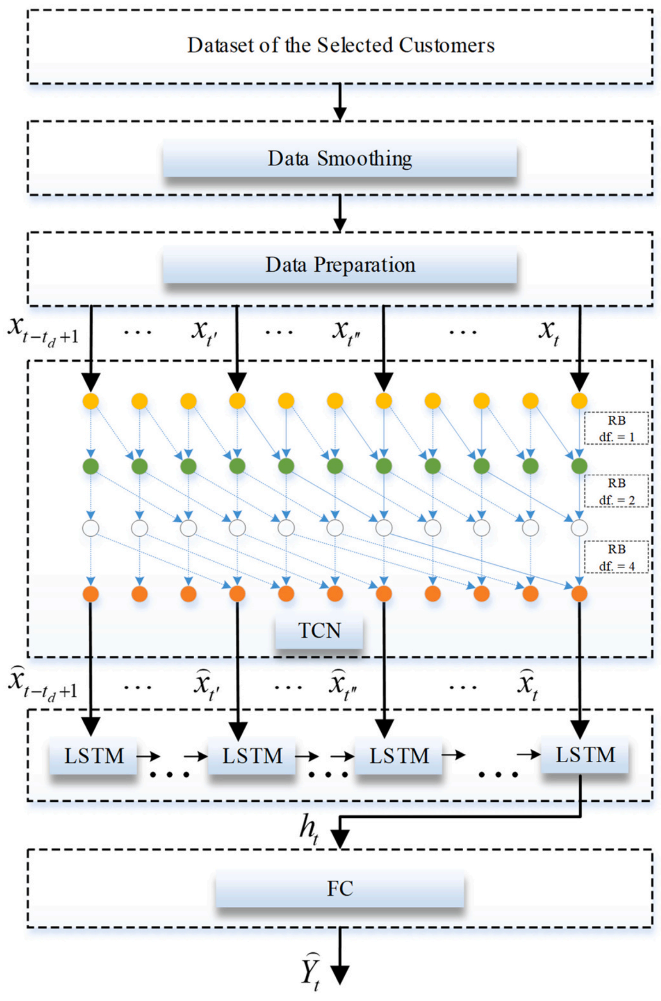
Dataset of the Selected Customers
Data Smoothing
Data Preparation
x_{t-t_d+1} ... $x_{t'}$ ... $x_{t''}$ ... x_t
RB df. = 1
RB df. = 2
RB df. = 4
TCN
$\hat{x}_{t-t_d+1}$ ... $\hat{x}_{t'}$ ... $\hat{x}_{t''}$ ... $\hat{x}_t$
LSTM
LSTM
LSTM
LSTM
h_t
FC
$\hat{Y}_t$

Fig. 2. Proposed TCN-LSTM model.

aggregating all customers' loads at each time step t, which is,

$$Y_t = \frac{1}{s_f}\sum\nolimits_{c=1}^{n_{TC}} l_t^c \in \mathbb{R}, \tag{16}$$

where a scaling factor s_f is used when the accumulated load is very large, which helps the DL model to converge faster. Finally, a DL model, $f(.)$, is used to learn the nonlinear mapping of X_t to Y_t, as below,

$$\widehat{Y}_t = f(X_t). \tag{17}$$

The DL model will be described in the next subsection.

4.2. *Offline model training*

The DL model for RTLE integrates two popular DL models, namely TCN and LSTM, as shown in Fig. 2. The TCN takes advantage of a special 1-D convolutional neural network (CNN) structure to learn low- and high-frequency dependencies of the input sequence, whereas, the LSTM extracts long-term interdependencies of the sequence. In the combined TCN-LSTM model, the TCN is used first for extracting short-term local features from the input sequence, and then the LSTM is trained with the extracted features, which improves the estimation accuracy. In addition, the combined TCN-LSTM model is efficient at capturing the nonlinear input-output relationship and providing a smoothed prediction.

The input sample X_t as obtained in Section 4.1 is forwarded to the TCN. Residual block (RB), the unit of TCN, applies the following transformations on X_t,

$$o = \text{Activation}(F(X_t) + X_t), \tag{18}$$

where F(.) is the residual mapping, which consists of two dilated causal 1-D convolution followed by normalization, rectified linear unit (ReLU), and dropout function; X_t is the identity mapping with an optional 1×1 convolution operation for matching the dimensionality of the inputs in the above summation. Unlike conventional convolution, causal convolution is used in TCN to ensure causality in time-series analysis, and dilated convolution is adopted to increase the receptive field. As an example, in Fig. 2, the operation of TCN is demonstrated with three stacked RBs having a filter size of 2 and a dilation factor (df) of 1, 2, and 4, respectively. The output of the TCN block is as below:

$$\hat{X}_t = \text{F}_{TCN}(X_t) \in \mathbb{R}^{n_f \times t_d}, \tag{19}$$

where $F_{TCN}(.)$ is the overall TCN transformation function on the input and n_f is the number of filters used in the convolution operation in RB.

The sequence output of the TCN layer is forwarded to an LSTM layer with several hidden units. LSTM consists of three gates, namely a forget gate, an input gate, and an output gate to avoid gradient disappearance and explosion problems existing in regular recurrent neural networks (RNNs). The LSTM unit takes its hidden output h_{t-1} from the previous timestep and the input $\hat{x}_t$ at the current timestep to calculate a new output (h_t) recursively and returns the final output as shown below:

$$h_t = F_{LSTM}\left(\hat{x}_t, h_{t-1}\right) \in \mathbb{R}^{n_h} \tag{20}$$

where, $F_{LSTM}(.)$ is the transformation function for an LSTM unit, and n_h is the number of hidden units.

The output of LSTM is forwarded to a fully-connected (FC) layer to generate the expected number of outputs from the model. During the training period, the parameters of all the layers are learned to minimize the mean absolute error (MAE) of the estimated output ($\widehat{Y}_t$) and the ground-truth (Y_t), as below,

$$MAE = \frac{1}{n}\sum_{t=1}^{n} |\widehat{Y}_t - Y_t|. \tag{21}$$

The use of MAE in the loss function can enhance the robustness of the training process against anomalies in historical SM data.

As numerous data samples are required for DL model training to capture the trends in the time series, the training dataset should contain data from several weeks for training the proposed TCN-LSTM model. Additionally, several hyperparameters of both the TCN and LSTM models need to be tuned to minimize the MAE.

4.3. *Data filtering and anomaly suppression for online load estimation*

With a trained DL model, RTLE can be performed for the feeder (or feeder section) of interest using the real-time measurements from the HRRSMs installed at the small subset of customers. However, the raw HRRSM data received in real time are subject to anomalies from sensor malfunction, communication delay and failure, or even false data injection attacks [32–34]. We propose an anomaly detection and correction method to enhance the robustness of RTLE, as will be described in this subsection.

To be consistent with the training dataset, the real-time raw data (test dataset) from HRRSMs of each selected customer $l_{t,rt}^c$ should be smoothed using the same exponential smoothing algorithm as described in (9):

$$x_{t,rt}^c = \alpha l_{t,rt}^c + (1-\alpha)x_{t-1,rt}^c, c \in C_s. \tag{22}$$

However, if the real-time data of some selected customers vary inconsistently with respect to most others (anomalies), they cannot represent the aggregated behaviors of the customers in the feeder (or

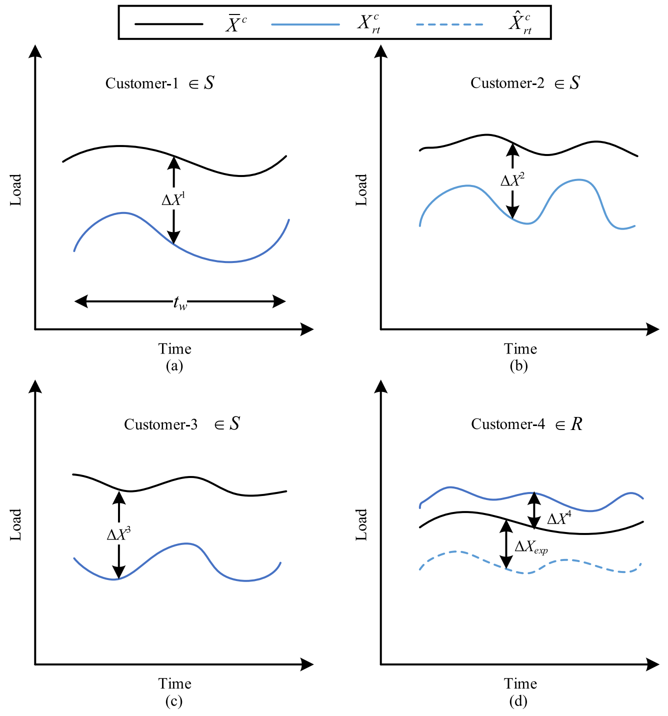

Fig. 3. The illustration of data robustification process, (a)-(c): three customers with consistent load curve, (d): one customer with inconsistent load curve.

feeder section) and may lead to large RTLE errors. In order to make the method robust, anomalies should be detected and suppressed. For illustration purpose, let us consider the dataset of four selected customers with HRRSMs in Fig. 3. The four subfigures show the load profile of these four individual customers. The solid blue curve X^c_{rt} shows the smoothed *real-time* data of the customer for a time window t_w and the black curve $\overline{X}^c$ shows the average *historical* data for the same time window. The two datasets for the t_w window can be expressed as below:

$$\left.\begin{aligned}\overline{X}^c &= \left[\overline{x}^c_{t-t_w+1}\cdots\overline{x}^c_{t-2}\ \overline{x}^c_{t-1}\ \overline{x}^c_{t}\right]\\ X^c_{rt} &= \left[x^c_{t-t_w+1,rt}\cdots x^c_{t-2,rt}\ x^c_{t-1,rt}\ x^c_{t,rt}\right]\end{aligned}\right\}, c\in C_s. \tag{23}$$

The difference between these two datasets is a dataset named load-deviation ΔX^c. From Fig. 3(a)-(d), it can be seen that the load-deviations of first three customers are positive ($\overline{X}^c > X^c_{rt}$) and with similar magnitudes, whereas, that of the fourth customer is negative. Here, the real-time dataset of the fourth customer is inconsistent with the dataset of the other three customers. In other words, for the given day, the loads of first three customers all have positive shifts from their respective historical profiles, while the load of the fourth customer has a negative shift from its historical profile. If the other customers in the feeder follow a similar trend as the first three customers, then the fourth customer is not a good representation of the current loading condition of the feeder (or feeder section), and will bias the RTLE result if used directly. In order to make it consistent with most other customers, the corrected dataset $\hat{X}^c_{rt}$can be obtained by using an expected load-deviation ΔX_{exp} for the fourth customer, as shown in Fig. 3(d). The procedure for obtaining ΔX_{exp} and $\hat{X}^c_{rt}$ is described below.

At any time instant, let us divide the set of selected customers with HRRSM, C_s, into two subsets: S is the set of customers having consistent deviation from average historical load and R is the set of customers having inconsistent deviation with respect to the customers of set S. The Euclidian distance metric similar to (6) among the load deviation curves of the selected customers can be considered to obtain S and R. The load-deviation ΔX^c of c^{th} customer can be expressed as below:

$$\Delta X^c = \overline{X}^c - X^c_{rt}. \tag{24}$$

The expected load deviation $\Delta X^c_{\exp}$ of all the customers can be obtained by taking the average of load-deviations of customers in set S, that is:

$$\Delta X_{exp} = \frac{1}{|S|}\sum_{c\in S}\Delta X^c. \tag{25}$$

For the aforementioned reasons, the load-deviations of customers in set R should be ΔX_{exp} instead of ΔX^c, $c \in R$. Therefore, ΔX_{exp} for a customer in set R can be expressed as below by substituting $\overline{X}^c$ from (24):

$$\Delta X_{exp} = \overline{X}^c - \hat{X}^c_{rt} = \Delta X^c + X^c_{rt} - \hat{X}^c_{rt}, c\in R. \tag{26}$$

The corrected dataset of c^{th} customer in R can be obtained from (26) as follows:

$$\hat{X}^c_{rt} = \Delta X^c + X^c_{rt} - \Delta X_{exp}, c\in R. \tag{27}$$

Using the above procedure, anomalies in the real-time HRRSM measurements can be suppressed in the corrected dataset which will make the trained DL model more robust for RTLE.

4.4. Online load estimation

The corrected dataset obtained in Section 4.3 is used as test dataset in the trained DL model. If the test samples are denoted as$\widetilde{X}_t$, the feeder (or feeder section) load of interest can be estimated as follows:

$$\hat{Y}_t = f(\widetilde{X}_t), \tag{28}$$

where, $f(.)$ is the trained TCN-LSTM DL model described in Section 4.2.

5. Case study

The effectiveness of the proposed RTLE method is verified by using a special dataset provided by the Pecan Street Inc. [38], which contains SM data totaling 645 customers with 1-min resolution. We use the load consumption of 84 consecutive days (12 weeks) from the month of June, July, and August in the summer season. As the Pecan Street dataset does not contain weather information and most of the customers are located around Austin, Texas, in the provided dataset, we have used the weather information from a station near Austin [39]. The resolution of the obtained weather information is 5 min, and we use linear interpolation to increase the resolution to 1-min as needed. It is assumed that all the customers come from the same section of a feeder, and the objective is to estimate the aggregated load (i.e., the load of the feeder section, referred to as "feeder load" below) based on real-time high-resolution data from a sparse subset of customers. The dataset is artificially divided into historical and real-time datasets for the verification of the proposed method. The data from the first 70 days (10 weeks) are considered historical data, which is further divided into the training and validation datasets, and the data from the last 14 days (2 weeks) are considered the real-time testing dataset. Although high-resolution data (1-min) is actually available for all customers, it is assumed that only low-reporting-rate SM data (15-min) is available initially for all of the customers, and both the *planning* stage (HRRSM placement) and the *operational* stage (RTLE based on sparse HRRSMs) of the proposed method will be verified.

1) *Planning* stage. Customer selection for HRRSM placement will be carried out. As it is assumed that HRRSMs have not been installed at this point, the proposed clustering method will only use features extracted from SM data of all customers with 15-min resolution.

2) *Operational* stage, *offline model training* substage. It is assumed that HRRSMs have been installed at a sparse subset of customers. The SM data of all customers with 15-min resolution will be aggregated to obtain the feeder load, which serves as the target for the training of the DL

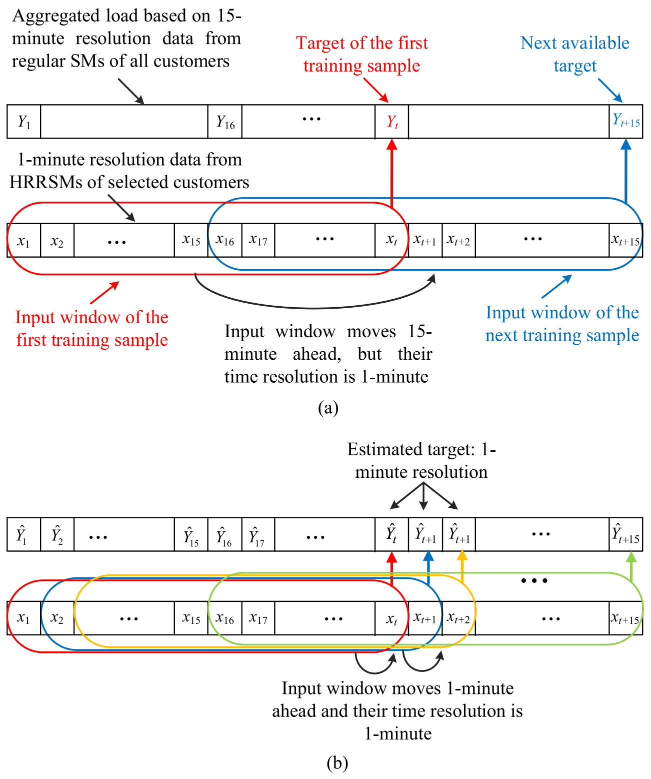
Aggregated load based on 15-minute resolution data from regular SMs of all customers
Target of the first training sample
Next available target
Y_1 Y_{16} Y_t Y_{t+15}
1-minute resolution data from HRRSMs of selected customers
x_1 x_2 x_{15} x_{16} x_{17} x_t x_{t+1} x_{t+2} x_{t+15}
Input window of the first training sample
Input window moves 15-minute ahead, but their time resolution is 1-minute
Input window of the next training sample
(a)
Estimated target: 1-minute resolution
$\hat{Y}_1$ $\hat{Y}_2$ $\hat{Y}_{15}$ $\hat{Y}_{16}$ $\hat{Y}_{17}$ $\hat{Y}_t$ $\hat{Y}_{t+1}$ $\hat{Y}_{t+1}$ $\hat{Y}_{t+15}$
Input window moves 1-minute ahead and their time resolution is 1-minute
(b)

Fig. 4. The training and test samples used in the DL model.
(a) Two consecutive training samples of the DL model are shown. The resolution of the input data x is 1-min, but two consecutive training samples are 15-min apart to be consistent with the resolution of the target samples Y. (b) Several consecutive test samples of the DL model are shown. The resolution of the input data x is 1-min, and two consecutive training samples are 1-min apart as the target Y is estimated every minute in RLTE.

model. As the HRRSM data of the selected customers is available at this stage, 1-min resolution data of the selected customers is used to generate the input vector for DL model training. However, the training samples resolution is 15-min to be consistent with that of the target samples. The input and target data of the training samples are shown in Fig. 4(a), where the samples are 15-min apart, but the input vector has 1-min resolution.

3) *Operational* stage, *online load estimation* substage. The HRRSM data of the selected customers and the weather information with 1-min resolution will be fed into the trained DL model to estimate the feeder section load. The test samples are shown in Fig. 4(b), where each time the sliding window moves 1-min ahead and the target load is estimated at every minute. In order to evaluate the accuracy of the estimated feeder load, the *ground-truth* feeder load with 1-min resolution should be known. For this purpose, the SM data of all customers with 1-min resolution will be aggregated to obtain the *ground-truth* feeder load for the comparison with the estimated feeder load.

Note that SM data of all customers with 1-min resolution is typically unavailable in practice (and also assumed to be unavailable by the proposed method), but thanks to the experimental high-resolution Pecan Street dataset, this data is available to us such that the 1-min-resolution *ground-truth* feeder section load can be obtained for the evaluation of the accuracy of the proposed method.

5.1. Baseline methods for comparison

As there has been little existing research on RTLE based on real-time SM data, the performance of our method is compared with several DL-based Day-Ahead Load Forecasting (DALF) methods, which is very common in literature and in practice [15–18]. In order to verify the benefit of incorporating sparse but real-time HRRSM data, the first obvious choice of baseline method is to use the same DL model (i.e., TCN-LSTM) as in our RTLE method, but to assume that the sparse real-time HRRSM data is unavailable and to perform standard DALF based on historical SM data only. To further evaluate our model, we have also used TCN, LSTM, Bidirectional LSTM (BiLSTM), and multilayer perceptron (MLP) for DALF as additional baseline methods [23] [16,40].

In the baseline DALF methods, the historical feeder load is used to perform day-ahead forecasting of the feeder load. The historical feeder load is obtained by summing the data of all customers at every 15-min interval, as it is assumed that the available data resolution of all customers is 15-min. In the training process of the baseline methods, a sliding time window is used to generate the input vector of the training samples containing h hours of data for d consecutive days, and the target is the data of the subsequent day ($d + 1$). In other words, the day-ahead feeder load is forecasted based on the data from the previous d days. Similar to the proposed RTLE method, the weather information is also taken as additional features to improve the accuracy of the baseline DALF methods. After tuning h and d, it is found that the input vector with 6 h of data from the previous 7 days yields the highest accuracy for the baseline methods. It should be noted that the baseline method can only estimate feeder load in the 15-min resolution, as it requires the dataset of all customers which is of the 15-min resolution. In comparison, an immediately noticeable advantage of the proposed method is that it increases the time resolution of estimated feeder load to the same resolution of the HRRSM data (in this simulation case, to 1-min), although only a small number of HRRSMs need to be installed along the feeder.

For measuring the performance of the methods, several commonly used metrics will be adopted, such as Mean Absolute Error (MAE) as given in (21), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). If the *ground-truth* feeder load is y_i and the estimated one is $\hat{y}_i$, then for n samples, the RMSE and MAPE metrics are defined below:

$$RMSE = \left(\frac{1}{n}\sum_{i=1}^{n}(|y_i - \hat{y}_i|)^2\right)^{1/2}, \tag{29}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{\left|\frac{1}{n}\sum_{i=1}^{n}y_i\right|} \times 100\%. \tag{30}$$

5.2. Customer selection for HRRSM installation

Before applying the K-medoid clustering algorithm, five features are computed from the historical SM data of each customer with 15-min resolution as described in Section 3.1. In feature selection, the value of α is set to 0.05 and the sample delay i_d is set to 12 (equivalent to 3-h time window for 15-min resolution) for the two-way smoothing as described by (2). These values are chosen because they can smooth the current data sample without losing important information. For instance, the actual load data and smoothed load data of the 32th customer (out of 645 customers) for a specific day are shown in Fig. 5. The fluctuation of the actual load data is significantly suppressed, e.g., the spike around the 600th minute; at the same time, the smoothed data still follows sustained trends promptly, e.g., the step increase after 1000 min.

After computing the features of all the customers, the appropriate number of clusters is chosen based on the Sum Square Error (SSE), the silhouette coefficient [36], and the need of the application. For our case, increasing the number of selected customers will increase the HRRSM installation cost. Based on the trend of SSE shown in Fig. 6(a), it is better to select a large number of clusters as the intra-cluster distance decreases with it. However, some clusters will have only a few customers or even single customer when the number of clusters is too large. Therefore, we consider the range of 15–25 clusters, and choose the one with the highest silhouette coefficient. As seen in Fig. 6(b), the silhouette

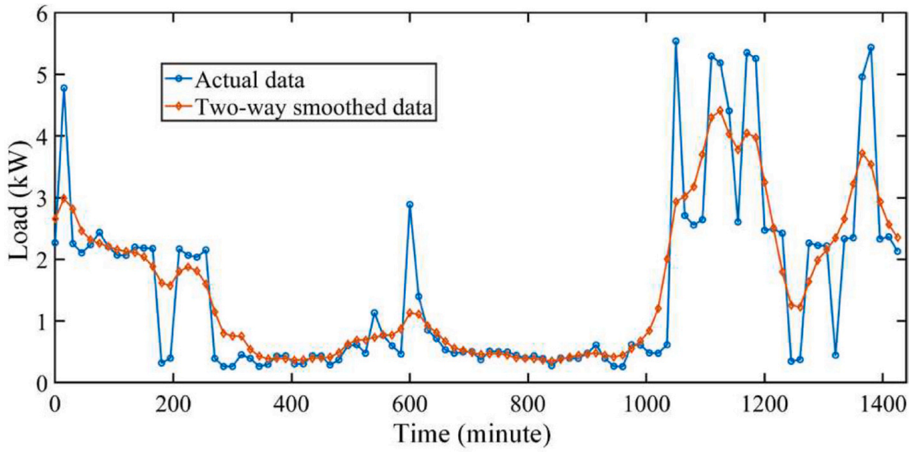

Fig. 5. Raw and two-way smoothed SM data of 32th customer for a random day with 15-min data resolution ($n = 96$).

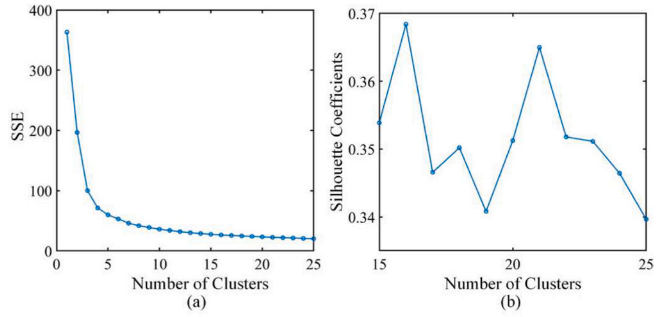

Fig. 6. Cluster selection indexes. (a) SSE vs. no. of clusters, (b) Silhouette coefficients vs. no. of clusters.

coefficient peaks when K is 16. At this time, the number of customers in each cluster is 65, 100, 49, 16, 2, 98, 32, 2, 21, 53, 32, 70, 6, 32, 5, and 62, respectively. It is noticed that some clusters are very large compared with the average size (i.e., $n_{TC}/K = 645/16 \approx 40$). Therefore, the clusters having more than 40 customers are sub-clustered such that each new cluster has no more than 40 customers. After sub-clustering, the number of clusters become 25, and the number of customers in each cluster becomes 30, 35, 38, 35, 27, 40, 9, 16, 2, 36, 35, 27, 32, 2, 21, 39, 14, 32, 20, 50, 6, 32, 5, 40, and 22. In order to illustrate the performance of clustering, the average loads of individual customers of three randomly selected clusters are shown in Fig. 7. From the figure, it can be observed that each cluster has customers with similar load profiles. For instance, customers in the 18*th* cluster have a significantly negative net load at midday probably due to high solar generation. Now, the selected customers (the medoid customers and the customers close to the medoids) from each cluster can be chosen based on (7) given a total number of customers to be selected (based on HRRSM installation budget in practice). In this case, a total of 60 customers are selected for HRRSM installation, which constitute less than 10% of all 645 customers. Next, load data series of the 60 selected customers are used in the TCN-LSTM training.

5.3. Performance evaluation of RTLE with normal dataset

Before training the DL model, the historical SM data of the selected customers with 15-min resolution is preprocessed based on the methods described in Section 4.1. The exponential smoothing method is applied to the raw SM data as in (9), with α being set to 0.05. The reason is explained with a selected (146th) customer's daily load profile. The actual load and exponential smoothed loads with three different values of α are shown in Fig. 8. Obviously, when α is 0.5, there is no significant effect of oscillation filtering, whereas the smoothing effect is strong but

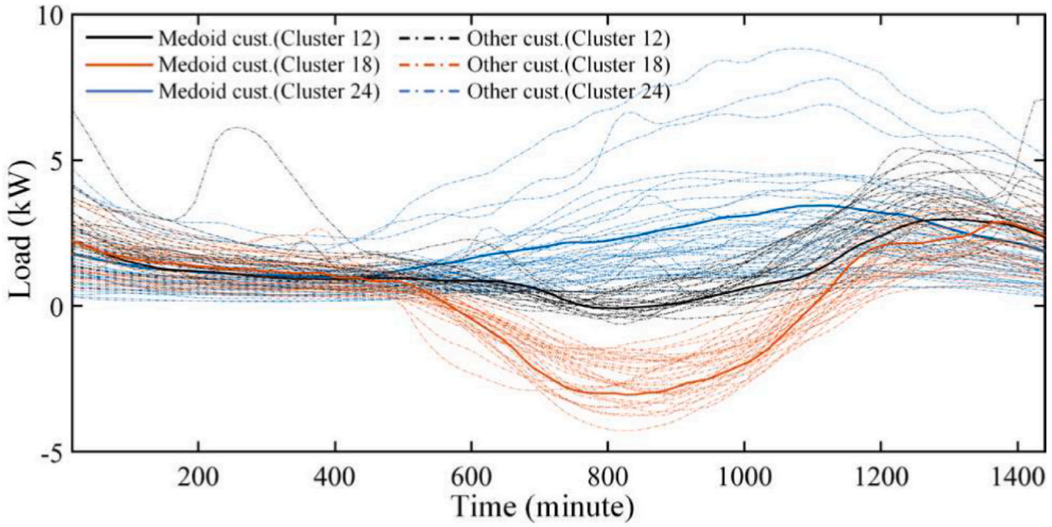

Fig. 7. Illustration of average load of individual customers for three clusters; medoid and other customers of each cluster are shown.

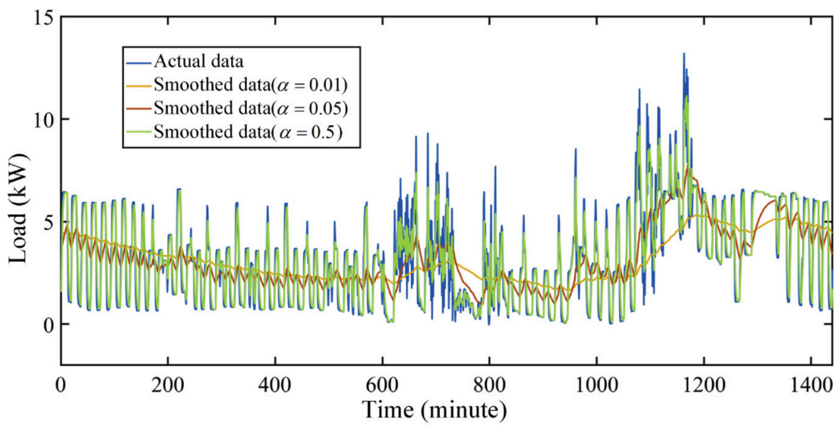

Fig. 8. Exponential smoothing of SM data of a selected customer (146th out of 645) with different values of α.

Table 2
Best tuned hyperparameters of the DL models.

Hyperparameters	*Proposed RTLE*	*Benchmark Models (Best Tuned Hyper-parameters using regular SM data only)*				
	TCN-LSTM	TCN-LSTM	TCN	LSTM	BiLSTM	MLP
No. of RB in TCN	4	3	3	–	–	–
Filter Size of 1D-CNN in RB	2	2	4	–	–	–
No. of Filters of 1D-CNN in RB	128	32	32	–	–	–
LSTM or BiLSTM	LSTM	BiLSTM	–	–	–	–
No. of LSTM Layers	1	1	–	2	4	–
No. of hidden units in LSTM layers	128	64	–	64	16	–
Flatten layer	–	–	–	–	–	Yes
No. of Dense layer	1	1	1	1	1	5
No. of Neuron in Dense layers	1	1	1	1	1	32,32, 16,16,1
Learning rate	0.001	0.001	0.001	0.001	0.001	0.005
Batch size	128	128	128	128	128	64
Optimization algorithm	Adam	Adam	RMS-prop	Adam	Adam	Adamax

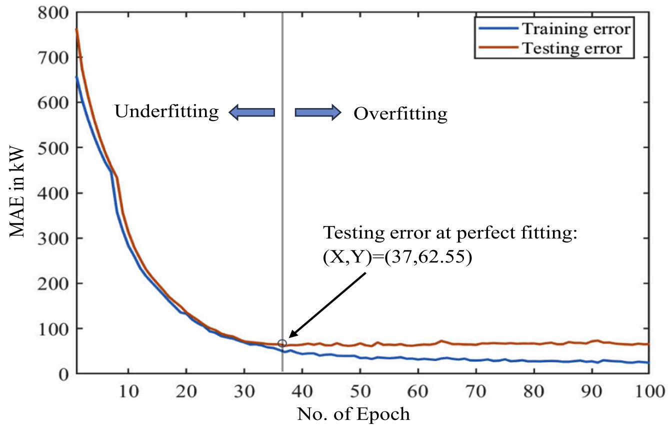

Fig. 9. Training and testing error vs. number of epochs.

sustained trends are followed with significant delay (e.g., around the 1300-th minute) when α is 0.01. When α is 0.05, a good tradeoff is achieved: the oscillations are finely smoothed (e.g., up to the 600th minutes), while sustained trends of the load are properly followed (e.g., around the 1300th minute). For the DL model training, the time window of input vector is set to 2 h (2×60 data samples) in (15).

After data preprocessing as described in Section 4.1, the TCN-LSTM model described in Section 4.2 is trained using the TensorFlow framework in Python. For tuning the hyperparameters of the TCN-LSTM model, the training dataset is divided into training and validation datasets (0.8:0.2) and the best hyperparameters listed in Table 2 are chosen by minimizing validation error using Bayesian optimization from a plausible range of search space [41]. The test dataset obtained as in Section 4.3 is fed into the trained TCN-LSTM model to estimate the feeder load in real time. To reduce overfitting in the DL models, we adopt a few well-known techniques such as layer normalization, dropout, kernel regularization, and early stopping. The training and testing errors versus the number of epochs are shown in Fig. 9. The figure indicates an early stopping point where both errors decrease to around 62 kW, signifying that the model is not overfitting the training dataset.

Table 3
Performance comparison between the proposed method and the baseline methods.

Error Metrics	*Proposed RTLE*	*Benchmark Models (Input vector uses regular SM data only)*				
	TCN-LSTM	TCN-LSTM	TCN	LSTM	BiLSTM	MLP
MAE (kW)	62.55	94.91	96.66	98.07	95.85	101.99
RMSE (kW)	83.06	121.36	124.81	124.68	121.94	130.64
MAPE (%)	7.26	10.95	11.15	11.31	11.05	11.78

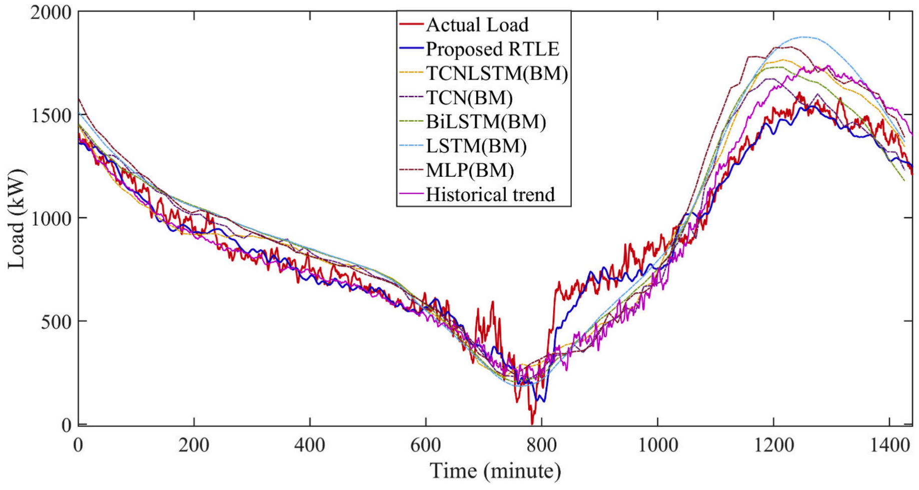

Fig. 10. Comparison between proposed method and the baseline method (BM) for Sunday. The historical trend (average of the load of Sunday) for Sunday is also drawn.

Table 4
Description of anomalies corrupting the test data.

Causes of anomalies	*Effects in SM data to mimic the anomalies*	*Duration of anomalies*
Delay in data reporting	Time-lagging the normal data.	30-min
Unreported data	Replacing normal data by last valid reported data.	2-h
Step/gradual changes in data	Increasing magnitudes of normal data by 4-times	2-h

The performance of the proposed method is compared with the baseline methods. The hyperparameters of the baseline methods are also optimally tuned using the Bayesian optimization and listed in Table 2. The overall error metrics for the entire test dataset are shown in Table 3. Evidently, the proposed method outperforms the DL-based DALF methods. Specifically, the MAE, RMSE, and MAPE metrics are improved by 34.09%, 31.56%, and 33.69%, respectively. For better illustration, the actual feeder load and the estimated ones by the methods in a test day are shown in Fig. 10 along with the historical trend. Noticeably, the feeder load of this day is inconsistent with the historical trend, especially after the midday. Note that in distribution systems with high renewable penetration and active customer participation, deviation of the average historical trend is common. From Fig. 10, it is observed that both the proposed and the baseline methods can estimate the feeder load reasonably well until midday. After midday, however, when the feeder load deviates from the historical trend, the baseline method continues to follow the historical trend and fails to predict the feeder load accurately, whereas the proposed method follows the actual feeder load effectively. The reason is that the baseline method exclusively relies on historical SM data and cannot adapt to the changes of operating conditions in real time, while the proposed method continuously receives update from the small group of HRRSMs, which effectively captures the actual trend of feeder load of the current day. It can also be noticed in Fig. 10 that the granularity for the baseline methods is 15-min, as it relies on historical SM data of all the 645 customers with 15-min resolution. In comparison, the proposed method tracks the feeder load with 1-min resolution, as it uses the 1-min HRRSM data of the 60 selected customers. Through this case study, it is demonstrated that using measurement data from sparsely but strategically installed HRRSMs, the load of a feeder or feeder section can be inferred with greatly improved accuracy and granularity compared with widely used DALF methods based on historical data from regular SMs.

5.4. Performance evaluation of RTLE under anomalies

In order to verify the robustness of the proposed RTLE method, this section will present results of simulation cases where anomalies are intentionally introduced into the real-time HRRSM dataset. Possible types of anomalies include consecutive zeros (unreported data), large spikes, and large step/ramp changes [32]. Anomalies may occur in the data from an individual SM due to sensor malfunction, or occur in the data from a group of SMs due to delay/failure of shared communication path or false data injection attacks. In the simulation, three types of anomalies, as shown in Table 4, are injected into the real-time (test) dataset of the selected customers at both individual-SM level and group-SM level. At individual-SM level, all the three types of anomalies are injected into 30 random customers out of the 60 selected customers. At group-SM level, several groups of customers are randomly picked without replacement from the selected customers and their data are corrupted by the three types of anomalies simultaneously. The RTLE results with and without the anomaly suppression method described in Section 4.3 are compared. The error metrics for the two cases are shown in Table 5. The estimation error with anomaly suppression is close to that of the anomaly-free case, whereas the estimation error without anomaly suppression is significantly higher. In order to further illustrate the performance of the proposed method, the actual load of a test day

Table 5
Performance of proposed method with and without anomaly suppression described in Section 4.3 in the presence of anomalies.

Test Condition		*MAE*	*RMSE*	*MAPE*
In the absence of anomalies		62.55 kW	83.06 kW	7.26%
In the presence of anomalies	With anomaly suppression	71.28 kW	95.24 kW	8.28%
	Without anomaly suppression	100.65 kW	134.84 kW	11.68%

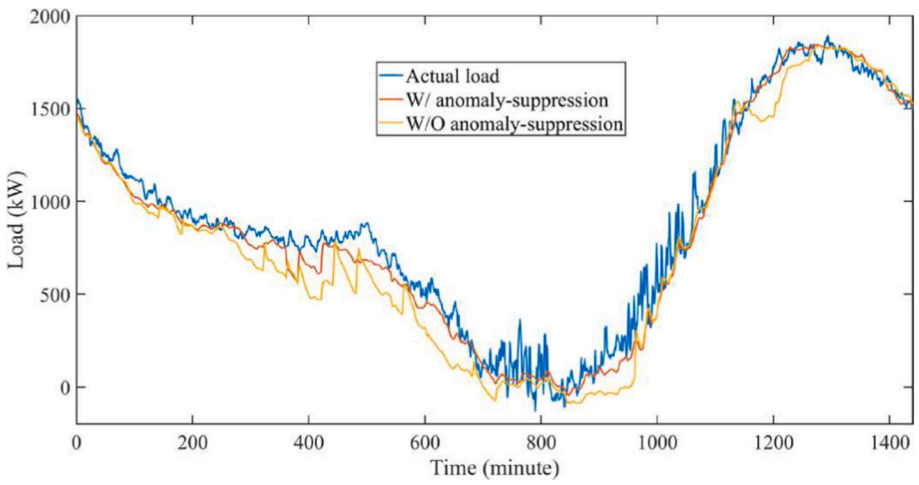

Fig. 11. Performance evaluation of the proposed method with and without anomaly-suppression, as described in Section 4.3, when the SM data incorporates anomalies.

Table 6
Comparison of proposed customer selection with random customer selection method.

Customer Selection Method	*MAE*	*RMSE*	*MAPE*
With proposed selection	62.55 kW	83.06 kW	7.26%
With random selection	77.05 kW	100.22 kW	8.91%

and the estimated one with and without the anomaly suppression method are shown Fig. 11. It is observed that at several minutes (e.g., the 400th–800th, and 1200th), when the real-time HRRSM data of several selected customers carry simultaneous anomalies, the proposed method helps suppress the effect of the anomalies on the estimated feeder load. Therefore, it can be concluded that the proposed method is robust against anomalies in real-time HRRSM data streams.

5.5. *Performance evaluation of customer selection strategy*

This subsection aims to verify the effectiveness of the customer selection strategy for HRRSM placement (i.e., *planning* stage) as described in Section 3. Although there are existing SM placement methods, they are not for HRRSM placement [42] [43]. Moreover, the objective of the existing methods [42] [43] is to make the distribution network observable for state estimation, which are based on the network topology and parameter information, not temporal load patterns of customers. For the RTLE application, the network model does not play a role, so the effect of existing SM placement methods is not different than random placement. Therefore, instead of using the proposed method to select the 60 customers, we randomly select 60 customers out of the 645 customers for HRRSM installation as the baseline for verifying the effectiveness of the proposed placement method. The MAE, RMSE, and MAPE are shown in Table 6 for both methods. Here, the error metrics for the random selection strategy are obtained by averaging the results of 20 different sets of 60 customers randomly selected for HRRSM placement. It is obvious that the customer selection method described in Section 3 achieves much better RTLE performance than the random selection. The performance degradation of RTLE under random customer selection is understandable as the randomly selected customers cannot effectively represent the aggregated behavior of all customers, thus their real-time data does not bring about significant benefit for RTLE. The customers selected using the proposed clustering method, by contrast, capture the behavior of all customers in a grouping manner, thus enabling highly accurate RTLE even though they only constitute a small portion of all customers along the feeder.

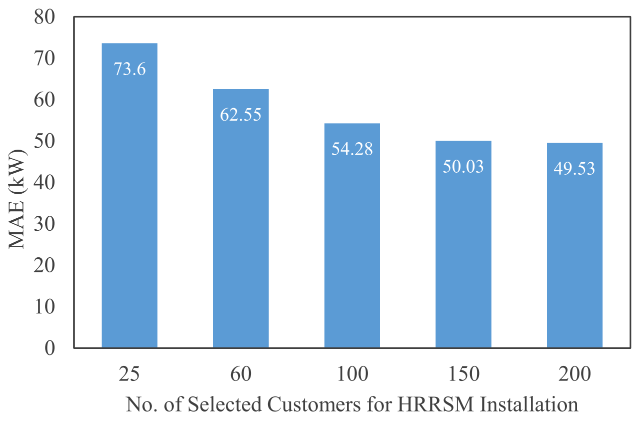

Fig. 12. MAE for the test days using the proposed method, when number of selected customers vary from 25 to 200.

5.6. *Performance evaluation under different numbers of selected customers*

Finally, we will study the impact of the number of selected customers on the performance of the proposed RTLE method. In the simulation cases of the previous subsections, this number has been fixed as 60, taking up 9.3% of the 645 customers. In Fig. 12, the MAE of RTLE is shown for different numbers of selected customers for HRRSM installation. Clearly, the performance of the proposed method improves with the increase of the selected customers. This is consistent with our expectation: the more HRRSMs are installed, the better representation of the loading condition of the feeder, and the more accurate RTLE becomes. However, the choice of the selected customer number is also dependent on the budgetary constraint for HRRSM installation and data communication /processing. Thus, for the economic objective, selecting few customers is more desirable. In practice, this tradeoff should be determined by the demand of RTLE accuracy and the available resources of the utility company. However, it is worth noting in Fig. 12 that even when the number of selected customers is 25 (3.88% of the 645 customers), the MAE of the proposed method is less than 75 kW, which is also already significantly better than the baseline DALF method with an average MAE of around 95 kW, as shown in Table 3. Furthermore, it is also observed that the incremental improvement of accuracy decreases as the number of selected customers increases. This implies that installation of a relatively small number of HRRSMs is possibly a good strategy that enables much more effective load tracking than state-of-the-art DALF methods at a fairly modest investment cost.

6. Conclusion

This paper proposes an innovative robust learning-based RTLE method using sparsely but strategically selected SMs with high reporting rates. In view of the fact that the collection of real-time high-resolution SM data from a sheer number of customers is not practical in the foreseeable future, the proposed method aims to estimate the load of a feeder or feeder section of interest using real-time measurements from very few customers. A clustering-based strategy for planning the installation of HRRSMs to achieve satisfactory RTLE results is also presented. Simulation results based on real-world SM data demonstrate the advantages of the proposed method in terms of the accuracy, robustness, and time granularity. It significantly outperforms the existing DALF methods, which only use historical data from regular SMs and cannot adapt to the deviation of the real-time loading condition from historical trends. A significant improvement in accuracy is observed even when the real-time data of very few customers are collected (as low as 3.88% or all customers), implying that the proposed method is economically competitive. It is also verified that the proposed method can effectively suppress the impact of various types of anomalies in the real-time HRRSM data streams, allowing consistently satisfactory performance under complicated measurement and communication environment in practice.

CRediT authorship contribution statement

Md. Zahidul Islam: Conceptualization, Data curation, Methodology, Validation, Visualization, Writing – original draft, Writing – review & editing. **Yuzhang Lin:** Conceptualization, Data curation, Methodology, Project administration, Resources, Supervision, Validation, Visualization, Writing – review & editing. **Vinod M. Vokkarane:** Methodology, Writing – review & editing, Supervision. **Nanpeng Yu:** Methodology, 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

The authors do not have permission to share data.

References

[1] Fang Z, Lin Y, Song S, Li C, Lin X, Chen Y. State estimation for situational awareness of active distribution system with photovoltaic power plants. IEEE Trans Smart Grid Jan 2021;12(1):239–50.

[2] Villanueva-Rosario JA, Santos-García F, Aybar-Mejía ME, Mendoza-Araya P, Molina-García A. Coordinated ancillary services, market participation and communication of multi-microgrids: a review. Appl Energy 2022;308:118332.

[3] Lotfi M, Almeida T, Javadi MS, Osório GJ, Monteiro C, Catalão JPS. Coordinating energy management systems in smart cities with electric vehicles. Appl Energy 2022;307:118241.

[4] Fang X, Misra S, Xue G, Yang D. Smart grid — the new and improved power grid: a survey. IEEE Commun Surveys Tutorials 2012;14(4):944–80.

[5] Cho Y, Lee E, Baek K, Kim J. Stochastic optimization-based hosting capacity estimation with volatile net load deviation in distribution grids. Appl Energy 2023;341:121075.

[6] Alzate E, Bueno-López M, Xie J, Strunz K. Distribution system state estimation to support coordinated voltage-control strategies by using smart meters. IEEE Trans Power Syst Nov 2019;34(6):5198–207.

[7] Jindal A, Singh M, Kumar N. Consumption-aware data analytical demand response scheme for peak load reduction in smart grid. IEEE Trans Indus Electron Nov 2018;65(11):8993–9004.

[8] Rizeakos V, Bachoumis A, Andriopoulos N, Birbas M, Birbas A. Deep learning-based application for fault location identification and type classification in active distribution grids. Appl Energy 2023;338:120932.

[9] Wang Y, Chen Q, Hong T, Kang C. Review of smart meter data analytics: applications, methodologies, and challenges. IEEE Trans Smart Grid May 2019;10(3):3125–48.

[10] Bhela S, Kekatos V, Veeramachaneni S. Enhancing observability in distribution grids using smart meter data. IEEE Trans Smart Grid Nov 2018;9(6):5953–61.

[11] Yildiz B, Bilbao JI, Dore J, Sproul AB. Recent advances in the analysis of residential electricity consumption and applications of smart meter data. Appl Energy 2017;208:402–27.

[12] Guo Z, Wang ZJ, Kashani A. Home appliance load modeling from aggregated smart meter data. IEEE Trans Power Syst Jan 2015;30(1):254–62.

[13] Jahangir H, et al. A novel electricity price forecasting approach based on dimension reduction strategy and rough artificial neural networks. IEEE Trans Indus Inform Apr. 2020;16(4):2369–81.

[14] Jiang Y, Liu C, Diedesch M, Lee E, Srivastava AK. Outage management of distribution systems incorporating information from smart meters. IEEE Trans Power Syst Sep 2016;31(5):4144–54.

[15] Syed D, et al. Deep learning-based short-term load forecasting approach in smart grid with clustering and consumption pattern recognition. IEEE Access 2021;9:54992–5008.

[16] Yin L, Xie J. Multi-temporal-spatial-scale temporal convolution network for short-term load forecasting of power systems. Appl Energy 2021;283:116328.

[17] Jiang Y, et al. Very short-term residential load forecasting based on deep-autoformer. Appl Energy 2022;328:120120.

[18] Fekri MN, Patel H, Grolinger K, Sharma V. Deep learning for load forecasting with smart meter data: online adaptive recurrent neural network. Appl Energy 2021;282:116177.

[19] Quilumba FL, Lee W, Huang H, Wang DY, Szabados RL. Using smart meter data to improve the accuracy of intraday load forecasting considering customer behavior similarities. IEEE Trans Smart Grid Mar 2015;6(2):911–8.

[20] Khan ZA, Jayaweera D. Smart meter data based load forecasting and demand side management in distribution networks with embedded PV systems. IEEE Access 2020;8:2631–44.

[21] Yang Y, Li W, Gulliver TA, Li S. Bayesian deep learning-based probabilistic load forecasting in smart grids. IEEE Trans Indus Inform Jul 2020;16(7):4703–13.

[22] Kong W, Dong ZY, Hill DJ, Luo F, Xu Y. Short-term residential load forecasting based on resident behaviour learning. IEEE Trans Power Syst Mar 2018;33(1):1087–8.

[23] Kong W, Dong ZY, Jia Y, Hill DJ, Xu Y, Zhang Y. Short-term residential load forecasting based on lstm recurrent neural network. IEEE Trans Smart Grid Jan. 2019;10(1):841–51.

[24] Xygkis TC, Karlis GD, Siderakis IK, Korres GN. Use of near real-time and delayed smart meter data for distribution system load and state estimation. In: *Proc.* 9th MedPower, Athens, Greece; Nov. 2014. p. 1–6.

[25] Lim B, Zohren S. Time-series forecasting with deep learning: a survey. Phil Trans R Soc A 2021. https://doi.org/10.1098/rsta.2020.0209.

[26] Haben S, Arora S, Giasemidis G, Voss M, Vukadinović Greetham D. Review of low voltage load forecasting: methods, applications, and recommendations. Appl Energy 2021;304:117798.

[27] Sanduleac M, Ciornei VI, Toma L, Plamanescu R, Dumitrescu AM, Albu M. High reporting rate smart metering data for enhanced grid monitoring and services for energy communities. IEEE Transactions on Industrial Informatics. 2021 [early access].

[28] Kelly M, Wells L. Inside-the-meter intelligence to become the norm [white paper]. Guidehouse Insights, Available: https://utilities.sense.com/wp-content/uploads/2022/08/Guidehouse-Insights-Inside-the-Meter-Intelligence-to-Become-the-Norm.pdf.

[29] Singh U, Zamani V, Baran M. "On-line load estimation for distribution automation using AMI data," *2016 IEEE Power and Energy Society General Meeting (PESGM)*. Jul. 2016. p. 1–5.

[30] Cheng G, Lin Y, Abur A, Gómez-Expósito A, Wu W. A survey of power system state estimation using multiple data sources: PMUs, SCADA, AMI, and beyond. IEEE Trans Smart Grid 2023 [early access].

[31] Zhang X, Eseye AT, Knueven B, Liu W, Reynolds M, Jones W. Curriculum-based reinforcement learning for distribution system critical load restoration. IEEE Trans Power Syst 2023;38(5):4418–31.

[32] Li Z, Liu J, Lin Y, Wang F. Grid-constrained data cleansing method for enhanced bus load forecasting. IEEE Trans Instrument Measur 2021;70:1–10.

[33] Won J, Ma CYT, Yau DKY, Rao NSV. Privacy-assured aggregation protocol for smart metering: a proactive fault-tolerant approach. IEEE/ACM Trans Network 2016;24(3):1661–74.

[34] Xu J, Wu Z, Zhang T, Hu Q, Wu Q. A secure forecasting-aided state estimation framework for power distribution systems against false data injection attacks. Appl Energy 2022;328:120107.

[35] Gelper S, Fried R, Croux C. Robust forecasting with exponential and holt–winters smoothing. J Forecast 2010;29(3):285–300.

[36] Al-Otaibi R, Jin N, Wilcox T, Flach P. Feature construction and calibration for clustering daily load curves from smart-meter data. IEEE Trans Indus Inform Apr 2016;12(2):645–54.

[37] Kalekar PS. Time series forecasting using holt-winters exponential smoothing. Kanwal Rekhi School of Information Technology. 2004.

[38] Pecan Street Database. [Online]. Available: http://www.pecanstreet.org/.

[39] NSRDB: National Solar Radiation Database. Available: https://nsrdb.nrel.gov.

[40] Ding N, Benoit C, Foggia G, Bésanger Y, Wurtz F. Neural network-based model design for short-term load forecast in distribution systems. IEEE Trans Power Syst 2016;31(1):72–81.

[41] Bergstra J, Yamins D, Cox D. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. Int Conf Mach Learn 2013:115–23.

[42] Liu J, Ponci F, Monti A, Muscas C, Pegoraro PA, Sulis S. Optimal meter placement for robust measurement systems in active distribution grids. IEEE Trans Instrument Measur 2014;63(5):1096–105.

[43] Damavandi MG, Krishnamurthy V, Martí JR. Robust meter placement for state estimation in active distribution systems. IEEE Trans Smart Grid 2015;6(4): 1972–82.