energies

MDPI

Article

OPEC: Daily Load Data Analysis Based on Optimized Evolutionary Clustering

Rongheng Lin *, **Zezhou Ye** and **Yingying Zhao**

State Key Lab of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

* Correspondence: rhlin@bupt.edu.cn; Tel.: +86-10-61198136

Received: 5 June 2019; Accepted: 9 July 2019; Published: 11 July 2019

Abstract: Customers' electricity consumption behavior can be studied from daily load data. Studying the daily load data for user behavior pattern analysis is an emerging research area in smart grid. Traditionally, the daily load data can be clustered into different clusters, to reveal the different categories of consumption. However, as user's electricity consumption behavior changes over time, classical clustering algorithms are not suitable for tracing the changes, as they rebuild the clusters when clustering at any timestamp but never consider the relationship with the clusters in the previous state. To understand the changes of consumption behavior, we proposed an optimized evolutionary clustering (OPEC) algorithm, which optimized the existing evolutionary clustering algorithm by joining the Proper Restart (PR) Framework. OPEC relied on the basic fact that user's energy consumption behavior would not abruptly change significantly, so the clusters would change progressively and remain similar in adjacent periods, except for an emergency. The newly added PR framework can deal with a situation where data changes dramatically in a short period of time, and where the former frameworks of evolutionary clustering do not work well. We evaluated the OPEC based on daily load data from Shanghai, China and the power load diagram data from UCI machine learning repository. We also carefully discussed the adjustment of the parameter in the optimized algorithm and gave an optimal value for reference. OPEC can be implemented to adapt to this situation and improve clustering quality. By understanding the changes of the users' power consumption modes, we can detect abnormal power consumption behaviors, and also analyze the changing trend to improve the operations of the power system. This is significant for the regulation of peak load in the power grid. In addition, it can bring certain economic benefits to the operation of the power grid.

Keywords: smart grid; behavior pattern; optimized evolutionary clustering

1. Introduction

It is generally acknowledged that there are changing cycles of users' electricity consumption at different time scales, such as days, weeks or years. In order to study the evolution of user consumption behavior in a more detailed way, we used daily data in this paper. Daily load data are important for operations of smart grid which reflect how the user's electricity consumption changes over a time period, e.g. days. These data are obtained by smart electric meters. There are abundant studies related to daily load data. According to the frequency of sampling, the daily load data include 96-points style, 48-points style, and 24-point style. The points reflect user's real-time power consumption. Thus, 96-points data means that there are 96 points of load data in 24 h, which is the data style used in this paper. Most studies are expected to unravel the power mode from the 96-point data and draw portraits for users. They use clustering methods by analyzing the statistical information or studying the shape of the 96-point curve. Li et al. [1] divided users into 3 categories: industry, public facilities, and residents. Zhang et al. [2] divided residents into office workers, elderly families, business users, etc. Lin et al. [3]

studied from the perspective of the population and household appliances. These methods mentioned above tend to define users by using daily load data. Benitez et al. [4] tried to merge 365 days of load data into a high dimension dataset for clustering. According to McLoughlin's research [5], we can also detect the relationship between the user's personal information (family size, income etc.) and their energy consumption behavior. Furthermore, in the literature, it is often assumed that load curve of a certain user tends to be clustered into a certain category which does not change over time.

The reality is that users might change their energy consumption behavior as time goes on. As mentioned before, daily load data are clustered to find certain user energy consumption behavior. To characterize the changing behaviors, an evolutionary clustering method is introduced in this paper. The changes of the behaviors are transformed into the evolution of the clusters.

This paper is organized as follows. Section 2 explores related works about evolutionary clustering and clustering for user behavior, and Section 3 explains why traditional clustering is insufficient and evolutionary clustering is preferred. Sections 4 and 5 give the definition of our new algorithm and aim to ensure it is understandable. We performed experiments to show the good performance of OPEC in Sections 6 and 7, and finally present a conclusion in Section 8.

2. Related Work

Clustering for user behavior is important to the design and operation of various services. Researchers divided residential users into different categories according to their electricity consumption behavior. Rhodes et al. [6] analyzed residential electricity demand by clustering their electricity consumption behavior. Oh and Lee [7] used a clustering algorithm to detect the normal behavior. Maia and Wang [8,9] grouped users that share similar behavioral pattern in online social networks.

However, in order to reflect the long-term changing trends of user behavior, the clustering results should reflect the status of the current data, and the clustering should not deviate drastically from the recent clustering results. Therefore, the relationship between the current data and historical data should be taken into consideration. Thus, the evolutionary clustering should be applied to user behavior pattern analysis instead of static clustering methods.

Evolutionary clustering is an emerging research area essential to important applications, such as clustering dynamic web and blog contents, monitoring of evolving communities [10], analyzing the relationships between the entities of the dynamic networks [11], clustering of multi-typed objects in a heterogeneous bibliographic network [12] and clustering seeking information on web for the dynamic recommender system [13–16]. Considering the problem of clustering data over time, one of the earliest methods for evolutionary clustering is proposed by Chakrabarti [17]. The evolving clustering method balanced two important objectives while performing the online clustering process. Firstly, the clustering at any point in time should remain faithful to the current data as much as possible. Secondly, the clustering should not shift dramatically from one timestep to the next. Chakrabarti's framework defined two indicators to evaluate the quality of evolutionary clustering. They are *SQ* (Snapshot Quality) and *HQ* (Historical Quality). Chakrabarti also discussed evolutionary versions of two widely used clustering algorithms within this framework: K-means and agglomerative hierarchical clustering. Ding et al. [18] proposed a novel hierarchical co-evolutionary clustering tree model (HCCT). It aimed to highlight the high-quality feature subsets, which can enrich the research of feature selection and classification in the heterogeneous big data. Chi et al. [19] expanded on this idea by proposing two frameworks that incorporate temporal smoothness in evolutionary spectral clustering.

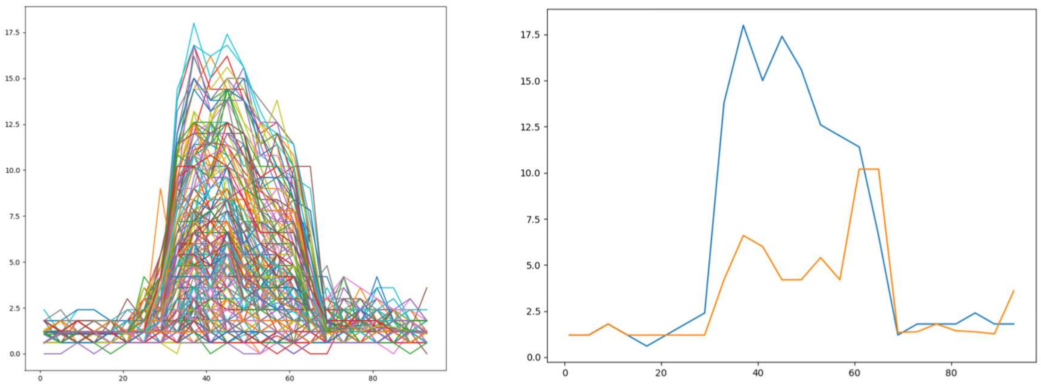
Following these works, the current study is different from the existing work in adding a temporal smoothness penalty to the cost function of a static clustering method. Xu et al. [20] used an evolutionary tree to describe the evolution process of data. By taking account of both the existing temporal smoothness and the structural smoothness, they proposed an evolutionary tree clustering framework and an evolutionary tree spectral clustering algorithm. To address large evolutionary data sets, where other existing clustering methods have poor performance, Wang et al. [21] proposed ECKF (Evolutionary Clustering based on low-rank Kernel matrix Factorization), a general framework for clustering large-scale data based on low-rank kernel matrix factorization. Xu et al. [22] proposed a framework that adaptively estimates the optimal smoothing parameter using a shrinkage approach. Day et al. [23] used Entropy-Based Evolutionary Clustering (EBEC) in an effort to cluster writing styles. The differential evolution (DE) [24] is also applied to deal with this problem. Amelio and Pizzuti [25] use a time-space enhanced clustering for community discovery. Al-Sharoa et al. [26] proposed a low-rank approximation based evolutionary clustering approach. The proposed approach provided robustness to outliers and results in smoothly evolving cluster assignments through joint low-rank approximation and subspace learning.

Shukri et al. [27] proposed a Multi-verse Optimizer (MVO) to optimize clustering problems which use the nature-inspired algorithms. Majdi et al. [28] focused on the capability of the evolutionary computation methods namely, genetic algorithm (GA) and particle swarm optimization (PSO) in design and optimizing the fuzzy c-means clustering (FCM) structure and their applications to predict the deformation modulus of rock masses. Furthermore, to deal with the problem that several objective functions usually need to be optimized, a multitude of multi-objective evolutionary clustering algorithms [29–33] have appeared.

These works apply evolutionary clustering for operating of various services, but their frameworks of evolutionary clustering do not work well when data change dramatically in a short period of time. Therefore, to understand the changes of consumption behavior, we apply evolutionary clustering to user daily load data analysis and present a new framework to optimize evolutionary clustering.

3. Problem Description

User's electricity consumption behavior changed as time went by, and the clustering results of load data changed as well. The evolution of electricity consumption behavior is transformed into the evolution of clustering results of load data. The load profile indicates the majority features in a series of daily load curves. There are several major methods for extracting load profile from load curves. The most commonly used method is the clustering method. We take a single load profile along with the original load curve, for example. In Figure 1, the left sub-figure indicates the load curves for a year, and the right sub-figure indicates the load profile. The x-axis represents the time points and the y-axis represents user's real-time power consumption (kW/h). The load profile has only two curves, but these two curves represent the characters of the load curves for a year.



(**a**) The load curves for a year. (**b**) The load profile.

Figure 1. Load Curves and Load Profile.

As the single user's load curve can be represented as some load profile and the user's electricity consumption model changes as the time goes, in this paper we try to characterize the changing state of the load profile.

Besides, in the multiple user scenario, if we try to cluster the users together, there would be several different communities due to different behaviors of the users. Like one single user, the group of users' activities might change. Therefore, if we watch the changes of the communities in a time sequence manner, the changing of the user's behavior would be revealed.

Since we need to cluster the load curve over time, dynamical clustering or multiple clustering is the easiest solution. However, the clustering is non-supervised learning, and every round of clustering might lead to different communities. It would be difficult to find the connection of those communities among different rounds of clustering. As shown in Figure 2, the left sub-figure shows the results of first-round clustering, and the right sub-figure is the second-round clustering. The communities generated by the first round are totally different from the second one. It would be hard to connect these two communities together. Even if we force them to connect together, it is hard to analyze the changing trends as they are not the same clusters, in fact.

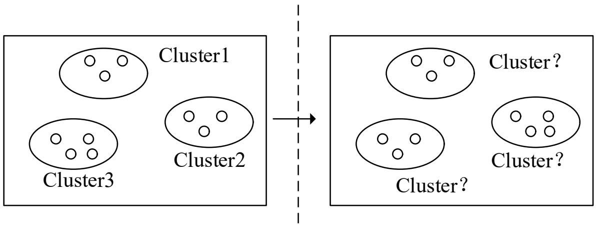


Figure 2. Dynamically clustering mapping problem.

To characterize the changing trend, evolutionary clustering is introduced. Evolutionary clustering would keep the evolution of clustering community in a consistent way, as every round of the clustering result would take the previous ones into consideration. As shown in Figure 3, the evolutionary clustering algorithm is based on the previous clustering result and clustering in an increment way.

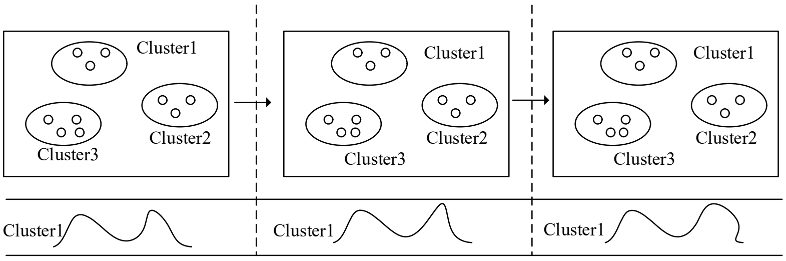


Figure 3. Evolutionary clustering with load profile changes.

The challenge of this paper is to find an efficient way to understand the changing principle of the load profile in an evolutionary way. To achieve this goal, optimized evolutionary clustering is proposed.

4. Evolutionary Clustering Math Definition

To illustrate the process of optimized evolutionary clustering, we define some basic symbols and introduce the proposed algorithm including distance calculation, clustering evaluation, and the new framework.

4.1. Basic Definition

Let t be the timestep and U_t be the daily load data, i.e., $x_{t,i}$ is the daily load data of user i at timestep t. There are some disagreements on what kind of data $x_{t,i}$ are preferred. Some researchers use statistical properties such as peak power consumption, average power consumption, while others use the full sample (96-point data for short). It is simple to build a model using statistical properties. However, statistical models lack of the timing features which are important for behavior pattern analysis. Therefore, we prefer the 96-point data.

$$U_t = \left\{x_{t,1}, x_{t,2} \ldots x_{t,N}\right\} \tag{1}$$

After clustering the load data U_t, we will get k clustering centroids. Let C_t be the set of centroids, and $c_{t,i}$ is the i-th centroid at timestep t.

$$C_t = \left\{c_{t,1}, c_{t,2} \ldots c_{t,k}\right\} \tag{2}$$

4.2. Distance Calculation

Measurements of similarity or distance between $x_{t,i}$ and $x_{t,j}$ are important in the clustering setting. Distance or similarity calculation differs in different situations. Distance measurement methods like the Euclidean distance are not concerned with the differences between the shapes of curves $x_{t,i}$ and $x_{t,j}$ while the cosine distance is concerned with this. The shape of curve $x_{t,i}$ is the key to the study of user behavior patterns. For example, Zhang et al. [2] analyzed several patterns of users which are clustered based on their different shapes of the 96-point data curves. These users behave differently during the day and night, which result in different shape of curves. We use $\text{dis}\left(x_i, x_j\right)$ to express the cosine distance between $x_{t,i}$ and $x_{t,j}$.

$$\text{dis}\left(x_i, x_j\right) = \frac{x_i x_j}{\| x_i \| \| x_j \|} \tag{3}$$

4.3. Clustering Evaluation

What we care most about here is the quality of clustering. There are two evaluation goals: the *SQ* (Snapshot Quality) and *HQ* (History Quality). These two measurements are proposed by Chakrabarti mentioned in Section 2, and they are important indicators for measuring evolutionary clustering. Firstly, the clustering at any point in time should remain faithful to the current data as much as possible. Secondly, the clustering should not shift dramatically from one timestep to the next. We use silhouette score for *SQ* evaluation and preserve cluster membership for *HQ* evaluation. Evolution rate can be observed through the change of *SQ* and *HQ*.

(1) Snapshot Quality: *SQ* returns the clustering quality at timestep t. The most common used *SQ* evaluation method is silhouette score. The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). Formula (4) shows how to calculate *SQ*, let $a(i)$ be the average dissimilarity of $x_{t,i}$ within the same cluster. Let $b(i)$ be the lowest average dissimilarity of $x_{t,i}$ to any other cluster.

$$SQ = \sum_{i=1}^{N} \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{4}$$

(2) History Quality: *HQ* returns the historical cost of the clustering at timestep t. We use PCM (Preserve Cluster Membership) [19] to calculate *HQ*. In PCM framework, the temporal cost is expressed as the difference between the current centroids and the historical centroids. In other words, if we get two current centroids C_t and $C_{t'}$. We assume that they have the same *SQ*. Let the centroids at timestep $t-1$ be C_{t-1}. If C_t is more similar to C_{t-1} than $C_{t'}$, then *HQ* (C_t) is higher than *HQ* ($C_{t'}$). As mentioned before, we use the sum of cosine distance between C_t and C_{t-1} to calculate *HQ* (C_t). *HQ* can be expressed as follows.

$$HQ = \sum_{i=1}^{k} \sum_{j=1}^{k} dis\left(c_{i,t}, c_{j,t-1}\right) \tag{5}$$

4.4. Proper Restart Framework (PR Framework)

Before introducing the framework, we emphasize that our target is to maximize *SQ* and *HQ*, but as stated by Chakrabarti [17], *SQ* and *HQ* are two conflict goals. Evolutionary clustering sacrifices *SQ* to gain *HQ* in some situations, and the data shift dramatically from timestep $t-1$ to t that any smoothing clustering method will reduce *SQ* greatly, thus, we would be better not using evolutionary clustering in the above situations. For example, two current centroids C_t and $C_{t'}$. $C_{t'}$ came from K-means while C_t came from evolutionary clustering. If *SQ* ($C_{t'}$) is 20% higher than *SQ* (C_t), there would be no need to use evolutionary clustering.

Based on above considerations, we design a piecewise function for parameter P_{cp}, as shown in formula (6):

$$P_{cp} = \begin{cases} const, & \frac{SQ'-SQ}{SQ'} < \beta \\ 0, & \frac{SQ'-SQ}{SQ'} \geq \beta \end{cases} \tag{6}$$

The goal of P_{cp}, which is decided by the changing of *SQ* over time, is to control when to use evolutionary clustering. Usually the evolutionary clustering will choose a suitable const parameter to fix it. However, if the value of *const* is too large, the quality of the cluster will drop sharply during the evolutionary process. In our experiments, the value of const is fixed to 0.3, which is of considering choosing a relatively small value. Parameter β is the upper threshold of relative degradation of the snapshot quality that we can accept. The time intervals for conducting the clustering depends on the snapshot quality. When the snapshot quality drops too much, framework PR will restart the evolutionary clustering process. P_{cp} will be used later. In conclusion, the proposed framework can balance *SQ* and *HQ* for evolutionary analysis.

5. Optimized Evolutionary Clustering

5.1. Optimized Evolutionary Clustering

In K-means clustering, there are several efficient heuristic algorithms that are commonly deployed. However, they converge quickly to a local optimum. In evolutionary clustering, the clustering result C_{t-1} is used to produce a local optimum C_t, which is close to C_{t-1}. In other words, some smoothing algorithms are used to balance C_t and C_{t-1}. However, any smoothing algorithm would sacrifice the *SQ*. In optimized evolutionary clustering, we introduce a parameter P_{cp} to improve the *SQ*. The basic idea of optimized evolutionary clustering is to control parameter P_{cp} to decide when to restart evolutionary clustering. The following formula shows how we use parameter P_{cp} to update centroids.

$$c_{t,j} = \gamma P_{cp} c_{t-1,f(j)} + \left(1 - \gamma P_{cp}\right) c'_{t,j} \tag{7}$$

Let $f(j)$ be the mapping function, which returns the nearest centroid at timestep $t-1$ according to the centroids j at timestep t. Let γ be a weight parameter between $c'_{t,j}$ and $c_{t-1,f(j)}$. γ is expressed as follows.

$$\gamma = \frac{members\ of\ c'_{t,j}}{members\ of\ c'_{t,j} + members\ of\ c_{t-1,f(j)}} \tag{8}$$

As shown in formula (7), the changing of parameter P_{cp} can control the smoothing progress. No smoothing method is used when P_{cp} = 0. When P_{cp} has a positive value, $c_{t,j}$ will become close to $c_{t-1,f(j)}$. As analyzed before, if *SQ* drops dramatically, we will stop using evolutionary clustering and set P_{cp} = 0. The parameter P_{cp} is decided by the current *SQ* like formula (6).

5.2. Flow Diagram

We apply Proper Restart framework to the evolutionary K-means algorithm using the flow diagram in Figure 4.

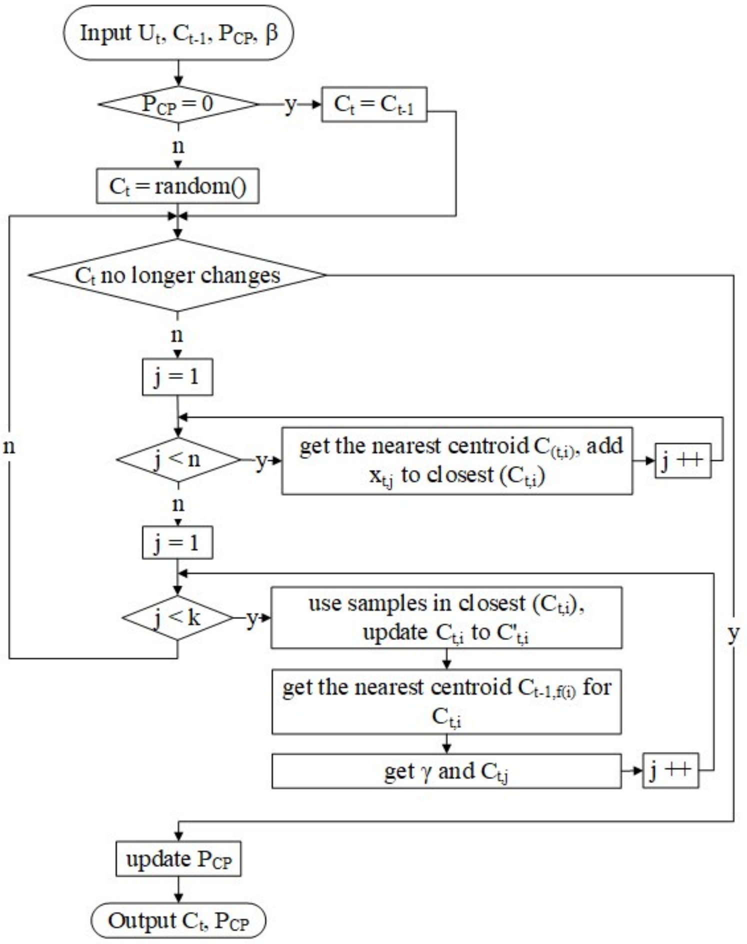

Input U_t, C_{t-1}, P_{CP}, β
$P_{CP} = 0$
y
$C_t = C_{t-1}$
n
C_t = random()
C_t no longer changes
n
j = 1
j < n
y
get the nearest centroid $C_{(t,i)}$, add $x_{t,j}$ to closest ($C_{t,i}$)
j ++
n
j = 1
j < k
y
use samples in closest ($C_{t,i}$), update $C_{t,i}$ to $C'_{t,i}$
get the nearest centroid $C_{t-1,f(i)}$ for $C_{t,i}$
get γ and $C_{t,j}$
j ++
n
y
update P_{CP}
Output C_t, P_{CP}

Figure 4. Flow diagram of optimized evolutionary clustering.

This block diagram shows how optimized evolutionary cluster works in time t. The input parameters include U_t, C_{t-1}, P_{cp} and β. C_{t-1} is the clustering centroids at timestep $t-1$. The parameter β is the threshold which is set to control the optimized evolutionary clustering. The value of β is between 0 and 1. If β is 0, there would be no evolutionary progress. In timestep $t-1$, P_{cp} should be updated to decide whether to restart evolutionary clustering. Output values include C_{t-1} and P_{cp}.

The main process is almost the same as K-means except the initial step and update step. In the initial step, P_{cp} plays an important role to decide whether a random initial is used. In the update step, we use formula (7) to update the new centroids. Finally, we update P_{cp} according to the current SQ.

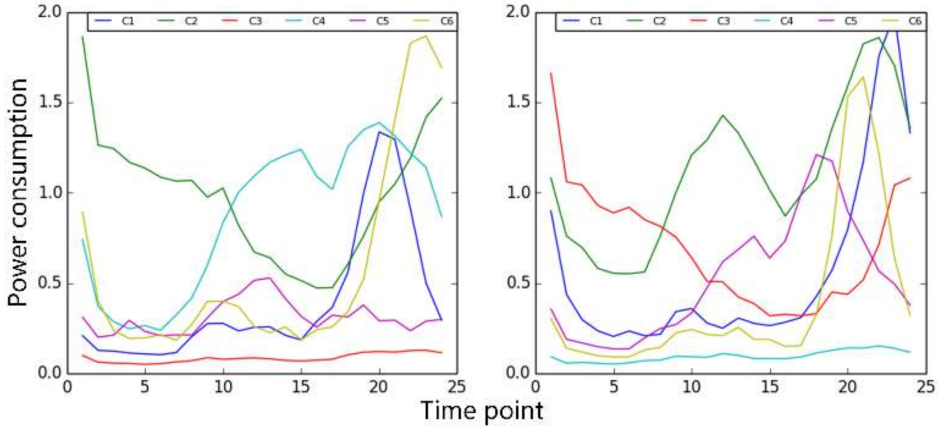
6. Experiments

In this section, to demonstrate the effectiveness of the proposed algorithm, firstly, we have compared three algorithms including K-means, evolutionary K-means and optimized evolutionary K-means. For each algorithm, the snapshot quality and history quality has been calculated and evaluated. Secondly, to study how parameter β affects the optimized evolutionary clustering, we performed an extensive study of our algorithms under different parameter settings, adjusting the optimal β. We show how well PR framework works for evolutionary clustering to reduce the deviation from history clusters and maintain high snapshot quality even if the data changes greatly.

For our experiments, we used two datasets. The first dataset is the one-year non-public collection of 96-point daily load data in Shanghai, China, which is collected by smart meters provided by Shanghai State Grid from about 1000 users' data. The timeline of the data is continuous. Each row contains information about the user id and timestep with 96 samples of a day. The second dataset is the power load diagram dataset LD2011_2014, which is provided by UCI machine learning repository by https://archive.ics.uci.edu/ml/machine-learning-databases/00321/. This open-source dataset records the power consumption of 370 customers every 15 min over four years, where the acquisition frequency, every 15 min, indicates that the daily load data is also 96-point. The large volume of the above two datasets makes the experimental results more convincing.

Furthermore, all rows at timestep t are gathered into U_t, and U_t will be clustered into K partitions C_t. According to the existing research [2], the classification of residential electricity use is roughly in the 5–10 category. In the following experiments, K is set to be 6 and the const in Equation (6) is set to be 0.3. Since the clustering algorithm is sensitive to outliers, the preprocessing processes include deduplication, using regression, decision tree induction to determine the most likely value to fill in the missing values, and using a data cleaning algorithm such as binning to smooth the noise.

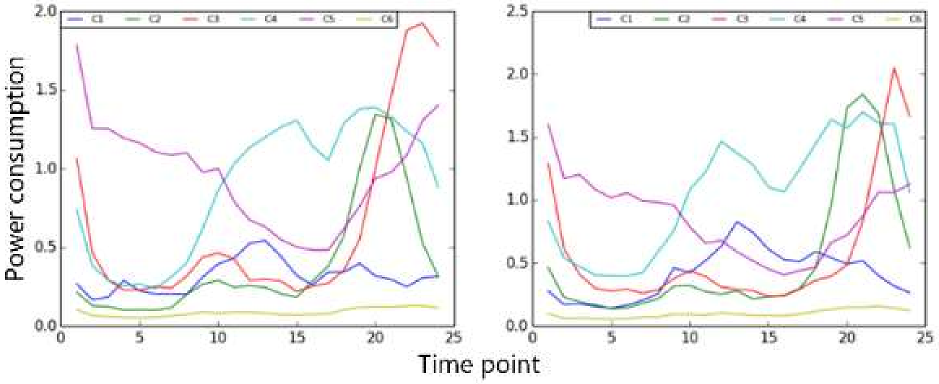
In the process of applying evolutionary clustering to daily load data, we observe that evolutionary clustering is useful to study behavior pattern evolution. Figure 5 shows the clustering centroid of two days' 96-point daily load data by using K-means. Figure 5a indicates the first day's centroid, and Figure 5b indicates the second day's centroids. Figure 6 shows the results of applying the evolutionary K-means clustering to the same data. Figure 6a indicates the first day's centroid, and Figure 6b indicates the second day's centroids. Each subfigure of Figures 5 and 6 represents the clustering centroids of 96-point daily load data in one specific day. The *x*-axis represents the time points and the *y*-axis represents user's real-time power consumption (kW/h). It is obvious that the centroids of Figure 6 between two days change more smoothly than Figure 5, which means evolutionary K-means clustering has higher history quality (*HQ*). When studying the user behavior, we tend to believe that two adjacent days should have the similar results.



(**a**) First day's centroids.

(**b**) Second day's centroids.

Figure 5. Two days' centroids of K-means.



(**a**) First day's centroids.

(**b**) Second day's centroids.

Figure 6. Two days' centroids of evolutionary K-means.

However, there are still some situations when evolutionary clustering is not sufficient. Figure 7 analyzes the current Snapshot Quality (*SQ*) of continuous 19 days' energy consumption data using both K-means and evolutionary K-means clustering. Figure 7 shows that the *SQ* of evolutionary clustering keeps up with K-means clustering for a period (from point 1 to point 8). However, after a certain time (point 8) the *SQ* of evolutionary clustering starts to drop. This kind of "deterioration" will last for a long period of time.

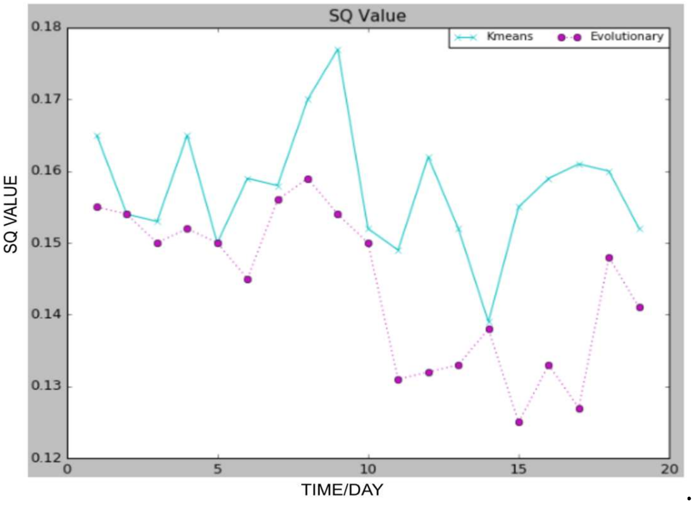


Figure 7. Nineteen days' snapshot quality of K-means and evolutionary K-means in Shanghai dataset.

According to the preliminary analysis, user power consumption behavior does not always change steadily. The behaviors change a lot along with some special events, such as large-scale climate change, holidays, blackout events. When using evolutionary clustering at such a special time point, the results of the two days are too different to be smoothed, so performing evolutionary clustering would be a negative option. The smoothing effect of evolutionary clustering will not only reduce the *SQ* at the current time point but also have a lasting negative effect on the *SQ* for the following period of time.

To optimize evolutionary clustering, one solution is to control the start time point. When meeting the time point mentioned above, we stop the old evolutionary clustering progress and start a new one. That is the core concept of Proper Restart framework.

6.1. Clustering Quality

Figure 8 shows the 19 days' daily results of snapshot quality and history quality of three algorithms in Shanghai dataset. The x-axis represents the day and the y-axis represents the snapshot quality and history quality. The average *SQ* and *HQ* over these days are described in Table 1.

In the following, we mainly explain about the impact of the restart events on clustering quality, which has not been explained clearly before.

As for the snapshot quality, we observe the following:

- The snapshot quality of K-means is higher than the other two, and the snapshot quality of optimized evolutionary clustering is almost as good as K-means, which can also be obtained from Table 1.
- From time point 0 to time point 8, the snapshot quality of the three algorithms is almost the same, and the process of optimized algorithm and evolutionary K-means are identical. At time 9, the snapshot quality of evolutionary clustering starts to drop, and at the same time the snapshot quality of the optimized algorithm rises to the same level as K-means. That is, time point 9 is the first restart point in the optimized algorithm.
- The negative effect of evolutionary clustering lasts for a period of time and then recovers to the normal level.

As for the history quality, we observe the following:

- The history quality of evolutionary clustering and the optimized algorithm is higher than K-means, which can also be obtained from Table 1.
- At the time point 9, the history quality of the optimized algorithm drops slightly, but it recovers quickly at time point 10.

From Table 1 we can clearly see that the optimized evolutionary clustering algorithm has better performance in comprehensive consideration of snapshot and history quality than K-means and evolutionary K-means, which means the optimized evolutionary clustering results remain faithful to the current data and the clustering deviates little from the recent clustering results. This proves the validity and novelty of our proposed algorithm.

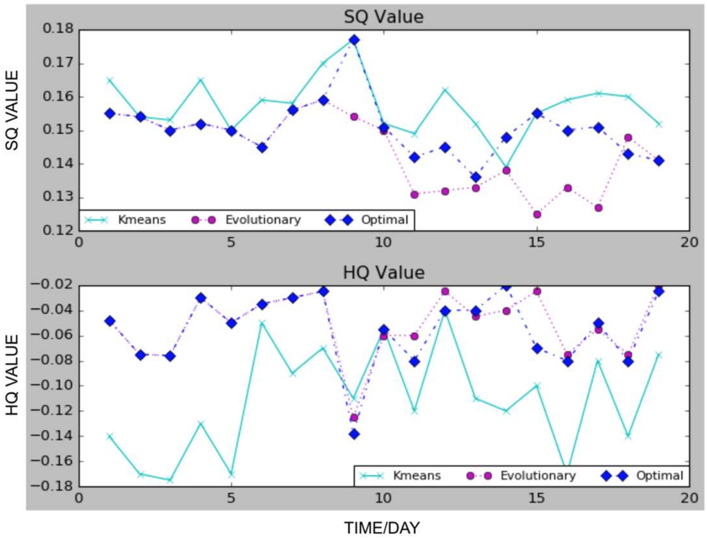


Figure 8. Nineteen days' Snapshot Quality (*SQ*) and History Quality (*HQ*) of three algorithms in Shanghai dataset.

Table 1. Quality comparison.

Algorithm	*SQ* (Snapshot Quality)	*HQ* (Historical Quality)
K-means	0.156	−0.105
Evolutionary Clustering	0.140	−0.048
Optimized Evolutionary Clustering	0.153	−0.049

In order to test the effectiveness of the algorithm over a longer period of time, we then tested over a continuous time period, where the start date of LD2011_2014 dataset and Shanghai dataset were 1 January 2012 and 1 January 2016, respectively. Figure 9 shows the average *SQ* over 100 days in Shanghai dataset. The x-axis represents the day and the y-axis represents the snapshot quality. What's more, this comparison is also conducted in dataset LD2011_2014, which is shown in Figure 10. The load data in 2012 are selected, and one year is sufficient to locate the problem. From Figures 9 and 10, we observe the following:

- The average snapshot quality of K-means and optimized evolutionary clustering are much higher than that of evolutionary clustering. As shown in Figure 10, there is a sudden drop around the 20th day and the average *SQ* of evolutionary clustering cannot be recovered. Thus, this should be the restart point.
- The average snapshot quality of evolutionary clustering decreases over time, while K-means and optimized evolutionary clustering are much more stable as time goes by.

- Experiments on datasets from different sources demonstrate the universality and superiority of the proposed algorithm.

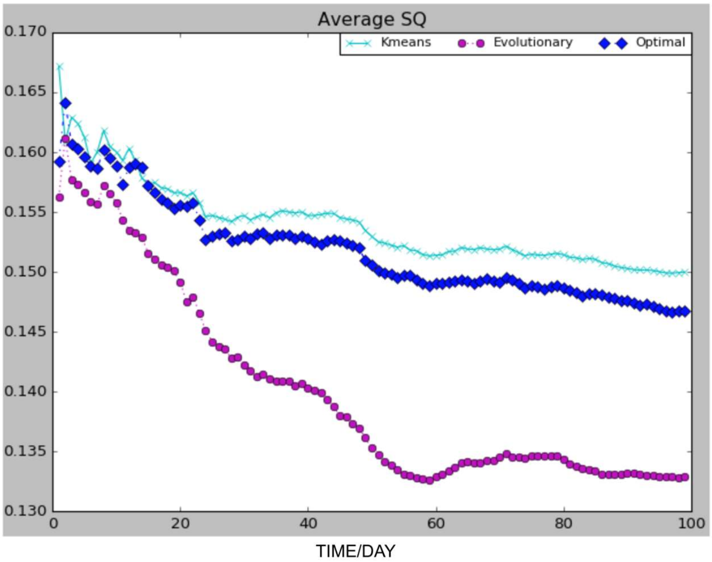


Figure 9. One hundred days' average *SQ* of three algorithms in Shanghai dataset.

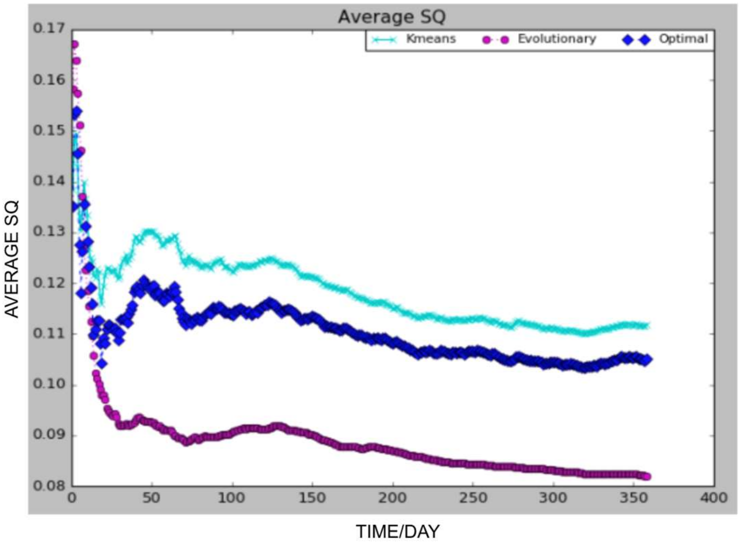


Figure 10. One-year average *SQ* of three algorithms in LD2011_2014 dataset.

A stable algorithm means that the average *SQ* stays stable as time goes on. Figures 9 and 10 show how the three algorithms perform. Optimized evolutionary clustering is as stable as K-means while evolutionary clustering's performance is poor.

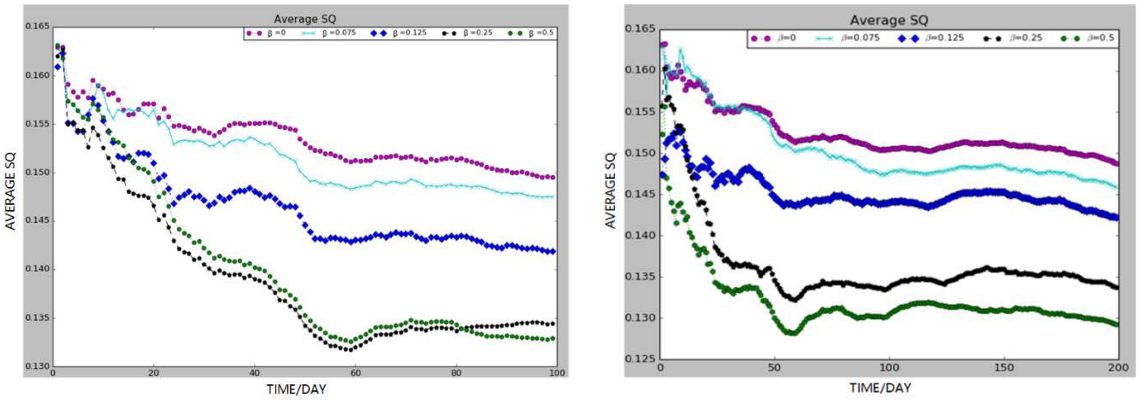
In the following experiments, we study how parameter β affects the optimized evolutionary clustering. We have further expanded the data scale to verify two important indicators (*SQ* and *HQ*) of evolutionary clustering, including the number of users (more than 100,000) and the expansion of time span (more than 300 days).

6.2. Effect of β on Snapshot Quality

The parameter β is the threshold which is set to control the optimized evolutionary clustering. The value of β is between 0 and 1. If β is 0, there would be no evolutionary progress.

Figures 11 and 12 show how the average snapshot quality of the optimized evolutionary clustering changes with the parameter β in different datasets. The x-axis represents the day and the y-axis represents the snapshot quality. From these figures, we observe the following::

- When β gets larger, the average SQ becomes lower.
- When β gets too large, such as 0.25 and 0.5, the average SQ is negative but it will not become worse. $\beta = 0.25$ means that the snapshot quality drops 25%.
- As the time span expands and the number of users increase, although these curves have some fluctuations, the 100-day, 200-day and one-year trend of three figures show that the average SQ is still decreasing. The relative positions between the curves remain relatively stable.



(**a**) 100 days' average SQ of optimized evolutionary using different β.

(**b**) 200 days' average SQ of optimized evolutionary using different β.

Figure 11. One hundred and 200 days' average SQ of optimized evolutionary using different β in Shanghai dataset.

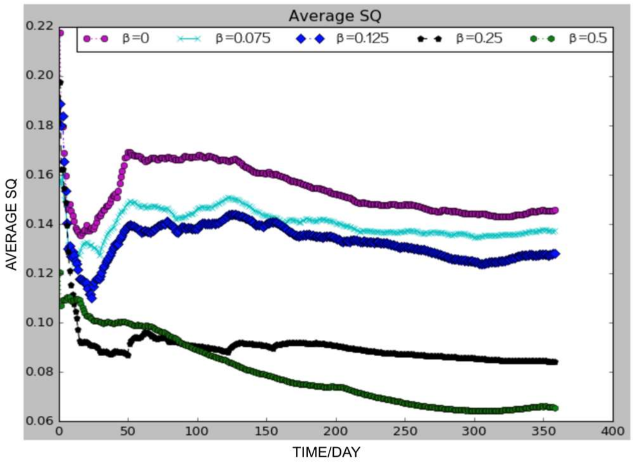


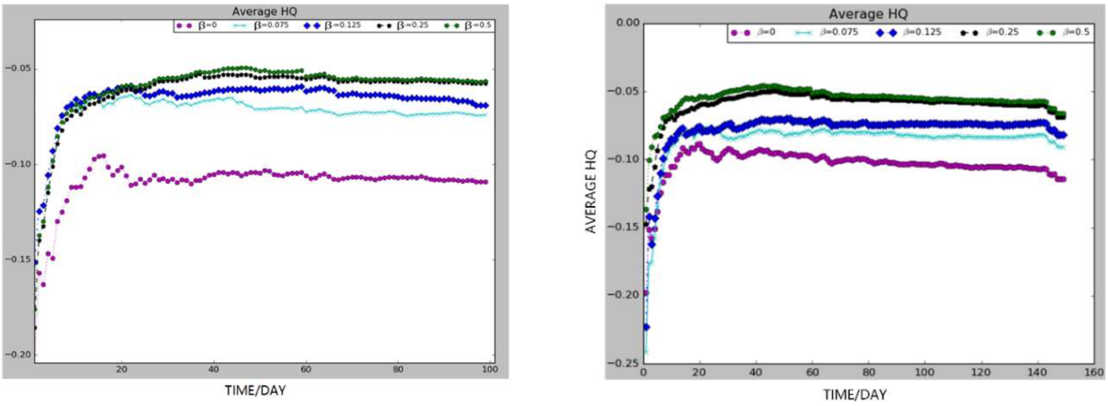
Figure 12. One-year average SQ of optimized evolutionary using different β in LD2011_2014 dataset.

6.3. Effect of β on History Quality

Figures 13 and 14 show how the parameter β affects the history quality in different datasets. The x-axis represents the day and the y-axis represents the history quality. From these figures we observe the following:

- When $\beta = 0$ no evolutionary clustering process is on, thus, the history quality is worse than the others.
- The larger the β, the higher the history quality. When β gets larger than 0.25, the history quality changes little.

- As the time span and the number of users increase, the history quality gradually fluctuates because the user's electricity consumption behavior is gradually changing over time due to seasonal changes and other factors mentioned before.



(**a**) 100 days' average *HQ* of optimized evolutionary using different *β*.

(**b**) 150 days' average *HQ* of optimized evolutionary using different *β*.

Figure 13. One hundred days' and 150 days' average *HQ* of optimized evolutionary using different *β* in Shanghai dataset.

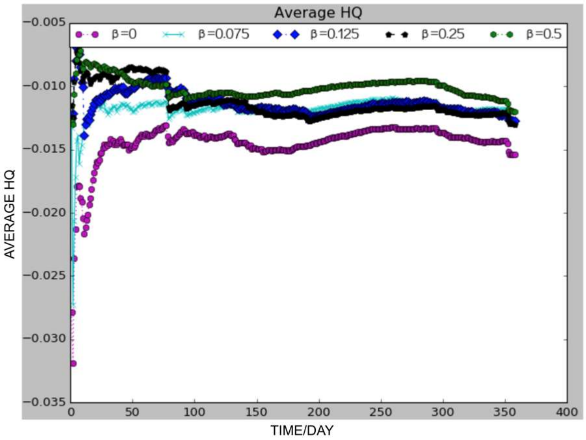


Figure 14. One-year average *SQ* of optimized evolutionary using different *β* in LD2011_2014 dataset.

From Figures 11–14, it can be seen that the value of parameter *β* around 0.075 to 0.125 would be ok, which can maximize two conflict goals, *SQ* and *HQ*. In the next part, we will provide a more convincing discussion about choosing the optimal *β*.

6.4. Adjusting Optimal *β*

To comprehensively demonstrate the optimal *β* when applying the optimized algorithm, a weighted sum of *SQ* and *HQ* (WSeH), is used as the evaluation indicator of choosing the optimal *β*. From the beginning, we have emphasized that maximizing two conflict measurements *SQ* and *HQ* is our goal, and the indicator, WSH, is the concrete goal.

$$\mathrm{WSH} = paraH * averHQ_{sta} + paraS * averSQ_{sta} \tag{9}$$

$$s.t. \quad paraH \geq 0,\ paraS \geq 0,\ paraS + paraH = 1 \tag{10}$$

The calculation formula of WSH is as above, where the values of paraH and paraH are user-defined with considering of evolution. In order to make *SQ* and *HQ*, the two indicators with different units can be compared and weighted. We must apply min-max standardization to the average HQ and average SQ before calculation. In our experiments, paraH is set 0.7, and our core concept is still evolutionary clustering of user consumption behavior, while the proposed optimized algorithm improves, keeping SQ based on guaranteeing the stable value of HQ.

Figure 15 shows the changes of WSH according to time when applying OPEC to LD2011 dataset with different values of β. It is clearly seen that in WSH, the proposed algorithm with β = 0.075 outperforms other parameter settings.

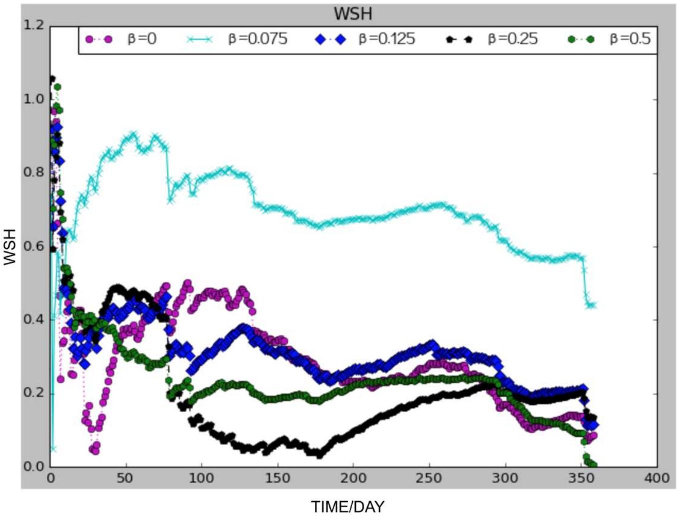


Figure 15. One-year average SQ of optimized evolutionary using different β in LD2011_2014 dataset.

It is worth noting that 0.075 is just for reference when users apply OPEC with a cold start, to get the best possible results. Thus, they can make smaller-grained parameter adjustments based on this benchmark, or they can revise the weights of SQ and HQ according to their own consideration. What is more, dynamically selecting the constant to optimize WSH (or momentarily WSH, without time averages) might give better results, which needs further improvement.

7. Case Study

In this section, we will apply optimized evolutionary clustering to analyze power data. There are several figures presenting the results; Figure 16 shows the evolutionary clustering result on a certain person. The x-axis represents time with the unit of day, and the y-axis represents the evolutionary clustering result. The result has 10 categories which are labeled from 1 to 10, and the size of the class measured by the label is the more power consumption the class represents. From the figure we conclude the following points:

- Optimized evolutionary clustering has good smoothing feature. We can observe from Figure 16 that the user action can be divided into two stages. In each stage, the clustering results are similar.
- Optimized evolutionary clustering acts greatly between stages. The point at 1st February 2015 differs significantly from the previous day, and there is no smoothing process because it does not require it.
- Optimized evolutionary clustering can find abnormal behaviors. The user action on 7th February 2015 is totally different from the points around it.

Then it comes the question, "does the user's action change significantly around the point of 1st February 2015?", we can dive deep into the user's power consumption curves to see the user used power changes around 1st February 2015. Figure 17 shows the user's two power consumption curve around 1st February 2015. The figure shows that the user tends to use more power before 1st February 2015 and it seems that the user used the power more frequently before 1st February 2015.

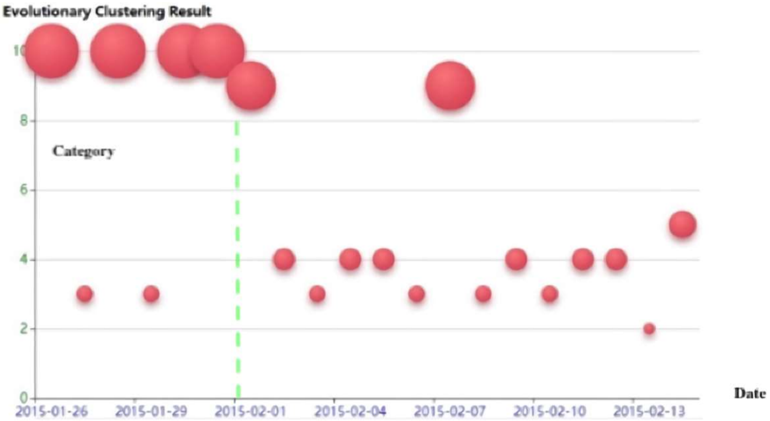


Figure 16. Evolutionary clustering result on a certain user.

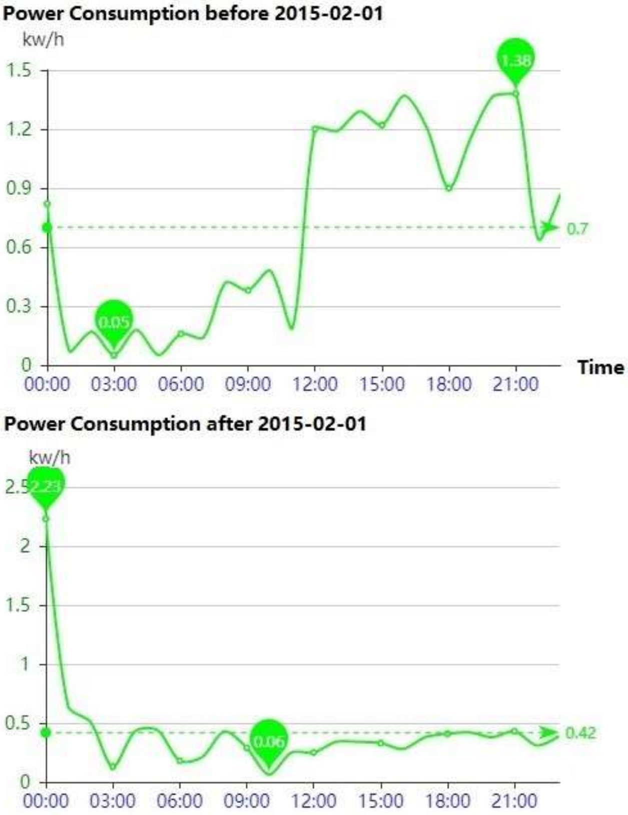


Figure 17. Power consumption curve around 1st February 2015.

In summary, the optimized evolutionary clustering is good at analyzing user power consumption mode which changes with time, and it can also identify users' abnormal actions. By understanding the characteristics and the evolutionary trends of users, we can provide the basis for the decision-making of load balancing in power grid.

8. Conclusions

In this paper, we apply evolutionary clustering to user daily load data analysis and present a new framework to optimize evolutionary clustering. In our new framework, we deal with the situation when data change dramatically in a short period of time, in which former frameworks of evolutionary clustering do not work well. Our framework named "Proper Restart" can decide when to restart the evolutionary progress and improve snapshot quality significantly.

By understanding the trends of the users' power consumption modes, we can detect users' possible power consumption modes and set the operation strategy to improve the operations of the power system, such as load balancing. It has significant guidance for the peak load regulation in the power grid, which can reduce the operating costs of the grid and bring significant social benefits.

In future works, we will consider other information as input features, such as description of the time point (morning, afternoon, Monday, Tuesday, weekdays, weekend, holiday, etc.). These features can be utilized, and they may contribute to the performance of evolutionary clustering. Dynamically selecting the constant to optimize WSH (or momentarily WSH, without time averages) might give better results, which needs further improvement. What is more, we will perform relevant experiments, comparing with suitable competitive methods (like varying "const" instead of β).

Author Contributions: R.L. designed the algorithm, performed the experiments, and prepared the manuscript as the first author. Z.Y. and Y.Z. assisted the project and managed to obtain the load data.

Funding: This research was funded by State Grid Corporation of China grant number 520940180016.

Acknowledgments: This work is supported by the State Grid Corporation of China under the project title: "The Improved Core Analysis Algorithms and Utilities for Smart Grid Big Data". (520940180016).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, X.; Jiang, X.; Qian, J.; Chen, H.; Song, J.; Huang, L. A classifying and synthesizing method of power consumer industry based on the daily load profile. *Autom. Electr. Power Syst.* **2010**, *10*, 012.
2. Zhang, S.; Liu, J.; Zhao, B.; Cao, J. Cloud computing-based analysis on residential electricity consumption behavior. *Power Syst. Technol.* **2013**, *37*, 1542–1546.
3. Lin, S.; Huang, N.; Zhao, L. Domestic daily load curve modeling based on user behavior. *Electr. Power Constr.* **2016**, *37*, 114–121.
4. Benítez, I.; Quijano, A.; Díez, J.L.; Delgado, I. Dynamic clustering segmentation applied to load profiles of energy consumption from Spanish customers. *Int. J. Electr. Power Energy Syst.* **2014**, *55*, 437–448. [CrossRef]
5. McLoughlin, F.; Duffy, A.; Conlon, M. A clustering approach to domestic electricity load profile characterisation using smart metering data. *Appl. Energy* **2015**, *141*, 190–199. [CrossRef]
6. Rhodes, J.D.; Cole, W.J.; Upshaw, C.R.; Edgar, T.F.; Webber, M.E. Clustering analysis of residential electricity demand profiles. *Appl. Energy* **2014**, *135*, 461–471. [CrossRef]
7. Oh, S.H.; Lee, W.S. An anomaly intrusion detection method by clustering normal user behavior. *Comput. Secur.* **2003**, *22*, 596–612.
8. Maia, M.; Almeida, J.; Almeida, V. Identifying user behavior in online social networks. In Proceedings of the 1st workshop on Social network systems, Glasgow, UK, 1 April 2008; pp. 1–6.
9. Wang, G.; Zhang, X.; Tang, S.; Zheng, H.; Zhao, B.Y. Unsupervised clickstream clustering for user behavior analysis. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, San Jose, CA, USA, 7–12 May 2016; pp. 225–236.
10. Falkowski, T.; Barth, A.; Spiliopoulou, M. Studying community dynamics with an incremental graph mining algorithm. In Proceedings of the AMCIS 2008 Proceedings, Toronto, ON, Canada, 14–17 August 2008.
11. Ahmed, R.; Karypis, G. Algorithms for mining the evolution of conserved relational states in dynamic networks. *Knowl. Inf. Syst.* **2012**, *33*, 603–630. [CrossRef]
12. Gupta, M.; Aggarwal, C.C.; Han, J.; Sun, Y. Evolutionary clustering and analysis of bibliographic networks. In Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining, Kaohsiung, Taiwan, 25–27 July 2011.
13. Rana, C.; Jain, S.K. An evolutionary clustering algorithm based on temporal features for dynamic recommender systems. *Swarm Evol. Comput.* **2014**, *14*, 21–30. [CrossRef]
14. Rana, C.; Jain, S.K. An extended evolutionary clustering algorithm for an adaptive recommender system. *Soc. Netw. Anal. Min.* **2014**, *4*, 164. [CrossRef]
15. Chen, J.; Uliji; Wang, H.; Yan, Z. Evolutionary heterogeneous clustering for rating prediction based on user collaborative filtering. *Swarm Evol. Comput.* **2018**, *38*, 35–41. [CrossRef]

16. Chen, J.; Wei, L.; Uliji; Zhang, L. Dynamic evolutionary clustering approach based on time weight and latent attributes for collaborative filtering recommendation. *Chaos Solitons Fractals* **2018**, *114*, 8–18. [CrossRef]
17. Chakrabarti, D.; Kumar, R.; Tomkins, A. Evolutionary clustering. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, Philadelphia, PA, USA, 20–23 August 2006; pp. 554–560.
18. Ding, W.; Lin, C.T.; Prasad, M. Hierarchical co-evolutionary clustering tree-based rough feature game equilibrium selection and its application in neonatal cerebral cortex MRI. *Expert Syst. Appl.* **2018**, *101*, 243–257. [CrossRef]
19. Chi, Y.; Song, X.; Zhou, D.; Hino, K.; Tseng, B.L. Evolutionary spectral clustering by incorporating temporal smoothness. In Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, San Jose, CA, USA, 12–15 August 2007; pp. 153–162.
20. Xu, X.; Liao, Z.; He, P.; Fan, B.; Jing, T. Evolutionary Tree Spectral Clustering. In *Advances in Computer Communication and Computational Sciences*; Springer: Singapore, 2019; pp. 259–267.
21. Wang, L.; Rege, M.; Dong, M.; Ding, Y. Low-rank kernel matrix factorization for large-scale evolutionary clustering. *IEEE Trans. Knowl. Data Eng.* **2010**, *24*, 1036–1050. [CrossRef]
22. Xu, K.S.; Kliger, M.; Hero Iii, A.O. Adaptive evolutionary clustering. *Data Min. Knowl. Discov.* **2014**, *28*, 304–336. [CrossRef]
23. Day, S.; Brown, J.; Thomas, Z.; Bass, L.; Dozier, G. Adversarial Authorship, AuthorWebs, and Entropy-Based Evolutionary Clustering. In Proceedings of the 2016 25th International Conference on Computer Communication and Networks (ICCCN), Waikoloa, HI, USA, 1–4 August 2016; pp. 1–6.
24. Chen, G.; Luo, W.; Zhu, T. Evolutionary clustering with differential evolution. In Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC), Beijing, China, 6–11 July 2014; pp. 1382–1389.
25. Amelio, A.; Pizzuti, C. Evolutionary clustering for mining and tracking dynamic multilayer networks. *Comput. Intell.* **2017**, *33*, 181–209. [CrossRef]
26. Al-Sharoa, E.; Al-khassaweneh, M.; Aviyente, S. Low-rank Estimation Based Evolutionary Clustering for Community Detection in Temporal Networks. In Proceedings of the ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 5381–5385.
27. Shukri, S.; Faris, H.; Aljarah, I.; Mirjalili, S.; Abraham, A. Evolutionary static and dynamic clustering algorithms based on multi-verse optimizer. *Eng. Appl. Artif. Intell.* **2018**, *72*, 54–66. [CrossRef]
28. Majdi, A.; Beiki, M. Applying evolutionary optimization algorithms for improving fuzzy C-mean clustering performance to predict the deformation modulus of rock mass. *Int. J. Rock Mech. Min. Sci.* **2019**, *113*, 172–182. [CrossRef]
29. Ma, J.; Wang, Y.; Gong, M.; Jiao, L.; Zhang, Q. Spatio-temporal data evolutionary clustering based on MOEA/D. In Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation, Dublin, Ireland, 12–16 July 2011; pp. 85–86.
30. Mukhopadhyay, A.; Maulik, U.; Bandyopadhyay, S. A Survey of Multiobjective Evolutionary Clustering. *ACM Comput. Surv.* **2015**, *47*, 61. [CrossRef]
31. Garcia-Piquer, A.; Bacardit, J.; Fornells, A.; Golobardes, E.; Herrera, A.F. Scaling-up multiobjective evolutionary clustering algorithms using stratification. *Pattern Recognit. Lett.* **2017**, *93*, 69–77. [CrossRef]
32. Zhao, F.; Fan, J.; Liu, H.; Lan, R.; Chen, C.W. Noise robust multiobjective evolutionary clustering image segmentation motivated by the intuitionistic fuzzy information. *IEEE Trans. Fuzzy Syst.* **2018**, *27*, 387–401. [CrossRef]
33. Zhao, F.; Liu, H.; Fan, J.; Chen, C.W.; Lan, R.; Li, N. Intuitionistic fuzzy set approach to multi-objective evolutionary clustering with multiple spatial information for image segmentation. *Neurocomputing* **2018**, *312*, 296–309. [CrossRef]