Information Sciences 429 (2018) 332–348

Contents lists available at ScienceDirect

Information Sciences

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

# CPLP: An algorithm for tracking the changes of power consumption patterns in load profile data over time

Imran Khan[a], Joshua Z. Huang[b], Zongwei Luo[a,*], M.A. Masud[b]

[a] Shenzhen Key Laboratory of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China
[b] College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China

ARTICLE INFO

*Article history:*
Received 29 January 2017
Revised 10 October 2017
Accepted 1 November 2017

*Keywords:*
Ensemble clustering
Power consumption patterns
Data stream
Load profile
Pattern change

ABSTRACT

In this paper, we propose a novel algorithm for tracking the Changes of Patterns in Load Profile (CPLP) data of factories. CPLP consists of two stages. The first stage is to cluster the load profiles in each time window and use the clusters to model the power consumption patterns. We propose a new ensemble clustering method to cluster the load profiles in consecutive time windows. It uses a hierarchical binary *k*-means algorithm to generate component clusterings and a new objective function to ensemble them to produce the final clustering. The second stage is to track the changes of patterns along the time windows. We propose a new method to detect the change of clusters from one window to the next one by using the distribution models of two related clusters in two neighboring windows. By using this method, we can link the clusters in the sequence of time windows to track the patterns. Experiments on synthetic and real-world load profile data have shown that the proposed algorithm was able to track the changes of power consumption patterns of different factory groups and identify the period of significant change, which are very useful for the smart grid applications.



## 1. Introduction

Load profile data provide information about power consumption behavior of selected customers over a given period. A load profile is a sequence of power consumption values measured by a smart meter at specified time intervals. A set of load profiles from selected customers can be used to analyze power consumption patterns and their changes over time. By tracking the changes of power consumption patterns from load profile data of factories, we can detect variations of power consumption in manufacturing processes, understand the work patterns of factories, and predict the productivity of factories. The information about the changes of power consumption patterns can find numerous applications in the power distribution industry such as determining optimal tariff rates, load forecasting, and energy demand management.

Clusters of load profiles in a given time window represent the power consumption patterns of users in the period. These patterns reflect power consumption behavior of users. Understanding the changes of power consumption patterns over time is very important for smart grid management. Since load profile data are streaming data, the approaches in concept drifting analysis of data streams can be adopted to analyze the changes of power consumption patterns in load profile data [6,30].

* Corresponding author. +8618319058199.
*E-mail addresses:* imran.khan@sustc.edu.cn (I. Khan), zx.huang@szu.edu.cn (J.Z. Huang), luozw@sustc.edu.cn (Z. Luo), masud@szu.edu.cn (M.A. Masud).

https://doi.org/10.1016/j.ins.2017.11.006
0020-0255/

In this paper, we present a concept drift approach for tracking the changes of power consumption patterns of manufacturing factories over time. The load profiles of selected factories are represented as a matrix where each row is a load profile of a factory, and each column is a set of power consumption measurements at a time slot. The matrix is vertically divided into a sequence of sub-matrices, each one representing the sub-load profiles of factories in a time window. Given this data representation, tracking the changes of power consumption patterns is achieved by finding the patterns in each time window and the changes of patterns along the sequence of windows. When solving this task, there are two challenges. The first one is to estimate the number of power consumption patterns and other one is to define the procedure for change detection [19].

This work is dedicated to proposing an algorithm for tracking the Changes of Patterns in Load Profiles (CPLP) of factories. The algorithm involves two steps. In the first step, we propose an ensemble clustering method to cluster the load profiles in each time window. A hierarchical binary $k$-means algorithm is used to generate multiple component clusterings. The output of this algorithm is a tree of clusters which represents the component clustering. We define a new stopping criteria for the tree generation process. By changing the input parameter value of tree generation, we generate multiple component clusterings from the same load profile data in a time window. We use a new objective function to ensemble the multiple component clusterings into a single clustering. In the second step, we track the changes of power consumption patterns along the time windows by using the obtained clustering results in all time windows. We define a new method to model two related clusters and check the change of distributions of two models. Two clusters are related if they have the maximum intersection of objects in neighboring windows. We define the survival pairs of clusters along time windows by obtained changes of distributions of related clusters to analyze the changes of power consumption patterns.

We present a series of experiments conducted on both synthetic and real data. Experiments on synthetic data have shown that the CPLP algorithm significantly outperformed the well-known state-of-the-art algorithms in almost all experiments. The experiments on real-world load profile data set demonstrated that the CPLP algorithm is able to track the changes of power consumption patterns of different factory groups and identify the period of significant change. The load profile data contains over 20,000 load profiles collected from manufacturing industries in Guangdong province of China in a period of six months in 2012. The power consumption measurements were collected at 15-minute interval. The results from this data are very important for the applications in the power distribution industry. The obtained results are useful to recommend good products to users and satisfy users demands as for as possible.

The rest of this paper is organized as follows. Section 2 introduces the load profile data and the research problem. Section 3 provides the related work. Section 4 presents the ensemble clustering method for load profile data. Section 5 presents the method for tracking the changes of power consumption patterns. Section 6 illustrates the experiments on synthetic data . Experimental results on real-world data are shown in Section 7. Some concluding remarks are given in Section 8.

## 2. Load profile data and research problem

The load profile data of $N$ factories and $J$ time slot measurements are represented as

$$X_{N\times J} = \begin{array}{c} \\ 1 \\ 2 \\ \cdot \\ \cdot \\ N \end{array} \begin{array}{c} \begin{array}{cccccc} Y_1 & Y_2 & . & . & Y_J \end{array} \\ \begin{bmatrix} x_{11} & x_{12} & . & . & x_{1J} \\ x_{21} & x_{22} & . & . & x_{2J} \\ . & . & . & . & . \\ . & . & . & . & . \\ x_{N1} & x_{N2} & . & . & x_{NJ} \end{bmatrix} \end{array} \tag{1}$$

where each row of $X$ represents a load profile of one factory and each column is a set of $N$ power consumption measurements at a time slot. The element $x_{ij}$ is the measurement of the $i$th factory at the $j$th time slot. By vector representation, $X$ can be represented as

$$X = \begin{bmatrix} X_1 \\ . \\ . \\ X_N \end{bmatrix} = [Y_1, \ldots, Y_J]$$

where $X_i$ is the vector of load profile of factory $i$ and $Y_j$ is the vector of consumption values of $N$ factories at time slot $j$. The sequence of $Y$ represents the data stream of $N$ factories.

We partition the sequence of $Y$ into a set of $B$ sub-sequences of equal length $h$ as $W_1, \ldots, W_B$ where $h$ is the number of time slots in the sub-sequence. The sub-sequence $W_{i_{N\times h}}$ is a sub-matrix of $X$ and is defined as

$$W_{i_{N\times h}} = \begin{bmatrix} w_{1,1} & . & . & . & w_{1,h} \\ . & . & . & . & . \\ . & . & . & . & . \\ w_{N,1} & . & . & . & w_{N,h} \end{bmatrix}$$

$X$ can be written as

$$X = [W_1 \quad W_2 \quad W_3 \quad W_4 \quad . \quad . \quad W_B] \tag{2}$$

Eq. (2) shows that matrix $X$ is partitioned into a set of sub-matrices of $B$ windows, which allows us to study power consumption patterns in the time windows and the changes of patterns in different windows.

**Problem Statement:** Given the sequence of sub-matrices of load profile data, find power consumption patterns in each window and the changes of power consumption patterns along the sequence of windows.

This problem can be solved in two stages, i.e., discovering the power consumption patterns in time windows and analyzing the changes of patterns along time windows.

## 3. Related work

Change detection in data streams has been widely investigated due to its broad application potential in all walks of science and technology, for example, fraud detection, market analysis, medical condition monitoring, and network traffic control [1]. Many studies have been applied to generate approaches for choosing, sampling, splitting, growing, and shrinking the distributions of the data in windows for optimal change detection [4]. For example, the Kullback–Leibler (KL) [9] divergence is used to measure the distance between the probability distributions of two different windows (old and recent) to detect possible changes. A well-known solution from statistics is to deal the two windows as two groups and apply Hotelling $t^2$ test to investigate whether the means of the two groups are the identical [13] or the multirank test for same medians [23]. There is a big collection of change detection techniques for observing a single variable [4]. The univariate change detection techniques are statically very sound but can not handle directly the multidimensional streaming data. Here, we are interested in detecting a change in the clusters of load profile multidimensional streaming data.

A large number of clustering techniques available with different underlying assumptions about the data [3,14]. Clustering techniques such as k-means [21], self-organizing maps (SOM) [29] and hierarchical clustering [2] have been used to analyze load profile data, for instance, identification of outliers and categorization of load profiles. Chicco et al. tested several clustering techniques to analyze a data set of 400 load profiles collected on weekdays [8]. Kim et al. used different clustering algorithms to analyze the load profile data of high-voltage power networks [18]. They found that the performance of a clustering algorithm depends on its capability to identify outliers or assign load profiles to a particular category. Among these methods, the hierarchical clustering technique was proved to be the most effective in assignment of load profiles to categories. Kwac et al. used the adaptive k-means algorithm to cluster the hourly power consumption data obtained from 220,000 homes in California with a total of 66 million individual (house-day) profiles [21]. k-means type algorithms require the number of clusters to be known, which is difficult to obtain from real data sets. Hierarchical clustering and SOM are not suitable for large data [5,32].

Clustering ensembles have emerged as an extensive method for improving robustness, stability, and accuracy of unsupervised classification solutions. Ensemble clustering integrates multiple component clusterings generated from samples of a given data set into a single clustering with a result which is usually much better than the results of individual component clusterings on the data set [16,17]. Due to this advantage, ensemble clustering becomes attractive in clustering complex data [15]. Three of the most widely used graph-based cluster ensemble algorithms are Cluster-based Similarity Partitioning Algorithm (CSPA), HyperGraph Partitioning Algorithm (HPGA), and Meta-CLustering Algorithm (MCLA) [28]. CSPA creates a similarity measure between data points but provides moderate performance and requires significant computation. HGPA generates the final ensemble clustering by using a hypergraph partitioning algorithm (HMETIS) to partition the hypergraph that represents the clusterings. The objective of this method is to create clusters that break the least number of hyperedges. MCLA works by breaking a group of related hyperedges into a single hyperedge. This provides better performance than HPGA and retains its low computational complexity. The graph-based ensemble methods are not suitable for the large data that contains large number of missing and noise values [12].

In the following sections, we present a new ensemble clustering method to discover power consumption patterns (clusters) and a change detection method to analyze the change of patterns.

## 4. Ensemble clustering method

In this section, we present a new ensemble clustering method to discover power consumption patterns modeled as clusters in each time window. We first describe a hierarchical binary $k$-means algorithm for generating the component clusterings. The advantage of this algorithm is that it does not require the number of clusters to be known in advance. Then, we present a new objective function for generating the clustering ensemble from the component clusterings.

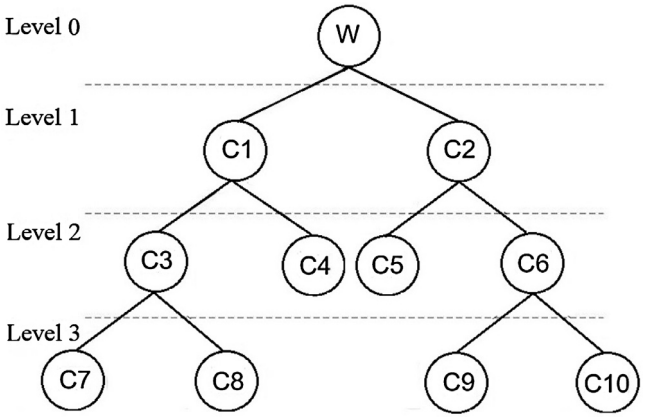


**Fig. 1.** A tree of clusters whose leaf nodes $C_4$, $C_5$, $C_7$, $C_8$, $C_9$ and $C_{10}$ form the component clustering.

### 4.1. Generation of component clusterings

The hierarchical binary *k*-means algorithm is used to generate the component clustering from a time window. Given data set $W_i$ in time window *i*, we first use the standard *k*-means algorithm to cluster $W_i$ into two clusters, make $W_i$ as the root of a tree and connect the two clusters to the root. Then, we use two stopping criteria to evaluate a cluster and determine whether the node is a leaf node. If the node is a leaf cluster (i.e., a power consumption pattern), clustering stops at that node. Otherwise, the cluster is further clustered into two sub-clusters using the standard *k*-means. This process continues until all leaf nodes are found and a tree of clusters is generated as shown in Fig. 1. The set of leaf clusters is taken as the component clustering.

In this hierarchical binary *k*-means clustering process, a critical step is to define the two stopping criteria for determining the leaf node. In the cluster tree, a node is a subset of load profiles. If the subset contains only one cluster, the node is a leaf node. Otherwise, it is an internal node which can be further clustered. To evaluate a leaf node, a mixture model is used to model the distribution of distances between the objects in the cluster and its centroid object. The mixture model with components of gamma distribution is defined as

$$f(x) = \sum_{i=1}^{c} \xi_i g_i(x, \alpha_i, \gamma_i), \quad x > 0 \tag{3}$$

where $x$ represents distances between the objects in the node (cluster) and its centroid object, $g_i(.)$ is the gamma distribution of the $i_{th}$ component with the shape and scale parameters $\alpha_i$ and $\gamma_i$, $\xi_i$ is the mixing coefficient and $c$ is the number of components in the mixture model.

Fitting this model, we estimate $c$ and use it to determine whether a node is a leaf node. If $c$ is equal to one, the node is a leaf node. Otherwise, we use the second criterion for determining a leaf node by computing the median value of the Probability Density Function (PDF) of best-fit selected mixture model by using Eq. (3). If the computed median value is smaller than the given threshold, the node is a leaf node. Otherwise, it is an internal node.

To solve the mixture model (3), we first use the fuzzy *c*-means (FCM) algorithm [31] to cluster the distance values into $c$ components and use the components to compute the initial values of parameters.

The density function of gamma distribution of component (cluster) $i$ is defined as

$$g_i(x, \alpha_i, \gamma_i) = \frac{\gamma_i^{\alpha_i}}{\Gamma(\alpha_i)} x^{\alpha_i - 1} e^{-\gamma_i x}, \quad 1 \le i \le c \tag{4}$$

where $\Gamma(\alpha)$ is the gamma function.

Given the set of distance values of component $i$, we use the maximum likelihood method [25] to compute the parameters $\alpha_i > 0$ and $\gamma_i > 0$ of gamma distribution (4). The likelihood function is defined as

$$L_{g_i}(\alpha_i, \gamma_i) = \frac{\gamma_i^{\alpha_i l_i}}{\Gamma^{l_i}(\alpha_i)} \prod_{x \in g_i} x^{(\alpha_i - 1)} e^{-\gamma_i \sum_{x \in g_i} x} \tag{5}$$

where $l_i$ is the number of distances in the $i_{th}$ component. The logarithm of Eq. (5) is given as

$$\log(L_{g_i}(\alpha_i, \gamma_i)) = l_i \alpha_i \log(\gamma_i) - l_i \log(\Gamma(\alpha_i)) + (\alpha_i - 1) \sum_{x \in g_i} \log(x) - \gamma_i \sum_{x \in g_i} x \tag{6}$$

We take the partial derivatives of Eq. (6) with respect to $\alpha_i$ and $\gamma_i$ and set the results equal to zero as

$$\frac{\partial}{\partial \alpha_i} log(L_{g_i}(\alpha_i, \gamma_i)) = l_i log(\gamma_i) - l_i \frac{\acute{\Gamma}(\alpha_i)}{\Gamma(\alpha_i)} + \sum_{x \in g_i} log(x) = 0$$

$$\Rightarrow -log(\gamma_i) + \frac{\acute{\Gamma}(\alpha_i)}{\Gamma(\alpha_i)} = \frac{1}{l_i} \sum_{x \in g_i} log(x) \tag{7}$$

and

$$\frac{\partial}{\partial \gamma_i} log(L_{g_i}(\alpha_i, \gamma_i)) = \frac{l_i \alpha_i}{\gamma_i} - \sum_{x \in g_i} x = 0$$

$$\Rightarrow \gamma_i = \frac{l_i \alpha_i}{\sum_{x \in g_i} x} \tag{8}$$

Combining Eq. (8) and (7) results in

$$log(\alpha_i) - \Psi(\alpha_i) = \log\left(\frac{1}{l_i} \sum_{x \in g_i} x\right) - \frac{1}{l_i} \sum_{x \in g_i} log(x) \tag{9}$$

where $\Psi(\alpha_i) = \frac{\acute{\Gamma}(\alpha)}{\Gamma(\alpha)}$ is the digamma function which can be approximated by [22].

$$\Psi(\alpha) = log(\alpha) - \frac{1}{2\alpha} - \frac{1}{12\alpha^2} - \frac{1}{120\alpha^4} + \ldots\ldots\ldots\ldots \tag{10}$$

Newton-Raphson method is used to solve (9) and get $\alpha_i$ which is then substituted into Eq. (8) to get $\gamma_i$. Using this method on all components produces the initial parameter values of all components in (3).

Given the initial parameter values, we use the EM algorithm [10] to solve the mixture model (3). EM algorithm involves two steps. The expectation step uses the current parameter values to create new clusters and the maximisation step updates the parameter values from the clusters generated in the expectation step using Eqs. (9), (8), and (10). Once EM algorithm converges, we get the $c$-component mixture model. Using this process, we generate multiple mixture models by changing $c$ from 1 to $c - max$ and evaluate them to select the best-fitted model. If the distance distribution contains only sparse values, the values, in general, are well fitted by one gamma component. If the distance distribution includes a mix of dense and sparse values, the values can normally be fitted by two distributions with two different gamma components. However, a large value of $c$-max can unnecessarily increase the execution time.

A variety of approaches have been proposed to estimate the number of components in a data set [26]. In our method, we use Akaike Information Criterion (AIC) [27] to evaluate the $c - max$ models produced by the EM algorithm and select the best fitted model with the minimum AIC value. AIC is defined as

$$AIC = 2P - 2log(L_k) \tag{11}$$

where $L_k$ is the maximum likelihood value of model $k$ and $P$ is the number of parameters in the mixture model. The mixture model is selected if its AIC value is minimal among other models. If the number of components $c$ in the selected model is 1, the node is a leaf node, and the clustering stops at that node. Otherwise, we compute the values of Probability Density Function (PDF) of the selected model using Eq. (3) and find the median value from the computed PDF values. If the median value is smaller than a given threshold $\sigma$, the node is a leaf node.

The median value is used instead of the mean value as a stopping criterion for the following consideration. A small median value indicates that the cluster has a dense distribution near the cluster center. The mean value is not used because if the shape parameter $\alpha$ of gamma distribution falls below zero, it represents an $L$-shaped distribution. In this case, the mean value is not an appropriate choice due to extreme values in the tail. On the other hand, a skewed distribution can be represented by a shape parameter $\alpha$ greater than one. In such case, the mean value of the distribution always appears in skewness.

The procedure for evaluation of a leaf node is given in Algorithm 1.

### 4.2. Ensemble clustering

Given a data set $W_i$ in time window $i$, multiple component clusterings can be generated from $W_i$ by the hierarchical binary $k$-means algorithm with different input values of the median threshold $\sigma$. Let $\lambda^1, \lambda^2, \ldots, \lambda^e$ be $e$ component cluster-

**Algorithm 1:** Evaluation of a tree node.

```
Input: Node, c-max, σ
Output: Leaf Node
Compute the distance values of Node;
Normalize distance values in the interval [0,1];
for c := 1 to c-max do
    if (c == 1) then
        Estimate the parameters of the gamma distribution parameters using (8), (9), and (10);
        Calculate AIC;
    else
        Initialize the EM algorithm using the FCM algorithm;
        Estimate the parameters of the mixture by EM algorithm using (8), (9), and (9);
        Calculate AIC;
Select the c-component model using (11) ;
if (c == 1) then
    Leaf Node ;
else
    Compute the PDF of c-component model using (3);
    if (Median(PDF) <= σ) then
        Leaf Node ;
    else
        internal node
Leaf node;
```

ings. We represent the cluster labels of $N$ objects in $e$ clusterings into matrix $E$ as

$$E = \begin{array}{c} \\ 1 \\ 2 \\ . \\ . \\ N \end{array} \begin{array}{c} \begin{array}{cccccc} \lambda^1 & \lambda^2 & . & . & \lambda^e \end{array} \\ \begin{bmatrix} l_{11} & l_{12} & . & . & l_{1e} \\ l_{21} & l_{22} & . & . & l_{2e} \\ . & . & . & . & . \\ . & . & . & . & . \\ l_{N1} & l_{N2} & . & . & l_{Ne} \end{bmatrix} \end{array} \tag{12}$$

where each row is an object (a factory) and each column is the set of cluster labels of $N$ objects in a clustering. Each cluster in the $e$ clusterings has a unique label. The set of unique cluster labels is listed as

$$L = \{ l_{11}, l_{21}, l_{12}, l_{32}, l_{13}, \ldots, l_{ke} \} \tag{13}$$

Taking two columns $\lambda^x$ and $\lambda^y$, we compute rand index (RI) [7] between clustering $\lambda^x$ and clustering $\lambda^y$ as

$$RI(\lambda^r, \lambda^y) = \frac{A + B}{A + B + C + D}. \tag{14}$$

where

- $A$ is the number of pairs of objects that are in the same set in $\lambda^x$ and in the same set in $\lambda^y$.
- $B$ is the number of pairs of objects that are in different sets in $\lambda^x$ and in different sets in $\lambda^y$.
- $C$ is the number of pairs of objects that are in the same set in $\lambda^x$ and in different sets in $\lambda^y$.
- $D$ is the number of pairs of objects that are in different sets in $\lambda^x$ and in the same set in $\lambda^y$.

The rand indeces of all pairs of clusterings are represented in matrix $R$ as

$$R = \begin{array}{c} \\ \lambda^1 \\ \lambda^2 \\ . \\ . \\ \lambda^e \end{array} \begin{array}{c} \begin{array}{cccccc} \lambda^1 & \lambda^2 & . & . & \lambda^e \end{array} \\ \begin{bmatrix} RI_{11} & RI_{12} & . & . & RI_{1e} \\ RI_{21} & R_{21} & . & . & RI_{2e} \\ . & . & . & . & . \\ . & . & . & . & . \\ RI_{e1} & RI_{e2} & . & . & R_{ee} \end{bmatrix} \end{array} \tag{15}$$

From $R$, we select a clustering as the reference clustering $\lambda^r$ by computing average of each row. The largest average value of the row $r$ give a reference clustering $\lambda^r$.

Given the column vector of the reference clustering $\lambda^r$ and the set of cluster labels $L$, we replace the cluster label of the first object in the reference clustering $\lambda^r$ with the first label in $L$ to generate a changed reference clustering $\lambda^{r'}$. Then, we compute the average of the rand indices between $\lambda^{r'}$ and all other clusterings as

$$RI_a(\Delta, \lambda^{r'}) = \frac{1}{(e-1)} \sum_{i=1}^{(e-1)} RI(\lambda^{r'}, \lambda^i). \tag{16}$$

where $\Delta$ is the set of clusterings excluding the reference clustering. If $RI_a(\Delta, \lambda^{r'}) > RI_a(\Delta, \lambda^r)$, we replace $\lambda^r$ with $\lambda^{r'}$ and $RI_a(\Delta, \lambda^r)$ with $RI_a(\Delta, \lambda^{r'})$. Otherwise, we keep both $\lambda^r$ and $RI_a(\Delta, \lambda^r)$ unchanged. We continue this process until all labels in $L$ are tested. After this iterative loop, the first object in the reference clustering is assigned a cluster label that maximizes $RI_a(\Delta, \lambda^r)$. The same iterative process is repeated until the last object is complete. Then, the process restarts from the first object of $\lambda^r$. In each loop on $N$ objects, the number of changes of object labels is recorded. The iterative process stops when no object changes its cluster label after a loop on $N$ objects. The reference clustering $\lambda^r$ is the final clustering ensemble.

The procedure for ensemble clustering is summarized in Algorithm 2.

**Algorithm 2:** Ensemble clustering.

**Input:** $\lambda^1, \lambda^2, \ldots, \lambda^e$
**Output:** $\lambda^f$
Select $\lambda^r$ by computing the matrix $R$;
$L = \cup_{i\in\{1,2,\ldots,e\}}\lambda^{(i)}$ ;
$kl$ = Length of $L$ ;
**while** *(flag == 1)* **do**
  flag = 0 ;
  **for** *i := 1 to length($\lambda^r$)* **do**
    Compute $RI_a(\Delta, \lambda^r)$ using $\lambda^r$ and $\lambda^1, \lambda^2, \ldots, \lambda^e$ with Equation (16) ;
    **for** *j := 1 to kl* **do**
      Temp = $\lambda^r[i]$;
      $\lambda^r[i] = L[j]$ ;
      Compute $RI_a(\Delta, \lambda^{r'})$ using $\lambda^{r'}$ and $\lambda^1, \lambda^2, \ldots, \lambda^e$ with Equation (16) ;
      **if** $(RI_a(\Delta, \lambda^{r'}) > RI_a(\Delta, \lambda^r))$ **then**
        $RI_a(\Delta, \lambda^r) = RI_a(\Delta, \lambda^{r'})$ ;
        $\lambda^r = \lambda^{r'}$ ;
        flag = 1;
      **else**
        $\lambda^r[i]$ = Temp ;
$\lambda^f = \lambda^r$.

## 5. Method for tracking the changes of power consumption patterns

Given the sequence of sub-matrices $W_1, W_2, \ldots, W_B$ in $B$ consecutive time windows, we can use the ensemble clustering method discussed in Section 3 to generate $B$ ensemble clusterings $\lambda^1$, $\lambda^2$, ... , $\lambda^B$, one for each time window. Let $\lambda^{fi}$ and $\lambda^{fi+1}$ be two ensemble clusterings of load profiles in two neighboring time windows $W_i$ and $W_{i+1}$, each consisting of a set of clusters. For any cluster $C_1$ from $\lambda^{fi}$, we can find a set of clusters in $\lambda^{fi+1}$ which have intersections of load profiles with $C_1$, as shown in Fig. 2. We take the cluster $C_2$ from $\lambda^{fi+1}$ which has the maximum intersection with $C_1$ and make them as the related pair of clusters in $W_i$ and $W_{i+1}$. In this way, all clusters in $\lambda^{fi}$ can find their corresponding related clusters in $\lambda^{fi+1}$.

For each pair of related clusters ($C_1$, $C_2$) in two neighboring time windows $W_i$ and $W_{i+1}$, we investigate the change of distributions of these two clusters from $W_i$ to $W_{i+1}$. Again, we use the mixture model (3) to model the distance distributions of the two clusters and Algorithm 1 to find the fitted models for $C_1$ and $C_2$. For each model with $c$ gamma components, the centers $\mu_{1\le i\le c}$ of components are computed. Then, the average of the squared Mahalanobis distances between the centers $\mu_i$ of $C_1$ and all load profiles of $C_2$ is computed as follows

$$M(C_1, C_2) = \frac{1}{c} \sum_{i=1}^{c} \left( \frac{1}{N_2} \sum_{x\in C_2} (x-\mu_i)^T \sum^{-1} (x-\mu_i) \right) \tag{17}$$

where $c$ is the number of components in $C_1$, $\sum^{-1}$ is the intra components covariance matrix of $C_1$ and $N_2$ is the number of load profiles in $C_2$. In the same way, we can compute $M(C_2, C_1)$. We define the change detection function as

$$CD(C_1, C_2) = \frac{max\{M(C_1, C_2), M(C_2, C_1)\}}{H} \tag{18}$$

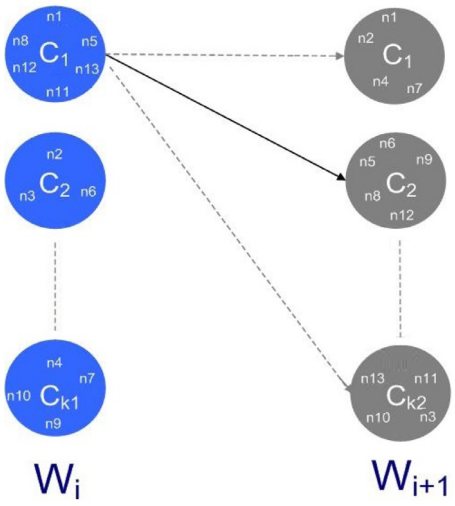


**Fig. 2.** $C_i$ represents the $i$-th cluster in a time window and $n_j$ represents the $j$th load profile (instance) in a cluster. Cluster $C_1$ in time window $W_i$ on the left intersects with clusters in time window $W_{i+1}$ on the right. $C_2$ has the maximum intersection (three load profiles $n_5$, $n_8$, and $n_{12}$) with $C_1$. Therefore, $C_1$ and $C_2$ are related clusters in $W_i$ and $W_{i+1}$.

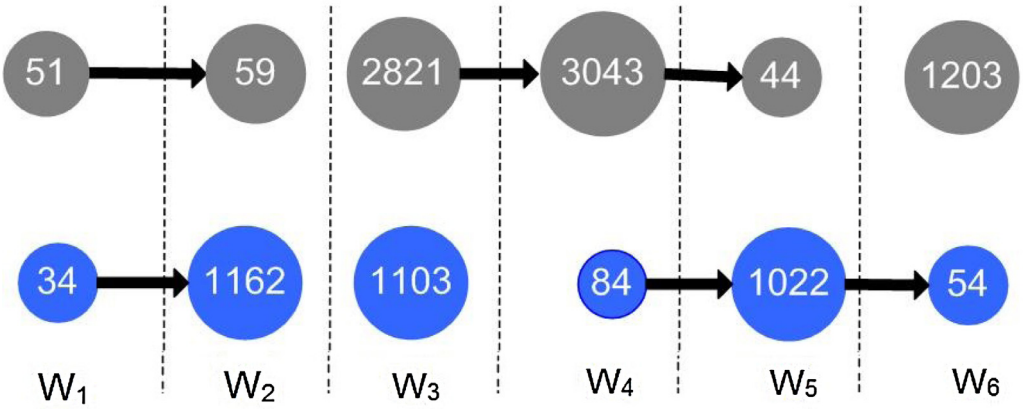


**Fig. 3.** Examples of survived clusters in six time windows.

where $H$ is the number of intersected load profiles in $C_1$ and $C_2$. A large $CD$ value indicates a significant change of distributions between $C_1$ and $C_2$. A small $CD$ value indicates that $C_1$ and $C_2$ have a similar distribution and $C_2$ is a survival of $C_1$. If window $W_{i+1}$ contains two clusters that are related to $C_1$ and the two pairs have the same maximum intersections, we use one of the pairs that have a smaller $CD$ value. Using (18) with a user-given threshold $\tau$, we can find all survived pairs of clusters between all neighboring time windows from all pairs of related clusters.

After all survived pairs of clusters in all neighboring time windows are found, we link the pairs of survived clusters along the time windows to obtain the chains of survived clusters in $[W_1, W_2, \ldots, W_B]$. Fig. 3 shows examples of survived clusters in six time windows. The numbers in the circles are the numbers of load profiles in clusters. We can see two two-month patterns ($W_1 - W_2$) with two survived clusters, two three-month patterns ($W_3 - W_4 - W_5, W_4 - W_5 - W_6$) of three survived clusters and two single clusters ($W_3$, $W_6$). Long chains reflect stable power consumption patterns and individual clusters indicate abrupt power consumption patterns.

To investigate the change activities of power consumption patterns in a given time window, we define two measures to evaluate the emerging rate (ER) and the fading rate (FR) of survived clusters as follows.

$$FR = \frac{1}{k_1}\sum_{i=1}^{k_1} CD\langle C_i, C_j\rangle \quad 1 \le j \le k_2 \qquad if \quad |C_i| > |C_j|$$

$$ER = \frac{1}{k_1}\sum_{i=1}^{k_1} CD\langle C_i, C_j\rangle \quad 1 \le j \le k_2 \qquad if \quad |C_i| < |C_j| \tag{19}$$

where $C_i \in W_i$ and $C_j \in W_{i+1}$ are two related clusters, $k_1$ is the number of clusters in $W_i$, $k_2$ is the number of clusters in $W_{i+1}$, and $|C_i|$ represents the number of load profiles (instances) in cluster $C_i$.

Algorithm 3 is intended to track the changes of power consumption patterns in load profile data. It includes all the required stages, namely ensemble clustering, change detection, survived cluster chaining and evaluation of the change in time windows. The algorithm takes a sequence of load profile sub-matrices, and three parameters $\tau$, $\sigma$, $c-max$ as inputs. The outputs of this algorithm include clusters in each time window, the set of pairs of survived clusters, the set of chains of survived clusters, and the rates of fading and emerging of clusters in each time window. The algorithm consists of two main steps. The first step is to generate the ensemble clustering from each sub-matrix. The second step is to compute all pairs of survived clusters in all neighboring windows, find the chains of survived clusters except for the last window and calculate the rates of fading and emerging of clusters in all time windows except for the first window.

**Algorithm 3:** CPLP Algorithm.

**Input:** $\{W_1, W_2, \ldots, W_B\}$, $\tau$, $\sigma$, $c-max$
**Output:** CD, FR, ER, $\{\lambda^{f_1}, \lambda^{f_2}, \lambda^{f_3}, \ldots, \lambda^{f_B}\}$
**for** *i := 1 to B* **do**
  **for** *j := 1 to e* **do**
    Set the values of $\sigma_j$ and $c-max_j$ ;
    Generate a cluster tree by applying binary *k*-means on $W_i$ using Algorithm 1 with $\sigma_j$ and $c-max_j$ ;
    Obtain a clustering $\lambda^j$ from the cluster tree ;
  Apply Algorithm 2 on $\lambda^1, \lambda^2, \lambda^3, \ldots, \lambda^e$ to obtain $\lambda^{f_i}$ ;
**for** *i := 1 to B − 1* **do**
  Select the *k* pairs of related clusters using $\lambda^{f_i}$ and $\lambda^{f_{i+1}}$;
  **for** *j := 1 to k* **do**
    Compute the change *CD*1 between *j*-th pair of related clusters using (18);
    **if** *(CD*1 $<\tau$*)* **then**
      Link the pair of related clusters;
      Append *CD*1 in *CD* ;
      Compute the fading rate *FR*1 or the emerging rate *ER*1 of clusters in $W_{i+1}$ using (19);
  Append *FR*1 in *FR*;
  Append *ER*1 in *ER*;
CD, FR, ER, $\{\lambda^{f_1}, \lambda^{f_2}, \lambda^{f_3}, \ldots, \lambda^{f_B}\}$

## 6. Experiments on synthetic data

The motivation for development of the CPLP algorithm is to cluster noisy power consumption data and compute the change between two data distributions. Synthetic data is often used to validate a clustering algorithm [24]. To better understand the properties of the CPLP algorithm, synthetic data with different structures containing data noise were first used to investigate the performance of ensemble clustering method on clustering accuracy in comparison with other ensemble clustering algorithms, the performance of change detection method in comparison with other change detection methods, and the impact of input parameters of the CPLP algorithm.

### 6.1. Clustering ensemble analysis

To investigate the performance of first part (ensemble clustering method) of the CPLP algorithm, we generated six synthetic data sets, each consisting of five hundred dimensions and three clusters. Each cluster was composed of fifty data points in five hundred dimensions with a normal distribution. We set the means of the three clusters in the main dimensions as 1, 5, 3, respectively and the same variance 1 for all clusters. The three clusters were generated independently and merged into one data set. Then, some noise features with a uniform distribution between 0 and 100 were added to the data set to replace the same number of features with cluster distributions. We generated six data sets by adding different percentages of noise features, i.e. 0.05, 0.1, 0.2, 0.5, 0.6, and 0.7, respectively. The introduction of noise features to the data sets increases the difficulty of clustering. The more the noise features were included in a dataset, the more difficult to cluster the data.

To evaluate the performance of hierarchical binary *k*-means algorithm in ensemble clustering, we choose two traditional clustering methods, *k*-means and hierarchical (complete) clustering algorithms. We used each method with different input parameters to generate component clusterings using each synthetic data sets. We generated ten component clusterings from each data set with each method. To integrate the component clusterings into clustering ensembles, we used three well-known consensus functions and one proposed ensemble method. Combining three clustering methods with four ensemble methods, we experimented twelve ensemble clustering techniques.

**Table 1**
Accuracies of clustering ensemble results of synthetic data sets with different percentages of noise features.

| Methods | Accuracy | | | | | |
|---|---|---|---|---|---|---|
| | Noise Features Ratio | | | | | |
| | 0.05% | 0.1% | 0.2% | 0.5% | 0.6% | 0.7% |
| HKIR | **0.92** | **0.90** | **0.90** | **0.89** | 0.87 | **0.87** |
| KMIR | 0.90 | 0.89 | 0.89 | 0.88 | **0.88** | 0.85 |
| HCIR | 0.89 | 0.89 | 0.87 | 0.85 | 0.85 | 0.83 |
| HKCSPA | **0.88** | 0.86 | 0.85 | **0.83** | **0.81** | **0.80** |
| KMCSPA | 0.86 | **0.87** | **0.86** | 0.82 | 0.80 | 0.80 |
| HCCSPA | 0.86 | 0.85 | 0.81 | 0.80 | 0.78 | 0.79 |
| HKMCLA | 0.90 | **0.91** | 0.88 | **0.85** | **0.86** | **0.86** |
| KMMCLA | **0.91** | 0.90 | **0.89** | 0.84 | 0.84 | 0.84 |
| HCMCLA | 0.89 | 0.88 | 0.85 | 0.82 | 0.83 | 0.83 |
| HKHGPA | 0.83 | 0.84 | 0.82 | 0.80 | 0.82 | 0.83 |
| KMHGPA | **0.84** | **0.82** | **0.83** | **0.82** | **0.80** | **0.80** |
| HCHGPA | 0.83 | 0.81 | 0.80 | 0.79 | 0.77 | 0.76 |
| KMAvg | 0.82 | 0.82 | 0.80 | 0.76 | 0.74 | 0.73 |

The component clusterings generation methods are hierarchical binary *k*-means denoted as HK, *k*-means denoted as KM, and hierarchical clustering (complete) denoted as HC. The four ensemble methods similarity-based consensus function denoted as CSPA [28], hyper graph-based consensus function denoted as HGPA [28], meta cluster-based consensus function denoted as MCLA [28], and the proposed rand index based ensemble method denoted as RI. The combinations of three component clusterings generation methods and four ensemble methods result in twelve ensemble clustering techniques denoted as HKIR, HCIR, KMIR, HKCSPA, HCCSPA, KMCSPA, HKMCLA, HCMCLA, HKHGPA, HCHGPA, and KMMCLA respectively. The baseline *k*-means clustering algorithm was applied on each data set several times with different initialization and average results were recorded. The obtained average results are denoted as KMAvg.

Table 1 shows the accuracies of clustering ensembles from each synthetic data set with different ratio of noise features by three component clusterings generation methods in combination of four ensemble methods. The best results of the four ensemble methods are marked in dark. The best result in all techniques for the same data set (with different noise features) is underlined. We can see that all ensemble results are better than the average of the data set clustering results. This indicates that the ensemble clustering performed much better on the synthetic data sets. In the ensemble results, the ensemble methods obtained a similar performance in clustering accuracy when the data sets contain less noise features. When the data sets contain more noise features, the accuracy of the ensembles clustering techniques that include HGPA method dropped significantly but accuracies of the results from other techniques reduced slightly. However, the results from the proposed ensemble clustering method (HKIR) method are more stable than the results of the other ensemble clustering techniques. These results clearly show that the proposed ensemble method (IR) is superior to the other ensemble methods (CSPA, HGPA, and MCLA) and the hierarchical binary *k*-means (HK) algorithm is useful to generate component clusterings without sacrificing clustering diversity at a large scale.

Generation of component clusterings is essentially focused on the diversity of component clusterings [20]. To investigate the quality and diversity of component clusterings, we used normalized mutual information (NMI) to measure the diversity [11]. To investigate the relationship between diversity of component clusterings and performance of clustering ensembles, we computed the NMI between each pair of component clusterings and the average of the NMI values for each ensemble clustering. The average NMI indicates the diversity of the component clusterings in the clustering ensemble. The smaller the average NMI, the more diverse the component clusterings. For each component clustering, we also measured the NMI value between the component clusters and the true labels of objects. The average NMI shows the quality of component clusterings in a clustering ensemble. The higher the average NMI, the better the clustering quality.

Fig. 4 shows the relations between diversity and quality of component clusterings generated by hierarchical binary *k*-means, *k*-means, and hierarchical (complete) clustering algorithms. The vertical axis is the diversity, and the horizontal axis is the quality. Each point in the figure was measured from a pair of component clusterings created with each clustering algorithm on the synthetic data set that contains 70% noise features. The vertical line in each figure represents the NMI value between the clustering results of an ensemble produced with the proposed ensemble method (IR) from ten component clusterings and the true labels. As it can be seen in the three figures, the hierarchical binary *k*-means algorithm generated higher quality component clusterings without sacrificing the diversity at a large scale.

### 6.2. Change detection analysis

We have demonstrated that the proposed change detection method outperforms the well-known change detection methods including, Hotelling $t^2$ test [13], Multirank test [23], and Kulback–Leibler (K–L) distance [9]. The reason behind the proposed finding is that the gamma mixture is usually a more reasonable model than the single Gaussian assumed for the

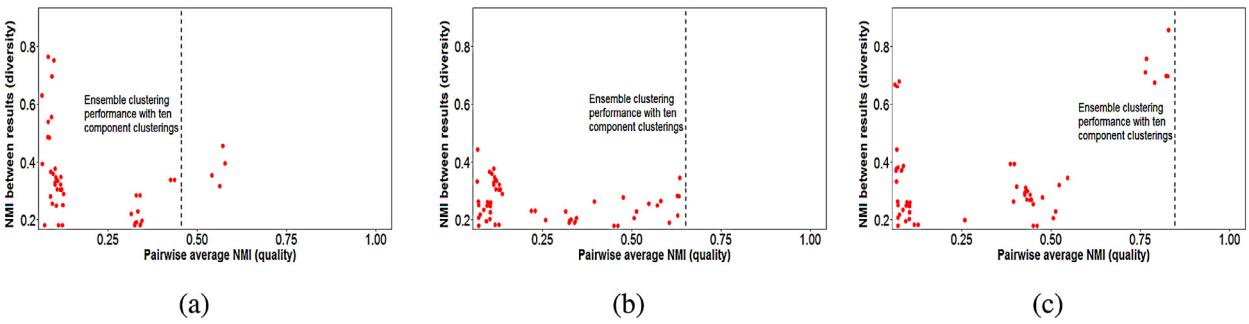


**Fig. 4.** Illustration of the relationships between the diversities and qualities of ten component clusterings which are computed from synthetic data set (70% noise features) with three three clustering algorithms respectively.

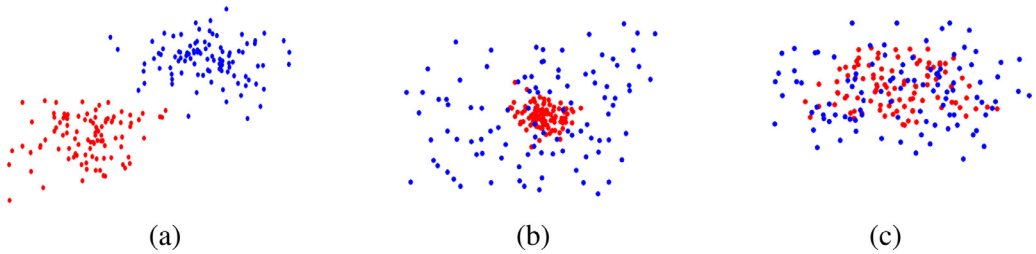


**Fig. 5.** Examples of synthetic data in windows $W_1$ (red), $W_2$ (blue) for the comparison of change detection criteria. (a) Translation (b) Random linear transformation (c) Change of variance. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

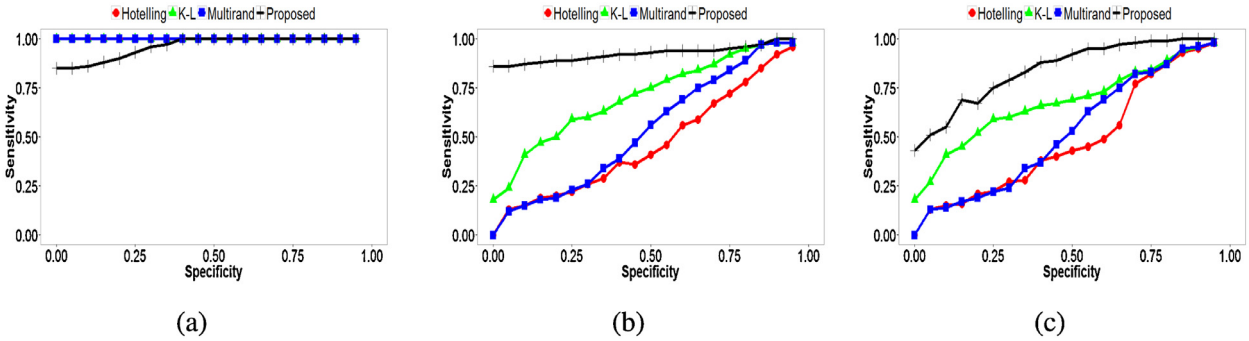


**Fig. 6.** Performance of four criteria on three types of changes in synthetic data. (a) Translation (b) Random linear transformation (c) Change of variance.

Hotelling test. The Hotelling criterion is not being able to detect the change in the variance of the data, while the proposed criterion is equipped to do so. The Multirank test relates the medians of the distributions in the two windows, but do not consider the changes in the variance. The K-L distance is not a true distance between distributions since it is not symmetric and does not satisfy the triangle inequality.

To support the proposed criterion, we provided a simulation example. We considered one hundred sampled data records as a window $W_1$ from a five-dimensional normal distribution with mean zero and a diagonal co-variance matrix $S$. The variances of the dimensions (features) were sampled from the positive half of the standard normal distribution. We denoted this distribution by $D_1$. The window $W_2$ was sampled once from $D_1$ (with the same covariance matrix) with a specific change type. Fig. 5 shows the examples of $W_1$ and $W_2$ with three types of changes in the space of the first two dimensions. Fig. 5(a) represents the translation change. In this change, we sampled a new mean from two-dimensions (2d), where d $\sim$ N(0,1). The window $W_2$ was sampled from $D_1$ with the new mean. Fig. 5(b) shows the random linear transformation. We generated a random matrix $R$ of size $5 \times 5$ by sampling each element from N(0,1). Then, we sampled $W_2$ from $D_1$ and multiply all objects by $R$. Fig. 5(c) represents the change of variance. We generated $W_2$ from a normal distribution with mean zero and co-variance matrix $S \times D$, where $D$ is the diagonal matrix with diagonal elements sampled from three-dimensions (3d) and d $\sim$ N(0,1).

Fig. 6 shows the calculation of four change detection criteria for the three changes. The curves show the sensitivity of methods (proportion of true changes correctly detected) against the specificity, where specificity is the true negative rate of

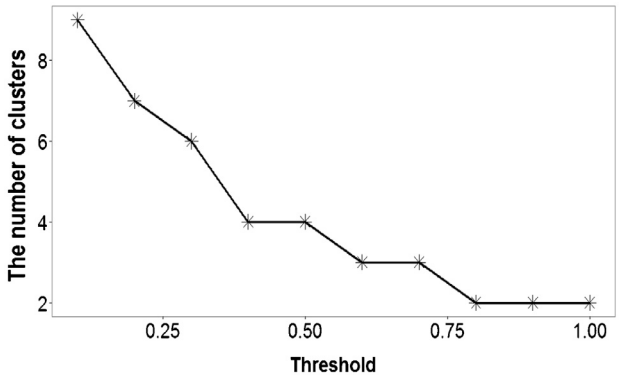


**Fig. 7.** The impact of $\sigma$ on generation of components clusterings.

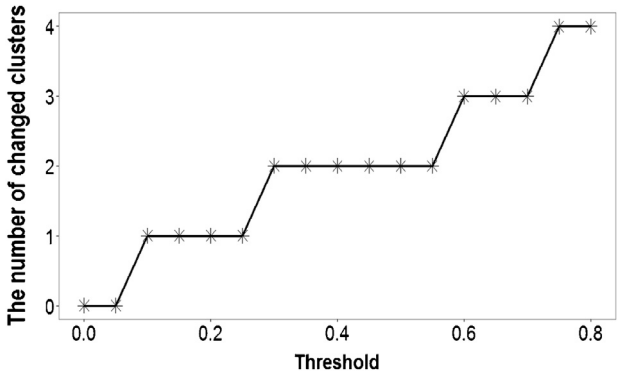


**Fig. 8.** The impact of $\tau$ on the number of changed clusters.

the methods (proportion of correctly labeled windows with no change). The proposed method and the K-L distance were used with three clusters. The thresholds were not applied because the evaluation was done by using the raw values. The graphs show the behavior of the four criteria. As it can be seen in the figure, the two criteria Hotelling and Multirank are superior to proposed and K-L for the mean change type, while the two other changes favor the proposed criterion. This is the reason we consider the proposed criterion for the experiments reported in the next section.

### 6.3. Impact of input parameters

In the CPLP algorithm, we need to specify the input parameters $\sigma$ and $\tau$. We use $\sigma$ to generate the component clusterings and $\tau$ to determine whether the two related clusters in consecutive windows meet the requirement of distribution change.

To investigate $\sigma$, we used a synthetic data set that contains one hundred data points in five hundred dimensions with a normal distribution. We ran the hierarchical binary $k$-means algorithm on the data set to generate a component clustering by building a cluster tree. We generated multiple cluster trees by using a unique threshold $\sigma$ for each one to produce different component clusterings. We generated multiple cluster trees by running the hierarchical binary $k$-means algorithm with different values of $\sigma$ and counted the number of clusters in each cluster tree which shows the diversity of a component clustering.

Fig. 7 shows the relationships between the number of clusters and the value of $\sigma$. We can see that the number of clusters decreases as $\sigma$ increases. When $\sigma$ is greater than 0.8, the number of clusters drops to two, which means no change in the number of clusters was identified. Therefore, $\sigma$ cannot be greater than 0.8 in this data set. To generate the better component clusterings of this data set, we set $\sigma$ in the value range of parameter $\sigma$ as [0.4,0.7].

To investigate $\tau$, we used two synthetic data set that contains one hundred data points in five hundred dimensions with a normal distribution. We added the percentage of noise features in two data sets 50% and 70%, respectively. Then, we applied the proposed ensemble clustering method on both data sets to generate clusterings. The change detection method was used to compute the change between related clusters from two data sets. The computed changes between related clusters of data sets are analyzed with different values of $\tau$. Fig. 8 shows the relationships between the number of changed clusters and the value of $\tau$. As shown in the figure, the number of changed clusters increases as $\tau$ increases. It is observed that when $\tau$ is less than 0.05, the number of changed clusters drops to zero, which means no cluster change was identified. Therefore,

**Table 2**
Summary of power consumption patterns in six months.

| Pattern types | January | February | March | April | May | June | Total |
|---|---|---|---|---|---|---|---|
| Six-month pattern | 9 | – | – | – | – | – | 9 |
| Five-month pattern | 3 | 3 | – | – | – | – | 6 |
| Four-month pattern | – | – | 4 | – | – | – | 4 |
| Three-month pattern | 1 | – | – | 2 | – | – | 3 |
| Two-month pattern | 1 | – | 1 | – | 3 | – | 5 |
| One-month pattern | 3 | 4 | 6 | 5 | 6 | 5 | 29 |
| Sum | | | | | | | 56 |

$\tau$ cannot be less than 0.05 in this data. To reveal cluster changes with high performance, the $\tau$ was set as default that is applicable for all consecutive time windows.

## 7. Experimental results on real-world data

In this section, we conducted a series of experiments on real-world load profile data to verify the performance of the CPLP algorithm. At first, a description is given to the load profile data and experiment settings. Then, the experiment results are provided with a detailed discussion.

### *7.1. Load profile data*

The load profile data used in experiments contain more than 20,000 load profiles collected from manufacturing factories in Guangdong province of China in the period from January to June, 2012. The power consumptions of each factory were measured at a 15-minute interval with a smart meter. For the measurements of a given day, the time range is from 00:00 to 23:59, which is normally used as the basic time window. Obviously, the load profile of the day contains 96 measurements.

To investigate changes of power consumption patterns over time, we used one month as the size of time window. The load profile data is divided into 6 time windows for the period of six months. We also separate workdays and weekends and removed the load profile data on Saturdays and Sundays and some public holidays because the power consumptions were different on these days and they were not concerned from production point of view.

The real load profile data had missing values, noise and anomalies. We identified the anomaly load profiles using visualization and removed them from the data. We aggregated the load profiles of Mondays in a month together by taking the average of corresponding time slot measurements to smooth the noise and remove the effect of missing values. In the same way, we also aggregated the load profiles of other workdays. Adding all workdays together, we obtained the load profile data in a month with 480 measurements, i.e., five days measurements. A one month workday load profile had approximate 1920 measurements.

### *7.2. Experimental settings*

The processed six-month load profile data were divided into six sub-matrices in six time windows corresponding to the months from January to June. Using CPLP to analyze this data, we set the value range of parameter $\sigma$ as [0,1], the value range of parameter $c - max$ as [2–5], the value of parameter $\tau$=0.9. For ensemble clustering, we generated ten component clusterings in each time window.

### *7.3. Results of changes of power consumption patterns*

A result produced by CPLP from the load profile data is shown in Fig. 9. Each column lists the clusters in the corresponding time window. The clusters are represented by circles. The number in a circle is the number of load profiles in the cluster. The clusters with double-line circles were emerging clusters in the corresponding time window. The clusters with small circles are the those that faded out in the next window. Following the link between clusters in the neighboring windows, we can find the chains of clusters in the six-month period.

Table 2 shows the summary of power consumption patterns in Fig. 9. It can be seen from the Table that more than half of the patterns (29 out of 56) are one-month patterns. These patterns were short-lived. However, there are nine six-month patterns, six five-month patterns, and four four-month patterns, which lasted for more than half of the six month period. Apparently, these power consumption patterns were stable patterns.

To investigate the characteristics of the long-live patterns, five six-month patterns, whose starting clusters are $C_{1,1}$, $C_{2,1}$, $C_{4,1}$, $C_{8,1}$, and $C_{6,1}$. The first index is the row number and the second index indicates the column number in Fig. 9. We also added one five-month pattern whose starting cluster is $C_{1,2}$. For each selected pattern, we computed the average of load profiles in the clusters and plotted the average load profiles of the six patterns in Fig. 10. The vertical axis is the average power consumption value (kW). The horizontal axis is time in a half hour interval.

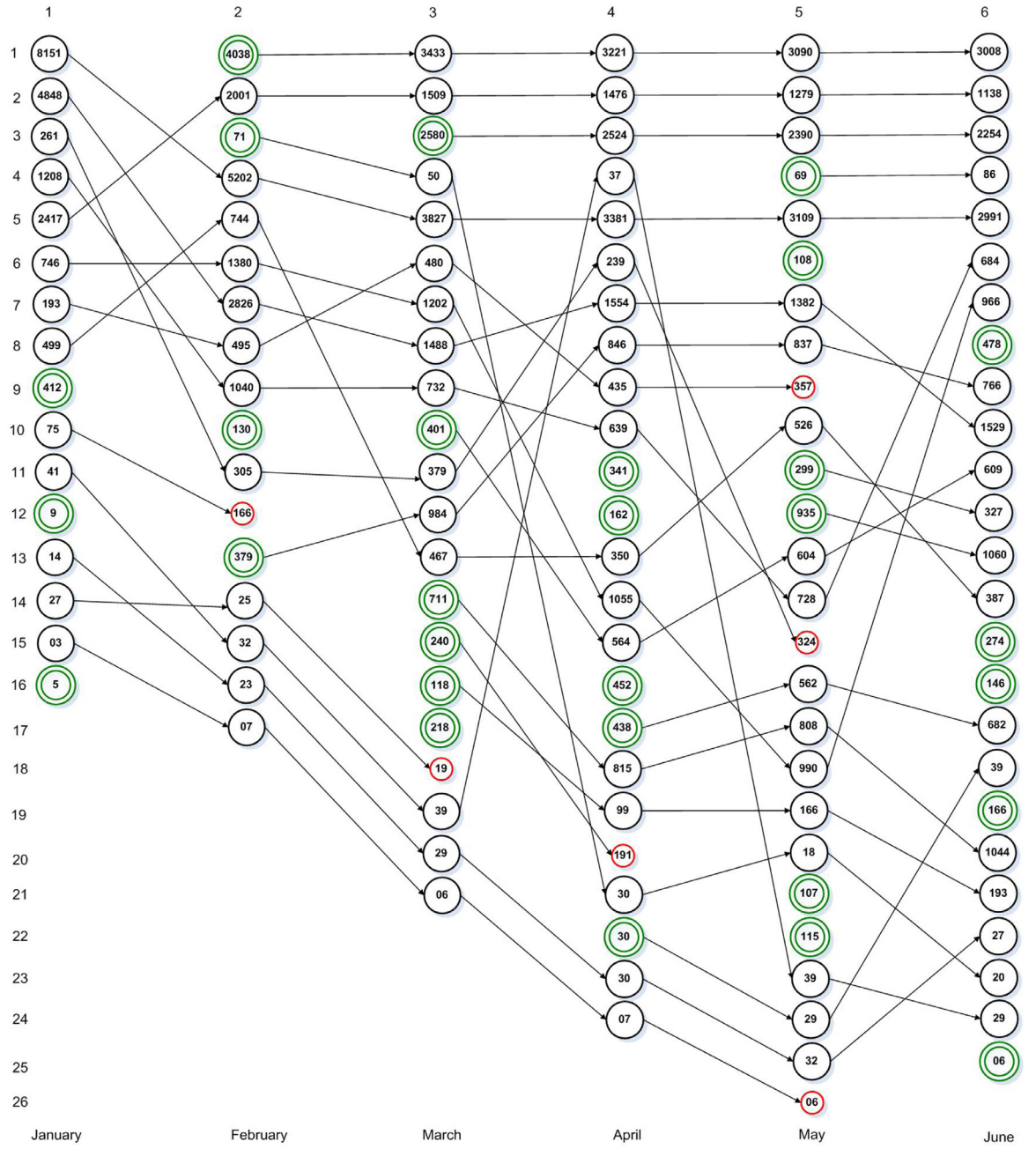


**Fig. 9.** A result produced by CPLP from the real load profile data. The circles represent clusters. The number in a circle is the number of load profiles in the corresponding cluster. The clusters with double circles were emerging clusters and those with small circles were the ones that faded out in the next window.

We can see from the variation of the five six-month patterns, all the power consumptions are periodically stable from February to June, except their values in January. Within a month, similar patterns repeated four times and there were typical rise from Monday and drop at Friday. A typical drop appeared in the middle of week. Most of the values had two peak values in a week. All the patterns were increased in the end of February or at the start of March. Actually, the abrupt decrease of power consumptions in late January is caused by The Spring Festival breaks in China. In other words, the drop is due to the fact that most factories are closed for the Chines New Year holidays. The long-life pattern always have stable pattern within a month but with little change between months. Actually, these patterns are the mainly power consumption components in a power grid.

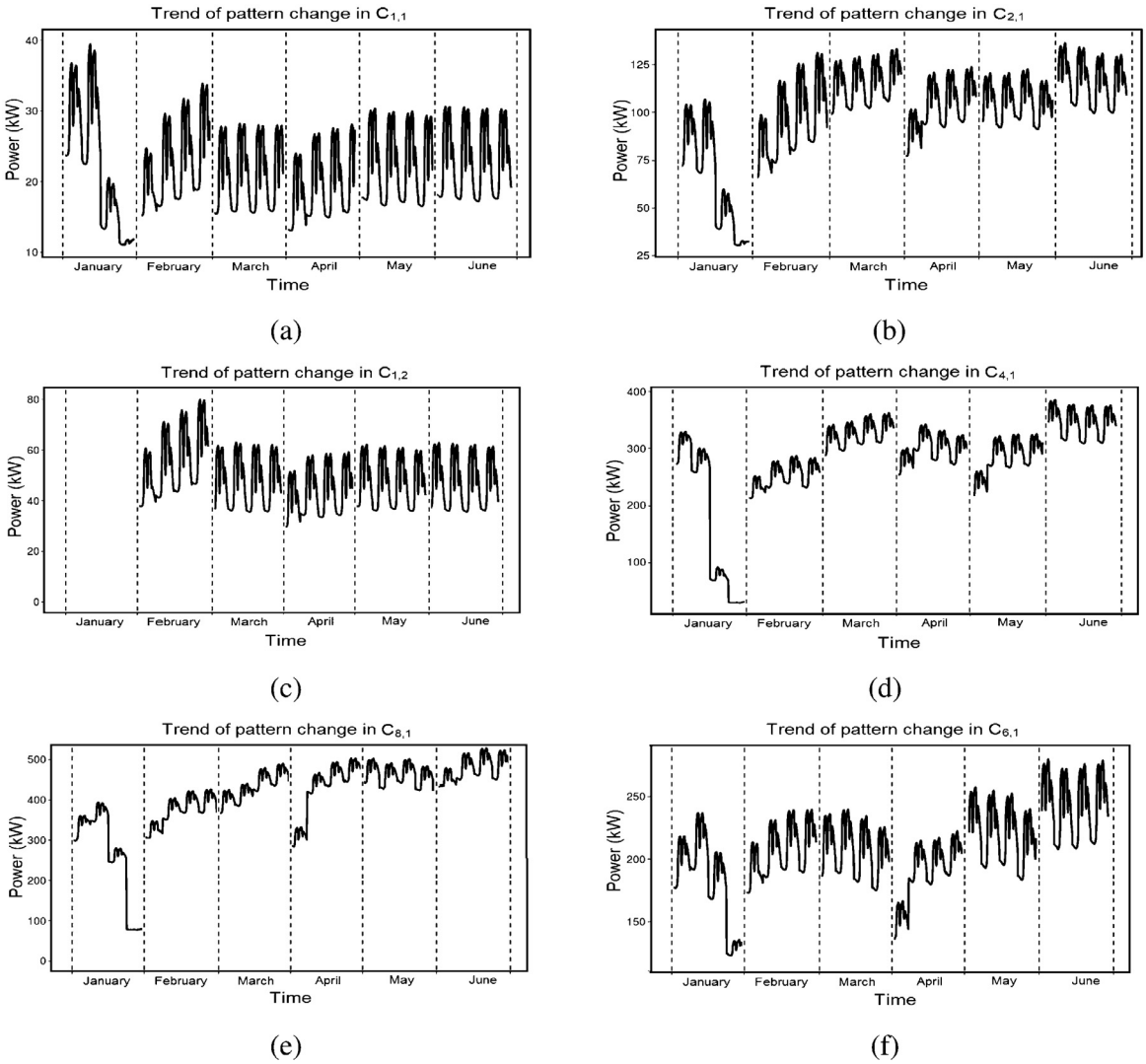


**Fig. 10.** The distributions of the average power consumption of 6 selected stable power consumption patterns in six months. The vertical axis represents the average power consumption (kW) and the horizontal axis represents the time in half hour.

To further investigate the reasons of the changes of power consumption patterns, we looked into factory types which may play a significant role. In Fig. 10(d), there exists inconsistent pattern change in the neighboring time windows. Diverse labour-intensive factories lead to such pattern change. Actually, most of the data were collected from the industries of communication equipment, plastic products, textile, electrical equipment, and metal product. These labour-intensive factories have a major share in power consumption in the region. Pattern $C_{4,1}$ contained a high portion of factories of communication equipment, textile, and plastic product factories. Production patterns of these factories may contribute to the pattern change.

Weather can also affect production activities in industrial sectors, therefore, causing the changes of power consumption patterns. We can observe the weather influence on the pattern change in $C_{1,8}$ from Fig. 10(e). The power consumption increases gradually due to the increase of temperature in the first half of the year from January to June. Some sudden changes of patterns from January to February and from March to April were caused by Chinese New Year and Qingming Festival in China. The holidays also have a significant impact on the power consumption patterns of factories. The pattern change in $C_{1,8}$ may be caused by weather and seasonal reasons. This pattern has a large number of metallic, textile, and garment factories.

To discover other reasons of pattern change, we looked into the maximum and minimum values of the average load profiles. As illustrated in Fig. 10(d)–(f), the patterns show that the power consumptions did not fall to zero, which reflect the production patterns of continuous manufacturing process. As shown in Fig. 10(a)–(c), since pattern change shows work in shift production pattern with small power consumption, they reflect the production patterns of discrete manufacturing process. Such patterns may be generated by production disturbances such as insufficient production orders, frequent change

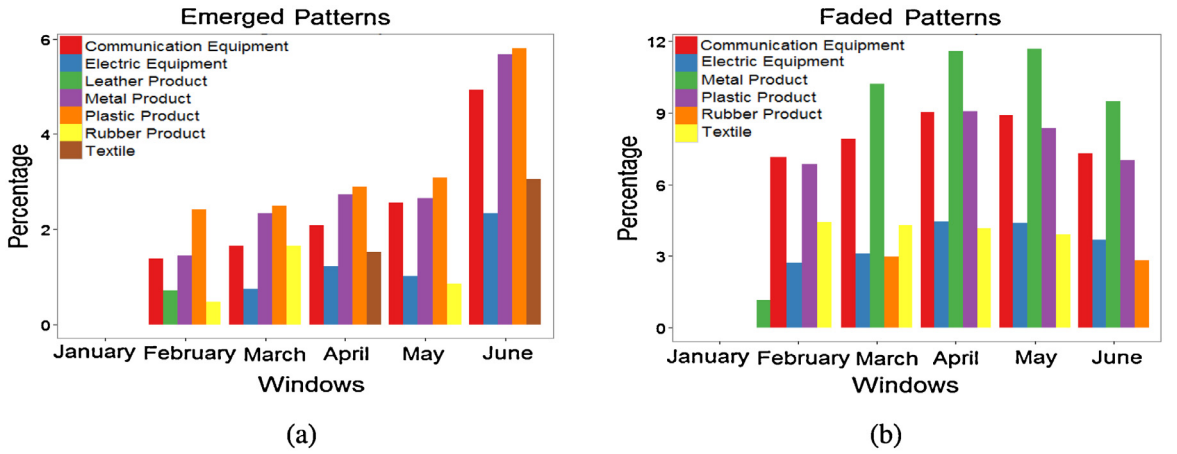


**Fig. 11.** Distributions of dominant factory types in the composition of faded and emerged patterns in six time windows.

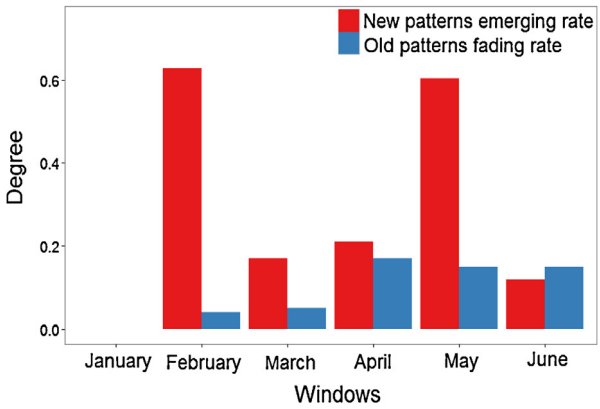


**Fig. 12.** Distributions of emerging and fading rates in six time windows.

of production processes, or partial operation of production lines due to maintenance. These patterns reflect either factories in small capacity or factories that production capacity was not entirely used due to insufficient production orders.

### *7.4. Change measures in time windows*

The number of fading and emerging clusters in a time window indicates the severity of changes of power consumption patterns in the corresponding month. This indirectly reflects the activities of industry in that month. We have defined the measures of fading rate and emerging rate in (19). Fig. 12 shows the distribution of fading rate and emerging rate in the six months computed from the load profile data. We can see that the emerging rate in February and May is high, which indicates that more new power consumption patterns emerged in these two months. The patterns of February are easy to understand, because a lot of production activities started after the Chinese new year. There was a significant change from April to May, where the maximum number of old patterns faded in April while the maximum number of new patterns emerged in May. These could be caused by the weather when temperatures went up or due to increased demands of overseas product orders during Christmas.

To further investigate the details of faded clusters and emerging clusters in each month, we plotted the proportions of dominant factory types in Fig. 11. We can see that communication equipment, plastic product, and metal product factory types played a significant role in the emerging new patterns. They gradually increased their proportions in the emerging patterns from January to June.

In China, plastics are progressively being used as a substitute for steel, wood, and traditional building materials. However, the plastic product factory type maintained the necessary presence in the fading old patterns as well as the emerging new patterns in each month. In the same way, textile factory type maintained consistent contribution in the fading old patterns from February to May then completely disappeared in June. Consequently, such flustered patterns in each month had a significant impact on pattern change in neighboring time windows.

## 8. Conclusion

In this paper, we have presented a new CPLP algorithm for tracking the changes of power consumption patterns of load profile data of factories. It contains two parts. In the first part, we have used a new ensemble clustering method to cluster the load profile data of each time window and use the clusters to model the power consumption patterns. We have developed a new hierarchical binary $k$-means algorithm to generate high quality component clusterings without sacrificing too much diversity. Then, we have defined a new objective function to ensemble component clusterings by maximizing the average similarity between the component clusterings and the final ensemble clustering. In the second part, we have used a new change detection method to compute the change of clusters from one window to the next one by using the distribution models of corresponding related clusters in each pair from two neighboring windows. The experimental results on synthetic data have shown that the CPLP algorithm outperformed the well-known state-of-the-art algorithms. The results on synthetic data also show that CPLP was more robust to noise. Finally, the experimental results on real-life load profile data demonstrated that CPLP can be used effectively for tracking the changes of power consumption patterns which are very important for applications in energy demand management, load forecasting, and dynamic tariff rate adjustment.

Our future work will develop a method that can automatically track the changes of patterns in the streaming data. Finally, we will test and improve our method on further real applications.

## Acknowledgments

This work was supported by the Science and Technology Innovation Committee Foundation of Shenzhen (Grant No. ZDSYS201703031748284), and National Natural Science Foundation of China (NSFC) Grant No. 61750110536. This work was partially supported by GDNSF fund (2015A030313782), and SUSTechStarup fund (Y01236215).

## References

[1] C.C. Aggarwal, Data Streams: Models and Algorithms, Vol. 31, Springer Science & Business Media, 2007.
[2] A. Albert, R. Rajagopa, Smart meter driven segmentation: what your consumption says about you, power systems, IEEE Trans. 28 (4) (2013) 4019–4030.
[3] A. Amini, H. Saboohi, T. Herawan, T.Y. Wah, Mudi-stream: a multi density clustering algorithm for evolving data stream, J. Network Comput. Appl. 59 (2016) 370–385.
[4] A. Bifet, R. Gavalda, Learning from time–changing data with adaptive windowing, in: SDM, volume Vol. 7, SIAM, 2007, pp. 443–448.
[5] A. Bivens, P. Alan, C. Palagiri, Network-based intrusion detection using neural networks, Department of Computer Science Rensselaer Polytechnic Institute Troy, New York 12 (1) (2002) 12180–13590.
[6] F. Cao, J.Z. Huang, J. Liang, Trend analysis of categorical data streams with a concept change method, Inf. Sci. 276 (2014) 160–173.
[7] C. Carpineto, G. Romano, Consensus clustering based on a new probabilistic rand index with application to subtopic retrieval, Pattern Anal. Mach. Intell., IEEE Trans. 34 (12) (2012) 2315–2326.
[8] G. Chicco, Overview and performance assessment of the clustering methods for electrical load pattern grouping, Energy 42 (1) (2012) 68–80.
[9] T. Dasu, S. Krishnan, S. Venkatasubramanian, K. Yi, An information-theoretic approach to detecting changes in multi-dimensional data streams, in: In Proc. Symp. on the Interface of Statistics, Computing Science, and Applications, Citeseer, 2006.
[10] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. 39 (1) (1977) 1–38.
[11] X.Z. Fern, C.E. Brodley, Random projection for high dimensional data clustering: a cluster ensemble approach, in: Proceedings of the Twentieth International Conference on Machine Learning (ICML), 2003, pp. 186–193.
[12] A.L. Fred, A.K. Jain, Combining multiple clusterings using evidence accumulation, IEEE Trans. Pattern Anal. Mach. Intell. 27 (6) (2005) 835–850.
[13] H. Hotelling, The generalization of students ratio, Ann. Math. Stat. 2 (3) (1931) 360–378.
[14] J. Hu, T. Li, H. Wang, H. Fujita, Hierarchical cluster ensemble model based on knowledge granulation, Knowl. Based Syst. 91 (2016) 179–188.
[15] L. Jing, K. Tian, J.Z. Huang, Stratified feature sampling method for ensemble clustering of high dimensional data, Pattern Recognit. 48 (11) (2015) 3688–3702.
[16] I. Khan, J.Z. Huang, K. Ivanov, Incremental density-based ensemble clustering over evolving data streams, Neurocomputing 191 (1) (2016) 34–43.
[17] I. Khan, J.Z. Huang, N.T. Tung, G. Williams, Ensemble clustering of high dimensional data with fastmap projection, in: Trends and Applications in Knowledge Discovery and Data Mining, Springer, 2014, pp. 483–493.
[18] Y.I. Kim, J.M. Ko, S.H. Choi, Methods for generating tlps (typical load profiles) for smart grid-based energy programs, in: Computational Intelligence Applications In Smart Grid (CIASG), IEEE Symposium on, 2011, pp. 1–6.
[19] L.I. Kuncheva, Change detection in streaming multivariate data using likelihood detectors, knowledge and data engineering, IEEE Trans. 25 (5) (2013) 1175–1180.
[20] L.I. Kuncheva, S.T. Hadjitodorov, Using diversity in cluster ensembles, in: Systems, Man and Cybernetics, IEEE International Conference on, Vol. 2, IEEE, 2004.
[21] J. Kwac, J. Flora, R. Rajagopal, Household energy consumption segmentation using hourly data, Smart Grid, IEEE Trans. 5 (1) (2014) 420–430.
[22] J.F. Lawless, Statistical Models and Methods for Lifetime Data, vol. 362, John Wiley & Sons, 2011.
[23] A. Lung-Yut-Fong, C. Levy-Leduc, O. Cappe, Robust changepoint detection based on multivariate rank statistics, in: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2011, pp. 3608–3611.
[24] G. Milligan, P. Isaac, The validation of four ultrametric clustering algorithms, Pattern Recognit. 12 (2) (1980) 41–50.
[25] D. Posada, T.R. Buckley, Model selection and model averaging in phylogenetics: advantages of akaike information criterion and bayesian approaches over likelihood ratio tests, Syst. Biol. 53 (5) (2004) 793–808.
[26] G. Schwarz, Estimating the dimension of a model, Annl. Stat. 6 (2) (1978) 461–464.
[27] A.K. Seghouane, S.I. Amari, The AIC criterion and symmetrizing the kullback-leibler divergence, Neural Networks, IEEE Trans. 18 (1) (2007) 97–106.
[28] A. Strehl, J. Ghosh, Cluster ensembles: a knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (2002) 583–617.
[29] S.V. Verd, M.O. Garcia, C. Senabre, A.G. Marin, Classification, filtering, and identification of electrical customer load patterns through the use of self-organizing maps, Power Syst., IEEE Trans. 21 (4) (2006) 1672–1682.
[30] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, Mach. Learn 23 (1) (1996) 69–101.
[31] L. Zhang, W. Lu, X. Liu, W. Pedrycz, C. Zhong, Fuzzy c-means clustering of incomplete data based on probabilistic information granules of missing values, Knowl. Based Syst. 99 (2016) 51–70.
[32] W. Zhong, R. Chow, J. He, Clinical charge profiles prediction for patients diagnosed with chronic diseases using multi-level support vector machine, Expert Syst. Appl. 39 (1) (2012) 1474–1483.