IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 30 April 2024, accepted 15 May 2024, date of publication 20 May 2024, date of current version 29 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3403128

RESEARCH ARTICLE

Graph Distance and Adaptive K-Nearest Neighbors Selection-Based Density Peak Clustering

YUQIN SUN, JINGCONG WANG, YUAN SUN, PENGCHENG ZHANG, AND TIANYI WANG
School of Mathematics and Physics, Shanghai University of Electric Power, Shanghai 201306, China
Corresponding author: Yuan Sun (combmathe@shiep.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 12071274.

ABSTRACT Density Peak Clustering (DPC) is known for its rapid identification of cluster centers and successful clustering tasks. However, traditional DPC encounters several issues, which include simplifications in local density and distance metrics, a non-robust single-allocation strategy and limited fault tolerance. To address these challenges, this study introduces an innovative density peak clustering algorithm, named Graph Distance and Adaptive K-Nearest Neighbors Selection-Based Density Peak Clustering (GAK-DPC). Our goal with the approach is to enhance the algorithm's adaptability to non-linear and complex data structures. We achieve this by replacing the traditional Euclidean distance with graph distance. Additionally, we redefine the method for computing local density based on information from K-nearest neighbor data points. By introducing the concept of natural neighbors, the neighborhood radius r is obtained when all instances in the dataset have at least one natural neighbor. Then for the current data point, the number of data points falling within a circle centered on it with radius r is counted as the K-value of that data point. Thus, we achieve the adaptive selection of the K-value. This adaptive K-value strategy takes into account the dataset's characteristics and inter-point neighbor relationships, which enhances the algorithm's adaptability and robustness. Finally, we optimize the secondary allocation strategy for sample points to improve the algorithm's fault tolerance. By conducting comparisons with traditional clustering algorithms on UCI datasets and synthetic datasets, we demonstrate the effectiveness of GAK-DPC.

INDEX TERMS Adaptive K-neighbors, allocation strategy, density peak clustering, graph distance, natural neighbors.

I. INTRODUCTION

Data mining stands as a cornerstone technology within the fields of both the information industry and artificial intelligence. Within the domain of data mining, cluster analysis stands as a profoundly significant technique. Its primary objective is to categorize data into separate groups or clusters by assessing the similarities among data points. This ensures that data points within a common cluster share high similarity, while data points belonging to separate clusters exhibit reduced similarity. Cluster analysis holds pivotal importance as a fundamental technique across various fields, including social studies [1], psychological research [2], biology [3], statistics [4], recognition of patterns [5] and information retrieval [6] and so on.

There are five types of traditional clustering algorithms: partition-based clustering algorithms, hierarchical clustering algorithms [7], density-based clustering algorithms [8], model-based clustering algorithms [9] and grid-based clustering algorithms [10]. For specific types of data or applications, each algorithm possesses its own set of advantages and disadvantages. Within density-based clustering, clusters are delineated as regions that have elevated density in comparison to the remaining dataset. Laio and Rodriguez proposed a new density-based clustering technique known as DPC [11]. The rest of the samples are allocated to cluster with the

The associate editor coordinating the review of this manuscript and approving it for publication was Claudia Raibulet.

highest nearby density after cluster centers are determined based on relative distance and local density of samples. While DPC performs well on various datasets, it still has some limitations. For example, DPC cannot handle data with high density variations; DPC's use of the Euclidean distance for density computation and density peak search is not suitable for streaming structures and there is a domino effect of DPC on point assignments. Moreover the choice of the truncation distance δ has a significant impact on the final result of DPC.

To address the aforementioned issues, several researchers have proposed various enhancements to the DPC algorithm. Xie et al. [12] introduced an improved DPC algorithm that relies on fuzzy weighted K-nearest neighbors. This algorithm computes point's local density by summing the distances to neighboring points within a specified truncation distance. The inclusion of distance information from nearby points effectively influences local data points' density and incorporates their contributions. Du et al. [13] presented the Density Peaks Clustering Based On K-Nearest Neighbors (DPC-KNN) algorithm, which accounts for the spatial structure of sample points by considering k-nearest neighbors. This approach provides a local density calculation that accurately reflects the spatial characteristics of the sample points. DPC-KNN addresses the issue of cluster loss by considering the spatial structure of sample points. Rui et al. [14] introduced the Shared-Nearest-Neighbor-based Clustering by fast search and find of Density Peaks (SNN-DPC) algorithm, which redefines the formulas for relative distance and local density by using the number of closest neighbors. They also employ a secondary allocation method based on shared nearest neighbors as part of allocation strategy. The SNN-DPC algorithm effectively enhances clustering performance. Du et al. [15] incorporated geodesic distance to adapt to manifold structures. Wang et al. [16] proposed a variational density peak clustering (VDPC) algorithm. The algorithm can systematically handle initial clusters spanning different density levels by categorizing data points into different density levels and integrating the advantages of DPC and DBSCAN to finally obtain robust clustering results. Wang et al. [17] proposed the multi-center density peak clustering (McDPC). The algorithm uses a hierarchical strategy to first obtain representative data points based on the assumptions in DPC and automatically categorizes these representative data points to different density levels. Finally, McDPC can merge micro-clusters of specific levels into one cluster when needed. This algorithm effectively solves the problem that DPC may not be able to recognize clusters with multiple density peaks. Guo et al. [18] proposed a density peak clustering with connectivity estimation (DPC-CE). The algorithm estimates the connectivity between them using a graph-based strategy by selecting points with large relative distances as local centers and incorporating the connectivity information into the distance calculation. Thus, it effectively solves the problems of incorrectly identifying clustering centers and the ''chain reaction'' phenomenon of DPC in the case of uneven density. Cheng et al. [19] introduced a dense member density peak clustering algorithm based on local cores, which is called the Dense members of Local Cores-based Density Peaks Clustering (DLORE-DP) algorithm. By using newly defined graph distances and local cores to handle manifold datasets, DLORE-DP enhances clustering efficiency and robustness against noise and intricate data. Wang et al. [20] proposed a density peak clustering algorithm guided by pseudo labels (PLDPC). The algorithm avoids manual pre-specification of parameters by applying the mutual information criterion, thus solving the time-consuming problem of DPC for determining the optimal parameters. Experimental results show that PLDPC outperforms three classical and eight state-of-the-art clustering algorithms in most cases. Wang et al. [21] introduced a novel density peaks clustering algorithm for automatic selection of clustering centers based on K-nearest neighbors. The algorithm overcomes the problems of DPC that need to determine the clustering centers manually when dealing with complex datasets as well as the poor performance in the case of varying densities or non-convexity by automatically selecting the clustering centers based on k-nearest neighbors. Vu et al. [22] proposed the Constrained Density Peak Clustering (CDPC) algorithm that aims to optimize the clustering results through ''must link'' and ''cannot link'' constraints. It combines constraints and k-nearest neighbor graph techniques to accurately filter peaks and find the center of each cluster. This method addresses the limitation that DPC has difficulty in clustering on datasets with both high and low density clusters. Xu et al. [23] proposed an automatic density peaks clustering based on a density-distance clustering index (ADPC) algorithm. The algorithm introduces a new clustering effectiveness metric called density-distance clustering (DDC), based on which the cut-off distance is automatically selected without additional parameters. Experimental results show that the algorithm is able to automatically determine the optimal number of clusters and cut-off distance and outperforms DPC, AP and DBSCAN.

In conclusion, each of the above papers improves DPC in terms of local density definition, similarity measure, cluster center selection and microcluster merging, respectively. However, it is difficult to simultaneously satisfy the requirements of adapting the algorithm to complex structured datasets, adaptively adjusting according to the dataset characteristics when calculating the local density and avoiding the propagation of data point errors. In this work, a new density peak clustering approach named GAK-DPC is presented. Firstly, by introducing graph distances, the relationships among data points are represented as the shortest paths in a graph. This allows the algorithm to capture interactions among data points more accurately, overcome the limitations of the distance metric and handle nonlinear and complex data structures efficiently. Secondly, a K-nearest neighbor-based approach for calculating local data point density is proposed. Unlike manual setting of neighborhood radius and K values,

this algorithm automatically adapts based on the intrinsic characteristics of the dataset when calculating local density. This enhances the versatility of the algorithm and allows the algorithm to better meet the different needs of various datasets, thus improving the quality of clustering. Finally, the allocation strategy for data points has been optimized to better handle exceptional cases within the dataset. By constraining the possible assignment of points, the algorithm more accurately identifies cluster boundaries and noisy points, which enhances the robustness of the algorithm. In summary, this paper improves the algorithm by proposing a novel adaptive method of calculating local density points and module combination innovation. It solves the problems that the cut-off distance of DPC is difficult to determine, the data point allocation is prone to cascading errors and it is difficult to deal with nonlinear and complex data. The experimental results show that the clustering quality of the improved DPC is greatly improved. The first part of this paper introduces the current status of research at home and abroad and introduces the algorithm GAK-DPC proposed in this paper. The second part describes the algorithmic steps of DPC and the concepts related to GAK-DPC. The third part describes the implementation process of GAK-DPC in detail. The last part validates the experimental results of GAK-DPC on UCI datasets and synthetic datasets. Finally GAK-DPC is analyzed for parameter sensitivity and time complexity.

II. PRELIMINARIES

A. DPC ALGORITHM

Two fundamental presumptions underlie DPC: (1) other data points with lower densities within the cluster surround the center of cluster; (2) relatively long distances among cluster centers. A decision graph can be made by examining each data point's distance value δ and local density parameter ρ. Under the Gaussian kernel, for each data point the determination of the local density parameter is carried out in the following manner:

$$\rho_i = \sum_{i \neq j} \exp[-(\frac{d_{ij}}{d_c})^2] \tag{1}$$

where d_{ij} is the Euclidean distance between points x_i and x_j; d_c is the cut-off distance, which is usually set to 1 to 2 percent of the distance in descending order.

The data point local density under the truncated kernel is specified as follows:

$$\rho_i = \sum_{i \neq j} \chi(d_{ij} - d_c) \tag{2}$$

where $\chi(x)$ is a logical judgment function, the function value is 1 if $x < 0$ and 0 otherwise. The original paper on DPC suggests that for larger-scale datasets, the clustering performance is better when using the truncated kernel calculation method. Conversely, for smaller-scale datasets, the Gaussian kernel calculation method yields more noticeable clustering results.

Each data point and data points with higher local densities have a minimum distance value, δ_i, of:

$$\delta_i = \begin{cases} \min\limits_{j:\rho_j > \rho_i} (d_{ij}), & \text{others} \\ \max\limits_{j}(d_{ij}), & \rho_i = \max(\rho) \end{cases} \tag{3}$$

where $j : \rho_i > \rho_j$ is assemblage of data points with local densities higher than data point x_i. As there are no places in the sample with the highest density, DPC designates this point as a density peak (cluster center) and artificially assigns its relative distance from the maximum. Two conditions must be met for the remaining density peaks: a significant relative distance δ and a high local density ρ. To achieve this, the initial paper of DPC finds such density peaks using a decision value and the following equation provides the definition of γ:

$$\gamma = \rho \times \delta \tag{4}$$

Using the decision values in (4), the user can determine the clustering centers. Alternatively, a decision diagram as shown in Figure 1 is constructed using the local density as the horizontal axis and the relative distance as the vertical axis. The clustering centers are manually selected peak density points with large values of ρ and δ.

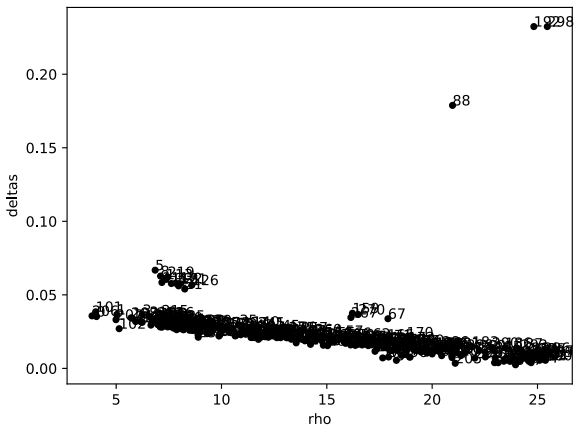

FIGURE 1. The decision graph of DPC on the spiral dataset.

After determining the cluster centers, labels are assigned to these cluster center points. Then, for data points without assigned labels, the clustering process is completed by assigning labels based on the nearest data points with a higher density that are already labeled. Algorithm 1 displays the steps involved in DPC.

B. BASIC KNOWLEDGE

Definition 1 (K-Nearest Neighbors): Given a dataset X containing n data points $\{x_1, x_2, \ldots, x_n\}$. Compute distances between data point x_i and the dataset X's remaining points, then sort these separations in ascending order. Let $d(x_i, k)$ represent the K-th distance and record the first K distances' indices. The equivalent data points are the K-nearest neighbors of x_i, which is described as:

$$NN_k(x_i) = \left\{x_j \in D(x_i, x_j) \leq d(x_i, k)\right\} \tag{5}$$

Algorithm 1 DPC Algorithm

Input: Dataset $X = \{x_1, x_2, \ldots, x_n\}$, parameter p
Output: $C = \{C_1, C_2, \ldots, C_m\}$
1: Calculate distance matrix $D_{n\times n}$
2: Determine d_c value
3: Compute ρ_i and δ_i for each data point in accordance with (1) and (3)
4: Make a decision graph and select cluster centers
5: Distribute non-cluster center points

Definition 2 (Reverse K-Nearest Neighbors): Given a dataset X containing n data points $\{x_1, x_2, \ldots, x_n\}$ and a query point q. Finding the K-nearest neighbor data points of the query point q or points which have the query data point q as one of their K-nearest neighbors, is the goal of reverse K-nearest neighbor. This can be described as:

$$RNN_k\ (q) = \{x \in D\,|q \in NN_k(x)\} \tag{6}$$

C. GRAPH DISTANCE

First, we calculate the K_1-nearest neighbor neighborhoods (K_1 is the first input parameter) for each point and construct a graph by connecting all neighboring points. Then we use Euclidean distance between respective connected points as the weight to label each arc in the graph. In the end, the graph distance between two points is estimated as the sum of the arc lengths along the shortest path connecting them and we compute the shortest paths between any two locations in the graph by using Floyd algorithm.

If data point i is one of the K-nearest neighbors of data point j, then by joining i and j, a graph G is defined over all data points. Calculate the edge length $d(i, j)$ between points i and j using the Euclidean distance, which is:

$$d(i, j) = \left\|x_i - x_j\right\| \tag{7}$$

The graph distance between points i and j is represented by the symbol $d_G(i, j)$, which is:

$$d_G(i,j) = \begin{cases} d(i,j), & \text{if i,j are linked by an edge} \\ \infty, & \text{otherwise} \end{cases} \tag{8}$$

Then, we replace every entry in the graph with the shortest pathways, as defined in (9).

$$d_G(i, j) = \min\{d_G(i, j), d_G(i, l) + d_G(l, j)\} \tag{9}$$

where $l = 1, 2, \ldots, N$. Eventually, a novel distance matrix $D_G = \{d_G(i, j)\}$ is generated, which lists the shortest path distances between any two graph locations.

III. THE PROPOSED ALGORITHM

Firstly, it replaces the Euclidean distance with graph distance, which enhances the algorithm's performance on manifold datasets. Secondly, it defines a K-nearest neighbor-based approach for calculating local point density and adaptively determines the value of K based on dataset characteristics. Lastly, we optimize the secondary allocation strategy for samples, which effectively solves issues that DPC did not consider spatial properties among data points and the one-step allocation problem.

A. DISTANCE MEASUREMENT

Most existing DPC algorithms are unable to discover manifold clusters because traditional Euclidean distance calculation methods cannot capture the complex relationships among data points and handle the nonlinear data structures well. According to [15], it is evident that geodesic distance can effectively handle manifold data and offer a more precise representation of the relative positions of data points. In [24], the author points out when there are a sufficient number of samples, geodesic distance can be estimated using graph distance. Therefore, in this paper, the graph distance introduced in section II is used for distance measurement. This type of distance calculation constructs a connectivity graph among data points by considering their K-nearest neighbor connections and calculates the shortest paths to measure the distances among them.

B. ADAPTIVE LOCAL DENSITY CALCULATION

In this work, we utilize the K-nearest neighbor information of points to establish local density. Doing so avoids the use of the cut-off distance d_c, which relies on the specific dispersion of data and is difficult to determine. Therefore how to choose the appropriate value of K becomes a problem we need to solve.

Inspired by DPC's approach, which establishes cut-off distance d_c by calculating the proportion of locations with a distance on average lower than d_c, typically ranging from 1% to 2% of the total points, this paper attempts to find a suitable neighborhood radius r for each dataset. For the current data point x_i, it calculates the quantity of data points that fall within a circle centered at x_i with a radius of r as x_i's K-value. Therefore, the K-value k_i for the current data point x_i is (10):

$$k_i = \sum_{j=1}^{n} I(d_G(i, j) \le r) \tag{10}$$

where n is the dimension of dataset and indicator function $I(\cdot)$ yields 1 in the case when the condition included in parenthesis is true and 0 in the other case.

Next, we obtain the local density of the point x_i by modifying k_i using the redefined density formula. The local density $\rho(x_i)$ of point x_i is:

$$\rho\ (x_i) = \sum_{j=1}^{k_i} \exp(-\,d_G\,(i, j)^2) \tag{11}$$

where $d_G(i, j)$ denotes the graph distance between x_i and its $j - th$ neighbor.

This method yields a more accurate local density estimation according to the distribution of surrounding points by

allowing an adaptive modification of the quantity of neighbors when computing the local density of points.

Natural neighbor [25] is a novel idea of neighbors, whose search algorithm can independently find neighbors without human intervention. Moreover, each point's neighbor count is mutually independent, which reflects a natural way of thinking. Applications include clustering evaluation [26], instance minimization [27] and outlier detection [28] illustrates its effectiveness.

Definition 3 (Natural Neighbors): Given a dataset X containing n data points $\{x_1, x_2, \ldots, x_n\}$. Natural neighbor of x_i is defined as follow:

$$x_j \in NN(x_i) \Leftrightarrow (x_i \in NN_k(x_j)) \wedge (x_j \in NN_k(x_i)) \tag{12}$$

Natural neighbor's underlying idea is that we continuously widen the range of objects we are searching for and we compute each time how many of those objects are neighbors of other objects. This process continues until either all objects are neighbors or the number of objects that are not neighbors of other objects remains constant. Introducing KD-tree [29] into the natural neighbor search algorithm can reduce its time complexity. According to [19], Algorithm 2 provides a description of the Natural Neighbor searching algorithm.

In typical situations, neighborhood of an object in a sparse region should be small, while the neighborhood in a dense region should be large. Using a fixed K value may lead to neighborhoods that are too crowded in some areas and too sparse in others. In Algorithm 2, the natural neighbors in the dataset are searched by iteratively adjusting the r until the exit condition is met, i.e., all instances in the dataset have at least one natural neighbor. The r represents the range of the number of natural neighbors rather than a direct radius value. However, it can be interpreted as an indirect neighborhood radius because it is the result of a number dynamically selected based on the dataset, which reflects the range of the number of natural neighbors for each data point. Thus it can be used to some extent as a measure of the spatial extent around the data point. Therefore, according to (10) and (11), we use the r obtained in Algorithm 2 after shrinking it by a factor of 100 as the neighborhood radius for calculating the local density. Too large a reduction radius may result in too few points in the local neighborhood, making the clustering results deviate from the actual data structure. On the contrary, too small a shrinkage radius may make the local neighborhood still contain too much noise and outliers. Through experiments, we found that shrinking the radius by a factor of 100 can balance these two extremes to some extent. Therefore, shrinking by 100 times is regarded by us as a relatively reasonable balance point, which can improve the accuracy and stability of the clustering results while maintaining the performance of the algorithm.This can better take into account the non-uniformity of data distribution, which reduces the risk of overfitting and enhances the algorithm's adaptability and performance.

Algorithm 2 NaN- Searching.

Input: Dataset X
Output: r
1: Initializing: $r = 1, nb(i) = 0, NN_0(i) = \phi, RNN_0(i) = \phi$
2: For the dataset X, find the rth nearest neighbor n of the current data point m
3: Record the number of nearest neighbors of n. Increase the nb value of n by 1 to indicate that n is a nearest neighbor point of m
4: Add the newly found nearest neighbor n to the set $NN_r(m)$ of rth nearest neighbors of point m
5: Add m to the set of reverse rth nearest neighbors $RNN_r(n)$ of point n
6: Iterate over the next data point, if the next data point exists then skip to step 2, if not then proceed to step 7
7: Compute the number of points with no neighbor (i.e., $nb(m) = 0$) Numb;
8: If the value of Numb remains unchanged then skip to step 10, if it still changes then proceed to step 9
9: Accumulate the value of r: $r = r + 1$ and jump to step 2
10: Output the r

C. CLUSTER CENTER SELECTION AND THE ALLOCATION OF THE REMAINING DATA POINTS IN CLUSTERING

In this study, cluster centers are chosen based on (3) and (4). Cluster centroids are selected manually, based on the γ value. Higher γ values correspond to higher ρ and δ values. A cluster center is more likely to form at this point.

Definition 4 (Inevitable Subordinate Point): When x_i is allocated to the corresponding cluster and x_j has not been assigned yet, this happens only if it fulfills:

$$\left|\left\{p \,|p \in KNN(x_i) \cap p \in KNN(x_j)\right\}\right| \geq l \times K_2 \tag{13}$$

It is considered that point x_j should be a part of the same cluster as data point x_i. Where K_2 is the second input factor, while l represents the proportion of shared neighbors to total neighbors that two data points need to satisfy in order to be grouped into the same cluster. The value of l provided in [14] is 1/2, while in reference [30], it is given as 3/4. For most datasets, the challenge lies in dealing with the overlapping regions between clusters, where they intersect and overlap. To reduce the risk of allocation errors, this paper has raised the conditions for reachability by increasing the value of l. Experimental results have shown that setting l to 18/23 achieves optimal results for most datasets. Specifically, when l is set to 0.753, it produces the aggregation dataset's best results. For the remaining unallocated data points, their neighbor characteristics are considered and they are assigned to the cluster that has the most relationships with its neighbors.

D. THE STEPS OF THE GAK-DPC

GAK-DPC adheres to the fundamentals of DPC and introduces improvements in crucial steps. The entire process is displayed in Algorithm 3.

Algorithm 3 GAK-DPC.

Input: $X = \{x_1, x_2, \ldots, x_n\}$ (n is the number of entries), number of neighbors K_1 and K_2

Output: result of clustering $C = \{C_1, C_2, \ldots C_m\}$ (m is the number of clusters)

1: Initialize dataset X
2: Calculate distance matrix $D^{n\times n} = \{d_{ij}\}^{n\times n}$ according to (9)
3: Apply Algorithm 2 to calculate neighborhood radius r
4: Calculate local density ρ according to (10) and (11)
5: Determine distance from the closest bigger density point δ according to (3)
6: Calculate decision value γ according to (4), and sort it in ascending order and note the new order of all elements
7: Create a decision graph and choose the cluster centers
8: Allocate non-cluster center points according to (13)
9: Data points that do not satisfy (13) are assigned to the cluster that has the most relationships with its neighbors
10:Output the clustering results

E. TIME COMPLEXITY ANALYSIS

For the dataset $X = \{x_1, x_2, \ldots, x_n\}$, the number of nearest neighbors K_1 and K_2, the time complexity of GAK-DPC proposed in this paper consists of the following parts: (1) Calculate the matrix of graph distances between all the data points $O(n^3)$; (2) Calculate the radius of the neighborhood according to Algorithm 2 $O(n \log n)$; (3) Calculate the local densities of all the points $O(n^2)$; (4) Determine the distances to the nearest points of larger densities δ $O(n^2)$; (5) Calculate the decision value and order it $O(n \log n)$; (6) The number of clusters in this part is counted as m. For the assignment of unavoidable slave points there is $O(mn^2)$, and for the assignment of possible slave points there is $O((K_2 + m)\, n^2)$. Thus, the overall computational complexity of the method proposed in this paper is $O(n^3)$.

IV. EXPERIMENT AND ANALYSIS

This experiment uses a set of UCI datasets and synthetic datasets to evaluate the clustering effect of GAK-DPC. Detailed information of the datasets used in this study is provided in Tables 1 and 2, which include sample size, attributes and classes for each dataset.

We compare the experimental results with several clustering algorithms, including SNN-DPC [14], McDPC [17], DPC-CE [18], DPC [11], DBSCAN [31] and K-Means [32]. For DBSCAN and K-Means algorithms, we use implementations available in the Python sklearn library [33]. SNN-DPC, McDPC, DPC-CE and DPC all employ publicly available source code, with DPC utilizing Gaussian kernel clustering. Among them McDPC and DPC-CE are implemented in MATLAB, other than that the algorithms are implemented in Python.

TABLE 1. Synthetic datasets.

Datasets	Size	Attributes	Classes
Atom	800	3	2
Aggregation	788	2	7
Spiral	312	2	3
Pathbased	300	2	3
Flame	240	2	2
Rings	1500	2	3
Chainlink	1000	3	2
Complex	3031	2	9
2circles	600	2	2
Halfkernel	1000	2	2
Twomoons	200	2	2
Threecircles	299	2	3
Fourlines	512	2	4

TABLE 2. Real-world datasets.

Datasets	Size	Attributes	Classes
Iris	150	4	3
Wine	178	13	3
Seeds	210	7	3
Segment	2310	19	7
Waveform	5000	21	3
Parkinsons	197	23	2
Spect Heart	267	44	2

A. CLUSTER EVALUATION INDEX

This study uses Adjusted Mutual Information (AMI) [34], Adjusted Rand Index (ARI) [34] and Fowlkes-Mallows Index (FMI) [35] as evaluation metrics. AMI and ARI have a range of values between -1 and 1, while FMI's range is between 0 and 1. The optimal results for all three metrics are achieved when the value is 1. When the value is close to 1, it indicates a greater clustering ability. When AMI and ARI are negative, the labels are dispersed separately, which indicates poor clustering.

At first, we apply the ''min-max normalization'' approach to preprocess the data. This preprocessing step not only eliminates the impact of various dimensions on the outcomes of the experiment but also reduces the algorithm's execution time. By normalizing the data, we ensure that features at different scales would contribute equally to the clustering analysis, thereby we enhance the algorithm's robustness and reliability. This preprocessing step is crucial to ensuring the fairness and accuracy of the experiments.

$$x'_{ij} = \frac{x_{ij} - \min(x_j)}{\max(x_j) - \min(x_j)} \tag{14}$$

where x_j is a primary data in the entire $j-th$ column, x_{ij} is the original data for the $i-th$ data point in the $j-th$ data entry and x'_{ij} is a re-scaled data for the $i-th$ data point in the $j-th$ data entry.

B. ALGORITHM PARAMETER SETTINGS

The parameter settings for each method utilized in the studies are shown in Table 3 based on the previously mentioned datasets. The parameters for each algorithm are selected based on their optimal values.

In this study, GAK-DPC requires the specification of two parameters: K_1, which is used when computing geodesic distances and K_2, which is used for allocating the remaining data points. We have designed a nested loop function, where K_1 acts as the outer loop and K_2 is the inner loop. The outer loop is kept constant while the inner loop is incremented by 1 from 3 until it reaches a threshold value of 40 when it jumps out of the inner loop. Then, the outer loop is also incremented by 1 from 3 until the traversal stops when both the inner and outer loops reach the threshold 40. At each parameter change, we calculate the ARI value of the dataset under the current parameter and record the current parameter. If a better solution than the current ARI value is found, the record is replaced to find the optimal solution. This range was chosen because for certain datasets, a small value of K may result in infinite computations which can lead to errors. Meanwhile, the results of the algorithm will not be much affected by a large value of K. Therefore, further exploration beyond this range may not be meaningful. In addition, in parentheses after the parameters we indicate the l values for each dataset, which is presented in the form of a score. SNN-DPC has a single integer parameter, K. The allowable range for these K values is also between 3 and 40 and they should be positive integers.

McDPC has four parameters, which are γ, θ, λ and pct. Parameters γ and θ perform ρ-cut and δ-cut, respectively, λ is the threshold for identifying micro-clusters and pct is used to generate decision diagrams. DPC-CE has two fixed parameters T_r and P_r, whose values do not require additional tuning for all datasets. For DPC, the authors provide an empirical rule that suggests modifying the parameter "d_c" to influence clustering outcomes. Even when the number of neighboring points falls within 1-2% of the overall amount of points, this experiment varies this proportion to achieve the greatest outcomes.

DBSCAN has two parameters, which are minpts (an integer) and ε (a floating-point number). K-Means takes the quantity of clusters in the dataset as its input parameter.

C. EXPERIMENTAL RESULTS ON SYNTHETIC DATASETS

We use a variety of synthetic datasets in this part to test different clustering strategies. These datasets vary according to the number of point clusters and total distribution. They serve to simulate different scenarios and allow for the comparison of the effectiveness of different clustering algorithms under diverse conditions.

Table 4 provides the clustering evaluation metric values for seven algorithms on thirteen synthetic datasets, including ARI, AMI and FMI. The results in bold indicate the optimal values for a particular metric within the same dataset. It is evident from the table that GAK-DPC performs the best when it comes to clustering on the majority of the datasets. Regarding the flame dataset, which is made up of two clusters and exhibits a balanced distribution, the algorithm's metrics may not be the best. However, the clustered images obtained from the experiments show that the results of this algorithm differ only slightly from those of DPC in the overlapping regions of the two clusters. The algorithm fails to accurately depict this data point. DPC employs a simple and efficient allocation approach that gives the cluster with the highest density ratio a data point and the closest allocated data point. While this strategy is effective, it is susceptible to the data point's structural characteristics and lacks robustness. In this study, GAK-DPC demonstrates greater clustering robustness than DPC on the majority of the datasets.

For the thirteen synthetic datasets, the clustering findings of the technique suggested in this work all obtain values above 0.95, which indicates that GAK-DPC can produce excellent clustering outcomes on complex-shaped datasets. In contrast, other clustering algorithms yield suboptimal results on certain datasets. For instance, on the spiral dataset, K-Means shows unsatisfactory clustering performance. On the twomoons dataset, only GAK-DPC, DPC-CE and DBSCAN exhibit relatively ideal clustering results. SNN-DPC and McDPC also demonstrate satisfactory clustering outcomes on only some of the datasets. Meanwhile, DPC performs well on the aggregation, flame and spiral datasets but did not excel on others.

Then, we will provide the clustering findings for a few of the experimental datasets. In the figures, points of the same color represent data points assigned to the same cluster, while distinct clusters are depicted in different colors. Additionally, except DBSCAN, McDPC and DPC-CE, cluster centers obtained by all other algorithms are represented by cross-shaped symbols.

According to the clustering outcomes shown in Figure 2, the aggregation dataset's clusters may be found using all six algorithms listed in the figure. However, these algorithms differ in their clustering results mainly in their assignment of data points at the intersection of the two rightmost ellipse clusters. SNN-DPC incorrectly divides some data points from the upper ellipse to the lower ellipse, whereas GAK-DPC, DPC-CE and DPC incorrectly assign only one data point at the articulation of the two ellipses. DBSCAN, while having some noise, correctly shape each cluster they identify.

Figure 3 displays the outcomes of various algorithms on the flame dataset. It can be observed that the clusters can be appropriately identified by GAK-DPC, SNN-DPC, DPC-CE, DPC and DBSCAN. GAK-DPC and SNN-DPC have slight variations in their assignments compared to DPC and DPC-CE. Only one point is misclassified at the upper-lower boundary. DBSCAN has some noise. McDPC fails to correctly identify cluster divisions, as it incorrectly divides the right end portion of the lower cluster into a third cluster, whereas the flame dataset has only two categories.

The pathbased dataset's clustering results from several techniques are shown in Figure 4. As the image illustrates,

TABLE 3. Experimental parameter values.

Data	GAK-DPC	SNN-DPC	McDPC	DPC-CE	DPC	DBSCAN	K-Means
Atom	5/1(18/23)	3	0.1/2/5/2	0.25/0.3	2.0	0.06/20	2
Aggregation	7/28(0.753)	15	0.5/0.1/2.9/4	0.25/0.3	3.9	0.04/6	7
Spiral	3/1(18/23)	5	0.1/0.03/3.5/2	0.25/0.3	2.0	0.04/2	3
Pathbased	6/29(18/23)	9	0.12/0.8/3.5/0.5	0.25/0.3	3.8	0.08/10	3
Rings	6/1(18/23)	4	0.1/1.8/0.3/2.5	0.25/0.3	1.9	0.07/20	3
Chainlink	5/1(18/23)	3	0.2/1/1/1.8	0.25/0.3	2.0	0.08/20	2
Complex	7/1(18/23)	3	0.1/0.7/20/2.1	0.25/0.3	1.4	0.08/20	9
Flame	8/14(18/23)	5	0.02/0.5/3/2.5	0.25/0.3	2.8	0.09/8	2
2circles	5/1(18/23)	37	0.2/2/3/2	0.25/0.3	2.0	3/3	2
Halfkernel	7/1(18/23)	3	0.5/5/7/4	0.25/0.3	1.0	3/4	2
Twomoons	7/9(18/23)	10	0.2/0.26/0.065/3.56	0.25/0.3	2.0	0.45/3	2
Threecircles	6/1(18/23)	5	0.1/1/0.08/2	0.25/0.3	1.7	0.27/9	3
Fourlines	5/1(18/23)	11	0.3/0.2/0.07/4	0.25/0.3	2.0	0.09/11	4
Iris	5/19(18/23)	15	0.03/0.5/1/0.8	0.25/0.3	0.2	0.12/5	3
Wine	20/28(18/23)	18	0.01/0.1/250/0.2	0.25/0.3	2.0	0.50/21	3
Seeds	8/8(18/23)	6	0.2/0.01/2/2	0.25/0.3	0.7	0.24/16	3
Segment	14/15(18/23)	7	0.1/0.5/37/1.4	0.25/0.3	1.5	0.15/2	7
Waveform	7/7(18/23)	7	0.1/1/5.2/0.73	0.25/0.3	0.1	0.38/5	3
Parkinsons	7/20(18/23)	5	0.2/0.009/48/2	0.25/0.3	1.2	0.50/17	2
Spect Heart	3/22(18/23)	32	0.2/0.1/1.5/5	0.25/0.3	2.0	1.5/8	2

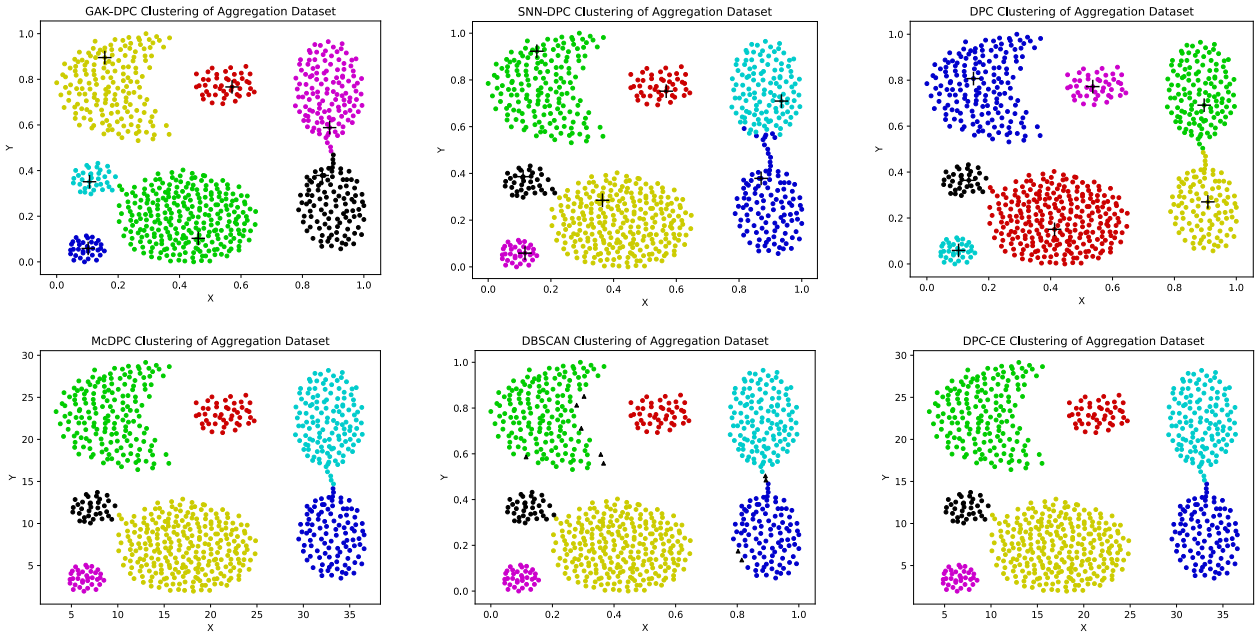

FIGURE 2. The clustering results on aggregation by 6 algorithms.

while GAK-DPC, SNN-DPC and DPC are able to identify cluster centers, DPC incorrectly divides one cluster into three. The half-ring cluster's left and right sides are wrongly associated with the other two clusters, so it leaves only a small portion of the top of the half-ring cluster. DPC-CE also incorrectly divides the semicircular ring into three parts and it incorrectly assigns data to the right-of-center clusters in more circular regions, leading to more incorrect division of data. SNN-DPC has some misclassifications at the intersection of the middle two clusters with the half-ring cluster. For DBSCAN, despite correctly assigning the other two clusters, the half-circle cluster as a whole is labeled as noise due to its significantly lower density compared to Minpts. Therefore, only GAK-DPC and McDPC correctly partition this dataset.

Figure 5 showcases the clustering results of various algorithms on the spiral dataset, highlighting the algorithms' ability to handle intertwined datasets. These algorithms all perfectly classified the dataset. The difference lies in the

TABLE 4. Comparison of clustering results on synthetic datasets.

Data	Index	GAK-DPC	SNN-DPC	McDPC	DPC-CE	DPC	DBSCAN	K-Means
Atom	ARI	**1**	**1**	**1**	0.2468	0.0955	**1**	0.1821
	AMI	**1**	**1**	**1**	0.3428	0.2157	**1**	0.2923
	FMI	**1**	**1**	**1**	0.6694	0.6463	**1**	0.6547
Aggregation	ARI	0.9978	0.9594	**1**	0.9978	0.9978	0.9779	0.7300
	AMI	0.9956	0.9500	**1**	0.9956	0.9956	0.9529	0.7935
	FMI	0.9983	0.9681	**1**	0.9983	0.9983	0.9827	0.7884
Spiral	ARI	**1**	**1**	**1**	**1**	**1**	**1**	−0.0060
	AMI	**1**	**1**	**1**	**1**	**1**	**1**	−0.0055
	FMI	**1**	**1**	**1**	**1**	**1**	**1**	0.3274
Pathbased	ARI	**1**	0.9294	**1**	0.4738	0.4717	0.9011	0.4613
	AMI	**1**	0.9001	**1**	0.5725	0.5212	0.8234	0.5098
	FMI	**1**	0.9529	**1**	0.6938	0.6664	0.9340	0.6617
Rings	ARI	**1**	**1**	0.9278	0.5167	0.2158	**1**	0.1148
	AMI	**1**	**1**	0.9421	0.7014	0.2815	**1**	0.1987
	FMI	**1**	**1**	0.9524	0.7378	0.4910	**1**	0.4401
Chainlink	ARI	**1**	**1**	0.0872	0.6225	0.2678	**1**	0.0879
	AMI	**1**	**1**	0.2133	0.7424	0.3564	**1**	0.0644
	FMI	**1**	**1**	0.6764	0.7888	0.6753	**1**	0.5435
Complex	ARI	**1**	0.6631	0.4166	0.4775	0.5124	0.2145	0.3458
	AMI	**1**	0.8389	0.6927	0.6626	0.7604	0.4352	0.6098
	FMI	**1**	0.7234	0.5141	0.5850	0.5969	0.5326	0.4514
Flame	ARI	0.9833	0.9502	0.7338	**1**	**1**	0.9388	0.4534
	AMI	0.9634	0.8975	0.7162	**1**	**1**	0.8234	0.3863
	FMI	0.9922	0.9768	0.8649	**1**	**1**	0.9712	0.7364
2circles	ARI	**1**	0.3164	**1**	**1**	0.0127	**1**	-0.0017
	AMI	**1**	0.3778	**1**	**1**	0.0921	**1**	-0.0012
	FMI	**1**	0.6891	**1**	**1**	0.6719	**1**	0.4983
Halfkernel	ARI	**1**	**1**	**1**	**1**	0.4132	**1**	0.0011
	AMI	**1**	**1**	**1**	**1**	0.4409	**1**	0.0008
	FMI	**1**	**1**	**1**	**1**	0.7163	**1**	0.5021
Twomoons	ARI	**1**	0.4068	0.0915	**1**	0.6064	**1**	0.1325
	AMI	**1**	0.4557	0.1239	**1**	0.5953	**1**	0.1022
	FMI	**1**	0.7175	0.5235	**1**	0.8057	**1**	0.5646
Threecircles	ARI	**1**	0.5310	**1**	**1**	0.0308	0.8739	0.0547
	AMI	**1**	0.6857	**1**	**1**	0.1792	0.8637	0.1632
	FMI	**1**	0.7160	**1**	**1**	0.4876	0.9193	0.4031
Fourlines	ARI	**1**	**1**	**1**	**1**	0.4680	**1**	0.4514
	AMI	**1**	**1**	**1**	**1**	0.6112	**1**	0.5617
	FMI	**1**	**1**	**1**	**1**	0.6045	**1**	0.6011

fact that the clustering centers of the GAK-DPC algorithm are closer to the endpoints, which aids in the allocation of the remaining data points. The clustering result charts of all synthetic datasets except the four mentioned above will be displayed in Figure 6.

D. EXPERIMENTAL RESULTS ON REAL-WORLD DATASETS

To further evaluate the clustering performance of GAK-DPC, this section conducts a comparative analysis, pitting GAK-DPC against six alternative algorithms across seven distinct real-world datasets featuring diverse structures and dimensions. Table 5 displays the results of clustering evaluation with bold data representing the best clusters.

From the table, it is evident that GAK-DPC delivers the most favorable clustering outcomes among the datasets presented. In particular, on the waveform dataset, the ARI of this algorithm is improved by 60.45% over DPC; on the wine dataset, the ARI of this algorithm is improved by 35.84% over DPC; on the parkinson dataset, the ARI of this algorithm is improved by 29.25% over SNN-DPC; in terms of performance metrics on both the seeds and segment datasets, the method performs at least 2% better than the other six algorithms. Combining the bar charts of clustering evaluation

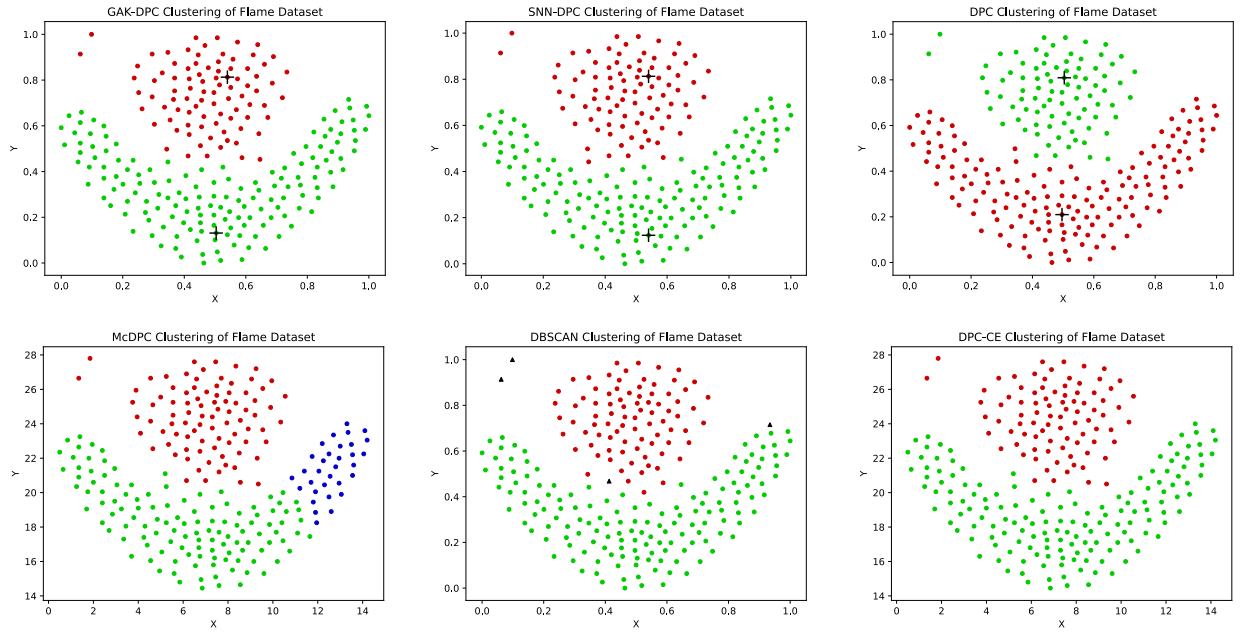

FIGURE 3. The clustering results on flame by 6 algorithms.

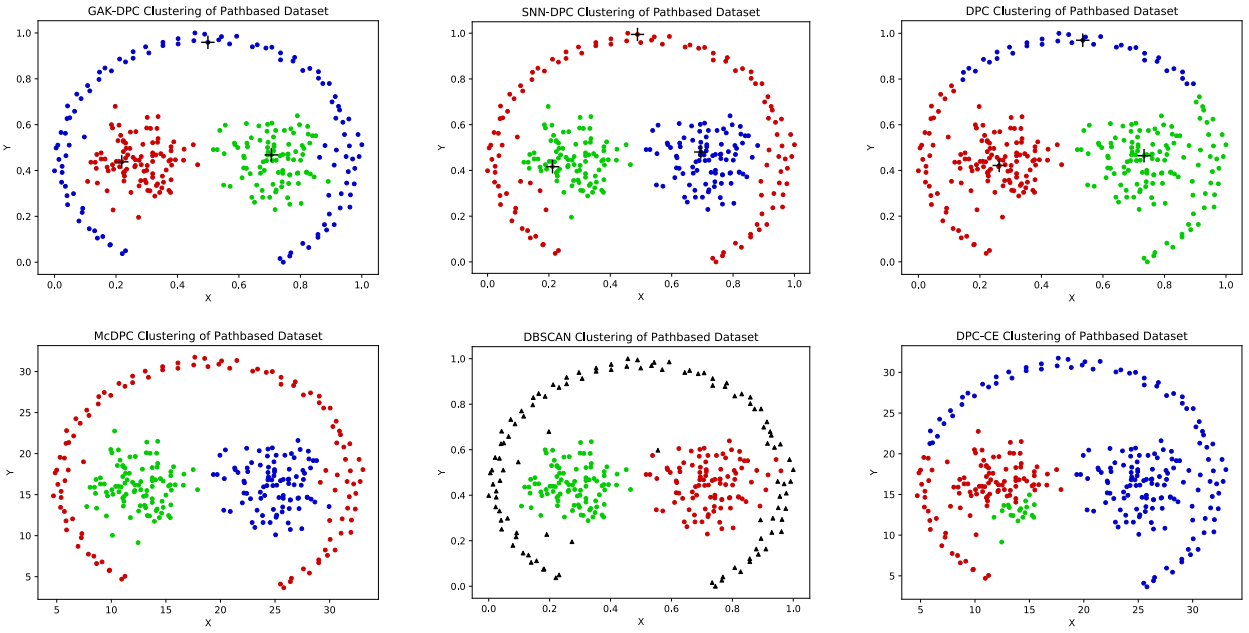

FIGURE 4. The clustering results on pathbased by 6 algorithms.

metrics in Figure 7 offers a more transparent and intuitive depiction of the overall clustering performance and versatility of GAK-DPC.

E. SENSITIVITY TESTS ON PARAMETERS

The algorithm in this paper involves two adjustable parameters, namely the number of nearest neighbors K_1 for calculating the geodetic distance and the number of nearest neighbors K_2 for assigning the remaining data points. In this section, the waveform dataset is used as an example for sensitivity testing of the algorithm parameters. The samples in this dataset usually have a certain degree of overlap in the feature space, i.e., samples of different categories may have similar feature distributions and the dataset is large in size. For DPC and some existing improved algorithms, the presence of overlapping samples when dealing with the waveform dataset increases the difficulty of clustering because the density of neighboring samples may be similar, which leads to

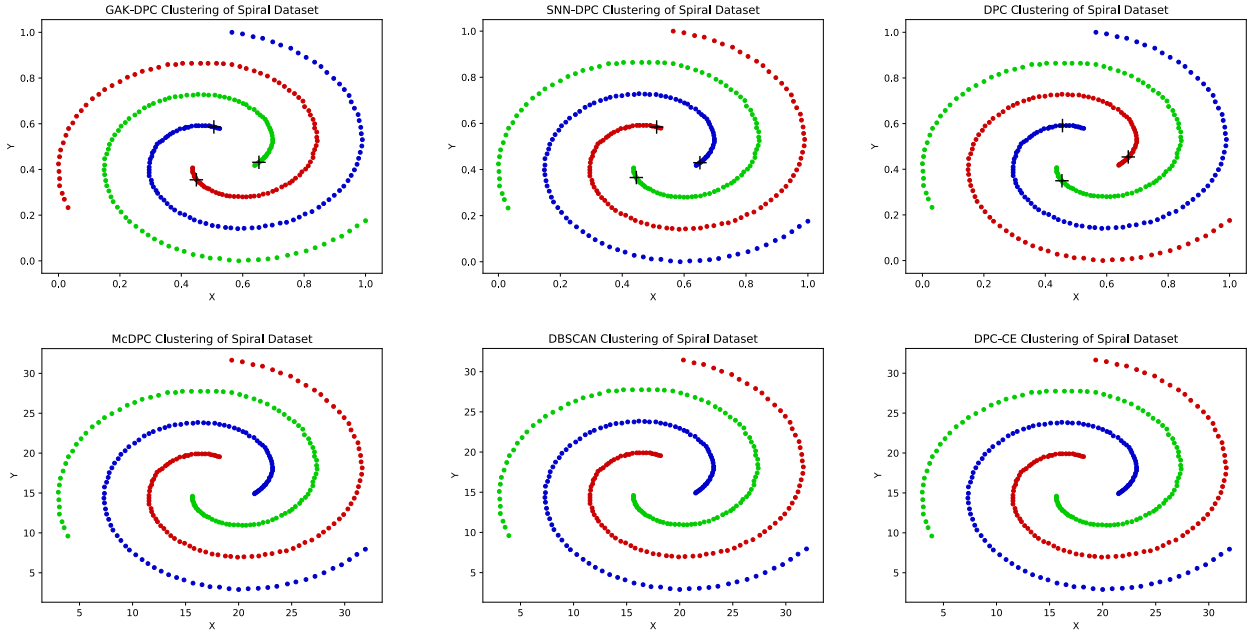

FIGURE 5. The clustering results on spiral by 6 algorithms.

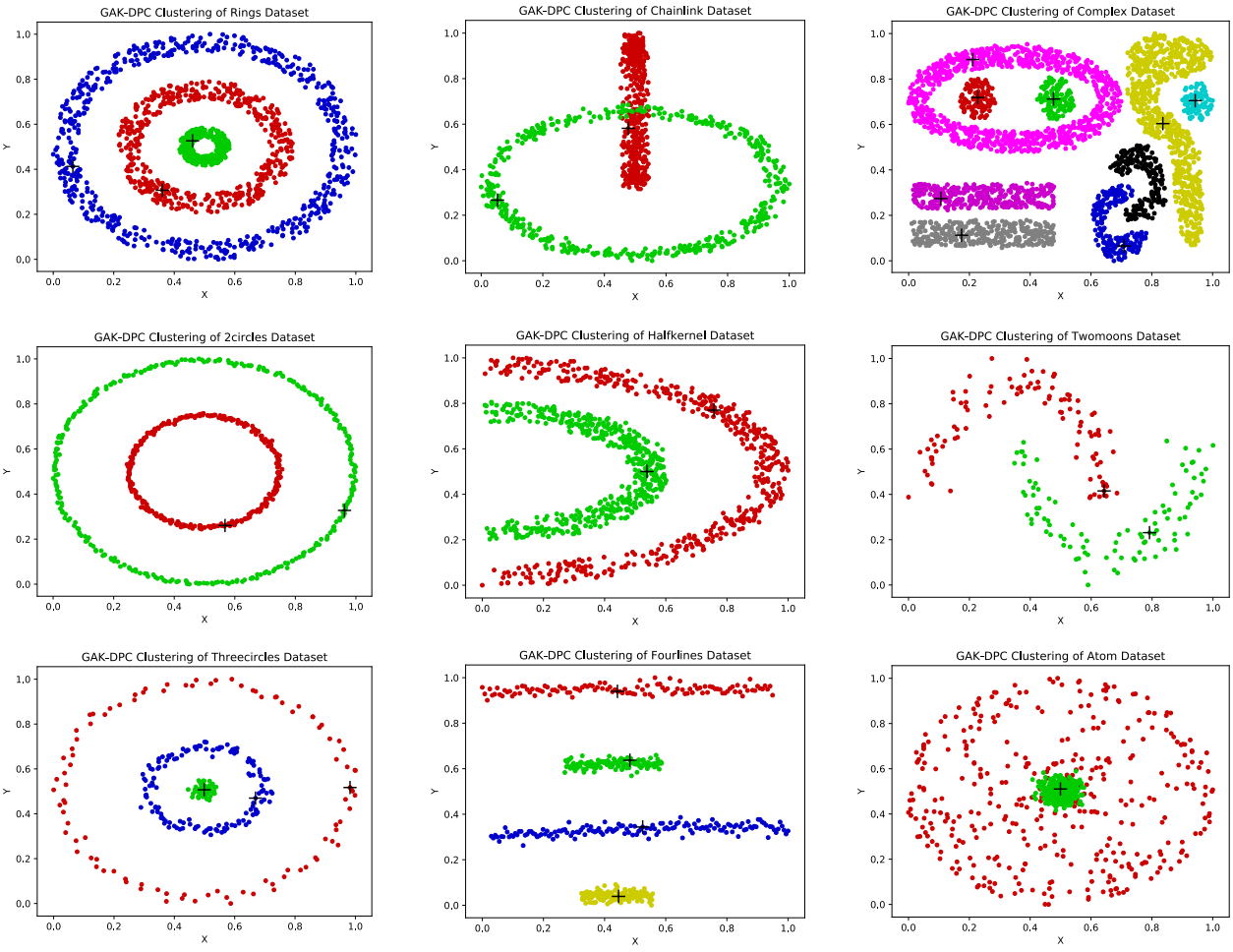

FIGURE 6. The clustering results of the remaining synthetic datasets.

TABLE 5. Comparison of clustering results on real-world datasets.

Data	Index	GAK-DPC	SNN-DPC	McDPC	DPC-CE	DPC	DBSCAN	K-Means
Iris	ARI	**0.9222**	**0.9222**	0.8858	0.7891	0.8857	0.6120	0.7163
	AMI	**0.9133**	0.9124	0.8689	0.8026	0.8606	0.5692	0.7331
	FMI	**0.9479**	**0.9479**	0.9234	0.8407	0.9233	0.7291	0.8112
Wine	ARI	**0.9134**	0.8992	0.3910	0.3715	0.6724	0.5292	0.8685
	AMI	**0.8810**	0.8735	0.4247	0.4172	0.7065	0.5484	0.8473
	FMI	**0.9425**	0.9330	0.6024	0.5834	0.7835	0.7121	0.9126
Seeds	ARI	**0.8361**	0.7890	0.7027	0.7687	0.7670	0.5291	0.7049
	AMI	**0.7910**	0.7509	0.6955	0.7826	0.7299	0.5302	0.6705
	FMI	**0.8903**	0.8589	0.8026	0.8184	0.8444	0.6711	0.8026
Segment	ARI	**0.5965**	0.5770	0.4135	0.2944	0.5891	0.4543	0.5049
	AMI	**0.7335**	0.6725	0.6056	0.4535	0.7143	0.4965	0.6102
	FMI	**0.6634**	0.6457	0.4939	0.4580	0.6600	0.5277	0.5758
Waveform	ARI	**0.4329**	0.4176	0.2956	0.2466	0.2698	0.0097	0.2536
	AMI	**0.4362**	0.3984	0.3512	0.2768	0.3261	0.0856	0.3630
	FMI	**0.6405**	0.6164	0.5416	0.5617	0.5292	0.4813	0.5037
Parkinsons	ARI	**0.3769**	0.2916	0.1497	0.0058	0.1256	0.0252	0.0520
	AMI	**0.2990**	0.1529	0.2130	0.0003	0.2478	0.0071	0.2129
	FMI	**0.8256**	0.8032	0.6458	0.7498	0.6187	0.5775	0.5957
Spect Heart	ARI	**0.2663**	0.1903	0.0468	0.0358	0.0273	0.1209	-0.0059
	AMI	**0.1399**	0.1117	0.1288	0.0556	0.0846	0.1113	0.0942
	FMI	**0.7329**	0.6819	0.4009	0.6848	0.5936	0.6336	0.5900

TABLE 6. ARI values of waveform dataset after clustering with different parameters.

K_1	3	4	5	6	7	8	9	10	11
ARI	0.2491	0.4224	0.1471	0.3394	0.4329	0.3114	0.3090	0.3090	0.3090
K_1	12	13	14	15	16	17	18	19	20
ARI	0.3087	0.2675	0.2568	0.2676	0.2676	0.2697	0.2697	0.2697	0.2697

K_2	2	3	4	5	6	7	8	9	10	11	12	13
ARI	0.1616	0.1615	0.3228	0.3711	0.4063	0.4329	0.4325	0.3032	0.2919	0.2463	0.2424	0.2399
K_2	14	15	16	17	18	19	20	21	22	23	24	25
ARI	0.1507	0.2090	0.2145	0.2170	0.2196	0.3939	0.4217	0.2071	0.4103	0.4075	0.3923	0.3767

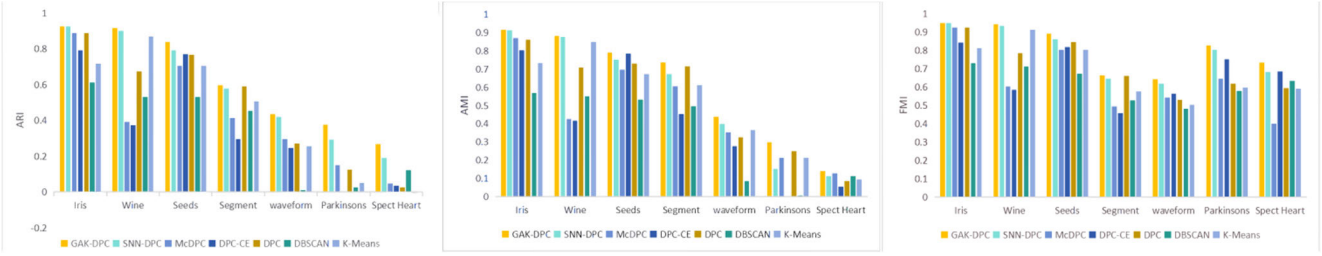

FIGURE 7. Cluster evaluation index comparison chart of GAK-DPC, SNN-DPC, McDPC, DPC-CE, DPC, DBSCAN, K-means algorithm on real-world datasets.

difficulties in accurately determining the center of clustering. In addition, when the amount of data is large, the performance of DPC may decrease and it is difficult to ensure the accuracy of the clustering results.

However, the algorithm proposed in this paper performs well in clustering the waveform dataset and has obvious advantages over other algorithms. Table 6 demonstrates the values of the Adjusted Rand Index (ARI) for the waveform dataset after clustering for different values of the parameters K_1 and K_2. The clustering effect is best when both K_1 and K_2 are taken as 7. Therefore, with K_2 set to 7, the test is conducted for the values of K_1 from 3 to 20 and the values of ARI

are recorded in the table under the variation of K_1 and its mean value is calculated as 0.2931 with a variance of 0.0040; while for K_2, tests are conducted for K_2 values from 2 to 25 with K_1 set to 7 and its mean value is calculated as 0.3013 with a variance of 0.0096. The calculated mean value can reflect the overall trend of clustering effect under different parameters, the mean value of ARI obtained by this paper's algorithm for clustering waveform dataset under different parameters is greater than the optimal value of ARI value for the majority of comparison algorithms in this paper's experiments. And the variance indicates the degree of dispersion of the data, which can help to assess the stability of the algorithm under different parameter settings. The variance of the above ARI data are smaller, which indicates that the performance of the algorithm under the change of the parameter value changes less and has a better stability.

In summary, the algorithm in this paper is robust to parameter selection and has some flexibility in parameter setting. It can be adjusted according to the characteristics of the dataset and the actual needs, while it is easy to find a relatively optimal combination of parameters.

V. CONCLUSION

To address the limitations of DPC, which has constraints related to distance measurement methods in data distribution, simplistic density definitions and a single allocation strategy prone to chaining issues, this paper introduces a new clustering algorithm called GAK-DPC. The proposed algorithm enhances traditional DPC in the following key aspects: Firstly, it replaces the traditional Euclidean distance in DPC with graph distance. This improvement allows for a more accurate description of the relationships between data points, particularly in non-Euclidean spaces. This approach finds more extensive applications and provides better adaptability to diverse data distributions. Additionally, it redefines the method for determining the local density of data points by using the K-nearest neighbor theory. Simultaneously, an adaptive K-value selection method is introduced. This method dynamically determines the K-value based on the dataset's characteristics. This adaptive K-value selection approach offers a more precise reflection of the proximity of data points and enhances the algorithm's adaptability and robustness. In this way, it provides a better consideration of data distribution diversity and the relationships among data points. Finally, to enhance the algorithm's fault tolerance, an optimization is performed on the secondary allocation strategy for sample points. Compared to the DPC's single allocation strategy, this approach significantly improves clustering accuracy and reduces the occurrence of misclassifications.

Across multiple synthetic and real-world datasets, this algorithm is compared to SNN-DPC, McDPC, DPC-CE, DPC, DBSCAN and K-Means. Clustering evaluation metrics and visualization of experimental results consistently shows that the proposed GAK-DPC outperforms the others on most datasets.

However, while the experimental results show advantages over other methods, the challenges of achieving a parameter-free algorithm and reducing algorithm complexity remain. Achieving these goals requires deeper research and methodological improvements to ensure that the algorithm performs well and is easy to apply in various scenarios. Furthermore, applying this algorithm to real-world power system data to address practical issues is an important future research direction. Such practical applications will further validate the utility of the algorithm and help solve real-world problems, which can provide more insights and opportunities for its application in the real world.

ACKNOWLEDGMENT

The authors would like to sincerely thank the anonymous reviewers and editors for their hard work and valuable suggestions. Their professional review has played a vital role in improving the quality of the paper.

REFERENCES

[1] F. T. Verleysen and A. Weeren, "Clustering by publication patterns of senior authors in the social sciences and humanities," *J. Informetrics*, vol. 10, no. 1, pp. 254–272, Feb. 2016, doi: 10.1016/j.joi.2016.01.004.

[2] G. Shanmugam, T. Thanarajan, S. Rajendran, and S. S. Murugaraj, "Student psychology based optimized routing algorithm for big data clustering in IoT with MapReduce framework," *J. Intell. Fuzzy Syst.*, vol. 44, no. 2, pp. 2051–2063, Jan. 2023, doi: 10.3233/jifs-221391.

[3] A. Gondeau, Z. Aouabed, M. Hijri, P. R. Peres-Neto, and V. Makarenkov, "Object weighting: A new clustering approach to deal with outliers and cluster overlap in computational biology," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 18, no. 2, pp. 633–643, Mar. 2021, doi: 10.1109/TCBB.2019.2921577.

[4] S. Yue, P. Wang, J. Wang, and T. Huang, "Extension of the gap statistics index to fuzzy clustering," *Soft Comput.*, vol. 17, no. 10, pp. 1833–1846, Oct. 2013, doi: 10.1007/s00500-013-1023-9.

[5] T. Yuan and W. Kuo, "A model-based clustering approach to the recognition of the spatial defect patterns produced during semiconductor fabrication," *IIE Trans.*, vol. 40, no. 2, pp. 93–101, Nov. 2007, doi: 10.1080/07408170701592556.

[6] H. Kim and J. Seo, "Cluster-based FAQ retrieval using latent term weights," *IEEE Intell. Syst.*, vol. 23, no. 2, pp. 58–65, Mar. 2008, doi: 10.1109/MIS.2008.23.

[7] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: An overview," *WIREs Data Mining Knowl. Discovery*, vol. 2, no. 1, pp. 86–97, Dec. 2011, doi: 10.1002/widm.53.

[8] J. Schneider and M. Vlachos, "Scalable density-based clustering with quality guarantees using random projections," *Data Mining Knowl. Discovery*, vol. 31, no. 4, pp. 972–1005, Mar. 2017, doi: 10.1007/s10618-017-0498-x.

[9] T. Chen, N. L. Zhang, T. Liu, K. M. Poon, and Y. Wang, "Model-based multidimensional clustering of categorical data," *Artif. Intell.*, vol. 176, no. 1, pp. 2246–2269, Jan. 2012, doi: 10.1016/j.artint.2011.09.003.

[10] V. Bureva, E. Sotirova, S. Popov, D. Mavrov, and V. Traneva, "Generalized net of cluster analysis process using STING: A statistical information grid approach to spatial data mining," in *Proc. 12th Int. Conf. Flexible Query Answering Syst.*, vol. 10333, 2017, pp. 239–248, doi: 10.1007/978-3-319-59692-1_21.

[11] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, Jun. 2014, doi: 10.1126/science.1242072.

[12] J. Xie, H. Gao, W. Xie, X. Liu, and P. W. Grant, "Robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors," *Inf. Sci.*, vol. 354, pp. 19–40, Aug. 2016, doi: 10.1016/j.ins.2016.03.011.

[13] M. Du, S. Ding, and H. Jia, "Study on density peaks clustering based on k-nearest neighbors and principal component analysis," *Knowl.-Based Syst.*, vol. 99, pp. 135–145, May 2016, doi: 10.1016/j.knosys.2016.02.001.

[14] R. Liu, H. Wang, and X. Yu, "Shared-nearest-neighbor-based clustering by fast search and find of density peaks," *Inf. Sci.*, vol. 450, pp. 200–226, Jun. 2018, doi: 10.1016/j.ins.2018.03.031.
[15] M. Du, S. Ding, X. Xu, and Y. Xue, "Density peaks clustering using geodesic distances," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 8, pp. 1335–1349, Aug. 2018, doi: 10.1007/s13042-017-0648-x.
[16] Y. Wang, D. Wang, Y. Zhou, X. Zhang, and C. Quek, "VDPC: Variational density peak clustering algorithm," *Inf. Sci.*, vol. 621, pp. 627–651, Apr. 2023, doi: 10.1016/j.ins.2022.11.091.
[17] Y. Wang, D. Wang, X. Zhang, W. Pang, C. Miao, A.-H. Tan, and Y. Zhou, "McDPC: Multi-center density peak clustering," *Neural Comput. Appl.*, vol. 32, no. 17, pp. 13465–13478, Feb. 2020, doi: 10.1007/s00521-020-04754-5.
[18] W. Guo, W. Wang, S. Zhao, Y. Niu, Z. Zhang, and X. Liu, "Density peak clustering with connectivity estimation," *Knowl.-Based Syst.*, vol. 243, May 2022, Art. no. 108501, doi: 10.1016/j.knosys.2022.108501.
[19] D. Cheng, S. Zhang, and J. Huang, "Dense members of local cores-based density peaks clustering algorithm," *Knowl.-Based Syst.*, vol. 193, Apr. 2020, Art. no. 105454, doi: 10.1016/j.knosys.2019.105454.
[20] Y. Wang, W. Pang, and J. Zhou, "An improved density peak clustering algorithm guided by pseudo labels," *Knowl.-Based Syst.*, vol. 252, Sep. 2022, Art. no. 109374, doi: 10.1016/j.knosys.2022.109374.
[21] Z. Wang, H. Wang, H. Du, S. Chen, and X. Shi, "A novel density peaks clustering algorithm for automatic selection of clustering centers based on K-nearest neighbors," *Math. Biosci. Eng.*, vol. 20, no. 7, pp. 11875–11894, May 2023, doi: 10.3934/mbe.2023528.
[22] V.-T. Vu, T. T. Q. Bui, T. L. Nguyen, D.-V. Tran, H.-Q. Do, V.-V. Vu, and S. M. Avdoshin, "Constrained density peak clustering," *Int. J. Data Warehousing Mining*, vol. 19, no. 1, pp. 1–19, Aug. 2023, doi: 10.4018/ijdwm.328776.
[23] X. Xu, H. Liao, and X. Yang, "An automatic density peaks clustering based on a density-distance clustering index," *AIMS Math.*, vol. 8, no. 12, pp. 28926–28950, Oct. 2023, doi: 10.3934/math.20231482.
[24] J. B. Tenenbaum, V. D. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000, doi: 10.1126/science.290.5500.2319.
[25] Q. Zhu, J. Feng, and J. Huang, "Natural neighbor: A self-adaptive neighborhood method without parameter K," *Pattern Recognit. Lett.*, vol. 80, pp. 30–36, Sep. 2016, doi: 10.1016/j.patrec.2016.05.007.
[26] D. Cheng, Q. Zhu, J. Huang, L. Yang, and Q. Wu, "Natural neighbor-based clustering algorithm with local representatives," *Knowl.-Based Syst.*, vol. 123, pp. 238–253, May 2017, doi: 10.1016/j.knosys.2017.02.027.
[27] L. Yang, Q. Zhu, J. Huang, and D. Cheng, "Adaptive edited natural neighbor algorithm," *Neurocomputing*, vol. 230, pp. 427–433, Mar. 2017, doi: 10.1016/j.neucom.2016.12.040.
[28] J. Huang, Q. Zhu, L. Yang, and J. Feng, "A non-parameter outlier detection algorithm based on natural neighbor," *Knowl.-Based Syst.*, vol. 92, pp. 71–77, Jan. 2016, doi: 10.1016/j.knosys.2015.10.014.
[29] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, Sep. 1975, doi: 10.1145/361002.361007.
[30] Z. Xinyuan and Y. Weiguo, "A density peak clustering algorithm with shared K-nearest neighbors and multi-allocation strategy," *Small Micro Comput. Syst.*, vol. 44, no. 1, pp. 75–82, 2023.
[31] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. KDD*, 1996, vol. 96, no. 34, pp. 226–231.
[32] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, vol. 1, 1967, pp. 281–297.
[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, and M. Blondel, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Jan. 2011.
[34] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, pp. 2837–2854, Oct. 2010.
[35] E. B. Fowlkes and C. L. Mallows, "A method for comparing two hierarchical clusterings," *J. Amer. Stat. Assoc.*, vol. 78, no. 383, pp. 553–569, Mar. 2012.

YUQIN SUN received the Doctor of Science degree in mathematics from Tongji University, in 2008. She is currently the Vice Dean of the School of Mathematics and Physics, Shanghai University of Electric Power, and a Professor with the School of Mathematics and Physics. Her research interests include combinatorial mathematics and graph theory, optimization theory energy power big data analysis, and scientific computing. Her main part-time jobs include the Executive Director of the Chinese Society of Educational Mathematics and a member of Shanghai Operational Research Society.

JINGCONG WANG was born in 1999. She is currently pursuing the M.S. degree with the School of Mathematics and Physics, Shanghai University of Electric Power, Shanghai, China. Her research interests include machine learning, clustering algorithms, and big data technologies.

YUAN SUN received the Doctor of Science degree in applied mathematics from the Department of Mathematics, Shanghai Jiao Tong University, in 2008. He is currently an Associate Professor with the School of Mathematics and Physics, Shanghai University of Electric Power. His research interests include machine learning, combinatorial design, and coding theory.

PENGCHENG ZHANG was born in 1999. He is currently pursuing the M.S. degree with the School of Mathematics and Physics, Shanghai University of Electric Power, Shanghai, China. His research interests include graph neural networks, optimization methods, and topology.

TIANYI WANG was born in 1998. He is currently pursuing the M.S. degree with the School of Mathematics and Physics, Shanghai University of Electric Power, Shanghai, China. His current research interests include machine learning, mathematics of computation, and algorithm optimization.