Energy & Buildings 251 (2021) 111375

Contents lists available at ScienceDirect

Energy & Buildings

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

A novel short-term load forecasting framework based on time-series clustering and early classification algorithm

Zhe Chen [a], Yongbao Chen [a,b,*], Tong Xiao [c], Huilong Wang [d,e], Pengwei Hou [f]

[a] School of Energy and Power Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China
[b] Shanghai Key Laboratory of Multiphase Flow and Heat Transfer in Power Engineering, Shanghai 200093, China
[c] Department of Mechanical Engineering, Tongji University, Shanghai 201804, China
[d] Department of Construction Management and Real Estate, Shenzhen University, Shenzhen 518052, China
[e] Sino-Australia Joint Research Center in BIM and Smart Construction, Shenzhen University, Shenzhen 518052, China
[f] China Construction Third Bureau First Engineering Co., LTD, Shanghai 201100, China

ARTICLE INFO

Article history:
Received 19 April 2021
Revised 1 August 2021
Accepted 17 August 2021
Available online 21 August 2021

Keywords:
Short-term load forecasting
Light gradient boosting machine (LightGBM)
Time series clustering
Early classification
Feature engineering

ABSTRACT

With the development of data-driven models, extracting information from historical data for better energy forecasting is critically important for energy planning and optimization in buildings. Feature engineering is a key factor in improving the performance of forecasting models. Adding load pattern labels for different daily energy consumption patterns resulting from different time schedules and weather conditions can help improve forecasting accuracy. Traditionally, pattern labeling focuses mainly on finding a day similar to the forecasting day based on calendar or other information, such as weather conditions. The most intuitive approach for dividing historical time-series load into patterns is clustering; however, the pattern cannot be determined before the load is known. To address this problem, this study proposes a novel short-term load forecasting framework integrating an early classification algorithm that uses a stochastic algorithm to predetermine the load pattern of a forecasting day. In addition, a hybrid multistep method combining the strengths of single-step forecasting and recursive multistep forecasting is integrated into the framework. The proposed framework was validated through a case study using actual metered data. The results demonstrate that the early classification and proposed labeling strategy produce satisfactory forecasting accuracy and significantly improve the forecasting performance of the LightGBM model.

1. Introduction

With the rapid development of urbanization and extensive usage of intermittent renewable energies, the energy balance between the supply side of the grid and the demand side of buildings presents a new problem worldwide. To address this problem, forecasting the energy demand of buildings quickly and accurately is important, as with short-term load forecasting (STLF) [1,2]. According to the modeling principles and input information, load forecasting models are usually classified into physical and data-driven models [3,4]. Physical models forecast energy consumption based on building characteristics and operational information such as HVAC system data and occupancy schedules. Many software packages have been developed based on physical models, such as DOE-2, EnergyPlus, eQuest, Trnsys, Dymola, and others [5]. However, the indispensable input information is usually difficult to collect for building energy forecasting.

Unlike physical models, thermal balance equations are not required for data-driven models; little physical building information is required. Data-driven models have become popular in building energy forecasting owing to their simplicity and flexibility. Autoregressive integrated moving average (ARIMA) and its variants are powerful time-series techniques based on the data-driven model and have been applied in many short-term load forecasting studies. Jeong et al. [6] combined seasonal autoregressive integrated moving average (SARIMA) with an artificial neural network (ANN) to forecast the seasonal and non-linear parts, respectively. Nie et al. [7] conducted similar research using an exponential kernel-based support vector machine (SVM) instead of an ANN. Both concluded that the hybrid model outperformed the conventional ARIMA model. Kim et al. [8] compared ANN with ARIMA and its variants, and concluded that the ANN model had the best prediction accuracy and robustness. In Ref. [9], time-series forecasts based on SARIMA with exogenous inputs (SARIMAX)

* Corresponding author at: School of Energy and Power Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China.
E-mail address: chenyongbao@usst.edu.cn (Y. Chen).

https://doi.org/10.1016/j.enbuild.2021.111375
0378-7788/

Nomenclature

Abbreviation

ANN	Artificial neural networks
ARIMA	Autoregressive integrated moving average
CNN	Convolutional neural network
CV-RMSE	Coefficient of variation of root mean square error
DBA	Dynamic time warping barycenter averaging
DNN	Deep neural network
DTW	Dynamic time wrapping
GBDT	Gradient-boosting decision tree
GLMM	Generalized linear mixed model encoding
LightGBM	Light gradient boosting machine
LSTM	Long short-term memory
MAE	Mean absolute error
MAPE	Mean absolute percentage error
MVscaled	Mean-variance scaled
RF	Random Forest
RNN	Recurrent neural network
SARIMA	Seasonal autoregressive integrated moving average
SARIMAX	SARIMA with exogenous inputs
STLF	Short-term load forecasting
SVM	Support vector machine
SVR	Support vector regression
WOE	Weight of evidence
XGBoost	Extreme gradient boosting
RH	Relative humidity
STD	Standard deviation

Subscript

cb	CatBoost encoded features
dry	Dry-bulb
dew	Dew-bulb
glmm	GLMM encoded features
n, t	The index of a series
iter	Iteration
smo	Smoothed data
d, i	Data of the ith day
h, i	Data of the ith hour in a day

Symbol

lag, t	$t$hour shifted feature
km	K-means
τ	Timestamp of variables
DTW	DTW distance
MV	Mean-variance scaled
v	Wind speed
w	Building load
z	Cluster label
$\widehat{z}$	Predicted cluster label
μ	Arithmetic average
σ	Variance, Parameter of gaussian kernel
$\mathbf{x}, \mathbf{y}$	Single time-series
x_i, y_i	Single point of time-series
Pr	Pressure
T	Temperature
W	Load feature
P, Q	Other features
hour	Hour of day
weekday	Day of week
month	Month of year
Sum	Sum of a group of numbers
Mean	Arithmetic average
$dist(*)$	Distance between time-series
$d_{euc}(*)$	Euclidean distance
$d_{dtw}(*)$	DTW distance
$d_{sbd}(*)$	Shape-based distance
$k_{gau}(*)$	Gaussian kernel
$C_t(*)$	Misclassification cost function
$P(*)$	Probability
c	Cluster

were shown to be less accurate than deep learning models such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs) [10,11]. Du et al. also developed deep learning models for univariate and multivariate time series forecasting in various domains[12,13]. Random forest (RF) is useful for dealing with high-dimensional forecasting. Fan et al. compared RF with other data-driven models and the results showed that RF and SVM were the two best models for the next-day load and peak load forecasting [14]. Johannesen evaluated data-driven models and concluded that RF performed best in STLF (30 min) [15]. In recent years, gradient boosting models have been proven effective for forecasting building energy consumption. Wang et al. [16] integrated deep learning models such as the temporal convolutional network (TCN) into the tree-based model LightGBM [17]; these models outperformed other artificial intelligence algorithms such as support vector regressor (SVR) and long short-term memory (LSTM). XGBoost [18] is another powerful tree-based model used to forecast multi-step heat loads; it was found to perform better than SVR and deep neural networks (DNNs) [19]. In addition to better performance, the importance of features used in tree-based ensemble models can be clearly explained through a score indicating the value of each feature as a splitting node in each tree. Zheng et al. [20] proposed a similar day selection method based on the feature importance score from the XGBoost algorithm output.

Adding features with practical meaning to energy consumption can improve forecasting accuracy. Given that building energy consumption tends to have several daily patterns, adding labels for different patterns can help improve forecasting accuracy. The most intuitive approach is to label the daily pattern of building energy consumption using a time-series clustering algorithm such as k-shape [21], k-means [22–24], or kernel k-means [25]. Using time-series clustering, the historical energy data of a building can be divided into groups with different patterns. However, the paradox in applying a load pattern to STLF is that we cannot label the pattern of the forecasting day before we know the actual load values. Thus, previous research has only applied clustering algorithms in the preprocessing step. Quintana et al. [26] found that k-shape clustering can help correctly label the primary space usage of a building to improve building performance benchmarking and analysis. Fan et al. [14] applied entropy-weighted k-means clustering analysis in the preprocessing step to detect outliers. Xue et al. [19] adopted cluster analysis to identify daily heating load patterns for both the training and test sets to evaluate the performance of the proposed forecasting method. To avoid the paradox, researchers have developed other methods to label the patterns of forecasting days. Traditionally, day type (weekday or weekend) is considered. Mandal et al. [27] proposed a similar day approach using the Euclidean distance of hourly load and temperature

between the forecasting day and historical days, and assigned the load data of similar days as an input of the neural network to improve accuracy. Mu et al. [28] calculated the similarity between two days according to day type, weather type, and maximum and minimum temperature, aggregated with different weights.

Given that STLF usually requires hourly forecast loads for the following hours, a multistep method is often used to improve accuracy, especially for non-time-series models. Multistep forecasting uses historical load data (1 h past, 2 h past, etc.) to make a prediction hour by hour. The longer the span to be predicted, the more calculation is required, resulting in a greater time cost. Xue et al. [19] compared the results of direct models (single-step XGBoost) with recursive models (multistep XGBoost); the results showed that the recursive models outperformed the direct models by a small margin (CV-RMSE of 10.52% vs. 10.73%) for 24 h-ahead load forecasting.

Based on our literature review, we arrived at several conclusions. First, tree-based models such as XGBoost and LightGBM are promising algorithms for STLF; further research should be conducted on feature engineering to improve their forecasting performance. Second, load pattern methods usually focus on day type and weather type, which are not equivalent to the actual load patterns that are critically important for building load forecasting. Because the forecasting day load is unknown, a stochastic method should be applied to predefine the energy load pattern of the forecasting days. Finally, a recursive multistep load forecasting method yields better performance, although at a greater calculation cost.

To address these problems, we propose a novel short-term load forecasting framework for the LightGBM algorithm. The framework originally proposes an early classification algorithm to accurately label the load patterns before forecasting. In the algorithm, the historical load of past several hours is used to generate the labels instead of day types or weather types in the traditional forecasting frameworks. More useful information is extracted as input features by using the algorithm. In the proposed framework, first, time-series clustering and an early stochastic classification algorithm are used to mine deeply hidden information from the historical load characteristics to create several new input features for the LightGBM model training. The dimension of input features is critically important for improving the forecasting performance of data-driven models. Second, a hybrid multistep forecasting method is adopted in the framework, combining the merits of the single-step and recursive multistep methods. With this framework, the forecasting performance of the LightGBM model in our case study was significantly improved. This framework can be generalized for other types of data-driven models, for short-term and long-term load forecasting. The remainder of this paper is organized as follows: Section 2 introduces the main algorithms used in this study and the framework we propose for STLF based on early classification. Section 3 presents a case study to validate the proposed STLF framework. Section 4 presents the results and discussion of the case study. Conclusions are presented in Section 5.

2. Methodology

The basic algorithms adopted in the proposed STLF are introduced in Sections 2.1–2.5. The details of the framework are presented in Section 2.6; the model performance evaluation metrics are presented in Section 2.7.

2.1. Basic data processing techniques

2.1.1. Categorical feature encoding

Encoding of categorical features helps to explain hidden information within them. Many categorical features serve as the inputs in prediction models, such as day type and weather type. We can obtain additional features through early classification, as described in Section 2.3. There are many encoding methods, including one-hot encoding, target encoding, and weight of evidence (WOE) [29]. In this study, we chose CatBoost encoding and the generalized linear mixed model (GLMM) encoder for their good compatibility with continuous targets and robustness in machine learning [29].

The CatBoost encoder is an encoding method enclosed in CatBoost [30] that can transfer categorical features into numerical features and help prevent overfitting. The encoding values *En* for each categorical feature are defined in Eq. (1):

$$En = \frac{Sum_i + Mean}{Count_i + 1} \tag{1}$$

where Sum_i denotes the sum of each category; *Mean* denotes the mean value of encoding categorical features; $Count_i$ denotes the count for each category.

The GLMM encoder has been proven to be one of the best encoding methods in the benchmark test of categorical feature encoding [29].

2.1.2. Data smoothing

There are many time-series input features for load forecasting, including dry-bulb temperature and relative humidity. These time series may contain noise and should be smoothed. One of the most widely used smoothing algorithms is the Kalman smoother [31,32], which smooths a time series by estimating its distribution. The fundamentals of the Kalman smoother are detailed in Ref. [33].

2.1.3. Data scaling

Depending on the characteristics of the time series and the algorithms used for clustering, scaling can help improve clustering performance. The most widely used scaler for time series is mean–variance scaling, which can transform time-series into the same mean and variance values. The time series $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ can be transformed into $\boldsymbol{x}' = (x'_1, x'_2, \cdots, x'_n)$, according to Eqs. (2)–(4):

$$x'_i = \frac{x_i - \mu}{\sigma} \tag{2}$$

$$\mu = \frac{1}{n}\sum_{i}^{n}(x_i) \tag{3}$$

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2} \tag{4}$$

where σ denotes the variance of the time series and μ denotes the arithmetic mean of the time series.

2.2. Time-series clustering

2.2.1. K-means algorithm

The K-means algorithm is a widely used clustering algorithm proposed by MacQueen [34]. Classic K-means can separate a set of data into *K* clusters according to the distance to each cluster centroid. The objective of K-means is to find the best partition with the minimum loss, as defined in Eq. (5):

$$P^* = \arg\min_{\{p_i\}} \sum_{j=1}^{K}\sum_{w_i \in p_j}^{n} dist\left(\left\|w_i - c_j\right\|^2\right) \tag{5}$$

where P^* denotes the best partition; w_i denotes the time series in the dataset; c_j denotes the cluster centroid of cluster *j*.

Because it is an NP-hard problem, the solution of K-means starts from a heuristic search.

Step 1: Input dataset $W(\mathbf{x}, \mathbf{y}, \cdots)$ and cluster number *K*.

Step 2: Randomly initialize K cluster centroids from $W(\mathbf{x}, \mathbf{y}, \cdots)$.

Step 3: Calculate every data point in the $W(\mathbf{x}, \mathbf{y}, \cdots)$ to K cluster centroids, assign data to the nearest cluster, and recalculate the centroid of each cluster.

Step 4: Iterate Step 3 until the centroids are stable.

The key to assigning data points into different clusters is calculating the centroids and the distance between two points. Thus, two instances of the time series can be formulated as follows:

$$\boldsymbol{x} = (x_1, x_2, \cdots, x_n) \tag{6}$$

$$\boldsymbol{y} = (y_1, y_2, \cdots, y_n) \tag{7}$$

where n denotes the dimension of a single time series.

Traditionally, Euclidean distance is calculated as the distance:

$$d_{euc}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_n - y_n)^2} \tag{8}$$

The arithmetic mean of the time-series coordinates is typically used to compute the cluster centroids when using the Euclidean distance. Although Euclidean distance is useful when calculating low-dimensional data points, important information may be lost if used with high-dimensional data points such as time-series data. For example, the Euclidean distance between $\mathbf{x} = (1, 2, 1, 2, 1, \cdots)$ and $\mathbf{y} = (2, 1, 2, 1, 2, \cdots)$ is greater than zero according to Eq. (8), even though the time series have the same pattern. Thus, the distance should be calculated by another method for time-series forecasting. Dynamic time wrapping (DTW), based on the Levenshtein distance, has been proven to be useful in time-series analysis [35,36]. It was first introduced to calculate the distance between two sound waves with different lengths [26]. DTW flexibly aligns two sets of time series so that the distance can be calculated more reasonably. There are three alignment rules:

Rule 1: The time series should be aligned in the same direction, either from left to right or from right to left.

Rule 2: One point in the time series w_1 should have at least one align point in w_2 and vice versa.

Rule 3: Backward alignment is not allowed.

Then, the distance is calculated as the cost of the optimal alignment path:

$$\begin{aligned} d_{dtw}(x_i, y_j) &= d_{euc}(x_i, y_j) \\ &+ \min\left[d_{dtw}(x_{i-1}, y_{j-1}), d_{dtw}(x_i, y_{j-1}), d_{dtw}(x_{i-1} + y_j)\right] \end{aligned} \tag{9}$$

Eq. (9) obtains the distance between two time series: $d_{dtw}(\mathbf{x}, \mathbf{y}) = d_{dtw}(x_n, y_n)$. Fig. 1 illustrates the DTW alignment results for two hourly load time series using the DTW package [37] (the time series are shifted 200 kW apart on the y-axis for better illustration). For DTW, centroids are often computed through dynamic time-warping barycenter averaging (DBA) [22], which helps to maintain the shape characteristics.

2.2.2. K-shape algorithm

Considering that the shape of a time series can express its pattern, a shape-based distance is applied in the K-shape clustering algorithm. Shape-based distance is not sensitive to time shifting and scaling, which is important when calculating distances. The shape-based distance is defined in Ref. [21].

2.2.3. Kernel K-means algorithm

Euclidean distance is not suitable for high-dimensional time series because the information is lost. Kernel functions such as a Gaussian kernel can map the data to a higher-dimensional space without much information loss.

$$k_{gau}(x, y) = exp\left(-\frac{||x - y||^2}{2\sigma^2}\right) \tag{10}$$

During kernel K-means clustering, data points are mapped into a hyper-dimensional space, and the cluster centroids are calculated based on the kernel function.

2.3. Early classification algorithm for time series

In practical application, the load patterns of forecasting days cannot be obtained in advance. However, most STLF tasks focus on peak load time (e.g., 13:00 pm −15:00 pm) during a day. Our motivation is to apply early classification algorithms using historical load data including several early hours to label load in the forecasting day with more input features.

$\mathbf{x}_t = (x_1, x_2, \cdots, x_t)$ is a segment of the time series $\mathbf{x}_t = (x_1, x_2, \cdots, x_t, \cdots, x_n)$. The target of early classification can be formulated using the following cost function [38]:

$$f(x_t) = \sum_{z \in Z} P(z|x_t) \sum_{\hat{z} \in Z} P\left(\hat{z}z, x_t\right) C_t\left(\hat{z}z\right) \tag{11}$$

where $C_t(\hat{z}|z)$ denotes the misclassification cost function with true class z labeled as $\hat{z}$.

The conditional probability $P(\hat{z}|z, \mathbf{x}_t)$ is unknown and can be estimated [38]. Thus, the optimal time t^* can be defined as

$$t^* = \underset{t \in \{1, \cdots, t_l\}}{\operatorname{argmin}} f(x_t) \tag{12}$$

where t_l denotes the latest allowable early classification time; that is, the forecasting start time.

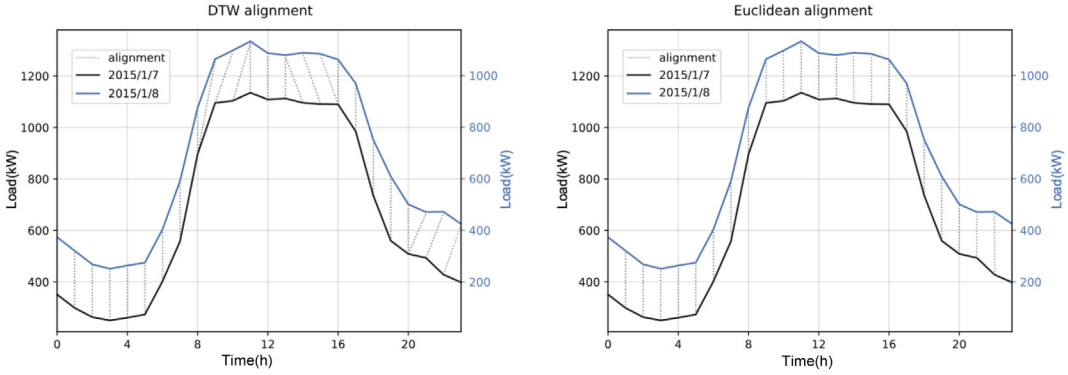

Fig. 1. DTW and Euclidean alignment of two load time series.

2.4. Tree-based model: LightGBM

LightGBM is a gradient-boosting framework based on the decision tree algorithm; that is, a gradient-boosting decision tree (GBDT) [39]. Using the histogram-based technique, LightGBM was designed to be fast, and was distributed based on its advantages in terms of faster training speeds, higher efficiency, lower memory usage, and parallel support for classification and regression. LightGBM was first released in 2017; a detailed theoretical introduction can be found in Ref. [17]. It was recommended by researchers as a promising forecasting algorithm in modern buildings with the development of big data technologies [40–42]. Thus, we adopted the LightGBM algorithm to investigate and validate our proposed forecasting framework.

2.5. Single-step and recursive multistep forecasting

STLF uses historical load data and other known information to predict the load in the following hours or days, which can be formulated as

$$\widehat{W} = f(W_t, P_t, Q) \tag{13}$$

where $\widehat{W} = (w_{t+1}, w_{t+2}, \cdots, w_{t+n})$ denotes the load from time $t+1$ to $t+n$ to be forecast; $W_t = (\cdots, w_{t-2}, w_{t-1}, w_t)$ denotes historical load data; $P_t = (\cdots, p_{t-2}, p_{t-1}, p_t)$ denotes other information, such as occupancy data; $Q = (\cdots, q_{t-2}, q_{t-1}, q_t)$ denotes the features that include both known and predictive data, such as weather.

Single-step and recursive multistep methods can be adopted to solve $\widehat{W}$.

2.5.1. Single-step STLF method

Developing forecasting models for $w_{t+1}, w_{t+2}, \cdots, w_{t+n}$, respectively:

$$w_{t+1} = f_1(W_t, P_t, Q) \tag{14}$$

$$w_{t+n} = f_n(W_t, P_t, Q) \tag{15}$$

To forecast the load of the following hours, the single-step method [43] is a compromise of building models for each forecasting point and is most widely used, especially for artificial intelligence algorithms such as SVM and ANN [44,45]. The model can be formulated as

$$w_{t+i} = f_i\big(W_t(\cdots, w_{t+i-m-1}, w_{t+i-m}), P_t(\cdots, p_{t+i-m-1}, p_{t+i-m}), Q(\cdots, q_{t-1}, q_t, q_{t+i})\big) \tag{16}$$

where $m > i$, meaning that the input features of load (W and P) are earlier than the forecasting period, which ensures every forecasting point with the same length has a valid feature. The upper part of Fig. 2 shows an example of using past 3 h and 12 h loads to forecast from $t+1$ to $t+3$. The disadvantage of this method is that some historical data are not used (w_{t-1} and w_t are not used for forecasting w_{t+1}).

2.5.2. Recursive multistep STLF method

Artificial intelligence algorithms such as SVM and ANN cannot address data with sequential information; the multistep method is applied to mitigate this weakness. The merit of multistep forecasting is that it can fully use historical data, which can increase forecasting accuracy [19]. Moreover, the approach is flexible, addressing different forecasting spans. The forecasting model is

$$w_{t+i} = f(W_{t-i+1}, P_t, Q) \tag{17}$$

The lower part of Fig. 2 shows an example of multistep STLF using past 6 h load data. When the first w_{t+1} is forecast, it is used as an input to forecast w_{t+2}. Thus, n forecasts are required to forecast from $t+1$ to $t+n$.

2.6. STLF framework based on time-series clustering and early classification

The STLF framework is shown in Fig. 3 and Fig. 4. The goal of the framework is to forecast the load after t for the following n hours $(w_{t+1}, w_{t+2}, .., w_{t+n})$. The raw input includes historical load, weather data, and other categorical features such as day of week and month of year. There are four main processes involved.

2.6.1. Data preprocessing

In this step, data cleansing techniques such as outlier detection and missing data imputation are adopted to ensure the reliability of the input data.

2.6.2. Early classification

Historical hourly load data are split into forecasting day load $W_1 = (w_{t-j+1}, ..., w_{t-1}, w_t)$ and non-forecasting data $W_2 = (w_0, w_1, ..., w_{t-j})$ for clustering and early classification to generate daily load patterns. In Fig. 3, only one forecasting day is illustrated for better understanding. Non-forecasting data are reshaped into a set of daily time series $W_2' = (W_{d,1}, W_{d,2}, \cdots, W_{d,m})$and then scaled time series with mean–variance scaling [46] to ensure that each time series has a similar scale. Subsequently, time-series clustering algorithms such as K-means and kernel K-means are used to classify historical daily time series into several clusters to obtain labels of daily data $L_2' = (l_{d,1}, l_{d,2}, \cdots, l_{d,m})$. Early classification is applied to predict the daily cluster label of the forecasting day $l_{d,m+1}$ using the first several load data points W_1 (e.g., load from 0:00 am to 9:00 am). The forecasted daily label and cluster labels are mapped to hourly data $L_1 = (l_{t-j+1}, ..., l_{t+k-1}, l_{t+k})$ and $L_2 = (l_0, l_1, ..., l_{t-j})$, respectively, as shown in the lower part of Fig. 3. Hourly labels are joined together as new input features, $L = (l_0, l_1, ..., l_{t+k})$.

2.6.3. Feature engineering

Different techniques are adopted according to the feature type. For categorical features such as cluster labels and day type, categorical feature encoding is used to better explain the features. For numerical features such as dry-bulb temperature and relative humidity, data smoothing techniques (e.g., Kalman smoother) and time shift techniques (e.g., shifting T_{dry} three hours ahead to obtain the dry-bulb temperature three hours ago $T_{dry,lag,3}$ to address the lag effect of building thermal mass [47]) are used.

2.6.4. Hybrid multistep forecasting

Based on the merits of single-step and multistep STLF, we propose a hybrid method that can improve accuracy and reduce the computation cost. The proposed hybrid method consists of four steps, as shown in Fig. 4.

Step 1: Implement single-step STLF with historical data as iteration 1 (12 h-ahead in the illustration, only load data are shown in the figure).

Step 2: Extract extra features according to the forecast results from Step 1. The load predicted by the single-step method, the load shifted 1 h-ahead [48], and the load difference are selected as new

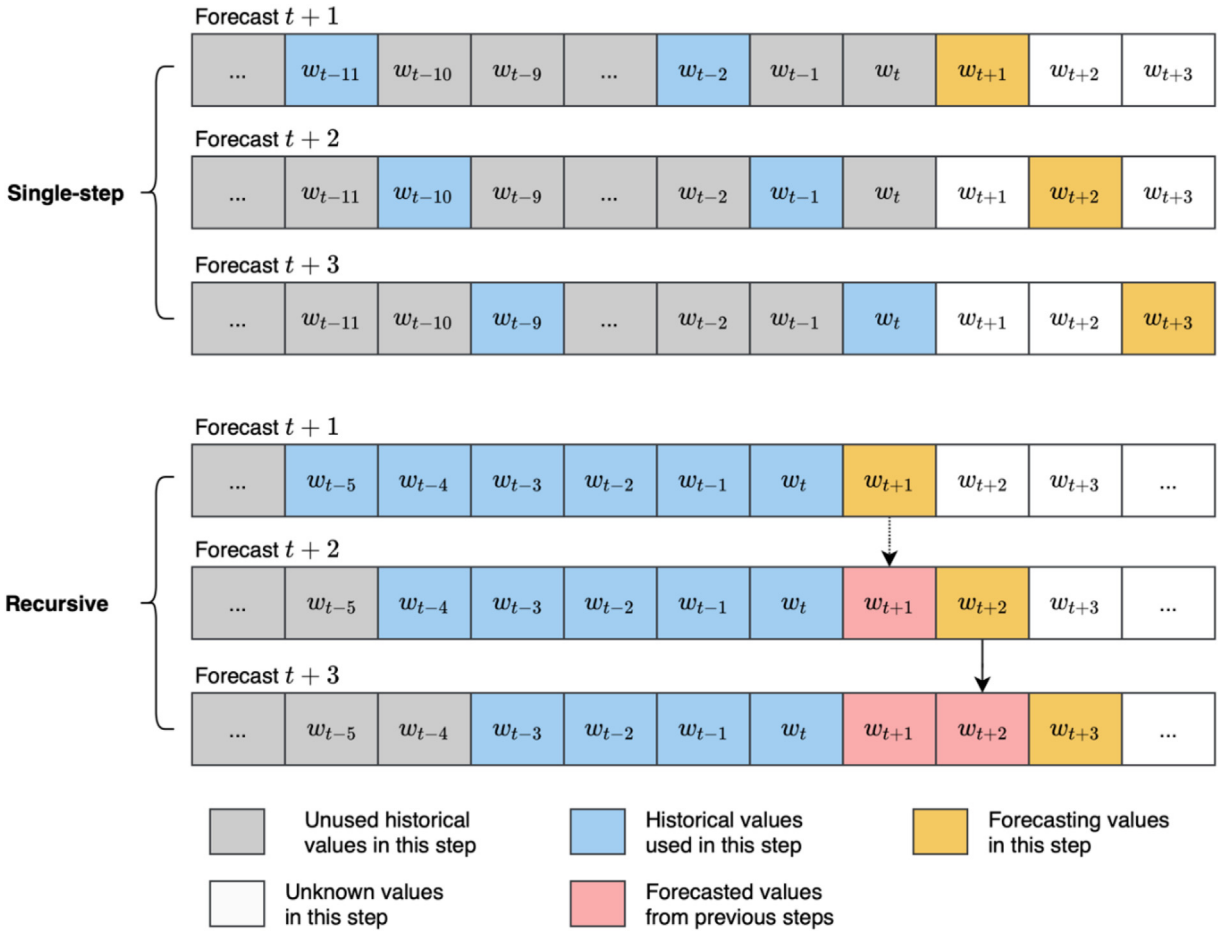

Fig. 2. Single-step and recursive multistep forecasting.

input features 1, 2, and 3, respectively. The data before t in the new features are filled with historical values (gray rectangle in Fig. 4).

Step 3: With three new features extracted from Step 2, re-forecasting is performed along with the old features.

Step 4: Update the forecast values in Step 2 and execute iteration 2; the number of iterations depends on the performance improvement.

2.7. Model evaluation

To evaluate the forecasting performance of our proposed models, three widely used metrics, mean absolute percent error (MAPE), mean absolute error (MAE), and coefficient of variation of root mean square error (CV-RMSE) were used. The CV-RMSE is a scale-independent indicator normalized by the RMSE. The CV-RMSE is recommended by the American Society of Heating, Refrigerating, and Air-Conditioning Engineers Guideline 14 [49], and has been adopted in many studies for STLF [50–52]. These metrics are defined in Eqs. (18), (19), and (20), respectively.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|w_i - \widehat{w}_i| \tag{18}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|w_i - \widehat{w}_i|}{w_i} \times 100\% \tag{19}$$

$$CV - RMSE = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(w_i - \widehat{w}_i\right)^2}}{\frac{1}{n}\sum_{i=1}^{n} w_i} \times 100\% \tag{20}$$

where w_i denotes the actual value, $\widehat{w}_i$ denotes the predicted value, and n denotes the number of forecasting points.

3. Case study

3.1. Raw data description

To validate the proposed model, we collected hourly historical electricity load data from 2015 to 2017 for a commercial building located in Shanghai (hot summer/cold winter weather zone). Weather data from a nearby weather station were also collected. Fig. 5 presents the daily peak load and the peak temperature of the building in 2017. The daily peak load has a strong positive correlation with the daily peak dry-bulb temperature during the cooling season (from July to September) and a negative correlation during the heating season (from December to February).

3.2. Data preprocessing

Data preprocessing is an important procedure for STLF; the treatment may differ according to the quality of the raw data. Poor data quality may cause significant performance losses in forecasting. Data cleansing techniques were applied to historical load and weather data in this section. The missing values were checked daily. The data for the entire day were removed when more than three values were missing for the day. Outlier detection was conducted according to the upper and lower bounds. Missing values and outliers were replaced by linear interpolation.

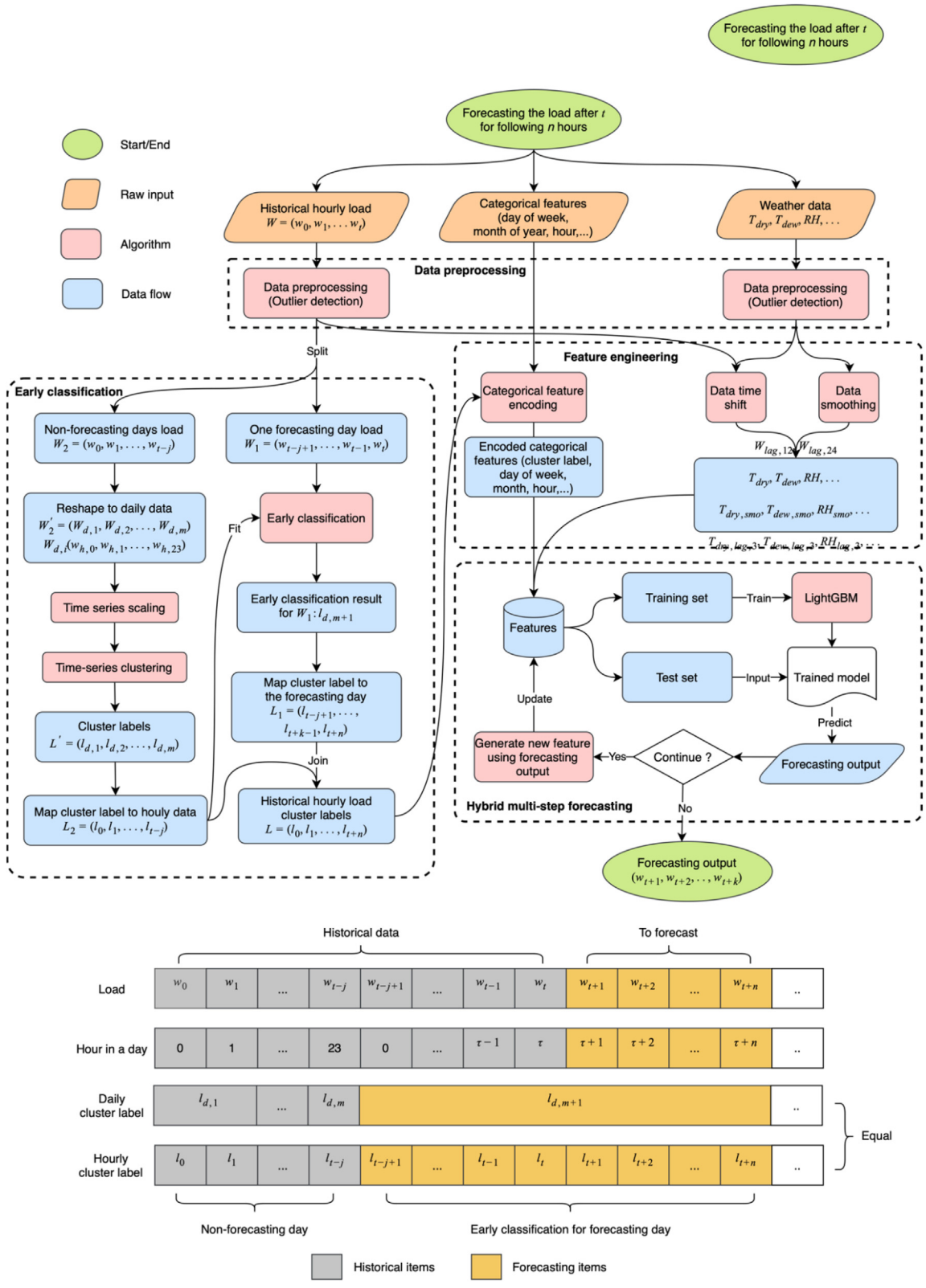

Fig. 3. Flow chart of the proposed k-step STLF framework.

3.3. Training set and test set split

To validate the proposed forecasting framework, we split the data into training and test sets. Two-year data from 2015 to 2016 were chosen as the training set. The training set is treated as the non-forecasting day and the test set is treated as the forecasting day. There were two functions of the training set in the proposed methodology. First, time-series clustering algorithms were applied to the training set to obtain the clustering label as a reference for early classification. Second, the LightGBM model was

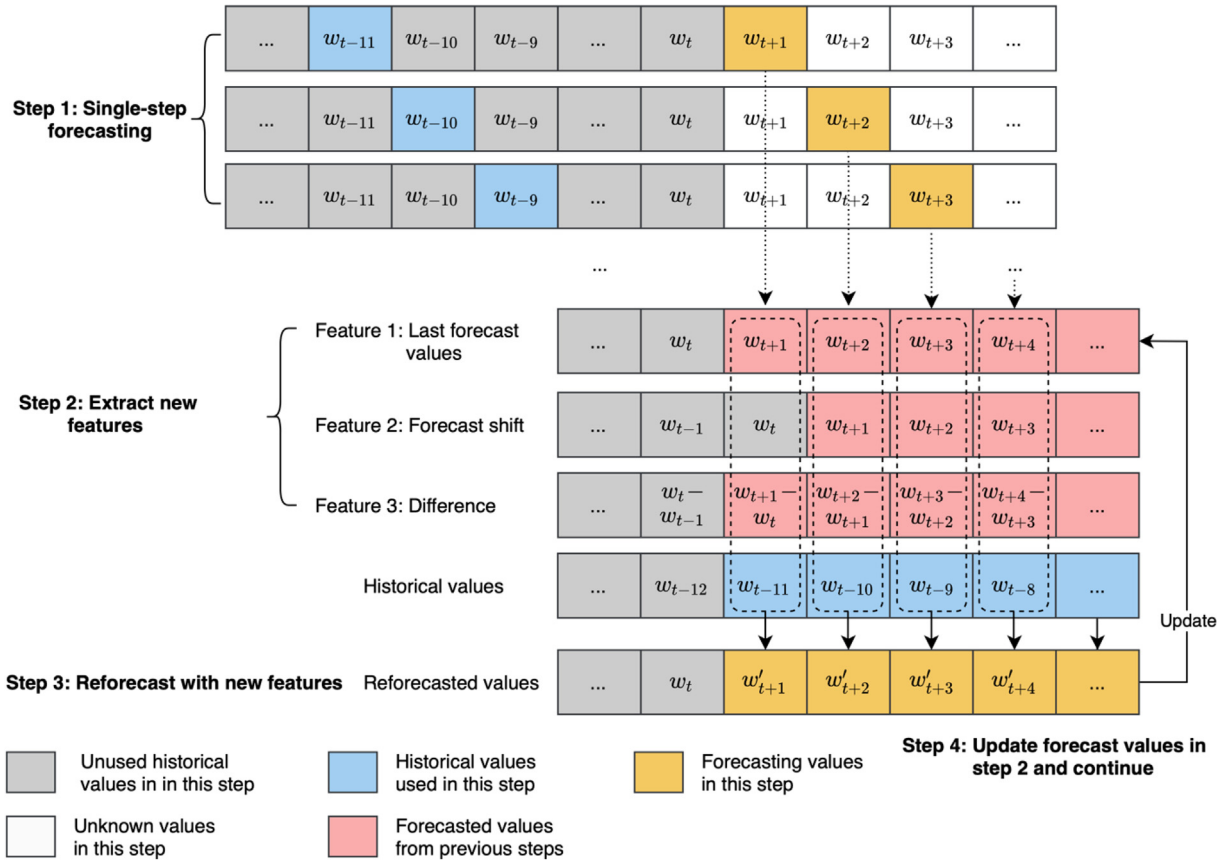

Fig. 4. Illustration of proposed STLF hybrid method.

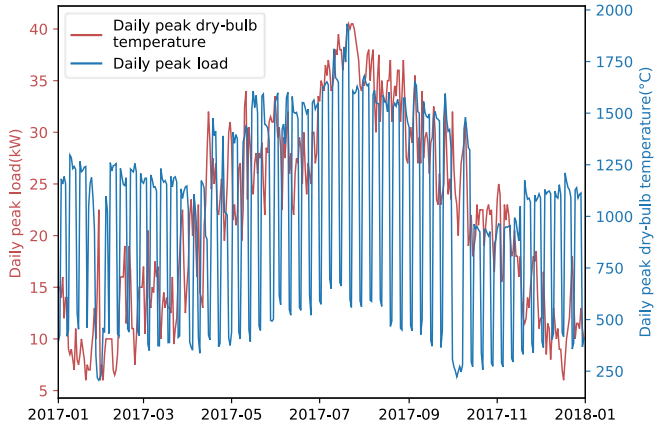

Fig. 5. Daily peak load and temperature of case study building in 2017.

trained on the training set along with the true labels generated by the clustering algorithms. The data of 2017 was used as the test set. When evaluating the test set, the predicted labels generated by the early classification algorithm were used as the input feature. In demand response buildings, the DR events usually occur between 11:00–17:00; for early classification, we make a reasonable assumption that the forecasting time is from 9:00–18:00 every day to ensure that the load data before 9:00 from the forecasting day are known before forecasting for early classification.

3.4. Time-series scaling and clustering

Different time-series clustering algorithms have different distance metrics and cluster centroid calculation methods with different meanings. In this case study, we chose five combinations of clustering algorithms and scaling methods, as shown in Table 1. Usually, the cluster number is determined using criteria such as

Table 1
Clustering algorithms and scaling methods.

Clustering method	Base clustering algorithm	Scaler	Labels of clusters
K-means, DTW	K-means with DTW	None	0, 1, 2
K-means, DTW, MVscaled	K-means with DTW	Mean-variance scaler	0, 1, 2
K-shape, MVscaled	K-shape	Mean-variance scaler	0, 1, 2
Kernel K-means	Kernel K-means	None	0, 1, 2
Kernel K-means, MVscaled	Gaussian Kernel K-means	Mean-variance scaler	0, 1, 2
Clustering method	Base clustering algorithm	Scaler	Labels of clusters
K-means, DTW	K-means with DTW	None	0, 1, 2
K-means, DTW, MVscaled	K-means with DTW	Mean-variance scaler	0, 1, 2
K-shape, MVscaled	K-shape	Mean-variance scaler	0, 1, 2
Kernel K-means	Kernel K-means	None	0, 1, 2
Kernel K-means, MVscaled	Gaussian Kernel K-means	Mean-variance scaler	0, 1, 2

the elbow method [53] and the average silhouette method [54]. With five clustering methods and the load distribution of our case building, the clusters were labeled as 0, 1, 2.

3.5. Early classification label update strategy

The goal of the case study is to evaluate the performance of the proposed STLF method based on the entire year. To fully utilize historical information and improve early classification accuracy, we developed a weekly update strategy; the early classification of the test set was performed weekly. The clustering labels from the previous week were assigned with true labels for the early classification for the following week.

3.6. Input features description

After early classification and feature engineering, all features required for the first iteration were collected. There are four categories of features: weather-related features, historical loads, day types, and clustering labels. All non-hourly data were remapped into hourly data according to their timestamps, as listed in Table 2. Extra features were extracted from the first iteration, including the predicted load in the last iteration, 1 h-shifted predicted load, and their difference.

3.7. Baseline description

To validate the effectiveness of early classification labels, we removed clustering labels from the input features and kept other features as the input. The forecasting result was regarded as the baseline for comparison with the proposed STLF framework of the case study.

4. Results and discussion

4.1. Clustering result

After clustering the training set with five clustering methods, we labeled the clusters $0, 1,$ or2 according to the counting days of each cluster (from the largest cluster count to the smallest), as listed in Table 3. Clustering results differ with different clustering algorithms. K-means with DTW exhibits the most uniform counts of each cluster, whereas kernel K-means with mean–variance exhibits the largest nonuniformity. Overall, the counts of each cluster were sufficiently large for the following early classification. To better illustrate the clustering result, the hourly load distribution along with the dry-bulb temperature for each cluster for each clustering method is shown in Fig. 6 (the load in the graph is without scaling). Different clustering methods can capture different distribution patterns. Clustering was divided into several subdistributions of the whole distribution, as illustrated in the lower right of Fig. 6. (Note that all six figures share the same X-axis and Y-axis) All clustering methods can extract the subdistribution with low load demand (cluster 1 in the figure); kernel K-means (DTW, MVscaled) continues to extract the subdistribution from the low load demand as cluster 2. The first four clustering methods can separate the dry-bulb temperature into two subdistributions (clusters 0 and 2 in the figure) with different boundaries. For K-means (DTW), K-means (DTW, MVscaled), and kernel K-means, cluster 2 extracts the part with a high dry-bulb temperature associated with the cooling season, whereas cluster 0 corresponds to the non-cooling season. K-shape with mean–variance scaling has the largest overlap of clusters 0 and 2, which may result in difficulty for the following early classification.

Table 2
Input features of LightGBM.

Feature category	Feature name	Description	Variable granularity
Weather-related feature	$T_{dry}, T_{dry,lag,3}, T_{dry,smo}$	Dry-bulb temperature	Hourly
	$T_{dew}, T_{dew,lag,3}, T_{dew,smo}$	Dew-bulb temperature	Hourly
	$RH, RH_{lag,3}, RH_{smo}$	Relative humidity	Hourly
	$Pr, Pr_{lag,3}, Pr_{smo}$	Atmospheric pressure	Hourly
	$v, v_{lag,3}, v_{smo}$	Wind speed	Hourly
Historical load	$W_{lag,12}, W_{lag,24}$	Electricity load	Hourly
Day type	$hour_{cb}, hour_{glmm}$	Hour of day	Hourly
	$weekday_{cb}, weekday_{glmm}$	Day of week	Daily
	$month_{cb}, month_{glmm}$	Month of year	Monthly
Clustering label	$day_{km,DTW,cb}, day_{km,DTW,glmm}$	Clustering label of K-means, DTW	Daily
	$day_{km,DTW,MV,cb}, day_{km,DTW,MV,glmm}$	Clustering label of K-means, DTW, MVscaled	Daily
	$day_{ks,MV,cb}, day_{ks,MV,glmm}$	Clustering label of K-shape, MVscaled	Daily
	$day_{kn,cb}, day_{kn,glmm}$	Clustering label of kernel K-means	Daily
	$day_{kn,MV,cb}, day_{kn,MV,glmm}$	Clustering label of kernel K-means, MVscaled	Daily
Extracted features during $(i+1)$ iteration$(i>1)$	$load_{iter,i}$	Forecasted load value in the last iteration	Hourly
	$load_{iter,i,lag,1}$	1 h-shifted forecasted load value	Hourly
	$(load_{iter,i} - load_{iter,i,lag,1})$	Difference between forecasted value and shifted values	Hourly

Table 3
Value counts for each clustering method (731 days in total).

Clustering method	Count of cluster 0	Count of cluster 1	Count of cluster 2	STD of counts
K-means, DTW	278	232	221	24.69
K-means, DTW, MVscaled	362	222	147	89.10
K-shape, MVscaled	306	218	207	44.30
Kernel K-means	292	234	205	36.17
Kernel K-means, MVscaled	507	169	55	191.93

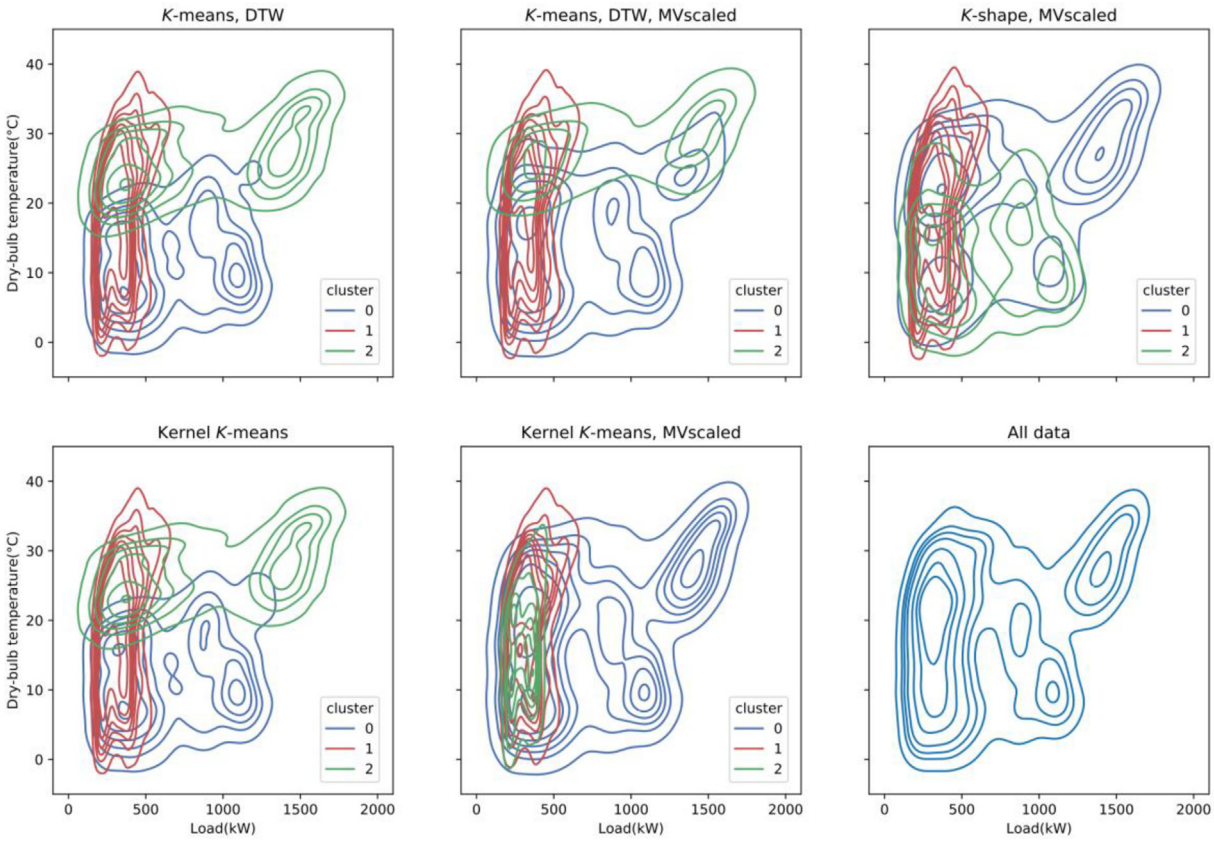

Fig. 6. Clustering result of the training set.

4.2. Early classification result

The cluster labels for the test set were generated through early classification in accordance with the concept that the more data we have, the higher the accuracy the early classification algorithm can yield. Fig. 7 shows the accuracy for different early classification times on the test set. The label update strategy was not applied here; all predicted labels were based on the training set. It is observed that all algorithms produce great accuracy improvement after 7:00 am, and four algorithms reach an accuracy of approximately 90% after 8:00 am. Thus, if the forecasting time begins after 8:00 am, the early classification label for the test set is sufficiently reliable as an input feature. K-means (DTW) and kernel K-means exhibit the best performance on the test set; the accuracy reached over 95% after 10:00 am. The accuracy of K-shape (MVscaled) is lower than that of other methods, due to its reliance on scaling compared with K-means and kernel K-means. The input for early classification was not scaled because the time series for the forecasting day was incomplete before forecasting. In the left part of Fig. 8, the Sankey diagram indicating the mapping relationship between true labels and early classification labels for each clustering method are shown (early classification time is from 0:00–8:00 am, label update strategy applied). The gray line width is proportional to the number of time series that flow from left to right. In the right part of Fig. 8, a heatmap of the arithmetic mean load for each early classification label is plotted on the test set.

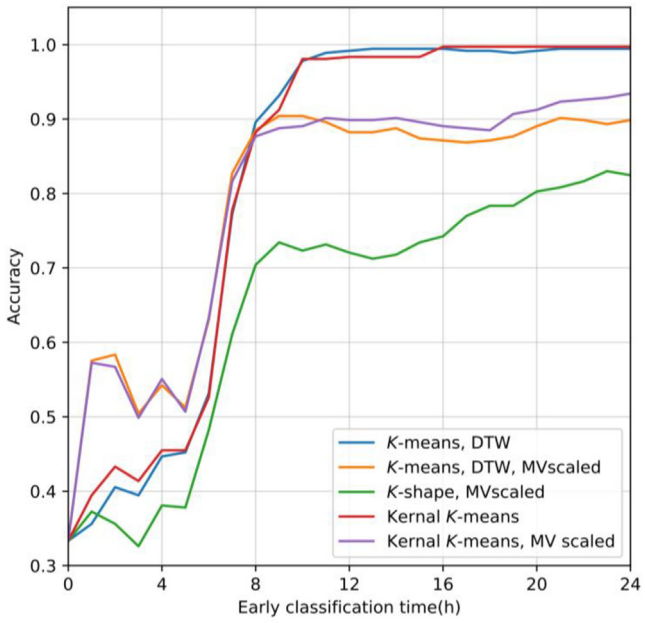

Fig. 7. Early classification accuracy of different clustering algorithms.

4.3. Single-step forecasting

CV-RMSE, MAPE, and MAE are used to evaluate the performance of LightGBM in the single-step forecasting. Three commonly used data-driven models, i.e., SVM, ANN, and RF, are used as reference models, as listed in Table 4. LightGBM performs best in three metrics overall. SVM results in the worst performance but the forecasting accuracy is still acceptable. ANN and RF have similar performance between LightGBM and SVM. With regard to these three metrics, the relative difference of CV-RMSE for different models is smaller compared with MAPE and MAE.

An ablation experiment was carried out to evaluate the effectiveness of feature processing techniques including data smoothing, time shift, and label encoding. As listed in Table 2, data

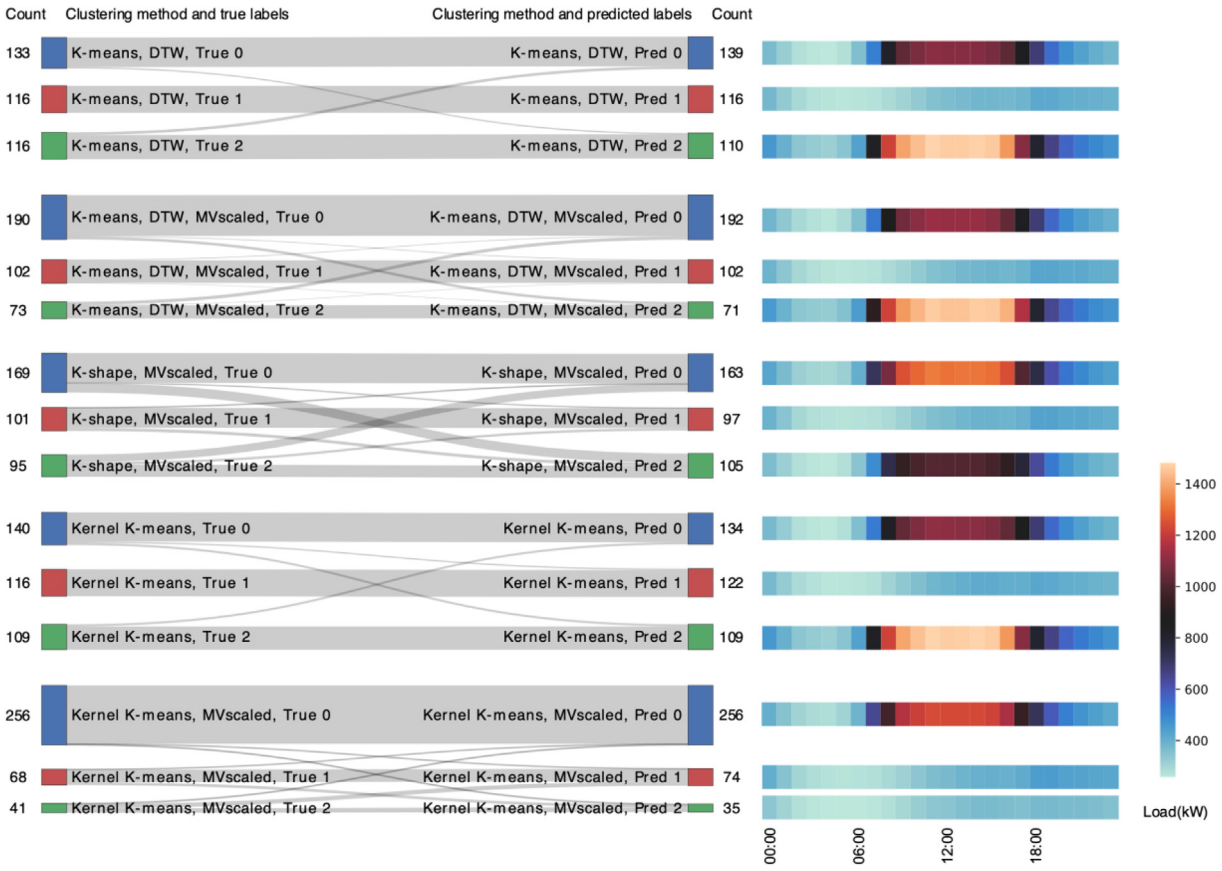

Fig. 8. Sankey diagram and heatmap for each clustering method.

Table 4
Metrics comparison of single-step forecasting.

	CV-RMSE (%)	MAPE (%)	MAE (kW)
LightGBM	9.49	6.92	53.29
SVM	11.64	11.60	78.09
ANN	10.30	9.70	64.14
RF	10.90	8.16	62.28

Table 5
Ablation experiment results of single-step forecasting for LightGBM.

	CV-RMSE (%)	MAPE (%)	MAE (kW)
LightGBM	9.49	6.92	53.29
w/o data smoothing	9.68	7.22	56.37
w/o time shift	9.82	7.21	56.12
w/o label encoding	10.13	7.82	58.11

smoothing and time shift were used to smooth weather-related features. Label encoding was applied to categorical features including day type and clustering label. The results of the ablation experiment are listed in Table 5. Three techniques improve forecasting performance to various extents. Label encoding can improve forecasting performance more than data smoothing and time shift.

4.4. Hybrid multistep forecasting

Fig. 9 presents the CV-RMSE, MAPE, and MAE results for the entire test set of hybrid multistep forecasting. Iteration 1 can be regarded as a single-step method as shown in Section 4.3. It is observed that CV-RMSE decreases rapidly in iteration 2 and

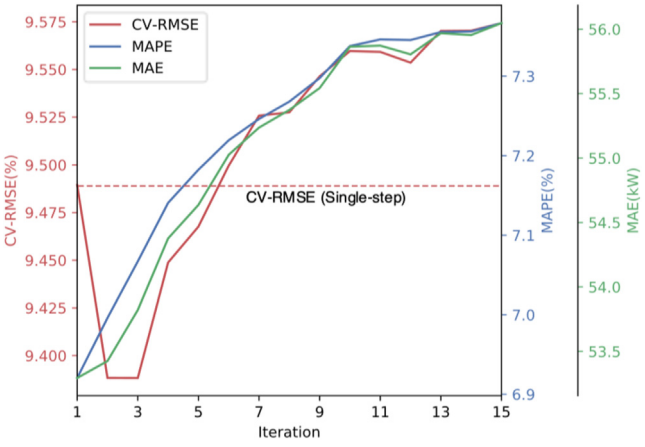

Fig. 9. Performance of hybrid multistep forecasting.

increases after iteration 3; MAPE and MAE increase gradually through the iterations. Thus, the proposed hybrid multistep method can improve the accuracy with only one additional iteration compared with single-step forecasting, with less computational cost than the recursive multistep method. The results of iterations 1 and 2 were compared with the baseline, as shown in Table 6. The CV-RMSE, MAPE, MAE, and standard variance of daily CV-RMSE of iteration 2 were 45.15%, 45.39%, 40.39%, and 73.01% lower than the baseline, indicating that forecasting performance is greatly improved through early classification labeling. Because iteration 2 has the lowest CV-RMSE, it is compared with the baseline and other results in the following analysis. Besides, CV-RMSE is

Table 6
Metrics comparison for baseline and forecast results with different classification labels.

Methods	CV-RMSE (%)	MAPE (%)	MAE (kW)	STD of daily CV-RMSE (%)
Baseline	17.12	12.41	89.40	26.01
Early classification label iteration 1 (single-step method)	9.49	6.92	53.29	6.91
Early classification label iteration 2	9.39	7.00	53.43	7.02
True classification label	7.72	5.95	46.92	7.43

calculated for every forecasting day (hourly load from 9:00–17:00). The daily CV-RMSE of iteration 2 is compared with the baseline in Fig. 10. With early classification labeling, predicting models can address time series that cannot be explained by traditional labels. The forecasting results with true classification labels are listed in Table 6, indicating the upper limit improvement of early classification and that forecasting performance can be improved with increased accuracy of early classification. The later the early classification time is, the higher the accuracy it can yield. The early classification time in this case study was 8:00 am to improve the forecast result. The average CV-RMSE at different forecasting times is compared in Fig. 11. Without early classification labels, CV-RMSE for the first several hours is considerably high, and decreases over time. In contrast, the curve of CV-RMSE in iteration 2 nearly flattens, indicating that early classification labels can significantly improve forecasting accuracy from 9:00–13:00. The red line representing CV-RMSE with true classification labels exhibits the opposite trend of the baseline. The gap between the red line and iteration 2 narrows over time, indicating that the effectiveness of early classification diminishes over time.

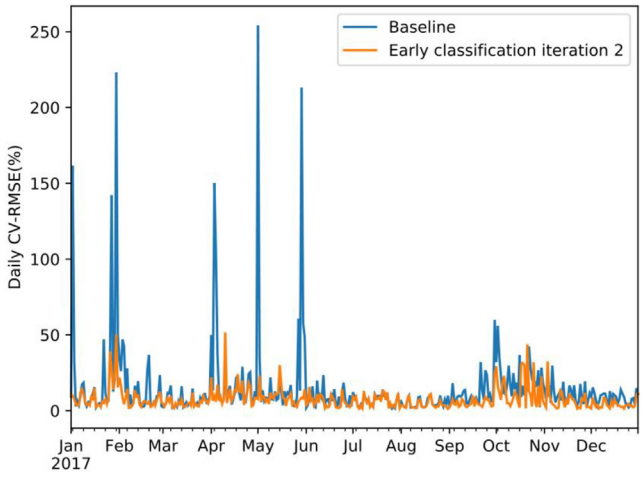

Fig. 10. Daily CV-RMSE results comparison.

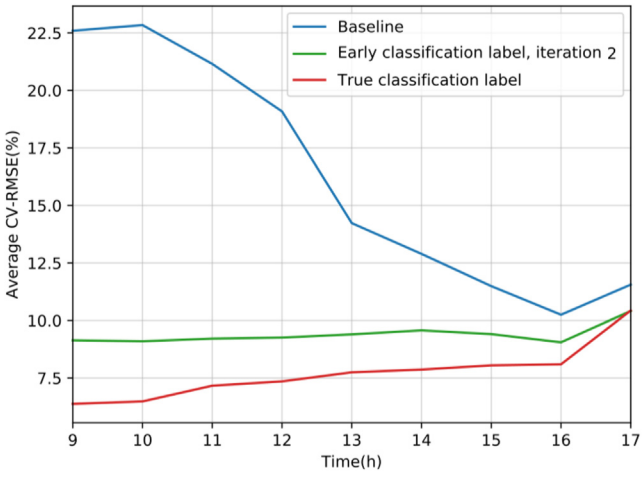

Fig. 11. Hourly average CV-RMSE on the test set (one year).

5. Conclusions

A literature review shows that traditional labeling methods for STLF usually consider day type and weather type, and the load pattern is neglected. To bridge this gap, this study presents a novel STLF framework that integrates time-series clustering and early classification techniques to supplement the feature engineering process. The LightGBM model is developed for STLF, as it is a powerful tool based on decision trees. To combine the strengths of single-step forecasting and recursive multistep forecasting, a hybrid multistep forecasting method is integrated into the proposed framework. To validate the performance of this framework, a case study was conducted based on hourly metered load data. From the results and discussion, the following conclusions are drawn.

- Different clustering methods can capture different subdistributions from the training set. The early classification accuracy differs for different clustering algorithms and the application of scaling. The accuracy is improved as more historical data are used for early classification. K-means (DTW) and kernel K-means demonstrate the best labeling accuracy, over 90% on the test set.
- LightGBM outperforms the other three data-driven models (i.e., SVM, ANN, and RF) in CV-RMSE, MAPE, and MAE. The ablation experiment shows that data processing techniques used in the proposed framework including data smoothing, time shift, and label encoding can improve the forecasting performance.
- The proposed hybrid multistep STLF method can improve CV-RMSE compared with single-step forecasting with only one additional iteration and less computational cost than the recursive multistep method. With early classification labels, the CV-RMSE, MAPE, MAE, and STD of daily CV-RMSE were 9.39%, 7.00%, 53.43%, and 6.91%, respectively, compared with 17.12%, 12,41%, 89.40%, and 26.01% for the baseline, respectively.
- Early classification can significantly improve STLF accuracy, especially during the first several hours from 9:00–13:00. When more data points are used in early classification, the predicted labeling results are more accurate and further improve the load STLF accuracies. Compared with forecasting results with true classification labels, the effectiveness of early classification diminishes over time.

Although the proposed framework demonstrates good performance in the case study, it has some restrictions. In the case study, building loads from 0:00–8:00 am were used to generate early classification labels; whole-day forecasting could not be implemented. In the future, past day loads will be considered for early classification labeling on STLF tasks that require whole-day forecasting. With the rapid development of big data and sub-metering techniques, early classification is a promising method

for application to more specific load pattern labeling, such as for lighting and HVAC.

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.

Acknowledgments

This work was funded by the China Postdoctoral Science Foundation (No. 2020 M681347).

References

[1] S.N. Fallah, M. Ganjkhani, S. Shamshirband, K. Chau, Computational Intelligence on Short-Term Load Forecasting: A Methodological Overview, Energies 12 (2019) 393, https://doi.org/10.3390/en12030393.
[2] S. Haben, G. Giasemidis, F. Ziel, S. Arora, Short term load forecasting and the effect of temperature at the low voltage level, Int. J. Forecast. 35 (4) (2019) 1469–1484, https://doi.org/10.1016/j.ijforecast.2018.10.007.
[3] Y. Sun, F. Haghighat, B.C.M. Fung, A review of the-state-of-the-art in data-driven approaches for building energy prediction, Energy Build. 221 (2020) 110022, https://doi.org/10.1016/j.enbuild.2020.110022.
[4] T. Hong, S. Fan, Probabilistic electric load forecasting: A tutorial review, Int. J. Forecast. 32 (3) (2016) 914–938, https://doi.org/10.1016/j.ijforecast.2015.11.011.
[5] D. Lee, Low-cost and simple short-term load forecasting for energy management systems in small and middle-sized office buildings, Energy Explor. Exploit. 39 (2) (2021) 637–656, https://doi.org/10.1177/0144598719900964.
[6] K. Jeong, C. Koo, T. Hong, An estimation model for determining the annual energy cost budget in educational facilities using SARIMA (seasonal autoregressive integrated moving average) and ANN (artificial neural network), Energy 71 (2014) 71–79, https://doi.org/10.1016/j.energy.2014.04.027.
[7] H. Nie, G. Liu, X. Liu, Y. Wang, Hybrid of ARIMA and SVMs for Short-Term Load Forecasting, Energy Procedia 16 (2012) 1455–1460, https://doi.org/10.1016/j.egypro.2012.01.229.
[8] Y. Kim, H. Son, S. Kim, Short term electricity load forecasting for institutional buildings, Energy Rep. 5 (2019) 1270–1280, https://doi.org/10.1016/j.egyr.2019.08.086.
[9] M. Cai, M. Pipattanasomporn, S. Rahman, Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques, Appl. Energy 236 (2019) 1078–1088, https://doi.org/10.1016/j.apenergy.2018.12.042.
[10] S. Hadri, Y. Naitmalek, M. Najib, M. Bakhouya, Y. Fakhri, M. Elaroussi, A Comparative Study of Predictive Approaches for Load Forecasting in Smart Buildings, Procedia Comput. Sci. 160 (2019) 173–180, https://doi.org/10.1016/j.procs.2019.09.458.
[11] N. Somu, G. Raman M R, K. Ramamritham, A deep learning framework for building energy consumption forecast, Renew. Sustain. Energy Rev. 137 (2021) 110591, https://doi.org/10.1016/j.rser.2020.110591.
[12] S. Du, T. Li, X. Gong, Y. Yang, S.J. Horng, Traffic flow forecasting based on hybrid deep learning framework, 2017 12th Int. Conf. Intell. Syst. Knowl. Eng. ISKE 2017 (2017) 1–6, https://doi.org/10.1109/ISKE.2017.8258813.
[13] S. Du, T. Li, Y. Yang, S.-J. Horng, Multivariate time series forecasting via attention-based encoder–decoder framework, Neurocomputing 388 (2020) 269–279, https://doi.org/10.1016/j.neucom.2019.12.118.
[14] C. Fan, F. Xiao, S. Wang, Development of prediction models for next-day building energy consumption and peak power demand using data mining techniques, Appl. Energy 127 (2014) 1–10, https://doi.org/10.1016/j.apenergy.2014.04.016.
[15] N.J. Johannesen, M. Kolhe, M. Goodwin, Relative evaluation of regression tools for urban area electrical energy demand forecasting, J Clean Prod 218 (2019) 555–564, https://doi.org/10.1016/j.jclepro.2019.01.108.
[16] Y. Wang, J. Chen, X. Chen, X. Zeng, Y. Kong, S. Sun, et al. Short-Term Load Forecasting for Industrial Customers Based on TCN-LightGBM. IEEE Trans Power Syst 2020:1–1. 10.1109/TPWRS.2020.3028133.
[17] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, et al., LightGBM: A Highly Efficient Gradient Boosting Decision Tree, Adv. Neural. Inf. Process Syst. 30 (2017) 3146–3154.
[18] T. Chen, C. Guestrin, XGBoost: A Scalable Tree Boosting, System (2016).
[19] P. Xue, Y. Jiang, Z. Zhou, X. Chen, X. Fang, J. Liu, Multi-step ahead forecasting of heat load in district heating systems using machine learning algorithms, Energy 188 (2019) 116085, https://doi.org/10.1016/j.energy.2019.116085.
[20] H. Zheng, J. Yuan, L. Chen, Short-Term Load Forecasting Using EMD-LSTM Neural Networks with a Xgboost Algorithm for Feature Importance Evaluation, Energies 10 (2017) 1168, https://doi.org/10.3390/en10081168.
[21] J. Paparrizos, L. Gravano, k-Shape: efficient and accurate clustering of time series, ACM SIGMOD Rec. 45 (1) (2016) 69–76, https://doi.org/10.1145/2949741.2949758.
[22] F. Petitjean, A. Ketterlin, P. Gançarski, A global averaging method for dynamic time warping, with applications to clustering, Pattern Recogn. 44 (3) (2011) 678–693, https://doi.org/10.1016/j.patcog.2010.09.013.
[23] A. Rajabi, M. Eskandari, M.J. Ghadi, L. Li, J. Zhang, P. Siano, A comparative study of clustering techniques for electrical load pattern segmentation, Renew. Sustain. Energy Rev. 120 (2020) 109628, https://doi.org/10.1016/j.rser.2019.109628.
[24] F. Blasques, M.H. Hoogerkamp, S.J. Koopman, I. van de Werve, Dynamic factor models with clustered loadings: Forecasting education flows using unemployment data, Int. J. Forecast. (2021), https://doi.org/10.1016/j.ijforecast.2021.01.026.
[25] I.S. Dhillon, Y. Guan, B. Kulis. Kernel k-means: spectral clustering and normalized cuts. Proc. Tenth ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., New York, NY, USA: Association for Computing Machinery; 2004, p. 551–6. 10.1145/1014052.1014118.
[26] M. Quintana, P. Arjunan, C. Miller, Islands of misfit buildings: Detecting uncharacteristic electricity use behavior using load shape clustering, Build. Simul. 14 (1) (2021) 119–130, https://doi.org/10.1007/s12273-020-0626-1.
[27] P. Mandal, T. Senjyu, N. Urasaki, T. Funabashi, A neural network based several-hour-ahead electric load forecasting using similar days approach, Int. J. Electr. Power Energy Syst. 28 (6) (2006) 367–373, https://doi.org/10.1016/j.ijepes.2005.12.007.
[28] Q. Mu, Y. Wu, X. Pan, L. Huang, X. Li, Short-term Load Forecasting Using Improved Similar Days Method, Power Energy Eng. Conf. 2010 (2010) 1–4, https://doi.org/10.1109/APPEEC.2010.5448655.
[29] F. Pargent. A Benchmark Experiment on How to Encode Categorical Features in Predictive Modeling n.d.:65.
[30] Transforming categorical features to numerical features - CatBoost. Documentation n.d. https://catboost.ai/docs/concepts/algorithm-main-stages_cat-to-numberic.html (accessed November 25, 2020).
[31] T. Proietti, A. Luati, Maximum likelihood estimation of time series models: the Kalman filter and beyond, Edward Elgar Publishing, Chapters, 2013, pp. 334–362.
[32] J. Wu, Q. Fang, Y. Xu, J. Su, F. Ma. Kalman filter based time series prediction of cake factory daily sale. 2017. 10.1109/CISP-BMEI.2017.8302108.
[33] P. Zarchan, H. Musoff, Fundamentals of Kalman Filtering: A Practical Approach, American Institute of Aeronautics and Astronautics, Incorporated, 2000.
[34] J. MacQueen, Some methods for classification and analysis of multivariate observations, The Regents of, the University of California, 1967.
[35] H. Sakoe, Dynamic-programming approach to continuous speech recognition, 1971 Proc Int Congr Acoust Bp, 1971.
[36] J. C, M.V.N.K. Prasad, S. Nickolas, G.R. Gangadharan, A novel probabilistic representational structures for clustering the time series data, Expert Syst. Appl. 145 (2020) 113119, https://doi.org/10.1016/j.eswa.2019.113119.
[37] G. Toni, Computing and visualizing dynamic time warping alignments in R: The dtw Package, J Stat Softw 31 (2009), https://doi.org/10.18637/jss.v031.i07.
[38] A. Dachraoui, A. Bondu, A. Cornuéjols. Early Classification of Time Series as a Non Myopic Sequential Decision Making Problem. In: Appice A, Rodrigues PP, Santos Costa V, Soares C, Gama J, Jorge A, editors. Mach. Learn. Knowl. Discov. Databases, vol. 9284, Cham: Springer International Publishing; 2015, p. 433–47. 10.1007/978-3-319-23528-8_27.
[39] Y. Chen, X. Xu, T. Koch, Day-ahead high-resolution forecasting of natural gas demand and supply in Germany with a hybrid model, Appl. Energy 262 (2020) 114486, https://doi.org/10.1016/j.apenergy.2019.114486.
[40] C.S. Bojer, J.P. Meldgaard, Kaggle forecasting competitions: An overlooked learning opportunity, Int. J. Forecast. 37 (2) (2021) 587–603, https://doi.org/10.1016/j.ijforecast.2020.07.007.
[41] S. Zhang, Y. Wang, Y. Zhang, D. Wang, N. Zhang, Load probability density forecasting by transforming and combining quantile forecasts, Appl. Energy 277 (2020) 115600, https://doi.org/10.1016/j.apenergy.2020.115600.
[42] L. Nespoli, V. Medici, K. Lopatichki, F. Sossan, Hierarchical demand forecasting benchmark for the distribution grid, Electr. Power Syst. Res. 189 (2020) 106755, https://doi.org/10.1016/j.epsr.2020.106755.
[43] B. Yang, L. Zhong, J. Wang, H. Shu, X. Zhang, T. Yu, L. Sun, State-of-the-art one-stop handbook on wind forecasting technologies: An overview of classifications, methodologies, and analysis, J Clean Prod 283 (2021) 124628, https://doi.org/10.1016/j.jclepro.2020.124628.
[44] A. Baliyan, K. Gaurav, S. Mishra, A review of short term load forecasting using artificial neural network models, Procedia Comput. Sci. 48 (2015) 121–125, https://doi.org/10.1016/j.procs.2015.04.160.
[45] Z. Tan, G. De, M. Li, H. Lin, S. Yang, L. Huang, Q. Tan, Combined electricity-heat-cooling-gas load forecasting model for integrated energy system based on multi-task learning and least square support vector machine, J. Clean Prod. 248 (2020) 119252, https://doi.org/10.1016/j.jclepro.2019.119252.
[46] A. Javed, B.S. Lee, D.M. Rizzo, A benchmark study on time series clustering, Mach Learn Appl. 1 (2020) 100001, https://doi.org/10.1016/j.mlwa.2020.100001.
[47] Y. Chen, P. Xu, Z. Chen, H. Wang, H. Sha, Y. Ji, Y. Zhang, Q. Dou, S. Wang, Experimental investigation of demand response potential of buildings: Combined passive thermal mass and active storage, Appl. Energy 280 (2020) 115956, https://doi.org/10.1016/j.apenergy.2020.115956.
[48] Y. Chen, P. Xu, Y. Chu, W. Li, Y. Wu, L. Ni, Y. Bao, K. Wang, Short-term electrical load forecasting using the Support Vector Regression (SVR) model to calculate

the demand response baseline for office buildings, Appl. Energy 195 (2017) 659–670, https://doi.org/10.1016/j.apenergy.2017.03.034.
[49] D.R. Landsberg, J.A. Shonder, K.A. Barker, J.S. Haberl, S.A. Judson, D.A. Jump, et al. ASHRAE Guideline 14-2014 n.d.:17.
[50] Y. Ding, Q. Zhang, T. Yuan, K. Yang, Model input selection for building heating load prediction: A case study for an office building in Tianjin, Energy Build. 159 (2018) 254–270, https://doi.org/10.1016/j.enbuild.2017.11.002.
[51] Z. Wang, T. Hong, M.A. Piette, Building thermal load prediction through shallow machine learning and deep learning, Appl. Energy 263 (2020) 114683, https://doi.org/10.1016/j.apenergy.2020.114683.
[52] K. Li, Z. Ma, D. Robinson, W. Lin, Z. Li, A data-driven strategy to forecast next-day electricity usage and peak electricity demand of a building portfolio using cluster analysis, Cubist regression models and Particle Swarm Optimization, J Clean Prod 273 (2020) 123115, https://doi.org/10.1016/j.jclepro.2020.123115.
[53] R.L. Thorndike, Who belongs in the family?, Psychometrika 18 (4) (1953) 267–276, https://doiorg/10.1007/BF02289263.
[54] P.J. Rousseeuw, Silhouettes: A graphical aid to the interpretation and validation of cluster analysis, J. Comput. Appl. Math. 20 (1987) 53–65, https://doi.org/10.1016/0377-0427(87)90125-7.