Clustering Study of Commercial and Industrial Power Load Profiles Based on Knndp-Pfcm with Weighted Fusion Similarity Measure

Tao Xie, Chaobin Zhang, Zhiqiang Tan, Jie Xu

***Abstract*—The volume of power load data has increased significantly with the proliferation of smart grids. Clustering of power load profiles plays a crucial role in identifying distinct consumption patterns and load characteristics. However, traditional clustering algorithms encounter challenges, such as difficulties in selecting suitable initial cluster centers and the inability of Euclidean distance to adequately capture the complexity of load profile data. These issues can result in suboptimal clustering performance. Therefore, this study proposes a novel clustering algorithm to address these challenges. The proposed algorithm uses an improved density peak clustering method to determine the initial cluster centers, incorporates an adaptive weighting approach to account for the dynamic nature of the load profiles, and introduces a novel similarity measure that combines Euclidean and Pearson distances to more effectively capture the relationships between the load profiles. The performance of the proposed algorithm is evaluated using two datasets of smart meter data from Southwest China. The computational results demonstrate that the proposed algorithm outperforms other algorithms in terms of the sum of squared errors within clusters (SSE), the Davies-Bouldin Index (DBI), the silhouette coefficient (SC), and the Calinski-Harabasz Index (CHI), as well as in algorithmic stability and iterative efficiency. The proposed algorithm also exhibits strong robustness when applied to datasets with different levels of noise perturbation. This study also explores the impact of different algorithm parameters and initial clustering centers on the clustering results through a comprehensive analysis. Overall, the results indicate that the proposed method exhibits high robustness and strong applicability in practical scenarios.**

***Index Terms*—Smart grid, Load curve, Data weighting, Fusion similarity, Power load clustering**

I. Introduction

With the continuous development of smart grids, the penetration rate of smart meters has been increasing, resulting in the accumulation of vast amounts of electricity consumption data. Power load profile clustering refers to the segmentation of users based on their power consumption characteristics, grouping users with similar consumption patterns. This technique helps to understand the electricity consumption behavior of users and assists with demand-side response and high-precision load forecasting [1]. In addition, power load profile clustering can be used for abnormal power usage detection [2], load control, distribution network planning, etc. Therefore, the analysis of power load profile clustering is of significant importance [3].

Numerous studies have explored clustering of power load profiles. For example, Hu *et al*. introduced a set of interpretable and distinguishable load features and proposed a two-step feature-based clustering method to classify photovoltaic (PV) and non-PV load profiles [4]. Additionally, a quantitative component-based method was applied to analyze load magnitudes and changes in load patterns. Yilmaz *et al*. proposed a clustering method based on five key features of load profile shapes, which was used to cluster daily load profiles into various categories [5]. They concluded that averaging the load data suppresses the diversity of electricity usage patterns. Ryu *et al*. proposed a compressed K-means clustering algorithm for annual load distribution using a deep convolutional autoencoder [6]. This method visualizes load patterns, daily characteristics, and seasonal variations as a two-dimensional load image, effectively capturing year-round load behavior through distinct vertical and horizontal features.

Currently, mainstream clustering algorithms are categorized into three types: hierarchy-based, density-based, and partition-based [7]. Among these, partition-based clustering algorithms, such as K-means and Fuzzy C-means (FCM), are the most widely used due to their computational efficiency and simplicity [8, 9]. However, classical partition-based clustering algorithms face a significant challenge: sensitivity to initial cluster centers [10]. Classical algorithms typically randomly select the initial clustering centers, which may increase the number of algorithm iterations and reduce the efficiency of the algorithm or even lead to inconsistent clustering results [11]. As a result, the stability of the algorithm is considerably affected.

To overcome this drawback, establishing fixed initial clustering centers through algorithmic approaches has become an important research direction for many improvement methods. Many researchers have focused on pre-selecting optimal initial clustering centers using optimization algorithms or other clustering methods. It is widely believed that selecting suitable initial centers can reduce the number of iterations required for convergence and improve the overall stability and accuracy of clustering algorithms. For instance, the density-based canopy algorithm selects the point with the highest density as the first clustering

Manuscript received August 13, 2024; revised February 14, 2025.

Tao Xie is a senior engineer of Chongqing University of Posts and Telecommunications, Chongqing 400065, China (e-mail: xietao@cqupt.edu.cn).

Chaobin Zhang is a postgraduate student of Chongqing University of Posts and Telecommunications, Chongqing 400065 , China (e-mail: zcbin0202@163.com).

Zhiqiang Tan is an engineer of Huaneng Chongqing Energy Sales Co., Ltd, Chongqing 401121 , China (e-mail: hncqnyxs@163.com).

Jie Xu is a senior engineer of State Grid Chongqing Electric Power Dispatch and Control Center, Chongqing 400014, China (e-mail: xuj00053@126.com).

center [12], while the remaining centers are automatically determined to ensure algorithmic stability. Zhang *et al.* Zhang et al. proposed a quantum-inspired genetic algorithm to identify optimal initial clustering centers, achieving a balance between clustering accuracy and computational efficiency [13]. Li *et al.* employed the K-means++ algorithm to select optimal initial clustering centers and used the powerful K-means++ algorithm for clustering [14], which can improve the stability and accuracy of the algorithm. The Density Peaks (DP) algorithm is a well-known density-based clustering algorithm, a method that does not require iteration and finds the clustering center at once. Liu *et al.* proposed a Fuzzy C-means algorithm based on density peaks [15], which addresses the sensitivity issue of the initial clustering center and improves the clustering accuracy.

Meanwhile, in traditional clustering algorithms, data points and features are treated equally. However, in practical applications, features of different dimensions often vary in importance and should be assigned different weights. Several clustering algorithms have been developed to address feature weighting. For example, an entropy-weighted K-means clustering algorithm (EWKM) [16] was proposed for clustering high-dimensional sparse data subspaces. Additionally, a feature parsimonious fuzzy c-means clustering algorithm [17] was proposed, which automatically computes feature weights and reduces the number of irrelevant features to improve algorithmic effectiveness and practicality. Yang *et al.* proposed a feature-weighted reduced possibility c-means algorithm (FWR-PCM) [18], which eliminates irrelevant features and reduces dimensionality, thereby enhancing clustering performance. In contrast, research on data weighting, particularly in the context of power load profiles, remains relatively scarce. This limited exploration may be due to the unique characteristics and specific significance of load profiles. Lin *et al.* proposed a hierarchical clustering algorithm based on weighted Pearson distance. This approach assigns fixed weights to data points according to their fluctuations and uses Pearson distance to measure both the overall profile and local similarity of the power load profiles [19].

In functional load curve clustering, traditional similarity measures, such as Euclidean distance, have notable limitations. These methods primarily assess the similarity of numerical distributions but fail to effectively capture trend similarities between load curves [20]. To address this limitation, several alternative similarity measures have been proposed. For example, methods such as Manhattan distance, Mahalanobis distance, dynamic time warping (DTW), edit distance, Pearson correlation, cross-correlation, and Fourier coefficient-based distance have been explored [21]. Yang *et al*. proposed an improved dynamic time warping distance (GD-DTW) for measuring the similarity between two unequal-length time series [22].

To address the limitations of traditional clustering algorithms in power load profile clustering, this study proposes a novel approach. First, the data are weighted based on the fluctuation characteristics of the load curve, and a new similarity measure is proposed by combining Euclidean distance and Pearson distance. Then, the enhanced K-nearest neighbors density peak (KNNDP) algorithm is employed to select the initial cluster centers. Finally, clustering is performed using the possibility fuzzy c-means (PFCM) algorithm, based on the improved similarity measure, to obtain the final results. The proposed WF-KNNDP-PFCM algorithm provides several innovations and contributions, as outlined below.

Optimization of Initial Clustering Centers: This study combines the KNNDP algorithm based on density clustering with the PFCM algorithm based on partitioning. By using KNNDP to select optimal initial clustering centers, the quality and stability of the clustering algorithm are significantly improved.

Fusion Similarity Measure: The Euclidean distance is employed to measure the numerical distribution characteristics of load curves, while the Pearson distance captures the trend change characteristics. An improved entropy weighting method is utilized to combine these two measures, resulting in a robust similarity measure for the clustering algorithm.

Novel Data Weighting Method: A new approach to data weighting is proposed. Based on the trend change characteristics between two neighboring points on the load curve and the change amplitude, data points are classified into non-load trend change points, ordinary load trend change points, and important load trend change points. Adaptive weighting according to the change amplitude helps accurately capture the load trend change characteristics.

The remainder of this paper is organized as follows. Section 2 provides an overview of clustering algorithms and related evaluation indices. Section 3 introduces the algorithms pertinent to the method proposed in this study. Section 4 delves into the practical application of the proposed algorithm, presenting experimental results and comparisons with other algorithms. Finally, Section 5 concludes the paper and suggests potential avenues for future research.

II. Background

A. Clustering Concepts

Clustering is a fundamental unsupervised data mining technique used to uncover inherent patterns within a dataset. The primary objective of clustering is to group data points into clusters based on their similarity, ensuring that data points within the same cluster exhibit high similarity, while points in different clusters have low similarity. The DP clustering algorithm, the PFCM algorithm, and the cluster evaluation metrics relevant to this study's methodology will be discussed in the following sections.

B. Density Peak Clustering Algorithm

The DP clustering algorithm is based on density peaks, which involves calculating the local density and relative distance of each sample to construct a decision diagram. A point is considered a cluster center if it satisfies both of the following conditions: its local density is higher than that of its neighboring points, and its relative distance to other points with higher local densities is greater. This means that these cluster centers are far from points with higher local densities.

The local density calculation in the DP algorithm can be performed using either a truncation kernel or a Gaussian kernel. The truncation kernel calculates the number of samples within a truncated distance from point x_i, whereas

the Gaussian kernel computes the local density as the sum of the Gaussian distances from all samples to point x_i. By definition, the truncation kernel is more suitable for clustering discrete data, while the Gaussian kernel is better suited for clustering continuous data.

Truncation kernel:

$$\rho_i = \sum_j \chi\left(d_{ij} - d_c\right), \chi = \begin{cases} 0, d_{ij} - d_c \geq 0 \\ 1, d_{ij} - d_c < 0 \end{cases} \tag{1}$$

Gaussian kernel:

$$\rho_i = \sum_{i \neq j} \exp\left[-\left(\frac{d_{ij}}{d_c} \right)^2 \right] \tag{2}$$

Where d_{ij} represents the Euclidean distance between samples x_i and x_j. d_c is the truncation distance, a parameter that needs to be set experimentally. Typically, d_c is chosen as the distance at the 2% position after sorting all sample distances in ascending order.

The relative distance is defined as the distance between the sample x_i and other points with higher local densities. If there are data points with higher local densities than x_i, the relative distance is the minimum Euclidean distance from x_i to any of these higher-density points. If x_i has the highest local density among all samples, the relative distance is the maximum distance from x_i to any other sample point. The specific formula is as follows:

$$\delta_i = \begin{cases} \min\limits_{j:\rho_j > \rho_i} \left(d_{ij}\right) \\ \max\limits_{j \neq i} \left(d_{ij}\right) \end{cases} \tag{3}$$

Decision Graph: The decision graph is constructed by plotting the local density on the x-axis and the relative distance on the y-axis. Points that have higher local densities and greater relative distances are more likely to serve as clustering centers, typically appearing in the upper right quadrant of the graph.

C. Possibility Fuzzy c-means Algorithm

The Possibility Fuzzy C-means (PFCM) algorithm, proposed by Pal et al., combines the advantages of fuzzy c-means and possibility c-means to address the limitations of FCM in handling noise and outliers. The algorithm defines an objective function that incorporates two types of memberships: possibility membership and fuzzy membership. Possibility membership quantifies the degree of absolute typicality of a point within a cluster, whereas fuzzy membership quantifies the relative degree of membership to a cluster.

The formula for the objective function is presented below:

$$\begin{aligned} function = & \sum_{j=1}^{n} \sum_{i=1}^{c} \left(c_F {\mu_{ij}}^m + c_p {t_{ij}}^\eta \right) \left\| x_j - v_i \right\|_A^2 + \\ & \sum_{i=1}^{c} \gamma_i \sum_{j=1}^{n} \left(1 - t_{ij}\right)^\eta, \sum_{k=1}^{c} \mu_{kj} = 1 \end{aligned} \tag{4}$$

Where C_F and C_P are two constants defined by the user to characterize the relative importance of the fuzzy and possibility terms in the algorithm. When noise and outliers are present in the data, a larger value of C_P enhances the algorithm's effectiveness. In addition, t_{ij} is an element in the typicality matrix, which indicates the typicality of the *j-th* element in the *i-th* cluster. η is the likelihood index (typically set to 2). γ_i is computed by the formula shown in 5, where K is typically 1. The process of minimizing the objective function, the typicality matrix, the membership matrix, and the center of the clusters is updated with the formulas shown in equations (6) to (8).

$$\gamma_i = K \frac{\sum_{j=1}^{n} {\mu_{ij}}^m \left\| x_j - v_i \right\|_A^2}{\sum_{j=1}^{n} \left(\mu_{ij}\right)^m} \tag{5}$$

$$t_{ij} = \left[1 + \left(\frac{c_p}{\gamma_i} \left\| x_j - v_i \right\|_A^2 \right)^{\frac{1}{\eta - 1}} \right]^{-1} \tag{6}$$

$$v_i = \frac{\sum_{j=1}^{n} \left(c_F {\mu_{ij}}^m + c_P {t_{ij}}^\eta \right) x_j}{\sum_{j=1}^{n} \left(c_F {\mu_{ij}}^m + c_P {t_{ij}}^\eta \right)} \tag{7}$$

$$\mu_{ij} = \left[\sum_{k=1}^{c} \left(\frac{\left\| x_j - v_i \right\|_A^2}{\left\| x_j - v_k \right\|_A^2} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{8}$$

The PFCM algorithm involves several parameters that must be initialized before iteration. These include selecting appropriate values for C_F, C_P, and m; setting the stopping threshold for the algorithm; defining the maximum number of iterations; and determining the initial clustering centers.

The detailed iterative steps of PFCM are as follows:

(1) Initialize the clustering centers and calculate γ_i and μ_{ij} according to Eqs. (5) and (8).
(2) Update the typicality value t_{ij} according to Eq. (6).
(3) Update the affiliation value μ_{ij} according to Eq. (8).
(4) Update the clustering center as well as γ_i, according to Eqs. (7) and (5).
(5) If the termination condition is met or the number of iterations exceeds the set limit, end the algorithm and output the membership matrix and the clustering centers. If the condition is not met, increment the iteration count and return to step 2.

D. Cluster Evaluation Index

1. Within-cluster sum of square error

$$SSE = \sum_{j=1}^{k} \sum_{x_i \in C_j} \| x_i - u_j \|^2 \tag{9}$$

Where x_i is a data point in the dataset, μ_j is the centroid of Cluster C_j. Theoretically, a smaller value is preferable. However, as the number of clusters increases, the index tends to show an overall downward trend. This does not necessarily indicate that the corresponding number of clusters is optimal.

2. Davies–Bouldin index

The Davies-Bouldin Index (DBI), also known as the categorization suitability index, was proposed by David L. Davies and Donald W. Bouldin to evaluate the effectiveness of clustering algorithms. This index measures clustering performance by calculating the average maximum similarity between each cluster and the most similar one. The specific formula is as follows:

$$DBI = \frac{1}{N} \sum_{i=1}^{N} \max_{i \neq j} \left(\frac{\overline{s_i} + \overline{s_j}}{\| w_i - w_j \|^2} \right) \tag{10}$$

Where N denotes the number of clusters, S_i represents the average distance from the data points within a cluster to the cluster's centroid, and $\| w_i - w_j \|^2$ denotes the distance

between the centroids of two clusters. A smaller DBI value indicates better clustering, characterized by compact clusters and a high degree of separation between clusters. Conversely, a larger DBI value suggests poorer clustering, indicated by dispersed clusters or a low degree of separation between clusters.

3. Silhouette Coefficient

The silhouette coefficient combines the cohesion and separation of clustering and is used to evaluate the effectiveness of clustering. Its value range is [-1,1]. If the silhouette coefficient is closer to 1, it means the clustering performance is better, and if the silhouette coefficient value is closer to -1, it means the clustering performance is worse. The specific formula is as follows.

$$I_{\mathrm{SIL}} = \frac{b_i - a_i}{\max\{a_i, b_i\}} \tag{11}$$

Where b_i denotes the minimum average distance of the *i-th* sample to all samples in other clusters, reflecting the degree of dispersion between the sample and other clusters, and a_i denotes the average distance of the *i-th* sample to other sample points in the cluster where it is located, reflecting the degree of closeness of samples in the same cluster. Averaging the silhouette coefficients of all samples yields the average silhouette coefficient, I_{SILmean}, which reflects the overall quality of the clusters. A value closer to 1 indicates better overall clustering performance, while a value closer to -1 indicates worse overall clustering performance.

4. Calinski-Harabasz

Also known as the variance ratio criterion, it is an internal index used to assess the quality of clustering results with the following formula.

$$\begin{cases} CH(k) = \dfrac{SSB}{SSW}\dfrac{m-k}{k-1} \\ SSW = \displaystyle\sum_{i=1}^{m} \left\| x_i - C_{pi} \right\|^2 \\ SSB = \displaystyle\sum_{j=1}^{k} n_j \left\| C_j - \overline{C} \right\|^2 \end{cases} \tag{12}$$

Where SSB (Sum of Squares Between) represents the overall inter-cluster dispersion, while SSW (Sum of Squares Within) represents the degree of compactness within each cluster. *m* denotes the number of samples, and *k* denotes the number of clusters. A higher value of the CH index denotes a better clustering result, with compactness of the samples within the clusters and high inter-cluster dispersion, and a lower value of the CH index denotes a poorer clustering result, with dispersed samples within clusters or low inter-cluster dispersion.

III. Methodology

A. K nearest-neighbors Density Peak Algorithm

Although the DP clustering algorithm provides greater stability in selecting initial clustering centers, it is not without its limitations. The effectiveness of the DP clustering algorithm diminishes when significant differences exist between clusters in the dataset. Furthermore, the calculation of local density is dependent on the choice of cut-off distance. However, in the DP algorithm, the truncation distance is determined by the global distribution of the data, neglecting local information. Consequently, when the dataset exhibits uneven density distributions, the performance of the DP clustering algorithm may degrade.

To address the shortcomings of the traditional DP algorithm, this study incorporates the concept of K nearest neighbors (KNN) into the DP algorithm. By redefining local density and proposing an adaptive method for calculating the local cut-off distance, the limitations of the manually set truncation distance are mitigated.

K nearest-neighbors: The K nearest neighbors of sample x_i are the set of K sample points that are closest to x_i, denoted by KNN(i):

$$KNN(i) = \left\{ x_1, x_2, ..., x_K \middle| \begin{array}{l} x_i \in D, \\ \forall p \in D - \{x_1, x_2, ..., x_K\}, \\ dxp \geq dxx_i \end{array} \right\} \tag{13}$$

K nearest neighbor distance: The K nearest neighbor distance is defined as the sum of the distances from x_i to all its K nearest neighbor objects. Its calculation formula is as follows:

$$K_{dist}(i) = \sum_{j \in KNN(i)} d_{ij} \tag{14}$$

K Nearest Neighbor Similarity:

$$dis_i^K = \frac{1}{K} \frac{\sum_{j=KNN(i)} \sum_{p=KNN(j)} d_{pj}}{K_{dist}(i)} \tag{15}$$

Where K refers to the number of nearest neighbors of sample x_i. The numerator is defined as the sum of the distances between the nearest neighbor objects of the sample x_i and its corresponding nearest neighbor objects, which reflects the degree of closeness of the sample x_i in the local range of samples, indicating that the selected initial clustering center must be the point with the highest density value.

Nearest neighbor density:

$$p_i^K = \exp\left(-\frac{dis_i^K}{d_i^c} \right)^2 \tag{16}$$

Adaptive cut-off distance:

$$d_i^c = \mu_i^K + \sqrt{\frac{1}{K-1} \sum_{j=1}^{K} \left(d_j^K - \mu_i^K \right)^2} \tag{17}$$

$$d_j^K = \max_{m \in KNN(j)} \{ d_{jm} \} \tag{18}$$

$$\mu_i^K = \frac{1}{K^2} \sum_{i=1}^{K} \sum_{j=1}^{K} d_{ij} \tag{19}$$

Where d_i^c is the local truncation distance of sample x_i, d_j^K is the maximum distance between sample point x_j and its K nearest-neighbors. μ_i^K is the average distance between two K nearest-neighbors of sample x_i. In equation (17), the second term is the standard deviation of the distance between the sample x_i and K nearest-neighbors samples. The degree of dissimilarity is introduced to enhance the standard deviation of the distances between the K nearest-neighbors sample points, which helps mitigate the influence of individual outliers on the cut-off distance. This approach makes the selection of initial clustering centers more adaptable and robust, especially for complex datasets. The cut-off distance d_i^c changes continuously due to different neighborhoods, which can adapt to the distribution of different datasets. Meanwhile, the subjectivity of the initial

manual selection of cutoff distance is eliminated, and it is not affected by the global data distribution.

To further reduce the subjectivity in the selection of initial clustering centers, the concept of decision value is introduced into the algorithm. The formula for calculating the decision value is as follows:

$$r_i = \rho_i \cdot \delta_i \tag{20}$$

Obviously, the larger the value of r_i, the greater the possibility of becoming the initial clustering center. In the algorithm, the decision value of each sample is calculated, and the samples corresponding to the k largest values are selected as the k initial clustering centers. This approach eliminates the subjectivity of human selection of initial clustering centers and achieves automatic selection of initial clustering centers. The process for selecting initial clustering centers using the KNNDP algorithm is outlined below.

Algorithm 1 Initialize centers

Input: D: dataset, K:nearest neighbor parameter,k: the number of clusters
Output: $m=\{m_1,m_2,...,m_k\}$: the initial cluster centers
1:Initialzation: $m=\emptyset, n=$length(D);
2:Construct the K nearest neighbor sets according to Eq.(13);
3:for each point $x_i \in$ D **do**
4: compute nearest neighbor similarity dis_i^K according to Eq.(15);
5: compute δ_i, p_i^K and d_i^c according to Eq.(3).Eq.(16) and Eq.(17);
6: compute the decision value $\boldsymbol{\gamma}_i$ according to Eq.(20);
7: $\gamma = \text{sort}(\gamma, \text{'descend'})$
8:end for
9:Select the points corresponding to the first k decision values as the initial clustering center
10:Return $m=\{m_1, m_2, ..., m_k\}$

B. Fusion Similarity Measure

Power load profiles are typically characterized as functional-type data, exhibiting high dimensionality and a strong temporal order. Traditional Euclidean distance, which focuses solely on the distribution of values at corresponding time points on the power load profiles, may fail to adequately capture the shape similarity between load profiles. In contrast, pearson distance can capture the trend-change characteristics of the power load profiles. To address these issues, a fused Euclidean-Pearson similarity measure based on data weighting is proposed. This approach aims to enhance clustering performance by integrating both distance measures.

The weighted Euclidean distance is calculated as follows.

$$d\_e = \sqrt{\sum_{i=1}^{n} w_{x_i} w_{y_i} \left(x_i - y_i\right)^2} \tag{21}$$

Pearson distance: Unlike the geometric mean distance, the Pearson distance places more emphasis on the synchronization between the two variables. The Pearson distance also reflects the trend similarity between the load profiles. The formula for calculating it is as follows:

$$\rho\left(DLC_1, DLC_2\right) = \frac{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)\left(y_i - \bar{y}\right)}{\sqrt{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}\sqrt{\sum_{i=1}^{n}\left(y_i - \bar{y}\right)^2}} \tag{22}$$

$$D\left(DLC_1, DLC_2\right) = 1 - \rho\left(DLC_1, DLC_2\right) \tag{23}$$

Where DLC_1 denotes the first load curve, DLC_2 denotes the second load curve, x_i and y_i represent the values on the i-th dimension of the two load curves. $\bar{x}$ and $\bar{y}$ are the mean values of the two load curves, respectively.

The formula for the weighted Pearson distance is as follows.

$$m\left(DLC_1\right) = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \tag{24}$$

$$COV\left(DLC_1, DLC_2, W\right) = \frac{\sum_{i=1}^{n} w_i \left(x_i - m\left(DLC_1\right)\right)\left(y_i - m\left(DLC_2\right)\right)}{\sum_{i=1}^{n} w_i} \tag{25}$$

$$\rho = \frac{COV\left(DLC_1, DLC_2, W\right)}{\sqrt{COV\left(DLC_1, DLC_1, W\right) COV\left(DLC_2, DLC_2, W\right)}} \tag{26}$$

$$d\_p = 1 - \rho\left(DLC_1, DLC_2\right) \tag{27}$$

The fusion similarity formula is shown below.

$$dis = W\_e \cdot d\_e + W\_p \cdot d\_p \cdot \gamma \tag{28}$$

Where d_e is the weighted Euclidean distance, d_p is the weighted Pearson distance, W_e is the weight corresponding to the weighted Euclidean distance, and W_p is the weight corresponding to the weighted Pearson distance, and the weight calculation is determined by the improved entropy weighting method, which will not be repeated in this paper. γ is the proportion coefficient, which is intended to balance the orders of magnitude of the two kinds of distances. The corresponding formula for the proportion coefficient is as follows:

$$r = \frac{d\_e.\max}{d\_p.\max} \tag{29}$$

C. Data Weight

Load curve clustering aims to group a large set of curves into clusters, where curves within the same cluster are similar, while those in different clusters differ significantly. However, not all data points contribute equally to the clustering process. Points where the load trend changes are particularly important and should be given higher weights to more effectively influence the clustering. In this paper, three types of load trends are defined based on the magnitude of change between neighboring points. These trends are classified and weighted based on the changes in load behavior associated with each point. The detailed definitions are presented below.

Load Trend: X_{i-1} and X_i are the two neighboring points on the load curve and $k_i = X_i - X_{i-1}$ is the difference between the two neighboring points. Based on the sign of k_i, three load trends are defined: ASCENSION, REDUCTION, and STEADY. The specific definition is shown in equation (30).

$$PT = \begin{cases} ASCENSION, k_i > 0 \\ REDUCTION, k_i < 0 \\ STEADY, \quad k_i = 0 \end{cases} \tag{30}$$

Trend change: X_{i-1}, X_i and X_{i+1} are three consecutive points on the load curve. $k_i = X_i - X_{i-1}$, $k_{i+1} = X_{i+1} - X_i$ is the difference between the two neighboring points. k_i and k_{i+1} have nine combinations, and the nine combinations are

shown in the table below.

TABLE I
DEFINITION OF LOAD TREND CHANGES

	$ki>0$	$ki<0$	$ki=0$
$k_{i+1}>0$	AA (Ascension -Ascension)	RA (Reduction - Ascension）	SA (Steady -Ascension)
$k_{i+1}<0$	AR (Ascension - Reduction)	RR (Reduction -Reduction)	SR (Steady - Reduction)
$k_{i+1}=0$	AS (Ascension - Steady)	RS (Reduction - Steady）	SS (Steady - Steady)

Load Trend Change Points: Based on the above definitions, the three consecutive points X_{i-1}, X_i, and X_{i+1} collectively represent a change in load trend. A point X_i is identified as a load trend change point if the differences k_i and k_{i+1} (whether positive or negative) exhibit a change in trend. The six types of load trend change points are outlined in Table I, and the specific criteria for their identification are provided by the following formulas:

$$\begin{aligned} x_i \in &\left\{x_i \middle| x_i \geq x_{i-1} \,\&\, x_i > x_{i+1}\right\} \\ &\bigcup\left\{x_i \middle| x_i > x_{i-1} \,\&\, x_i \geq x_{i+1}\right\} \\ &\bigcup\left\{x_i \middle| x_i \leq x_{i-1} \,\&\, x_i < x_{i+1}\right\} \\ &\bigcup\left\{x_i \middle| x_i < x_{i-1} \,\&\, x_i \leq x_{i+1}\right\} \end{aligned} \tag{31}$$

Important Load Trend Change Points: The significance of load trend change points varies, with those showing larger changes deserving more attention and importance in the clustering process. Therefore, based on the distance D from X_i to the line formed by X_{i-1} and X_{i+1}, load trend change points are categorized into common and important types. D represents the perpendicular distance from X_i to the line segment connecting X_{i-1} and X_{i+1}. If D exceeds a threshold value ε, X_i is classified as an important load trend change point; otherwise, it is classified as a common load trend change point. The formulas for calculating D and ε are as follows:

$$D = \left(x_{i-1}, x_i, x_{i+1}\right) > \left| x_{i-1} + \frac{x_{i+1} - x_{i-1}}{2} - x_i \right| \tag{32}$$

$$\varepsilon = \left(L_{\max} - L_{\min}\right) \times epsonRate \tag{33}$$

Where L_{max} represents the maximum value of the whole load curve and L_{min} represents the minimum value of the whole load curve. epsonRate represents the threshold ratio, which can be adjusted to control the number of important load trend points.

Data weighting: To enhance the influence of specific points on the clustering process, appropriate weighting of the data points is essential. There are two methods for weighting data points. The first is fixed weighting, where a predetermined weight is assigned to both normal and important load trend points. However, this approach may not be optimal; for instance, during continuous fluctuations in the load profile over time, fixed weights may lose significance as all points receive the same weight. Additionally, assigning fixed weights to points during minor fluctuations in the load profile may result in an excessive number of weighted points.

Alternatively, adaptive weighting adjusts the weight based on the magnitude of the load profile change. This method helps mitigate the aforementioned issues by providing differentiated weights that reflect the varying importance of points in the load profile. Points with larger changes receive higher weights, thereby exerting greater influence on the clustering results. The weighting formulas for ordinary and important load trend change points are as follows:

$$W_{\text{normal}} = 1 + D / \left(L_{\max} - L_{\min}\right) \tag{34}$$

$$W_{\text{important}} = 1.5 + D / \left(L_{\max} - L_{\min}\right) \tag{35}$$

Load curve weight matrix: According to the above formula, the weight matrix W of the load curves can be calculated, assuming that there are N load curves used for clustering and each curve has a total of M points. Then, the load weight matrix of size $N \times M$ can be generated. The specific weight matrix generation algorithm is shown below.

Algorithm2. Weight Matrix Generation Algorithm
Input: D: dataset ={$load_1$,$load_2$,$load_3$…,$load_N$}, $load_i$={x_i} i=1,2,3…M,epsonRate =0.1
Output : Weight Matrix = $W_{N\times M:}$The weight matrix of all curves
1:Initialize weight matrix W with all 1.
2:for each curve $load_i$ ∈ D **do**
3: compute threshold ε according to Eq.(33);
4: **for** point x_i,i=2,3…M-1 ∈ D **do**
5: compute x_i-x_{i-1} ,x_i-x_{i+1};
6: compute vertical distance d = (x_{i-1},x_i,x_{i+1}) according to Eq.(32);
7: **if** x_i satisfies Eq.(31) and vertical distance d >ε **then**
8: compute w_i according to Eq.(34);
9: **elif** x_i satisfies Eq.(31) and vertical distance d <ε **then**
10: compute w_i according to Eq.(35);
11: **end if**
12: **end for**
13: end for
14:Return Weight Matrix = $W_{N\times M}$

D. Proposed Methodology

Fig. 1 illustrates the flowchart of the algorithm proposed in this paper, which is divided into two primary parts: the initialization of clustering centers and data weighting, followed by the application of Possibility Fuzzy C-means for clustering analysis.

The algorithm mainly includes the following key steps:

Step 1: Initialize the clustering centers. The KNNDP algorithm is used to calculate the local density and relative distance for each sample, selecting the samples with the highest decision values as the initial cluster centers.

Step 2: Generate a data weight matrix. Based on the fluctuation characteristics of load profiles, adaptive weights are assigned to different time intervals, generating a weight matrix that captures the trend features of each load profile.

Step 3: Similarity Measure Calculation. A fused similarity measure combining Euclidean distance and Pearson correlation is introduced to overcome the limitations of Euclidean distance in capturing the similarity of load profiles.

Step 4: Clustering using the proposed algorithm. Clustering is performed by optimizing membership degrees and cluster centers with the new similarity measure until convergence.

Step 5: Analysis of clustering results. The final clustering results are analyzed to interpret the distinct load patterns identified.

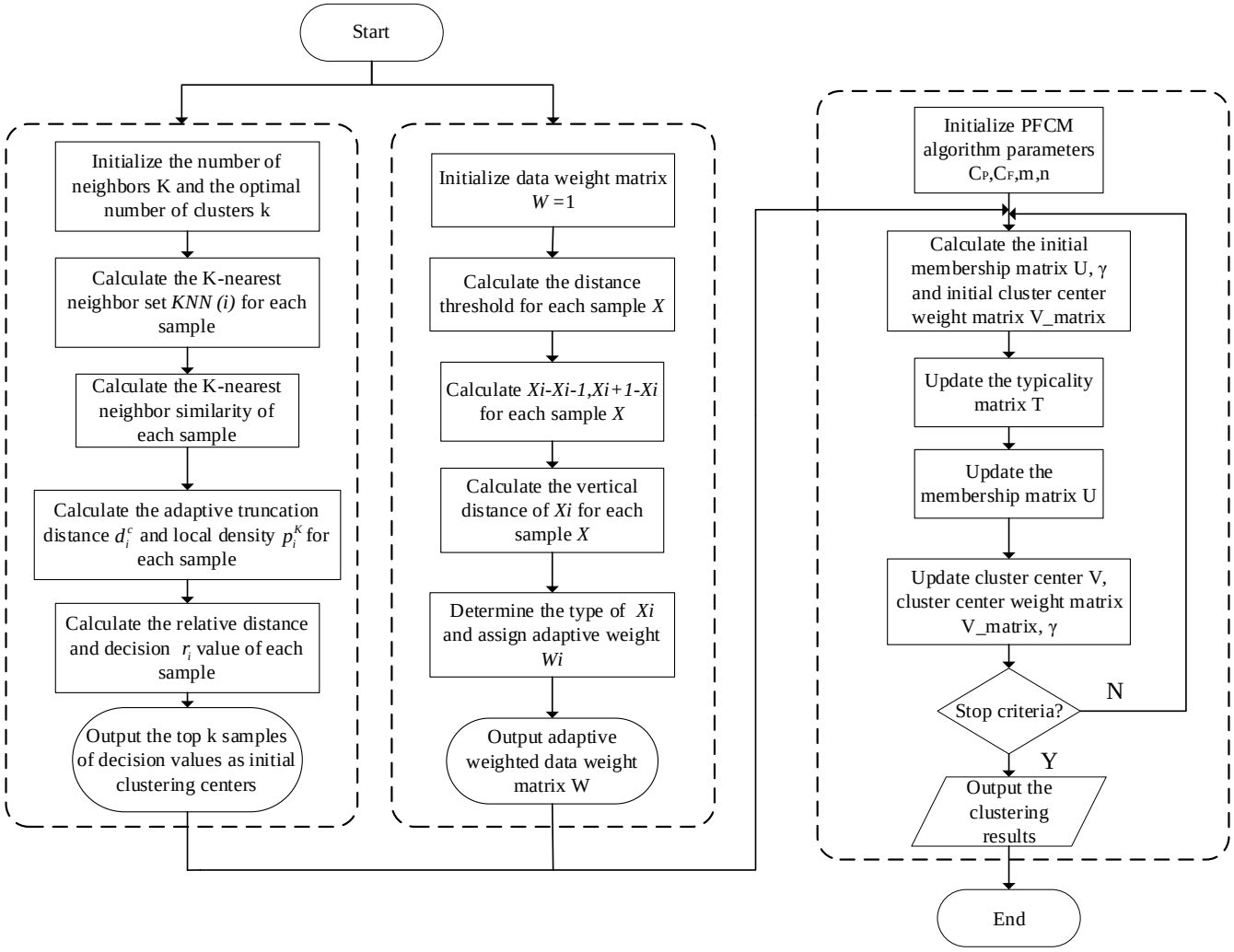

Fig. 1. The flowchart of the proposed algorithm

IV. Experiment

A. Datasets

All the experimental data in this paper come from real load profile data collected from January 1, 2023, to December 31, 2023, in a region of southwest China. These data were gathered by smart meters at 15-minute intervals, resulting in 96 data points per day from 00:00 to 23:45. For cluster analysis, two datasets are used: one containing data from 1,000 users and another from 462 users, with the typical load profile of user calculated as the annual average of their data.

B. Experimental Parameter Setting

The parameter settings for the proposed algorithm are presented in Table II. The dataset contains load curves from various industries and users, which may have significant differences in amplitude. Since the goal of clustering is to classify load curves based on their morphological characteristics, it is necessary to normalize each load curve by its maximum daily electricity consumption. Without normalization, clustering would primarily reflect differences in load amounts, while the trend characteristics would remain similar. Min-max normalization, a widely used data scaling technique, is applied. The formula for this normalization is provided in equation (36). After normalization, each load curve is scaled to a consistent range.

$$x_{ij} = \frac{x_{ij} - \min(x_i)}{\max(x_i) - \max(x_i)} \tag{36}$$

TABLE II
Algorithm Parameter Setting

Parameter	Value
C_f	1
C_p	3
m	2
η	2
Max_ iterations	1000
epsonRate	0.1

To validate the effectiveness of the proposed method, a comparative analysis was conducted with several established algorithms, including K-means, FCM, K-means++, DP-FCM, Interpretable Feature Extraction K-means (IFE-K-means), Online FCM (OFCM) [23], Single-Pass FCM (SPFCM) [24], and Membership Scaling FCM (MSFCM) [25]. These algorithms were chosen to represent both classical and novel algorithms for clustering. For all algorithms, the fuzzy index m was set to 2, and the number of clusters ranged from 3 to 8. The OFCM and SPFCM require the dataset to be divided into s subsets, and S was set to 10 in the comparison experiments. Since all algorithms, except DP-FCM, involve random initialization of the initial matrices, each algorithm was run 100 times for each number of clusters, and the average value of each internal evaluation index was calculated.

C. Dataset 1

C1. Evaluation of Clustering Algorithms

Fig. 2 illustrates the graphs of the evaluation indices with the numbers of clusters for each algorithm. In these graphs, smaller DBI and SSE values indicate better clustering performance, while larger CH and SC values indicate better clustering performance. The red line represents the algorithm proposed in this paper. As seen in Fig. 2, the DBI index of the proposed algorithm outperforms other algorithms, except when the number of clusters is 4. The SSE index surpasses other clustering algorithms when the number of clusters is 3, 5, and 8, with minimal differences from the optimal value.

Table III shows average values for different clustering numbers across various algorithms. As seen in the table, the proposed method outperforms other methods in terms of average DBI, SSE, and CH evaluation indices. Specifically, the DBI index is reduced by an average of 13.56%, the SSE index by 1.95%, and the CH index by 8.64% compared to other methods. The SC indices are slightly lower than those of K-means++ and MSFCM.

To summarize, the clustering performance of the proposed method surpasses that of other methods across several evaluation indices.

Fig. 3 represents the internal evaluation indices of different similarity measures under different numbers of clusters, and the red line represents the method proposed in this paper and compared with several other similarity measures, including Euclidean distance (Eu), Pearson distance (Pearson), weighted Euclidean distance (W-Eu), Euclidean-Pearson distance (Eu-Pearson), and weighted Pearson distance (W-Pearson). From Fig. 3, it can be seen that the method proposed in this paper has significantly lower DBI and SSE indices compared with several other methods. In terms of the SC index, the method proposed in this paper is comparable to the Eu-pfcm method, with significant advantages at k = 5, 7, and 8. In addition to that, it can be found that the clustered internal evaluation indexes of Eu, Pearson, and Eu-Pearson after weighting are better than those before unweighting. Overall, the proposed method performs significantly better than the other methods. In order to quantify the effectiveness of the proposed methods, Table IV shows the mean values of evaluation indicators for different numbers of clusters, and the improvement of the proposed methods compared to the other methods was assessed. Specifically, the DBI index decreased by an average of 25.15%, the SSE index decreased by an average of 2.64%, the CH index increased by 6.62%, and the SC index improved by an average of 16.36%, compared to other similarity measures.

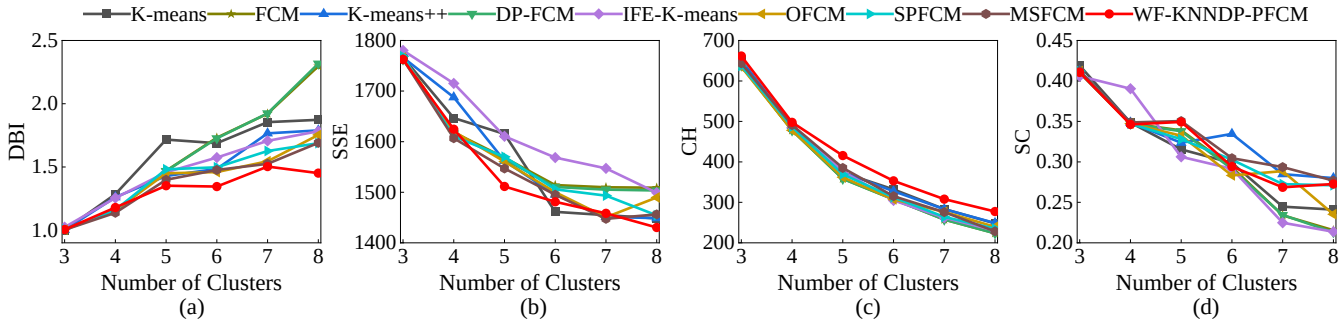

Fig. 2. Evaluation indices for each algorithm with different numbers of clusters.

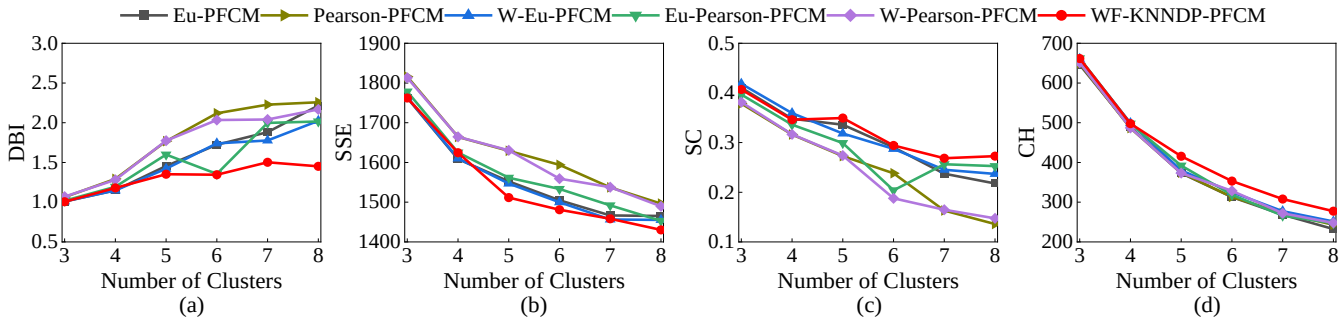

Fig. 3. Evaluation indices for different numbers of clusters with different similarity measures.

TABLE III
AVERAGE EVALUATION INDICES OF DIFFERENT CLUSTERING ALGORITHMS

Index	K-means	FCM	K-means++	DP-FCM	IFE-K-means	OFCM	SPFCM	MSFCM	WF-KNNDP-PFCM
DBI	1.569433	1.597152	1.455271	1.599523	1.466470	1.394680	1.410310	1.371692	1.305932
SSE	1567.118	1580.945	1568.646	1577.151	1620.154	1564.531	1567.058	1552.597	1544.746
CH	395.3075	377.1797	395.6184	377.1094	383.3275	383.1265	384.0417	389.8856	418.8841
SC	0.311233	0.306586	0.330617	0.305659	0.305420	0.316487	0.322261	0.331034	0.323604

TABLE IV
AVERAGE VALUE OF CLUSTERING EVALUATION INDICES FOR DIFFERENT SIMILARITY MEASURES

Index	Eu-pfcm	Pearson-pfcm	W-Eu-pfcm	Eu-Pearson-pfcm	W-Pearson-pfcm	WF-KNNDP-PFCM
DBI	1.605698	1.788037	1.520125	1.530302	1.727831	1.305927
SSE	1560.109	1622.866	1555.492	1573.901	1615.758	1544.746
CH	386.8884	389.7609	397.2471	397.3116	393.3434	418.8841
SC	0.306686	0.250787	0.310956	0.290833	0.245314	0.323604

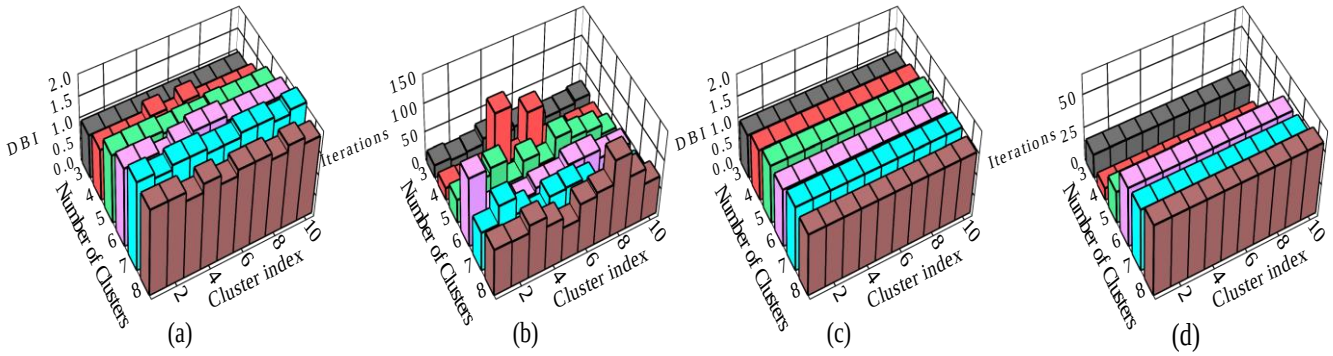

Fig. 4. Bar charts of DBI index and iteration times for different initialization centers.

C2. Analysis of Algorithm Stability

The proposed method is used to determine the initial clustering centers through the use of KNNDP, which are then compared with those generated through random initialization. The algorithm is executed ten times, and the DBI evaluation indices and the number of iterations for each run are recorded.

As shown in Table V, the DBI index of the KNNDP-initialized clustering centers decreased to varying degrees compared to the randomly initialized clustering centers. The DBI index decreased by an average of 15.05%, and the number of iterations decreased by an average of 38.03%.

Fig. 4 presents the histogram for each iteration, with Figs. 4(a) and (c) illustrate the variation in the DBI index and Figs. 4(b) and (d) show the changes in the number of iterations. As observed in the figure, the clustering results with randomly initialized centers display significant fluctuations in both the DBI index and the number of iterations, reflecting poor algorithm stability. In contrast, after using the KNNDP algorithm for initialization across 10 iterations, the clustering results remain consistent, demonstrating a notable improvement in stability compared to random initialization. This suggests that using KNNDP for initial cluster center selection enhances the stability of the clustering process, accelerates the convergence of iterations, and improves overall clustering effectiveness.

TABLE V
DBI EVALUATION INDEX AND ITERATION TIMES FOR DIFFERENT INITIALIZATION CENTERS

k	KNNDP		Random initialization	
	DBI	Iterations	DBI	Iterations
3	1.0063	18	1.0071	22.4
4	1.1794	9	1.2589	35.8
5	1.3520	23	1.4784	50.1
6	1.3451	45	1.6726	55.3
7	1.5030	49	1.7711	63.8
8	1.4508	54	1.9144	86.8

C3. Analysis of Algorithm Convergence

In cluster analysis, the optimal number of clusters is typically determined using the elbow method. The elbow method works as follows: as the number of clusters increases, the SSE index gradually decreases. Initially, the SSE index decreases rapidly as the number of clusters increases, but at a certain point, the rate of decline slows down, creating an inflection point. This inflection point indicates the optimal number of clusters. However, in some cases, the inflection point may not be very clear. Therefore, the DBI index is also used to assist in determining the optimal number of clusters. Fig. 5 illustrates the trends of both SSE and DBI as the number of clusters varies from 3 to 8. From the figure, it can be observed that the SSE decreases significantly between 3 and 5 clusters, while the decrease slows considerably when the number of clusters increases from 5 to 6. Based on the elbow method, it can be concluded that the optimal number of clusters is 5.

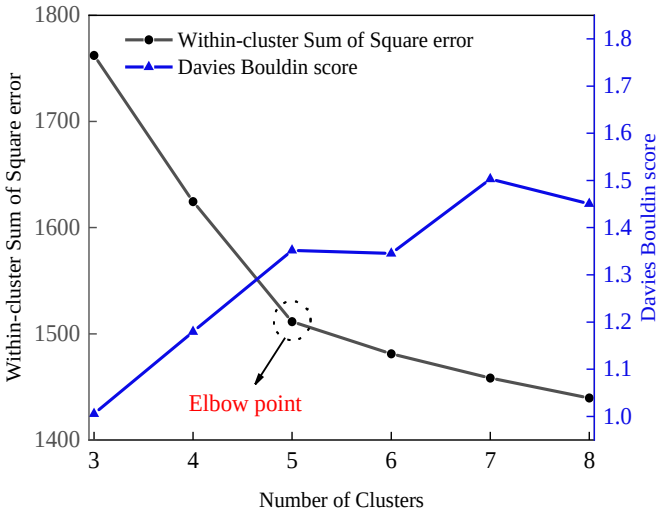

Fig. 5. Trends of average within-cluster distance and DBI index while cluster number K ranging from 3 to 8.

Fig. 6 illustrates the SSE convergence curves for various clustering algorithms with a cluster number of 5. As illustrated in Fig. 6, there are substantial differences between the algorithms with regard to their optimization capabilities and the convergence speed of SSE. The SSE values of all algorithms steadily decrease and eventually converge as the number of iterations increases, suggesting that they all operate efficiently. The proposed algorithm demonstrates superior performance, characterized by a low initial SSE value, a rapid descent rate, convergence of the SSE value after approximately 10 iterations, and a low final SSE value. These results suggest that the initial clustering center selected by the KNNDP algorithm is close to the final clustering center, thereby enhancing both the convergence rate and the clustering effectiveness. The K-means and K-means++ algorithms also exhibit a rapid convergence speed, with K-means++ further enhancing the convergence speed through improved initialization of the cluster centers.

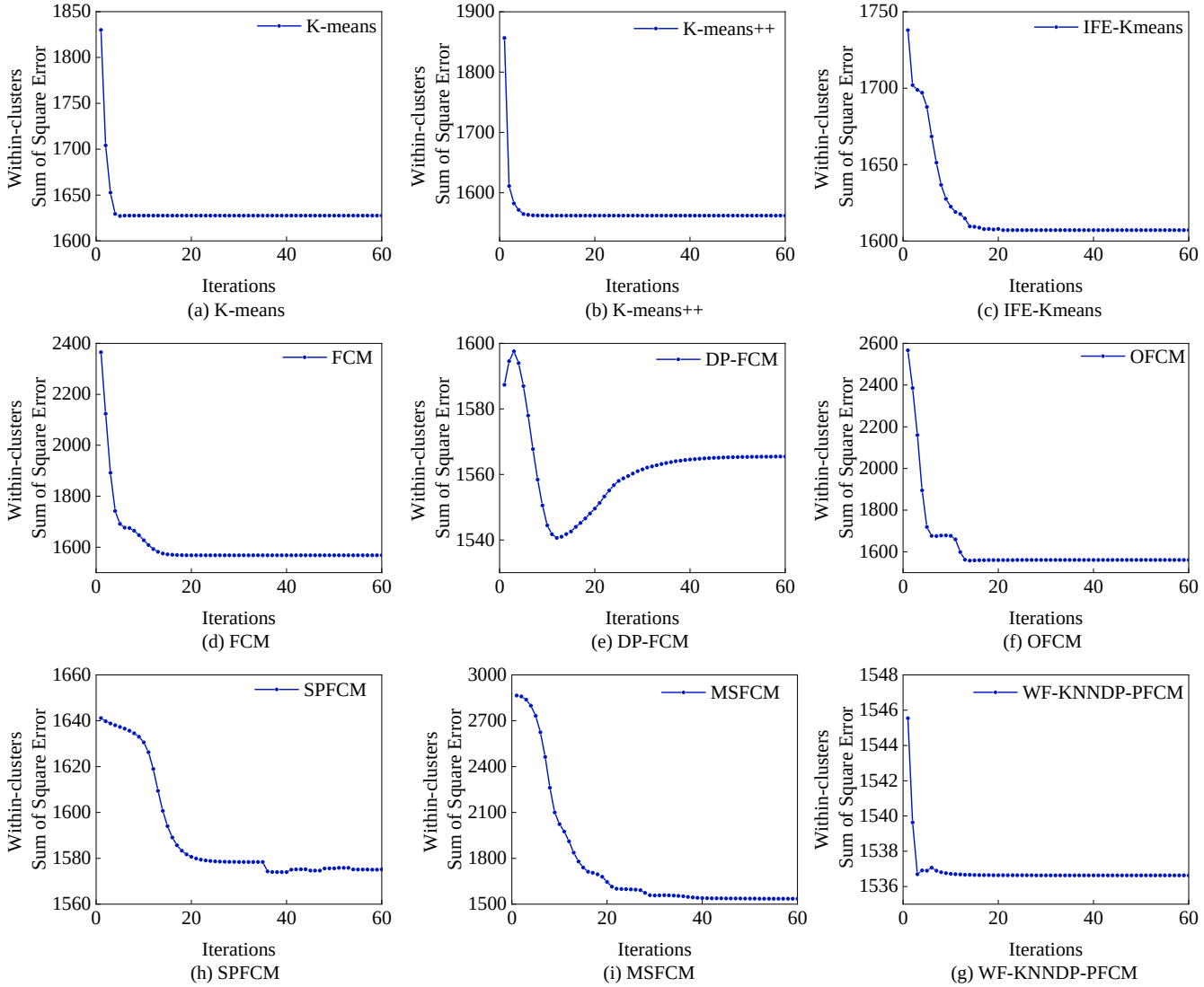

Fig. 6. SSE convergence curves for different clustering algorithms (k = 5).

Although both algorithms converge quickly, their final SSE values are slightly higher than those of the proposed algorithm. Fuzzy clustering algorithms generally have higher initial SSE values and slower convergence speeds, such as FCM, MSFCM, SPFCM, and OFCM. Among these, MSFCM and SPFCM converge more slowly compared to FCM, but they achieve lower final SSE values. This improvement in clustering results comes at the expense of some convergence speed. Overall, each of the other algorithms demonstrates certain advantages, either in terms of convergence speed or final clustering performance. However, the method proposed in this paper outperforms the others by achieving the best balance between iteration rate and final clustering effect, effectively optimizing both convergence speed and clustering performance.

C4. Analysis of Algorithm Robustness

To assess the robustness of the proposed algorithm, random perturbations of size r (where r=5%, 10%, 15%) are introduced to the load curves in the dataset. These perturbations simulate load fluctuations caused by random factors in the real-world power load collection process. Various clustering algorithms are then applied to the perturbed dataset for comparative analysis, using clustering evaluation indices such as SSE and DBI to assess robustness.

As shown in Table VI, the clustering quality evaluation indices of the different algorithms generally decrease as the degree of load profile perturbation increases. When the perturbation level is low (r = 5%), the proposed algorithm significantly outperforms the other algorithms, with the SSE and DBI indices being 3.2% and 11.5% lower, respectively. Additionally, the CH and SC indices are 14.09% and 7.05% higher, indicating better clustering quality.

At a medium perturbation level (r = 10%), the proposed algorithm continues to outperform others in terms of SSE, DBI, and SC indices. Specifically, the SSE and DBI indices are 3.02% and 11.6% lower, on average, compared to the other algorithms, while the CH index is 12.34% higher. However, the SC index shows a slight decrease compared to the other algorithms.

When the perturbation level is high (r = 15%), the clustering indices of all algorithms show a significant decrease, and the performance advantage of the proposed algorithm is further weakened. Specifically, the SSE and DBI indices are 2.32% and 5.21% lower than the other algorithms, respectively. The CH index remains 15.3% higher, indicating relatively better inter-cluster dispersion, while the SC index continues to be lower than the other algorithms. In summary, the proposed algorithm demonstrates strong clustering performance under low and medium perturbation levels.

TABLE VI
CLUSTERING INDICES OF DIFFERENT ALGORITHMS WITH RANDOM PERTURBATIONS

r/%	Index	K-means	K-means++	IFE-K-means	FCM	DP-FCM	OFCM	SPFCM	MSFCM	WF-KNNDP-PFCM
5	SSE	1732.561	1647.051	1724.711	1669.0188	1664.631	1655.551	1670.242	1636.591	1622.905
	DBI	1.746576	1.520175	1.610369	1.5597908	1.552567	1.521761	1.577568	1.473786	1.408361
	CH	361.2376	355.2882	333.7887	332.09964	331.0842	334.1064	339.7306	356.6236	390.8766
	SC	0.335386	0.304293	0.262101	0.3148430	0.312565	0.307383	0.304087	0.325672	0.328514
10	SSE	1946.227	1941.388	1950.184	2024.4439	2041.803	1975.691	1959.123	1948.543	1915.434
	DBI	1.922239	1.942195	2.161601	2.1292341	2.088036	1.879694	1.869035	1.882463	1.776937
	CH	0.293304	0.264696	0.215022	0.1983812	0.192286	0.245367	0.247637	0.259796	0.194459
	SC	324.4112	322.3353	296.6399	248.46709	247.3702	268.6126	269.9933	278.8566	318.1544
15	SSE	2180.559	2236.106	2263.408	2391.3244	2286.744	2356.079	2338.600	2375.421	2251.226
	DBI	2.031703	2.169541	2.555782	2.3261538	2.443628	2.490306	2.255088	2.143516	2.187836
	CH	0.214833	0.241806	0.166803	0.1969947	0.143019	0.179477	0.193000	0.195389	0.156224
	SC	269.9431	281.3606	257.0781	193.59804	198.8304	214.4714	204.4046	208.9494	258.4972

However, its robustness diminishes under high perturbations, particularly in terms of the SC index. This decline in robustness may be due to the adaptive weighting algorithm, which can excessively weight certain points under high perturbation, thus impacting the overall clustering performance. Therefore, while the proposed algorithm exhibits excellent robustness for low and medium perturbations, its performance significantly degrades when faced with large perturbations.

C5. Analysis of Different Paraments

The number of K-nearest neighbors is a crucial parameter in the proposed algorithm. It dictates the number of neighbors each data point relies on when selecting the initial clustering centers, which in turn directly influences both the accuracy of the initial center selection and the subsequent convergence of the clustering process. Thus, analyzing the impact of different K values on clustering results, particularly through indices like SSE and DBI, can help reveal their influence on clustering performance, including quality, convergence speed, and stability.

By comparing clustering performance indices across different K values, the optimal K value can be identified to achieve better clustering performance and increased stability. Therefore, various K values are tested within the optimal cluster range to comprehensively assess their impact on clustering results.

TABLE VII
EVALUATION INDICES FOR DIFFERENT K VALUES

K	DBI	SSE	CH	SC	Iterations
10	1.547116	1561.600	402.2576	0.261281	35
20	1.547116	1561.600	402.2584	0.261281	40
30	1.547116	1561.600	402.2589	0.261281	41
40	1.547116	1561.600	402.2574	0.261281	41
50	1.353373	1536.638	415.5580	0.349371	23
60	1.353373	1536.638	415.5581	0.349371	23

As shown in Table VII, the experimental results for clustering performance indices and the number of iterations indicate that both the clustering performance and the number of iterations vary significantly as the value of K increases. Specifically, the values of DBI and SSE decreased significantly, indicating an improvement in clustering performance and more reasonable clustering results. Meanwhile, the CH and SC values increased, suggesting enhanced compactness and separation of the clusters. Although the performance metrics are similar for certain values of K (e.g., K = 10, 20, 30), the number of iterations varies, indicating that the number of K-nearest neighbors significantly affects the convergence speed. In summary, selecting the appropriate number of K-nearest neighbors not only results in more reasonable, compact, and well-separated clustering results but also significantly reduces the number of iterations, thereby increasing the convergence efficiency of the proposed algorithm.

Algorithm parameters have a crucial influence on the final clustering results. The PFCM algorithm, in particular, involves several parameters, such as C_f, C_p, m, and η. Variations in these parameters can significantly influence evaluation indices like SSE, DBI, and SC. By analyzing the clustering results under different parameter configurations, it becomes clear how parameter selection affects clustering quality and structure. To explore the impact of different parameter configurations on clustering performance, six different combinations of parameters are evaluated in this paper, and the specific parameters for each combination are shown in Table VIII.

TABLE VIII
DIFFERENT COMBINATIONS OF ALGORITHMIC PARAMETERS

Combination	C_f	C_p	m	η
1	1	3	2	2
2	1.5	2	2	2.5
3	2	5	3	3
4	1	2	1.5	2
5	2	3	2	2.5
6	1.5	5	3	3

TABLE IX
EVALUATION INDICES FOR DIFFERENT COMBINATIONS

Combination	DBI	SSE	CH	SC
1	1.352012	1531.526	415.5365	0.349402
2	1.348600	1480.623	349.9034	0.295070
3	1.369598	1495.677	330.6745	0.294500
4	1.296669	1601.362	329.8569	0.335391
5	1.352449	1480.452	351.3088	0.295203
6	1.353133	1495.605	329.1885	0.293847

Table IX presents significant variations in evaluation indices across different parameter combinations. For instance, combination 1 exhibits strong performance across most indices, particularly in the CH and SC indices, indicating superior clustering performance. Other combinations exhibit distinct strengths and weaknesses. For instance, Combination 2, although performing better in the DBI and SSE indices, shows slightly weaker results in the CH and SC indices. Similarly, Combination 4 has the lowest DBI value among all combinations and demonstrates superior performance in the SC index, but its SSE value is the highest.

TABLE X
NUMBER OF SAMPLES IN EACH CLUSTER FOR DIFFERENT METHODS

Label	KNNDP_center	Random_center1	Random_center2	Eu_PFCM
1	217	197	215	200
2	60	134	91	99
3	224	234	236	219
4	243	289	205	251
5	255	145	253	230

TABLE XI
SILHOUETTE COEFFICIENTS OF EACH CLUSTER FOR DIFFERENT METHODS

Label	KNNDP_center	Random_center1	Random_center2	Eu_PFCM
1	0.3359	0.3341	0.3314	0.3624
2	0.0197	0.0229	0.2870	-0.0585
3	0.3827	0.3328	0.3394	0.3812
4	0.5475	0.4680	0.1900	0.5027
5	0.2159	0.2708	0.2196	0.2638

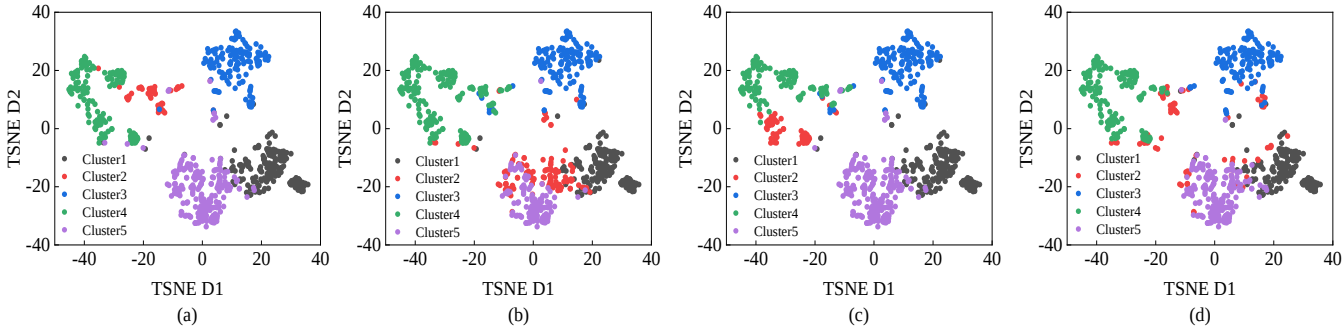

Fig. 7. Visualization of t-SNE dimensionality reduction using different clustering methods when k =5.

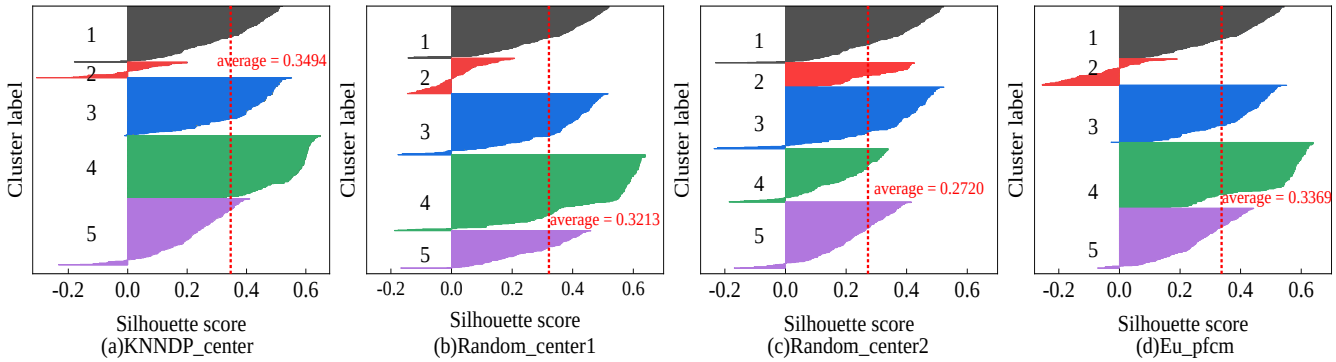

Fig. 8. Silhouette coefficient graph of different clustering methods when k =5.

TABLE XII
EVALUATION INDICES FOR DIFFERENT METHODS

	KNNDP_center	Random_center1	Random_center2	Eu_PFCM
DBI	1.352012	1.572361	1.545324	1.655531
SSE	1531.526	1590.047	1561.536	1552.448
CH	415.5365	378.1162	400.7972	373.0799
SC	0.349402	0.321308	0.271962	0.336908

These differences indicate that the parameter configuration has a direct impact on the clustering results, and different parameter combinations significantly change the accuracy, separateness, and compactness of the clusters. Therefore, optimizing the parameter configuration is the key to improving the clustering performance.

C6. Analysis of Different Clustering Results

Figs. 7 and . 8 show the t-SNE visualizations of clustering results with different initial centers and similarity measures for k = 5, along with the silhouette coefficients for each cluster. Fig. 7(a) presents the clustering results using initial centers selected by the KNNDP algorithm. Fig. 7(b) and Fig. 7(c) show results with randomized initial centers (Random_center1 and Random_center2), both using the proposed similarity measure. Fig. 7(d) depicts clustering results based on Euclidean distance with KNNDP centers. Distinct colors represent different clusters, while Tables X and XI provide sample sizes and average silhouette coefficients for each method.

From Fig. 7, Clusters 1, 2, and 3 show similar distributions across different methods, while significant differences are observed in Clusters 4 and 5. Table X reveals variations in sample sizes for Clusters 4 and 5. Random_center1 merges Clusters 2 and 4 from KNNDP into one cluster and splits Cluster 5, while Random_center2 maintains similar clustering for Clusters 1 and 3 but distributes Clusters 2 and 4 differently. KNNDP-based clustering consistently shows more stable and clear results. In Fig. 8, silhouette coefficients for KNNDP_center are higher than those for

Random_center1 and Random_center2, indicating better clustering quality.Random_center methods show many negative silhouette coefficients, reflecting poor clustering performance, whereas KNNDP results in higher average silhouette coefficients for clusters 1, 3, and 4.

Figs. 9 to 12 display the normalized actual load curves and centroids for each cluster. The centroid curves of Clusters 1, 2, 3, and 5 are similar, with minor differences in amplitude and timing. However, Clusters 3 and 4, using KNNDP, are more compact with less noise compared to the random methods. Random_center1 and Random_center2 show similar centroids, indicating poor cluster separation. In contrast, KNNDP's clustering results show greater differentiation between clusters and higher compactness within each cluster. Overall, KNNDP-based initialization improves clustering stability by producing clearer, more distinct clusters with lower intra-cluster noise. This results in better separation, as reflected by lower DBI and SSE values and higher CH and SC values.

A comparison of Fig. 7(a) and Fig. 7(d) reveals the impact of different similarity measures. Cluster 2, using Euclidean distance, groups edge points from other clusters, resulting in poor separation and a negative silhouette coefficient. The centroid of Cluster 2 also fails to reflect the actual load trends. Table XI and Fig. 11(b) show that the proposed similarity measure outperforms Euclidean distance in terms of clustering quality, as reflected in higher silhouette coefficients and better separation for Clusters 3 and 4.

In conclusion, the proposed method, incorporating KNNDP-based initialization and a weighted fusion similarity measure, offers improved clustering performance by enhancing intra-cluster compactness, reducing noise, and achieving better cluster separation.

C7. Analysis of Power Load Characteristics

Based on the above analysis, it is evident that the clustering results obtained using the proposed algorithm outperform the other algorithms. Therefore, a detailed analysis of the actual power consumption curves from the clustering results is conducted. As shown in Fig. 9, the following power usage patterns can be identified:

- Typical double-peak power consumption patterns (Cluster 5)
- Typical single-peak power consumption patterns (Cluster 1)
- Multi-peak power consumption patterns (Cluster 3, Cluster 4)
- Single-peak power consumption pattern (Cluster 2)

Of the various power consumption patterns, Cluster 2 and Cluster 5 are the more common power consumption patterns that match the work patterns of most commercial and industrial businesses. Cluster 1 peaks at 9:00 a.m. and continues until 8:00 p.m. Cluster 5 peaks at 9:00 a.m., similar to Cluster 1, but unlike Cluster 1, Cluster 5 declines for a period of time at 12:00 p.m., peaks again at 2:00 p.m., and then declines at 5:00 p.m., a typical double-peak pattern for a midday lunch break.

Cluster 2 is similar to the single peak pattern in that it peaks around 2:00 and then gradually declines until 12:00 when it slowly rises again; it is likely to be a manufacturing facility that operates at night.

Cluster 3 and Cluster 4 have more fluctuating patterns of electricity consumption. Cluster 3 is on a downward trend between 0:00 and 6:00 o'clock. It rises briefly between 06:00 and 10:00 and continues to rise between 10:00 and 16:00, reaching its peak for the day. After that, there is a constant fluctuation until 24:00.

Cluster 4 is very similar to Cluster 3 in that it rises for a period of time and then falls rapidly, but Cluster 4 fluctuates more frequently and does not have high power usage for most of the day, like Cluster 3.

These insights into the load profiles help in understanding the electricity consumption behaviors, aiding in better load management and optimization strategies for different groups.

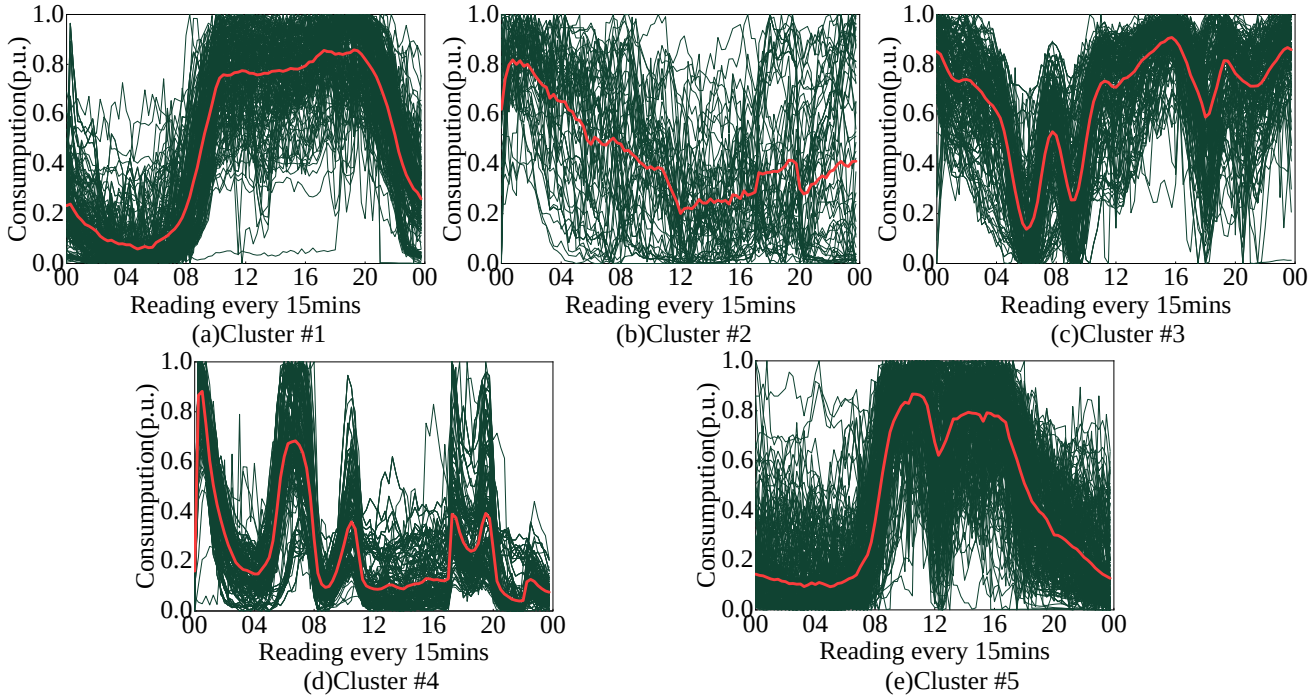

Fig. 9. Clustering results of KNNDP_center when k =5.

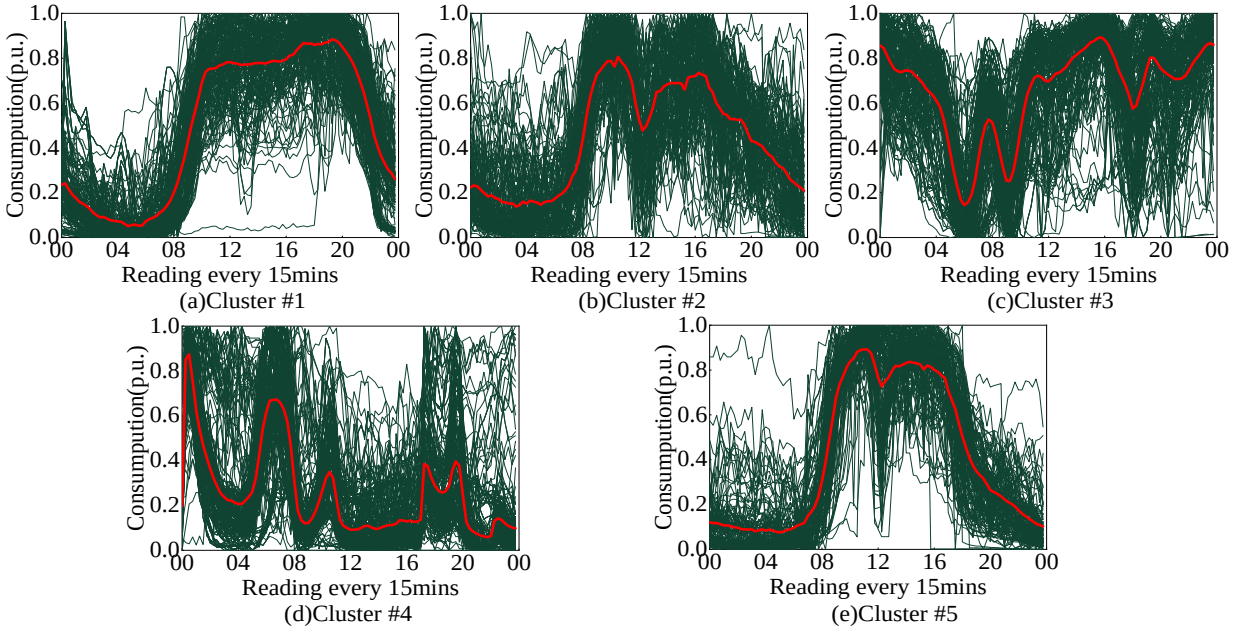

Fig. 10. Clustering results of Random_center1 when k =5.

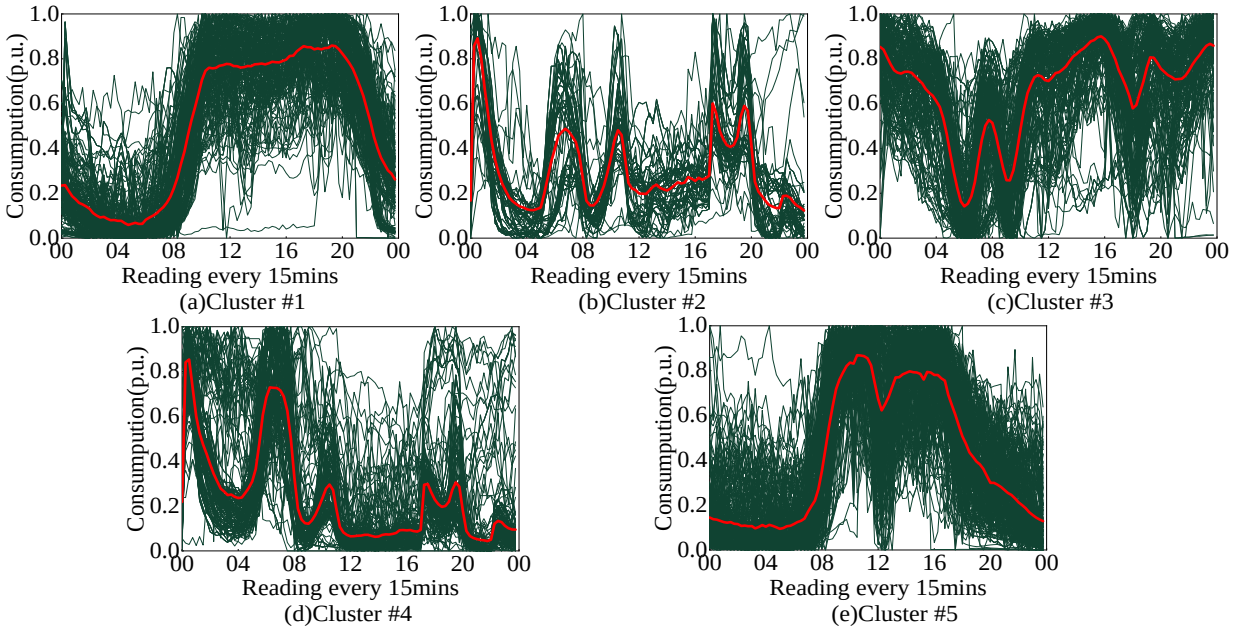

Fig. 11. Clustering results of Random_center2 when k =5.

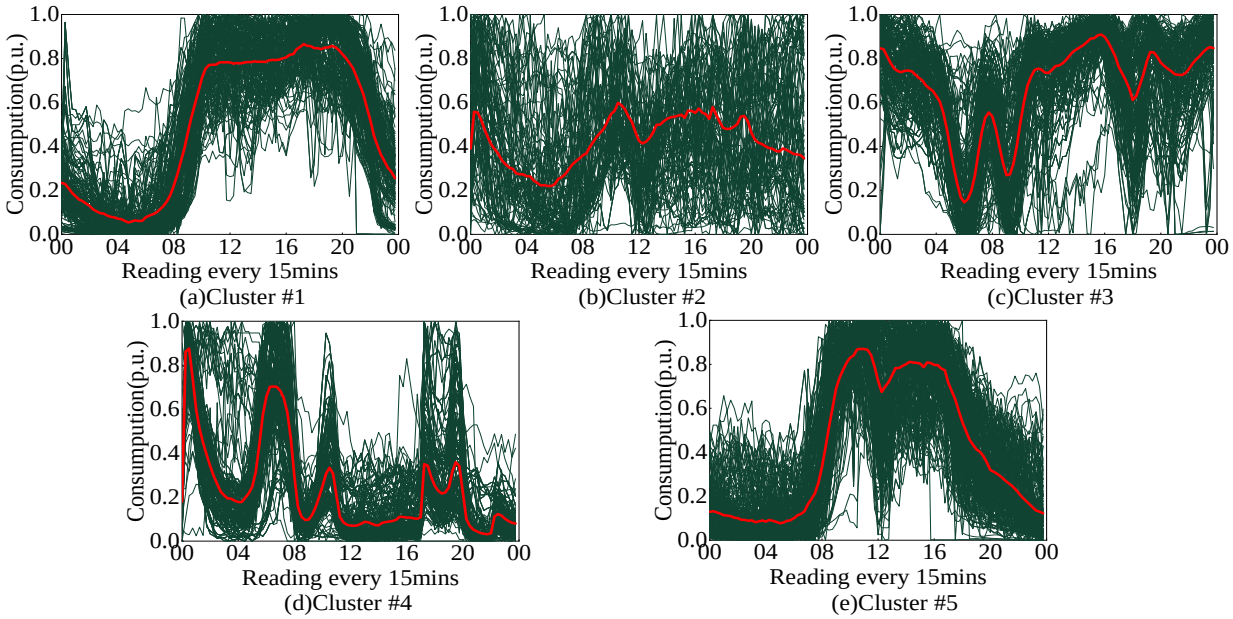

Fig. 12. Clustering results of Eu_Pfcm when k =5.

D. Dataset 2

D1. Evaluation of Clustering Algorithms

To further validate the effectiveness of the proposed algorithm, the experiment was repeated on an additional dataset. Fig. 13 illustrates the values of various evaluation indices across different numbers of clusters. As shown, the DBI and SSE values achieved by the proposed algorithm significantly outperform those of other clustering algorithms, except when the number of clusters is four. The CH index consistently outperforms all other algorithms across all numbers of clusters, and the SC metric performs best in most cluster configurations.

Table XIII presents a comprehensive summary of the average values of various evaluation indices for different clustering algorithms. From Table XIII, it is evident that the proposed algorithm surpasses other algorithms in average DBI, SSE, CH, and SC indices. The proposed algorithm reduces the DBI index by 15.63% and the SSE index by 3.5% compared to alternative algorithms. Additionally, the CH and SC indices improve by 18.29% and 16.57%, respectively, highlighting the superior performance of the algorithm.

In conclusion, these results confirm that the proposed clustering algorithm outperforms other methods, offering better clustering quality across multiple evaluation indices.

Fig. 14 demonstrates that the proposed similarity measure outperforms other methods in terms of DBI and SSE, with the exception of when the number of clusters is 4. Similarly, it achieves significantly better CH and SC values, except when the number of clusters is 4 or 7.

Table XIV further confirms the effectiveness of the proposed method, showing superior average values for DBI, SSE, CH, and SC across all numbers of clusters. Specifically, the proposed method reduces the DBI index by 11.38% and the SSE index by 2.4%, while increasing the CH and SC indices by an average of 11.42% and 13.49%, respectively, compared to other methods. Additionally, Table XIV demonstrates that the evaluation metrics using weighted similarity surpass those without weighting, confirming the effectiveness of the proposed method in this study.

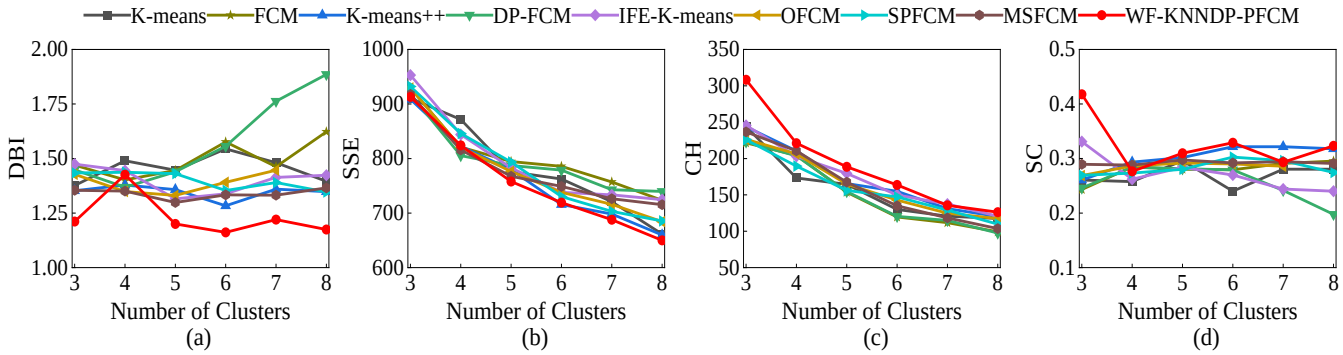

Fig. 13. Evaluation indices for each algorithm with different numbers of clusters.

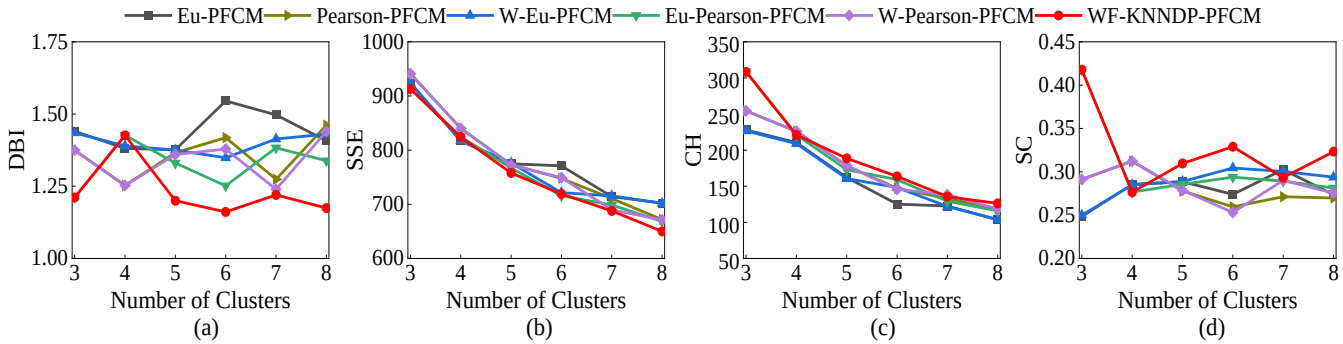

Fig. 14. Evaluation indices for different numbers of clusters with different similarity measures.

TABLE XIII
EVALUATION INDICES FOR DIFFERENT CLUSTERING ALGORITHMS OF DATASTET2

Index	K-means	FCM	K-means++	DP-FCM	IFE-K-means	OFCM	SPFCM	MSFCM	WF-KNNDP-PFCM
DBI	1.455523	1.496423	1.346441	1.576632	1.401574	1.384424	1.398475	1.339121	1.232193
SSE	783.8324	801.8062	764.0312	795.3383	796.0156	777.3572	781.7441	781.9386	758.6911
CH	158.3195	151.7235	170.9976	152.2864	172.7058	163.7924	159.5983	161.9372	190.5694
SC	0.269203	0.279022	0.302821	0.254932	0.271811	0.282777	0.282605	0.291986	0.324906

TABLE XIV
AVERAGE VALUE OF CLUSTERING EVALUATION INDICES FOR DIFFERENT SIMILARITY MEASURES

Index	Eu_pfcm	Pearson_pfcm	W_eu_pfcm	Eu_pearson_pfcm	W_pearson_pfcm	WF-KNNDP-PFCM
DBI	1.441234	1.357762	1.398747	1.323479	1.340923	1.232191
SSE	783.7333	780.8563	776.2546	764.5662	777.4135	758.6925
CH	158.6377	176.2249	161.6961	184.5369	176.9357	190.5694
SC	0.280512	0.280313	0.287031	0.307421	0.283453	0.324906

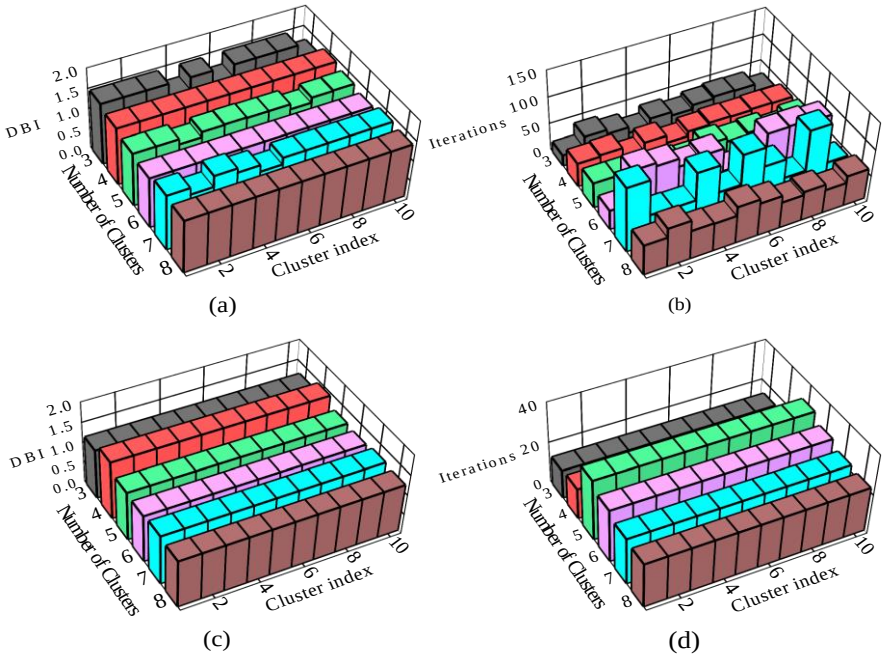

Fig. 15. Bar charts of DBI index and iteration times for different initialization centers.

D2. Analysis of Algorithm Stability

Table XV presents the DBI values and number of iterations for Dataset 2, comparing the performance of using the KNNDP algorithm to select initial cluster centers with that of random initialization. As shown in the table, initializing with KNNDP results in better DBI indices and fewer iterations across different numbers of clusters compared to random initialization. Specifically, the average DBI index is reduced by 10.20%, while the average number of iterations decreases by 59.14%, demonstrating the effectiveness of KNNDP in improving clustering performance.

TABLE XV
DBI EVALUATION INDEX AND ITERATION TIMES FOR DIFFERENT INITIALIZATION CENTERS

k	KNNDP		Random initialization	
	DBI	Iterations	DBI	Iterations
3	1.2113	14	1.4307	34.9
4	1.4259	15	1.4259	40.4
5	1.2000	30	1.3062	44.6
6	1.1613	26	1.2584	73.3
7	1.2200	23	1.3639	85.5

Fig. 15 presents the variation in DBI across iterations and illustrates the changes in the iteration count. As observed, the DBI values and iteration counts fluctuate considerably for the 10 clustering centers initialized randomly, in contrast to the consistent outcomes achieved using the KNNDP algorithm. This consistency indicates that the KNNDP algorithm yields a higher level of stability compared to random initialization. Thus, initializing clustering centers with the KNNDP algorithm not only enhances the stability of the clustering process but also reduces the required iteration count, contributing to improved clustering efficiency and performance.

D3. Analysis of Algorithm Convergence

Fig. 16 illustrates the trends of the DBI and SSE indices for Dataset 2 as the number of clusters increases. According to the elbow method and the observed trends, the inflection point occurs at 5 or 6 clusters. However, since the DBI value is lower at 6, this is considered the optimal number of clusters.

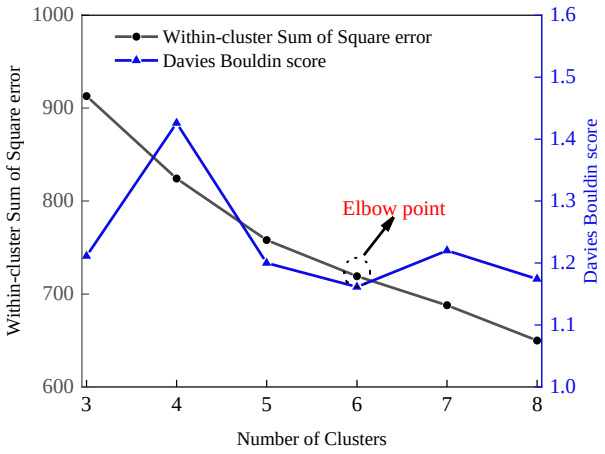

Fig. 16. Trends of average within-cluster distance and DBI index while cluster number k ranging from 3 to 8.

Fig. 17 illustrates the SSE convergence curve for the second dataset. The results show that the proposed algorithm performs well, with both its initial and final SSE values being lower than those of the other algorithms. The K-means algorithm converges the fastest, reaching convergence in fewer than ten iterations. However, its final SSE value is higher, suggesting that it may converge to a local optimum. The K-means++ algorithm, while requiring slightly more iterations, achieves a lower final SSE value than K-means, indicating an improvement in avoiding local optima. Fuzzy clustering algorithms, such as FCM, OFCM, and MSFCM, exhibit higher initial SSE values and slower convergence speeds. The DP-FCM algorithm reduces the initial SSE value by improving the initial clustering centers, though its iteration speed remains slower. In summary, the algorithm proposed in this paper outperforms all other algorithms in terms of convergence effectiveness and demonstrates better clustering accuracy and convergence speed.

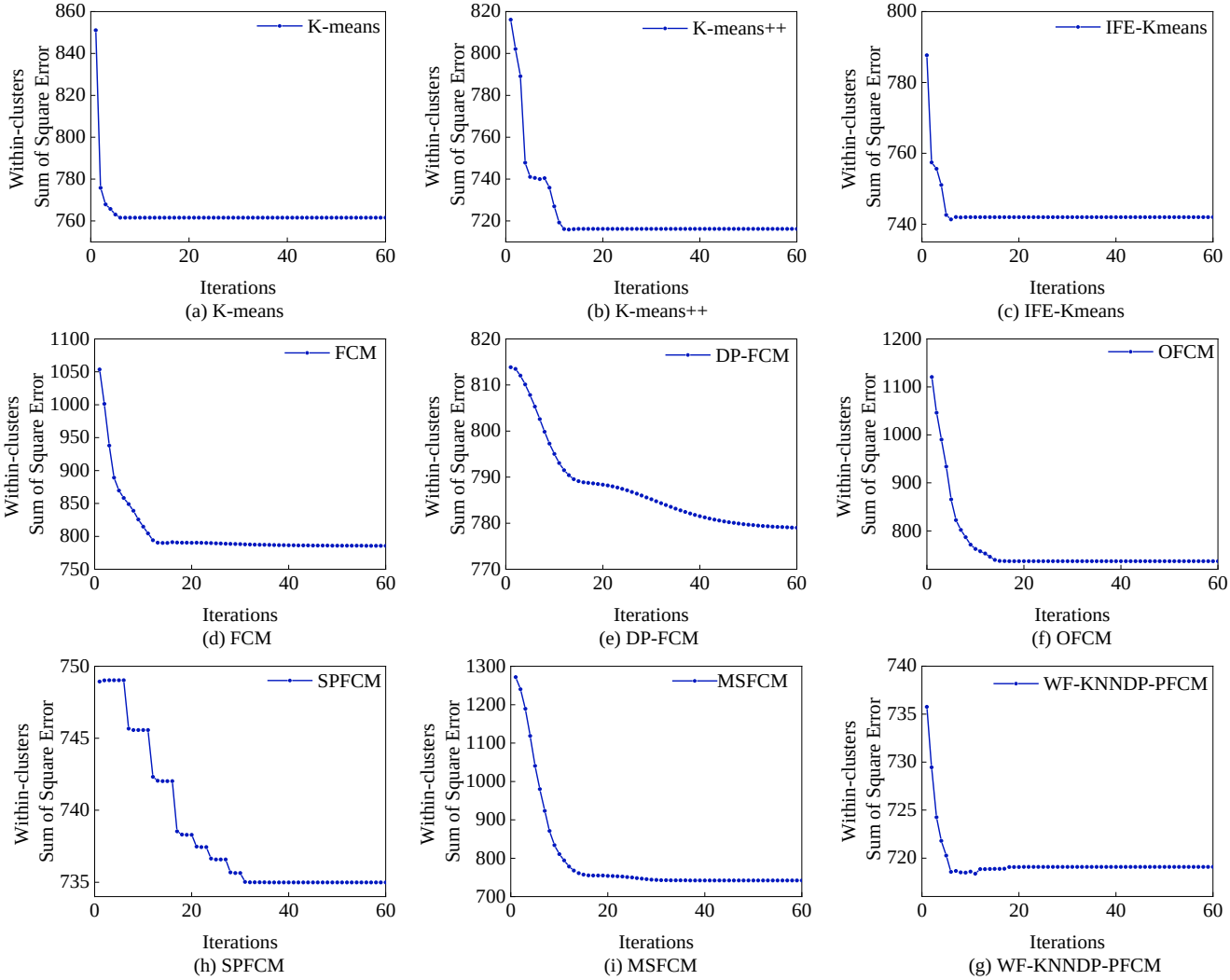

Fig. 17. SSE convergence curves for different clustering algorithms (k = 6).

TABLE XVI
CLUSTERING INDICES OF DIFFERENT ALGORITHMS WITH RANDOM PERTURBATIONS

r/%	Index	K-means	K-means++	IFE-Kmeans	FCM	DP-FCM	OFCM	SPFCM	MSFCM	WF-KNNDP-PFCM
5	SSE	802.2382	781.5552	822.1179	861.6233	827.4254	790.6215	811.2552	796.1234	761.6619
	DBI	1.647676	1.413709	1.766698	1.611991	1.564964	1.467967	1.605911	1.478096	1.220757
	CH	124.1108	139.8232	129.1899	107.7807	111.8542	133.8724	122.0148	120.8016	153.2932
	SC	0.229449	0.271341	0.201596	0.251178	0.251120	0.260964	0.250458	0.268631	0.306136
10	SSE	948.3732	882.9309	927.7822	976.2041	858.4407	928.8511	938.4304	957.6302	886.4289
	DBI	1.842816	1.667926	2.050694	2.062706	1.756672	1.834252	1.917575	1.706574	1.526744
	CH	103.2721	121.6084	113.6077	89.76891	102.2129	110.2845	98.86763	94.34256	121.5125
	SC	0.165465	0.216401	0.162861	0.191672	0.197795	0.202391	0.198491	0.203729	0.228258
15	SSE	1044.416	1043.175	1071.202	1182.781	1021.824	1140.933	1127.932	1135.789	1080.725
	DBI	2.142197	2.056513	2.380635	2.540991	3.033249	2.253901	2.346158	2.218686	2.065741
	CH	107.6176	98.87705	99.04407	63.50561	71.68734	84.95625	73.95903	69.52874	94.43683
	SC	0.167591	0.163659	0.133747	0.178327	0.050013	0.140668	0.143095	0.165678	0.154021

D4. Analysis of Algorithm Robustness

Table XVI presents the evaluation indices of each clustering algorithm for the second dataset at various perturbation levels. When the perturbation level is low ($r \leq 10\%$), the proposed algorithm outperforms all other algorithms across all indices. Specifically, at r = 5%, the SSE and DBI indices are 6.5% and 28.5% lower, while the CH and SC indices are 24.76% and 24.4% higher, respectively. At r = 10%, SSE and DBI indices are 4.61% and 21.49% lower, and CH and SC indices are 17.58% and 19.74% higher, respectively. At r = 15%, although SSE and DBI indices remain lower than those of the other algorithms, the SC index approaches the average. In general, the proposed algorithm performs well under small and medium perturbations, but its advantage diminishes under large perturbations, particularly for the SC index.

D5. Analysis of Different Paraments

Table XVII presents the clustering performance indices and the number of iterations for different numbers of K-nearest neighbors on dataset 2. As shown, K = 30 emerges as the optimal choice. It achieves superior results across several clustering performance metrics while maintaining a moderate number of iterations, striking a balance between clustering quality and algorithmic efficiency. While increasing K further reduces the number of iterations, it results in a noticeable degradation in clustering quality.

Therefore, K = 30 is identified as the optimal number of K-nearest neighbors, providing both high clustering quality and acceptable algorithmic efficiency.

TABLE XVII
EVALUATION INDICES FOR DIFFERENT K VALUES

K	DBI	SSE	CH	SC	Iterations
10	1.547117	1561.600	402.2577	0.261281	35
20	1.547117	1561.601	402.2585	0.261281	40
30	1.547117	1561.601	402.2589	0.261281	41
40	1.547117	1561.600	402.2576	0.261281	41
50	1.353374	1536.638	415.5581	0.349371	23
60	1.353374	1536.639	415.5582	0.349371	23

The parameter combinations used in this experiment are consistent with those in Table VIII. As shown in Table XVIII, the experimental results of different parameter combinations show that there are significant differences in the DBI, SSE, CH, and SC indices for different parameter combinations. Among them, DBI and SSE of combination 4 are the lowest among all combinations, while CH and SC are the highest among all combinations, which indicates that the clustering effect of combination 4 is the best. The clustering performance of combination 1 is second only to combination 4, while the clustering effect of combination 3 is the worst among all combinations. Therefore, optimizing the parameter configuration is the key to improving the clustering performance.

TABLE XVIII
EVALUATION INDICES FOR DIFFERENT COMBINATIONS

Combination	DBI	SSE	CH	SC
1	1.352012	1531.526	415.5365	0.349402
2	1.348600	1480.623	349.9034	0.295070
3	1.369598	1495.677	330.6745	0.294500
4	1.296669	1601.362	329.8569	0.335391
5	1.352449	1480.452	351.3088	0.295203
6	1.353133	1495.605	329.1885	0.293847

D6. Analysis of Different Clustering Results

Figs. 18 and 19 provide visualizations of t-SNE dimensionality reduction for different initial cluster centers and classical Euclidean distances, focusing on the case where the number of clusters is set to 6. These figures also display the silhouette coefficients for each cluster. Fig. 18(a) shows the clustering results based on initial cluster centers selected by the KNNDP algorithm, Fig. 18(b) presents the results from randomly initialized cluster centers, and Fig. 18(c) depicts the results using Euclidean distance-based clustering.

TABLE XIX
NUMBER OF SAMPLES IN EACH CLUSTER FOR DIFFERENT METHODS

Label	KNNDP_center	Random_center	Eu_pfcm
1	140	137	118
2	71	51	94
3	41	43	41
4	35	80	50
5	163	129	95
6	26	36	78

TABLE XX
SILHOUETTE COEFFICIENTS OF EACH CLUSTER FOR DIFFERENT METHODS

Label	KNNDP_center	Random_center	Eu_pfcm
1	0.3013	0.2561	0.3579
2	0.6144	0.5108	0.4160
3	0.4384	0.4226	0.4653
4	0.2905	0.1216	-0.010
5	0.2336	0.2347	0.2758
6	0.1781	0.0413	0.1189

TABLE XXI
EVALUATION INDICES FOR DIFFERENT METHODS

Label	KNNDP_center	Random_center	Eu_pfcm
DBI	1.16132	1.53268	1.54531
SSE	719.088	744.546	771.042
CH	163.666	152.517	125.334
SC	0.32901	0.25376	0.28421

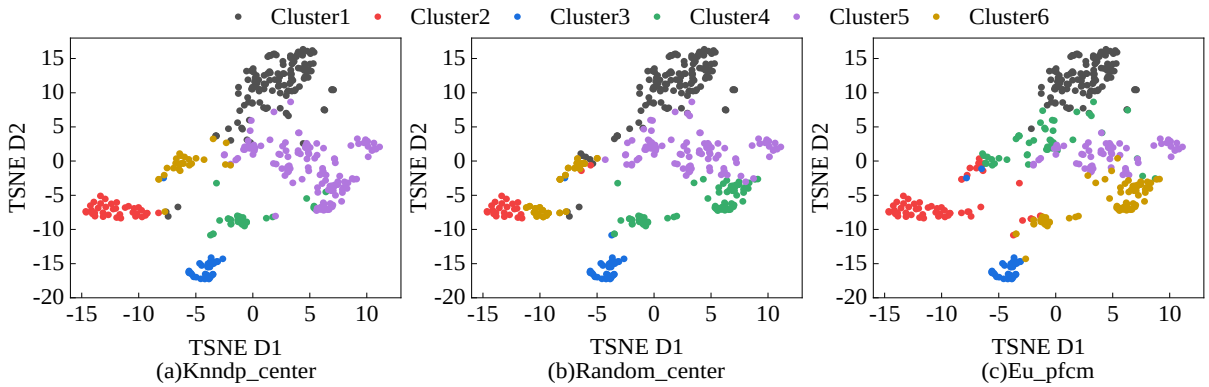

Fig. 18. Visualization of t-SNE dimensionality reduction using different clustering methods when k=6.

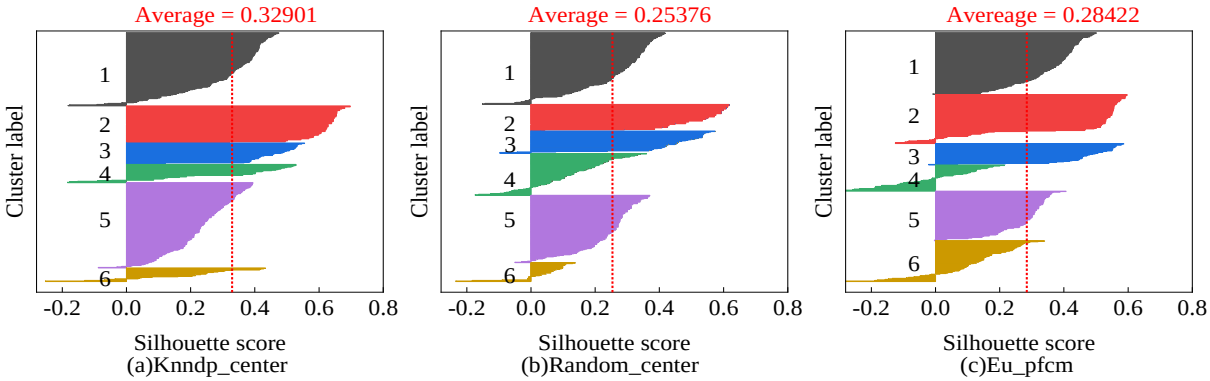

Fig. 19. Silhouette coefficient graph of different clustering methods when k=6.

Additionally, Table XIX presents the distribution of sample counts across categories for each clustering method, Table XX summarizes the average silhouette coefficient for each cluster, and Table XXI provides key clustering evaluation indices.

As illustrated in Fig. 18, the dataset is partitioned into six distinct clusters. Clusters 1, 2, and 3 exhibit similar distributions across the different clustering methods, with the primary differences observed in the allocation of Clusters 4, 5, and 6. Table XIX further highlights significant discrepancies in sample sizes for Clusters 4, 5, and 6.

As shown in Fig. 18, the Random_center method divides Cluster 5 from the KNNDP_center method into two separate clusters, merging one segment with Cluster 4 and reallocating some samples from Cluster 2 to Cluster 6. In contrast, the clustering results based on Euclidean distance show significant differences from the other two methods. Specifically, this approach creates a new cluster by reassigning samples from Clusters 1, 5, and 6 (as identified in the KNNDP_center method), while also grouping portions of Clusters 4 and 5 into a distinct cluster.

As shown in Fig. 19, the average silhouette coefficient for KNNDP_center is significantly higher than those of the other two clustering methods, indicating superior clustering performance. In contrast, the silhouette coefficient plots for both Random_center and Eu_pfcm show a substantial number of clusters with sample silhouette coefficients below zero, leading to much lower overall average silhouette coefficients compared to KNNDP_center. When combined with other clustering evaluation indices presented in Table XXI, these results suggest that initializing cluster centers using KNNDP enhances clustering effectiveness.

Figs. 20-22 present the normalized actual load curves and centroids for each cluster under different clustering methods. A comparison of the actual load curves reveals that the centroid curves of Clusters 1, 2, 3, and 5 exhibit similar trends, with only slight differences in amplitude and peak timing. However, the compactness within each cluster varies.

Notably, Cluster 2 in the Euclidean distance-based clustering exhibits considerable noise, with many load curves deviating from the centroid trend. This is reflected in its average silhouette coefficient, which, as shown in Table XX, is the lowest among all methods. Additionally, the clustering results from both Random_center and Euclidean distance show clusters with similar curve trends. For example, Clusters 2 and 6 in the Random_center method have nearly identical peak times, differing only in peak magnitudes. This suggests that samples in Cluster 6 would be more appropriately assigned to Cluster 2. Furthermore, Cluster 6 in the Random_center method contains numerous load curves that do not align with its centroid, leading to suboptimal clustering performance.

Similarly, the centroid curves of Clusters 4 and 5 in the Euclidean distance clustering results exhibit comparable trends and amplitudes, suggesting redundancy in the clustering process. In summary, both random initialization and Euclidean distance-based methods produce clusters with overlapping centroid curves and high intra-cluster noise; the KNNDP-based initialization facilitates clearer differentiation of trends among clusters and reduces intra-cluster noise, resulting in more distinct and meaningful clusters.

In conclusion, using KNNDP to initialize cluster centers and applying the proposed similarity measurement method significantly enhances algorithm stability and improves clustering accuracy. Moreover, the integration of the data-weighted fusion similarity measure further refines the clustering results by more effectively capturing the underlying patterns in the load profiles, which traditional distance-based methods often overlook. These combined improvements lead to a clearer differentiation between clusters, reduced intra-cluster noise, and more meaningful clustering outcomes. Ultimately, the proposed method not only ensures more reliable and consistent clustering results but also facilitates more accurate identification of distinct power consumption patterns.

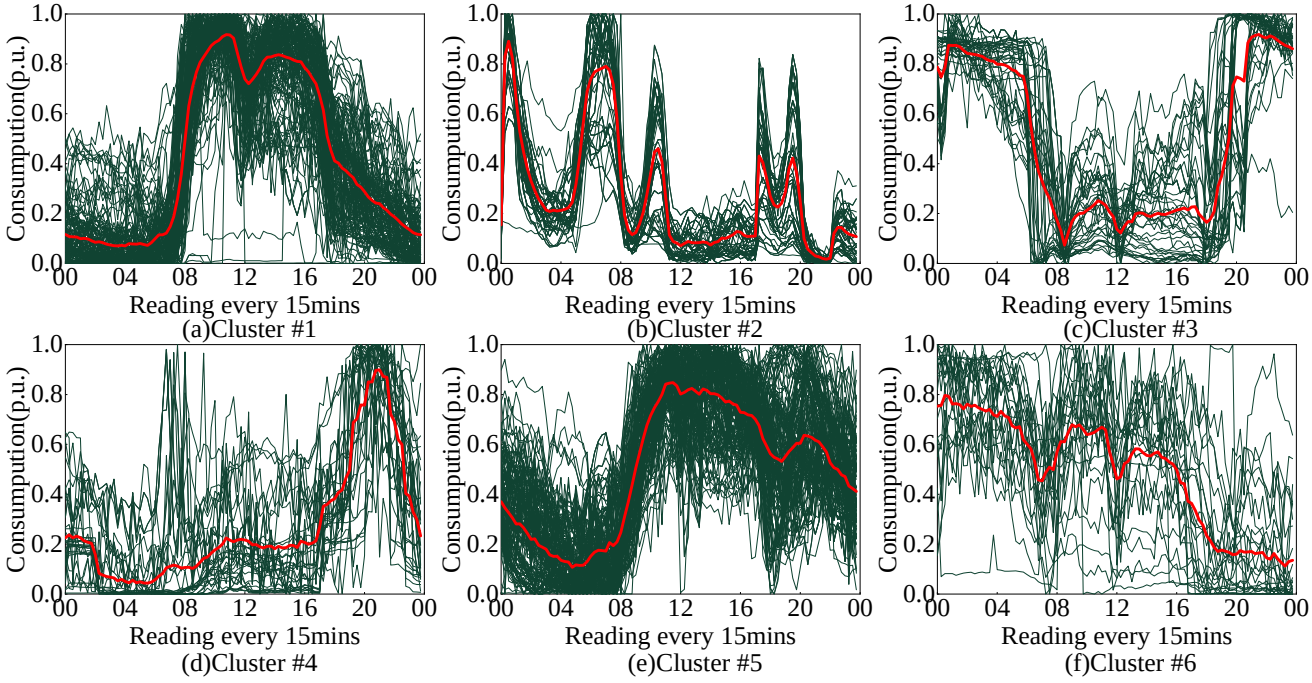

Fig. 20 . Clustering results of KNNDP_center when k=6.

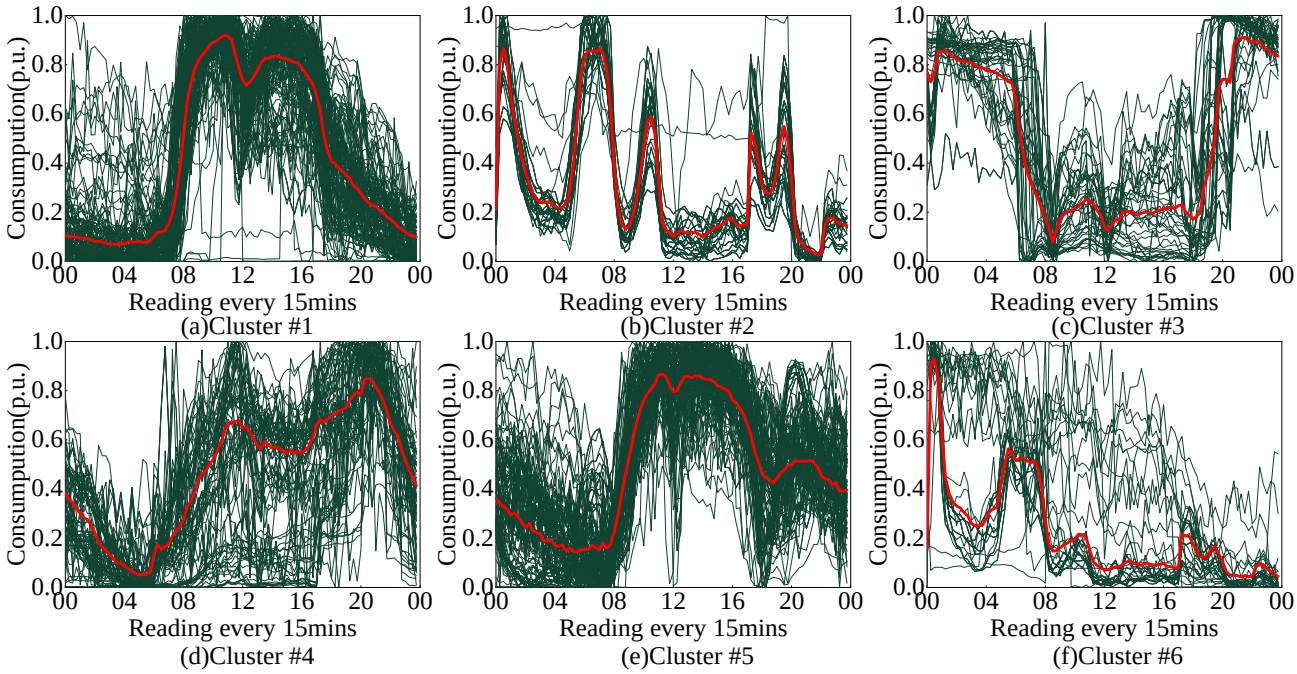

Fig. 21 . Clustering results of Random_center when k=6.

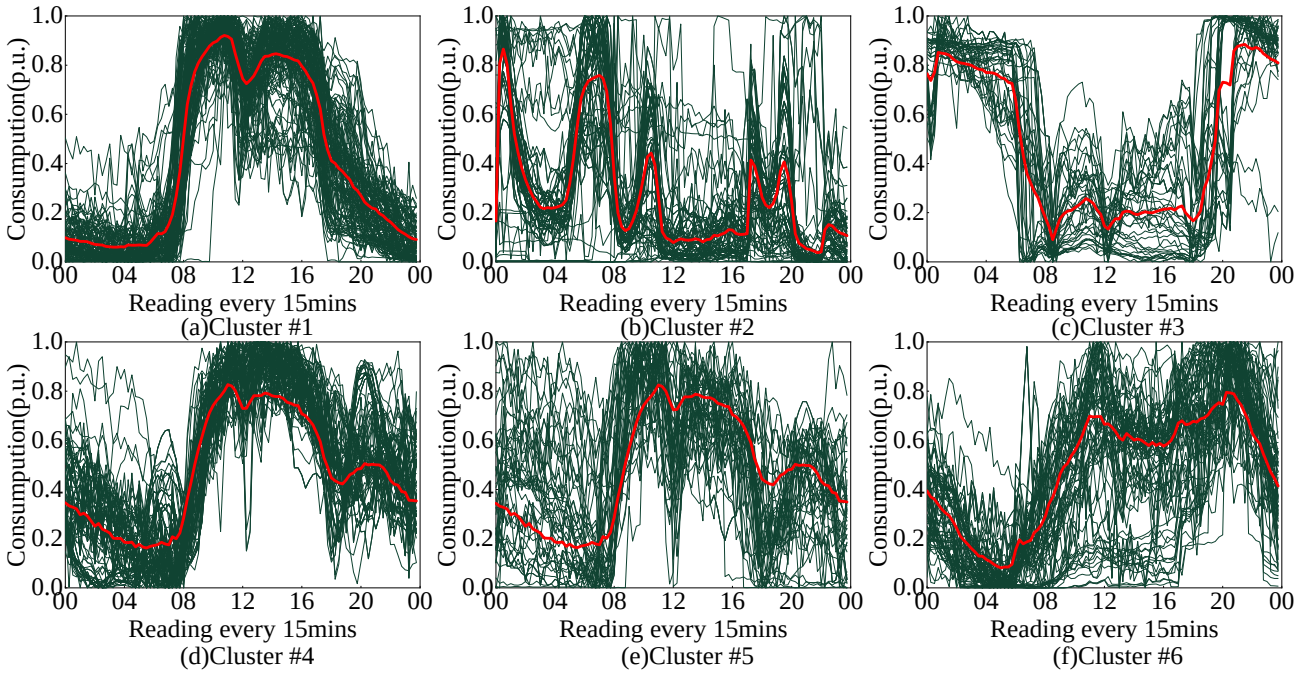

Fig. 22 . Clustering results of Eu_pfcm when k=6.

D7. Analysis of Power Load Characteristics

From the above analysis, it can be seen that the clustering effect of the method proposed in this article is the best. Therefore, an in-depth analysis was conducted on the actual load curve of the clustering results. From Fig. 20, the following power consumption modes can be observed.

- Typical double-peak power consumption patterns (Cluster 1)
- Multi-peak power consumption patterns (Cluster 2)
- Typical nighttime power consumption patterns (Cluster 3)
- Typical two-shift power consumption patterns (Cluster 5)

Cluster 1 represents the most common load consumption pattern, typical for the majority of commercial and industrial enterprises. At 9:00 AM, the load rises sharply as equipment like lighting, office machinery, and air conditioning are activated, reaching a peak. During the lunch break, the load decreases, rising again to a second peak at around 2:00 PM. After work ends, the load rapidly drops, resulting in a bimodal pattern with morning and afternoon peaks. This pattern aligns with the daily routines of office staff.

Cluster 5 displays a characteristic two-shift load pattern, commonly seen in the manufacturing sector. The morning shift begins with a gradual load increase at 8:00 AM, maintaining high levels throughout the shift. The evening shift starts after the shift change at 6:00 PM, with the load returning to high levels and sustaining until around 4:00 AM the next day.

Cluster 3 is a typical nighttime load pattern, frequently

observed in venues like hotels, nightclubs, and bars. The load begins to rise at 8:00 PM, remaining high and relatively stable until around 4:00 AM.

Cluster 4 represents a single-peak nighttime load pattern, with the load rising at 6:00 PM, peaking at 8:00 PM, and then gradually declining. Cluster 6, on the other hand, is less common, as its load remains high throughout the day with a moderate decline at night, likely due to the specific operational characteristics of certain enterprises.

V. Conclusion

This paper addresses the limitations of traditional algorithms in power load profile clustering by proposing the WF-KNNDP-PFCM algorithm. The proposed algorithm effectively overcomes the shortcomings of conventional clustering methods when applied to power load profiles. Experimental results demonstrate that the WF-KNNDP-PFCM algorithm outperforms existing algorithms in terms of clustering effectiveness, stability, iteration efficiency, and robustness. Compared to methods that use randomly initialized clustering centers, the KNNDP algorithm generates more effective initial cluster centers, significantly reducing the number of iterations, improving clustering stability, and enhancing overall algorithm performance. Furthermore, the combination of data weighting and the fusion similarity measure optimizes clustering quality. The results also highlight the significant impact of different initial clustering centers on clustering quality, further validating the effectiveness of the proposed algorithm.

Although the experimental results demonstrate the superior performance of the proposed algorithm, potential improvements remain:

(1) The PFCM algorithm involves multiple parameters that influence clustering performance. Integrating the algorithm with meta-heuristic methods to identify optimal parameter combinations could further enhance its performance.

(2) The clustering algorithm requires the number of clusters as an input parameter. Future work could develop an algorithm that automatically determines the optimal number of clusters.

(3) The current analysis focuses on population-level clustering results. Given the volatility in individual electricity consumption patterns, future research should conduct deeper analyses of user-specific consumption characteristics.

References

[1] Z. Jiang, R. Lin, F. Yang, and B. Wu, "A Fused Load Curve Clustering Algorithm Based on Wavelet Transform," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 5, pp. 1856-1865, May, 2018.

[2] K. Zheng, Q. Chen, Y. Wang, C. Kang, and Q. Xia, "A Novel Combined Data-Driven Approach for Electricity Theft Detection," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 3, pp. 1809-1819, Mar, 2019.

[3] C. Si, S. Xu, C. Wan, D. Chen, W. Cui, and J. Zhao, "Electric Load Clustering in Smart Grid: Methodologies, Applications, and Future Trends," *Journal of Modern Power Systems and Clean Energy*, vol. 9, no. 2, pp. 237-252, Mar, 2021.

[4] M. Hu, D. Ge, R. Telford, B. Stephen, and D. C. H. Wallom, "Classification and characterization of intra-day load curves of PV and non-PV households using interpretable feature extraction and feature-based clustering," *Sustainable Cities and Society*, vol. 75, Dec, 2021.

[5] S. Yilmaz, J. Chambers, and M. K. Patel, "Comparison of clustering approaches for domestic electricity load profile characterisation - Implications for demand side management," *Energy*, vol. 180, pp. 665-677, Aug 1, 2019.

[6] S. Ryu, H. Choi, H. Lee, and H. Kim, "Convolutional Autoencoder Based Feature Extraction and Clustering for Customer Load Analysis," *IEEE Transactions on Power Systems*, vol. 35, no. 2, pp. 1048-1060, Mar, 2020.

[7] K. Zheng, Q. Chen, Y. Wang, C. Kang, and Q. J. I. T. o. I. I. Xia, "A novel combined data-driven approach for electricity theft detection," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 3, pp. 1809-1819, 2018.

[8] S. Surianarayanan, A. Menaga, and K. Kannan, "An Efficient Approach on Groupings in Rainfall Prediction around Sathanur Dam, India, by K-Means, Hierarchical and Fuzzy Clustering Techniques," *IAENG International Journal of Computer Science*, vol. 51, no. 11, pp. 1769-1780, 2024.

[9] F. Yanuar, R. D. Putri, D. Devianto, A. Zetra, A. R. Putri, and Y. Asdi, "People's Welfare Clustering Using Fuzzy C-Means with Spatial Information," *IAENG International Journal of Computer Science*, vol. 51, no. 7, pp. 785-790, 2024.

[10] X. Wang, T. Wang, H. Xiang, and L. Huang, "A Parallel Genetic K-Means Algorithm based on the Island Model," *Engineering Letters*, vol. 32, no. 8, pp. 1632-1643, 2024.

[11] R. Suganya, R. J. I. J. o. S. Shanthi, and R. Publications, "Fuzzy c-means algorithm-a review," *International Journal of Scientific and Research Publications*, vol. 2, no. 11, pp. 1, 2012.

[12] G. Zhang, C. Zhang, and H. Zhang, "Improved K-means algorithm based on density Canopy," *Knowledge-Based Systems*, vol. 145, pp. 289-297, Apr 1, 2018.

[13] F. Di Martino, and S. Sessa, "A novel quantum inspired genetic algorithm to initialize cluster centers in fuzzy C-means," *Expert Systems with Applications*, vol. 191, Apr 1, 2022.

[14] H. Li, and J. Wang, "Collaborative annealing power k-means plus plus clustering," *Knowledge-Based Systems*, vol. 255, Nov 14, 2022.

[15] X.-y. Liu, J.-c. Fan, and Z.-w. Chen, "Improved fuzzy C-means algorithm based on density peak," *International Journal of Machine Learning and Cybernetics*, vol. 11, no. 3, pp. 545-552, Mar, 2020.

[16] L. Jing, M. K. Ng, and J. Z. Huang, "An Entropy Weighting k-Means Algorithm for Subspace Clustering of High-Dimensional Sparse Data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 8, pp. 1026-1041, Aug, 2007.

[17] M.-S. Yang, and Y. Nataliani, "A Feature-Reduction Fuzzy Clustering Algorithm Based on Feature-Weighted Entropy," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 2, pp. 817-835, Apr, 2018.

[18] M.-S. Yang, and J. B. M. Benjamin, "Feature-Weighted Possibilistic c-Means Clustering With a Feature-Reduction Framework," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 5, pp. 1093-1106, May, 2021.

[19] R. Lin, B. Wu, and Y. Su, "An Adaptive Weighted Pearson Similarity Measurement Method for Load Curve Clustering," *Energies*, vol. 11, no. 9, Sep, 2018.

[20] X. Zhang, J. L. Ramirez-Mendiola, M. Li, and L. Guo, "Electricity consumption pattern analysis beyond traditional clustering methods: A novel self-adapting semi-supervised clustering method and application case study," *Applied Energy*, vol. 308, Feb 15, 2022.

[21] S. Koehler, R. Rongstock, M. Hein, and U. Eicker, "Similarity measures and comparison methods for residential electricity load profiles," *Energy and Buildings*, vol. 271, Sep 15, 2022.

[22] Z. Yang, S. Jiang, F. Yu, W. Pedrycz, H. Yang, and Y. Hao, "Linear Fuzzy Information-Granule-Based Fuzzy C-Means Algorithm for Clustering Time Series," *IEEE Trans Cybern*, vol. 53, no. 12, pp. 7622-7634, Dec, 2023.

[23] P. Hore, L. O. Hall, D. B. Goldgof, Y. Gu, A. A. Maudsley, and A. Darkazanli, "A Scalable Framework For Segmenting Magnetic Resonance Images," *Journal of Signal Processing Systems for Signal Image and Video Technology*, vol. 54, no. 1-3, pp. 183-203, Mar, 2009.

[24] T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall, and M. Palaniswami, "Fuzzy c-means algorithms for very large data," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 6, pp. 1130-1146, 2012.

[25] S. S. Zhou, D. Li, Z. Zhang, and R. Ping, "A New Membership Scaling Fuzzy C-Means Clustering Algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 9, pp. 2810-2818, Sep, 2021.