


## Research Article

# Density Peaks Clustering Based on Feature Reduction and Quasi-Monte Carlo

Zhihui Hu,<sup>1</sup> Xiaoran Wei,<sup>2</sup> Xiaoxu Han,<sup>3</sup> Guang Kou ,<sup>1</sup> Haoyu Zhang,<sup>1</sup> Xueyi Liu,<sup>4</sup> and Yefei Bai<sup>2</sup>

<sup>1</sup>Artificial Intelligence Research Center, Defense Innovation Institute, Beijing, China

<sup>2</sup>Ocean College, Zhejiang University, Zhoushan, China

<sup>3</sup>College of Intelligence and Computing, Tianjin University, Tianjin, China

<sup>4</sup>College of Science, China Jiliang University, Hangzhou, China

Correspondence should be addressed to Guang Kou; kg5188@163.com

Received 19 July 2021; Revised 3 November 2021; Accepted 7 December 2021; Published 6 January 2022

Academic Editor: Jiangbo Qian

Copyright © 2022 Zhihui Hu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Density peaks clustering (DPC) is a well-known density-based clustering algorithm that can deal with nonspherical clusters well. However, DPC has high computational complexity and space complexity in calculating local density  $\rho$  and distance  $\delta$ , which makes it suitable only for small-scale data sets. In addition, for clustering high-dimensional data, the performance of DPC still needs to be improved. High-dimensional data not only make the data distribution more complex but also lead to more computational overheads. To address the above issues, we propose an improved density peaks clustering algorithm, which combines feature reduction and data sampling strategy. Specifically, features of the high-dimensional data are automatically extracted by principal component analysis (PCA), auto-encoder (AE), and t-distributed stochastic neighbor embedding (t-SNE). Next, in order to reduce the computational overhead, we propose a novel data sampling method for the low-dimensional feature data. Firstly, the data distribution in the low-dimensional feature space is estimated by the Quasi-Monte Carlo (QMC) sequence with low-discrepancy characteristics. Then, the representative QMC points are selected according to their cell densities. Next, the selected QMC points are used to calculate  $\rho$  and  $\delta$  instead of the original data points. In general, the number of the selected QMC points is much smaller than that of the initial data set. Finally, a two-stage classification strategy based on the QMC points clustering results is proposed to classify the original data set. Compared with current works, our proposed algorithm can reduce the computational complexity from  $O(n^2)$  to  $O(Nn)$ , where  $N$  denotes the number of selected QMC points and  $n$  is the size of original data set, typically  $N \ll n$ . Experimental results demonstrate that the proposed algorithm can effectively reduce the computational overhead and improve the model performance.

## 1. Introduction

With the advent of the era of big data, the importance of data mining is increasingly prominent [1]. As an unsupervised learning method, clustering is widely used in many different fields including image processing, medicine, and archaeology. There are various classical clustering algorithms, such as K-means [2], DBSCAN [3], and AP [4]. According to different standards, clustering algorithms are classified into different categories. Generally speaking, clustering algorithms are divided into partition-based methods, hierarchy-

based methods, density-based methods, and grid-based methods.

In recent years, a new density peaks clustering (DPC) algorithm has been proposed [5]. It is a typical density-based clustering algorithm with excellent advantages. One advantage is that the DPC relies on the decision graph to select the clustering center. Specifically, DPC draws the decision graph of the data set by defining local density  $\rho$  and distance  $\delta$ . Then, DPC determines the cluster centers based on the decision graph. The obtained cluster centers have two characteristics: (1) The local density of the cluster centers is

large and the density of its neighborhood is not greater than itself. (2) The distance between the cluster centers and other data points with a higher density is relatively large. Hence, the cluster centers are data points with high local density and high distance, which are called density peaks. Another advantage is that DPC can not only deal with clusters of arbitrary shape but also does not need to determine the number of categories in advance.

Although DPC has achieved good performance in many situations, it still has some drawbacks. Firstly, DPC needs to calculate the local density and distance of each data point, which makes the computational complexity  $O(n^2)$ . The expensive computational overhead limits the application of DPC in large-scale data sets. To address this issue, the study in [6] proposed a distributed density peaks clustering algorithm (EDDPC). EDDPC aggregates large-scale data sets into MapReduce and integrates local results to approximate the final results. However, EDDPC is a distributed algorithm and is not suitable for single CPU scenarios. The study in [7] proposed a density-based and grid-based clustering algorithm (DGB). Instead of calculating distances between all data, only a smaller number of grid points are calculated. However, DGB is only suitable for dealing with high-dimensional data set. In general, the data distribution in high-dimensional space may be more complex and contain more noise. Although [8, 9] are proposed to filter the noise, additional operations increase the computational overhead.

To address the above problems, an improved density peaks clustering algorithm combining feature reduction and data sampling strategy is proposed in this paper. Firstly, the original data feature space is compressed by some classical feature reduction methods. Then, the low-dimensional feature data are sampled by super-uniformly Quasi-Monte Carlo sequence, and the selected high-density Quasi-Monte Carlo points are used to replace the original data points for clustering. Finally, we perform a two-stage strategy to determine the category for the original data. The proposed method has the following advantages:

- (1) The proposed algorithm reduces the computational complexity from  $O(n^2)$  to  $O(Nn)$ , where  $N$  and  $n$  represent the number of selected QMC points and the size of original data set, respectively. In general, there is  $N \ll n$
- (2) Through feature reduction, the proposed algorithm reduces the noise from the original data and decreases the complexity of high-dimensional feature space
- (3) Extensive experiments have demonstrated the effectiveness of our proposed algorithm in terms of computational overhead and model performance

## 2. Related Work

**2.1. Feature Reduction.** Feature reduction indicates mapping the data from the high-dimensional feature space to a low-dimensional space. The features of the high-dimensional data will be extracted by linear or nonlinear transformation.

Hence, efficient low-dimensional features of the original data set can be obtained by various feature reduction methods. An ideal low-dimensional feature should retain the classification information as much as possible and filter the noise.

Generally speaking, feature reduction can be divided into linear and nonlinear feature reduction methods. Principal component analysis (PCA) is a classical linear feature reduction method [10]. PCA transforms a group of variables that may correlate with linearly uncorrelated variables by orthogonal transformation. Auto-encoder (AE) and t-distributed stochastic neighbor embedding (t-SNE) are nonlinear feature reduction methods. AE can be regarded as a self-supervised manner that consists of the encoder and the decoder [11]. The input data will be mapped to the hidden layer by the encoder, while the decoder transforms the hidden layer features back to the input. Its goal is to combine some high-order features to reconstruct itself. The t-SNE is a machine learning method basing stochastic neighbor embedding (SNE) for feature reduction [12]. t-SNE maps high-dimensional data to two or more dimensions and alleviates the congestion problem in the process of feature reduction. All the above methods have been applied in many fields [13–15].

**2.2. Density Peaks Clustering.** Density peaks clustering (DPC) is proposed in [5], and it can efficiently deal with arbitrary shape data sets without specifying the cluster number  $k$  in advance. The cluster center selected by DPC has two characteristics: (1) the local density of the cluster center should be larger than the local density of its neighbors; (2) Data points with low local density should be far away from other data points with high local density. To describe these characteristics, DPC defines two concepts for each data point  $x_i$ : the local density  $\rho_i$  and the minimum distance  $\delta_i$ . The local density  $\rho_i$  is formulated as

$$\rho_i = \sum_j X(d_{ij} - d_c), \quad (1)$$

$$X(x) = \begin{cases} 1 & x < 0 \\ 0 & x \geq 0 \end{cases},$$

where  $d_{ij}$  represents the distance between  $x_i$  and  $x_j$ .  $d_c$  is the intercept, which is the only artificially defined parameter in DPC. In the code provided by [5],  $d_c$  is formulated as

$$d_c = d_{N_d} \times 2\%, \quad (2)$$

where  $d_{N_d}$  is the size of the distance matrix, which defines the distance between any data point pairs. When the data set is small, the Gaussian kernel function is used to calculate  $\rho_i$ .  $\rho_i$  is formulated as

$$\rho_i = \sum_j \exp\left(-\frac{d_{ij}^2}{d_c^2}\right), \quad (3)$$

In addition,  $\delta_i$  is formulated as

$$\delta_i = \begin{cases} \min_{j: p_j > p_i} d_{ij} & \exists j \text{ s.t. } p_j > p_i \\ \max_j : d_{ij} & \text{otherwise} \end{cases}. \quad (4)$$

DPC draws the decision graph based on  $\rho_i$  and  $\delta_i$ . Then, DPC selects the data points with both  $\rho_i$  and  $\delta_i$  as the cluster centers and assigns the remaining data points to the nearest class. DPC is a simple and efficient algorithm, and a series of works have been carried out [16–22]. However, DPC requires a huge computational overhead. The computational complexity of the DPC is  $O(n^2)$ , which makes it unsuitable for large-scale data set. To address this problem, a feasible strategy is to sample the data set [23]. Our work is based on the sampling strategy to reduce the computational overhead.

**2.3. Quasi-Monte Carlo.** As a statistical test method, the Monte Carlo method has been widely used in machine learning. The Quasi-Monte Carlo method is similar to the Monte Carlo method, but there are theoretical differences between them. The superiority of the Quasi-Monte Carlo method is to generate the deterministic super-uniformly distributed sequence (called low-discrepancy sequence in mathematics) instead of the pseudo-random sequence generated by the Monte Carlo method. The Quasi-Monte Carlo method has been widely used in the field of machine learning [24, 25]. Specifically, the study in [24] utilizes the Quasi-Monte Carlo method to reduce the computational overhead that occurs in the parameter optimization process of neural networks. The study in [25] generates the Quasi-Monte Carlo sequence to perform the feature map and obtains the low-rank features. Similarly, we generate Quasi-Monte Carlo sequence for data sampling. Next, we briefly describe the Quasi-Monte Carlo sequence.

The Quasi-Monte Carlo Random sequence is a deterministic super-uniformly distributed sequence with low deviation. It has the property that any long subsequences are uniformly distributed in the feature space. Recently, the most widely used Quasi-Monte Carlo Random sequence mainly includes Halton sequence [26], Faure sequence [27], and Niederreiter’s  $(t, s)$  sequence [28]. In our work, the Halton sequence is selected to perform the sampling strategy.

The Halton sequence is one of the standard low-discrepancy sequences, which is used to generate super-uniformly distributed random numbers. Compared with pseudo-random numbers generated by the Monte Carlo method, it is mathematically proved that the volatility of the Halton sequence is smaller. Specifically, the approximate error of the Halton sequence is determined by the degree of difference of the sequence  $\{x_1, \dots, x_N\}$ . The approximate error is formulated as the following equation:

$$\left| \frac{1}{N} \sum_{k=1}^N f(x_k) - \int_{I^s} f(x) dx \right| \leq V(f) D_N, \quad (5)$$

where  $|\frac{1}{N} \sum_{k=1}^N f(x_k) - \int_{I^s} f(x) dx|$  is the error term,  $V(f)$  is the Hardy-Krause variation of the function  $f$ , and  $D_N$  is the deviation of  $(x_1 \cdots x_N)$ .

Because the order of  $D_N$  is  $O((\log N)^{s-1}/N)$ , the approximate error order of the Quasi-Monte Carlo method is  $O(1/N)$ . Similarly, the error order of the pseudo-random sequence is  $O(1/\sqrt{N})$ . Compared with the above error orders, the error order of the Quasi-Monte Carlo method is smaller than that of the Monte Carlo method. Note that the above discussion only gives the upper limit of approximate error. In fact, the convergence rate of the Halton sequence is much faster than the rate obtained by the upper limit. Generally speaking, the Quasi-Monte Carlo method greatly speeds up the convergence compared with the Monte Carlo method, and the random numbers generated by the Quasi-Monte Carlo method are more uniform.

The Monte Carlo method generates the pseudo-random numbers, and the Quasi-Monte Carlo method generates the quasi-random numbers. Figure 1 shows the comparison between the quasi-random numbers and the pseudo-random numbers on a two-dimensional plane. As shown in Figure 1, the pseudo-random numbers are not uniformly distributed in some places. However, the Halton sequence is highly uniformly distributed in the whole space. Intuitively, the Quasi-Monte Carlo method may be more comprehensive, while the Monte Carlo method has more blank areas. Hence, this paper adopts the Halton sequence to sample the original data and further proposes a new density peaks clustering algorithm.

### 3. Description of the Algorithm

In this section, a novel improved density peaks clustering algorithm based on the Quasi-Monte Carlo method (QMC-DPC) is proposed to improve the performance of DPC. Specifically, the proposed method includes two components: the feature reduction module and the data sampling module.

**3.1. The Feature Reduction Module.** In this module, we aim to reduce the feature dimension of data sets. The original data set  $X \in R^{n \times m}$  will be transformed to  $X' \in R^{n \times d}$  by various feature reduction methods, where  $d \leq m$ . Our goal is to retain the original information as much as possible while reducing the dimension of the data.

In practice, we utilize linear and nonlinear feature reduction methods, including PCA, AE, and t-SNE, respectively. Firstly, we perform the zero-mean normalization for  $X$ . For  $x_1, x_2, \dots, x_n \in X$ , we calculate the mean  $\bar{x} = 1/n \sum_{i=1}^n x_i$  and the standard deviation  $s = \sqrt{1/n - 1 \sum_{i=1}^n (x_i - \bar{x})^2}$ . Hence, we can obtain the normalized numbers  $\tilde{x}_i = x_i - \bar{x}/s, i = 1, \dots, n$ . Then, PCA, AE, and t-SNE are implemented on the normalized data set  $\tilde{X}$ . For PCA, we choose the number of principal components that are smaller than the original dimension of the data set (except for the two-dimension data set). We keep the original dimension for the two-dimension data set. For AE, we set the AE with three layers, including an encoder, a decoder, and a hidden layer. The dimension of encoder and decoder is equivalent to  $d$  and the number of hidden layer units is equivalent to  $d$ . For the input data  $X$ , we select the

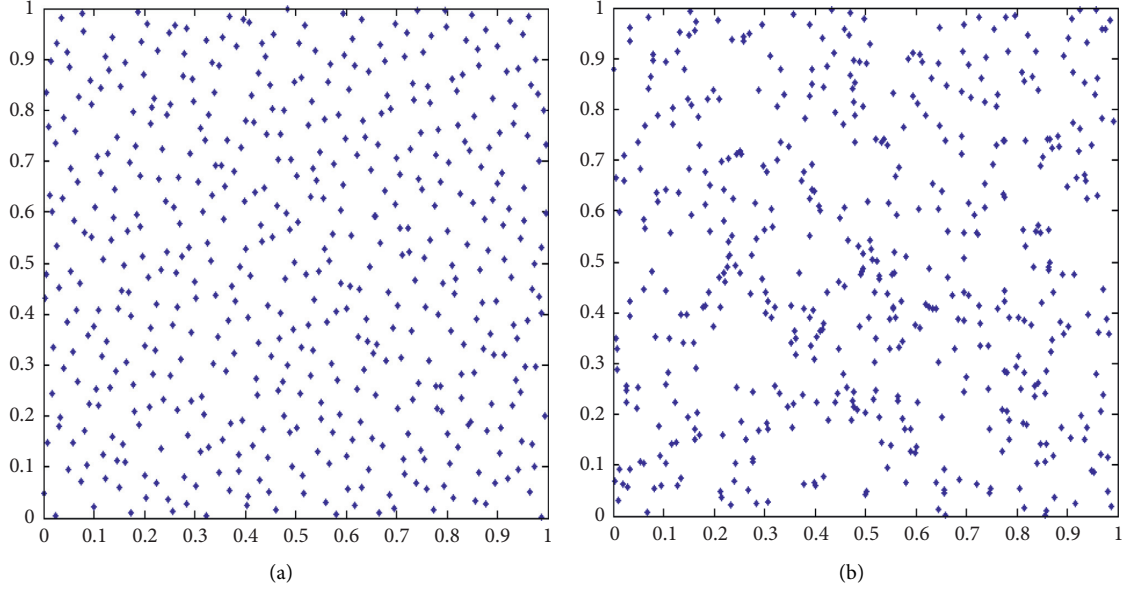


FIGURE 1: 500 random points of pseudo-random numbers and the quasi-random numbers. (a) Quasi-random numbers (Halton sequence). (b) Pseudo-random numbers.

hidden layer features as the  $X'$ . For t-SNE, the similarity between data points is measured by probability instead of Euclidean distance. Specifically, the similarity of data points in the original feature space is calculated by Gaussian joint probability, while the heavy-tailed student t-distribution is used in the low-dimension to measure the similarity. Then, we minimize the KL divergence to obtain the reduced features  $X'$ . Figure 2 shows the obtained two-dimensional features of PCA, AE, and t-SNE on Waveform and Landsat. The original dimensions of Waveform and Landsat are more than 20. From Figure 2, it can be seen that the low-dimensional features that map from higher-dimensional data are distinguishable. In Section 4, we will discuss how to select the feature reduction method by experimental analysis.

**3.2. The Data Sampling Module.** Although we compress the feature dimension of data sets through the feature reduction module, the computational complexity of the DPC is still  $O(n^2)$ . In this module, we aim to reduce the time overhead of DPC. Hence, an improved Density Peaks Clustering algorithm based on super-uniformly Quasi-Monte Carlo sequence (QMC-DPC) is proposed. In summary, we utilize the super-uniformly Quasi-Monte Carlo sequence to sample the low-dimensional feature space of the data set. Then, the representative Quasi-Monte Carlo points are used to calculate  $\delta_i$  and  $\rho_i$  instead of the original data. Generally speaking, the number of selected Quasi-Monte Carlo points is much smaller than the size of original data set  $n$ . The detailed description of QMC-DPC is given in the following.

Specifically, we first define two basic concepts as follows:

- (1) Circular data unit  $C$ : the circle with the Quasi-Monte Carlo points as the center and radius  $r$
- (2) Unit density  $C_{u,d}$ : the number of data points contained in circular data unit  $c$

Assume that  $X' = \{x'_1, x'_2, \dots, x'_n\}$ ,  $X' \in \mathbb{R}^{n \times d}$  is the low-dimensional feature data set obtained by the feature reduction module. We randomly generate  $N_0$  Quasi-Monte Carlo points in the feature space. With the Quasi-Monte Carlo points as the centers, the corresponding  $C$  are determined under the appropriate  $r$  (When  $d$  is small, the parameter  $r = d/2\sqrt{N_0}$  after experiments). Then, according to whether  $C$  contains data points or not, the circular data units are divided into two categories: non-empty unit set and empty unit set, where a nonempty unit set  $C_{ne} = \{C | C_{u,d} > 0\}$  and empty unit set  $C_e = \{C | C_{u,d} = 0\}$ . Next, since the empty unit set indicates that it does not contain any data, empty unit set and corresponding Quasi-Monte Carlo points are eliminated. The effect is shown in Figure 3.

As shown in Figure 3, the remaining nonempty Quasi-Monte Carlo points are distributed around the sample points, while the removed empty Quasi-Monte Carlo points are far from the sample points. Hence, the distribution of the original data set can be sampled by nonempty Quasi-Monte Carlo points. Furthermore, the local density of the original data set can be estimated by the unit density  $C_{u,d}$ . Therefore, it is reasonable to utilize the nonempty Quasi-Monte Carlo points to calculate local density  $\rho$  and minimum distance  $\delta$  instead of the original data points. Next, for all the nonempty Quasi-Monte Carlo points (assuming that the number of the nonempty Quasi-Monte Carlo points are  $N$ , there is generally  $N \ll N_0$ ), the distance of the nonempty Quasi-Monte Carlo point pairs is calculated to obtain the distance matrix  $A$ :

$$A = \begin{bmatrix} d_{11} & \cdots & d_{1N} \\ \vdots & \ddots & \vdots \\ d_{N1} & \cdots & d_{NN} \end{bmatrix} = \begin{bmatrix} 0 & \cdots & d_{1N} \\ \vdots & \ddots & \vdots \\ d_{N1} & \cdots & 0 \end{bmatrix}, \quad (6)$$

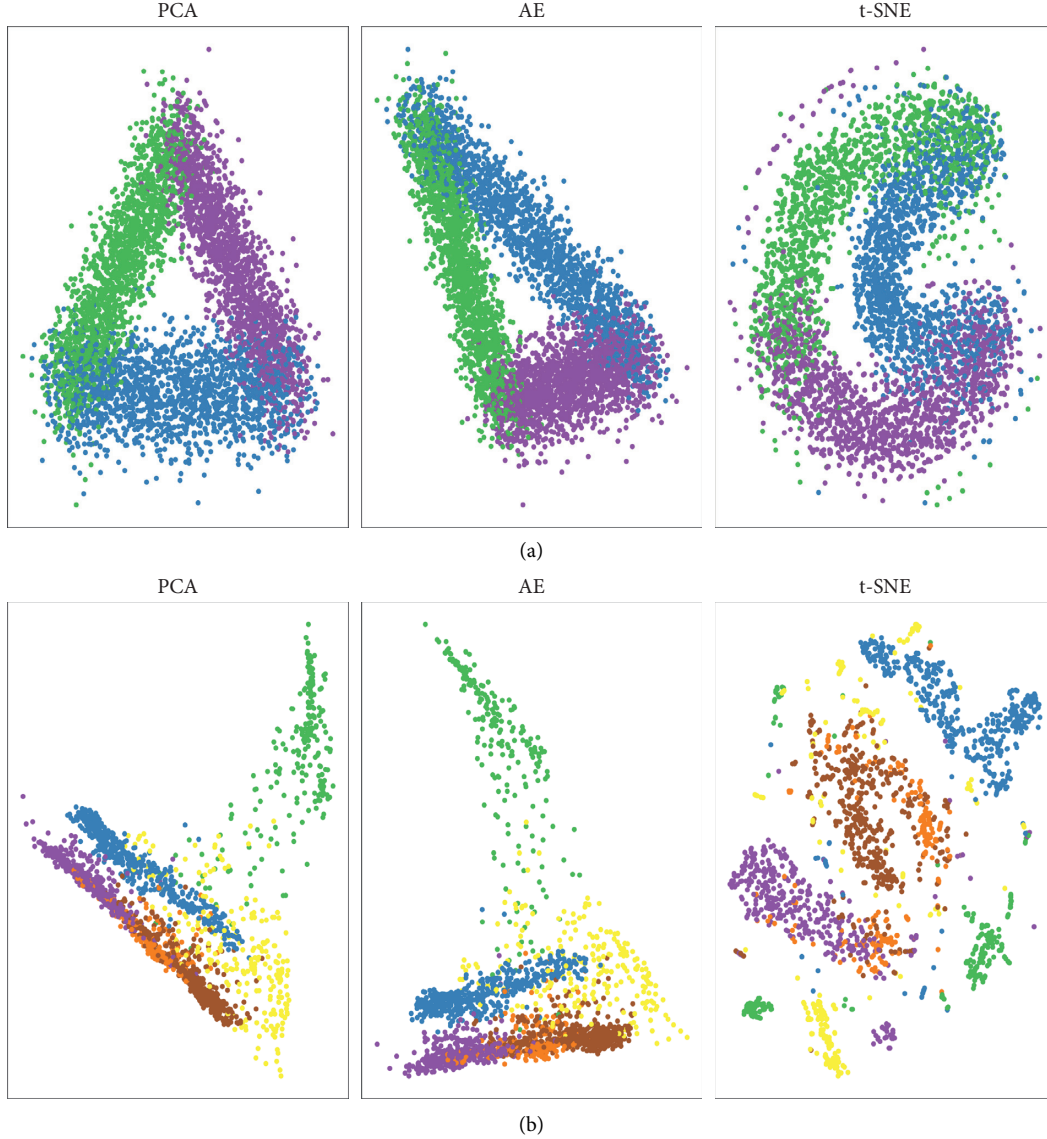


FIGURE 2: The obtained two-dimensional features of PCA, AE, and t-SNE. (a) The two-dimensional features of waveform. (b) The two-dimensional features of Landsat.

where  $A \in \mathbb{R}^{N \times N}$  is a symmetric matrix with diagonal elements that are zero.  $d_N$  is the ascending order of all elements in  $A$ . When  $N$  is too small, the  $d_c = d_N \times 2\%$  may be zero which indicates that the function of intercept  $d_c$  is eliminated. Hence, we remove the zero elements in  $A$  and take the first 2% distance from the remaining  $N^2 - N$  elements as the  $d_c$ . Then, we use equations (3) and (4) to calculate  $\rho_i$  and  $\delta_i$  of each nonempty Quasi-Monte Carlo point and draw the decision graph. Figure 4 shows the decision graph of QMC-DPC and DPC on Waveform.

As shown in Figure 4, the density peaks obtained by QMC-DPC are easier to distinguish than that of DPC, especially on the low-dimensional features generated by AE and t-SNE. Meanwhile, the number of data points in the decision graph of QMC-DPC is smaller than DPC. Specifically, QMC-DPC (PCA), QMC-DPC (AE), and

QMC-DPC (t-SNE), respectively, calculate 2742, 2499, and 2989 data points in the decision graph, while DPC calculates 5000 data points in the decision graph. The above discussion further proves the effectiveness of the Quasi-Monte Carlo sampling method. Specifically, it can be summarized as the following three aspects: (1) Combined with the super uniformity of the Quasi-Monte Carlo sequence, the data sampling is more comprehensive, so as to reduce the bias. This conclusion is described by Figure 1. (2) The number of selected nonempty Quasi-Monte Carlo points is small, which greatly reduces the time and space overhead. This conclusion is described in Figure 3. (3) Based on  $\delta$  and  $\rho$ , data points located in dense areas are difficult to distinguish, because their  $\delta$  and  $\rho$  are similar. On the contrary, Quasi-Monte Carlo points essentially sample the local density, and the distinction between selected nonempty Quasi-Monte

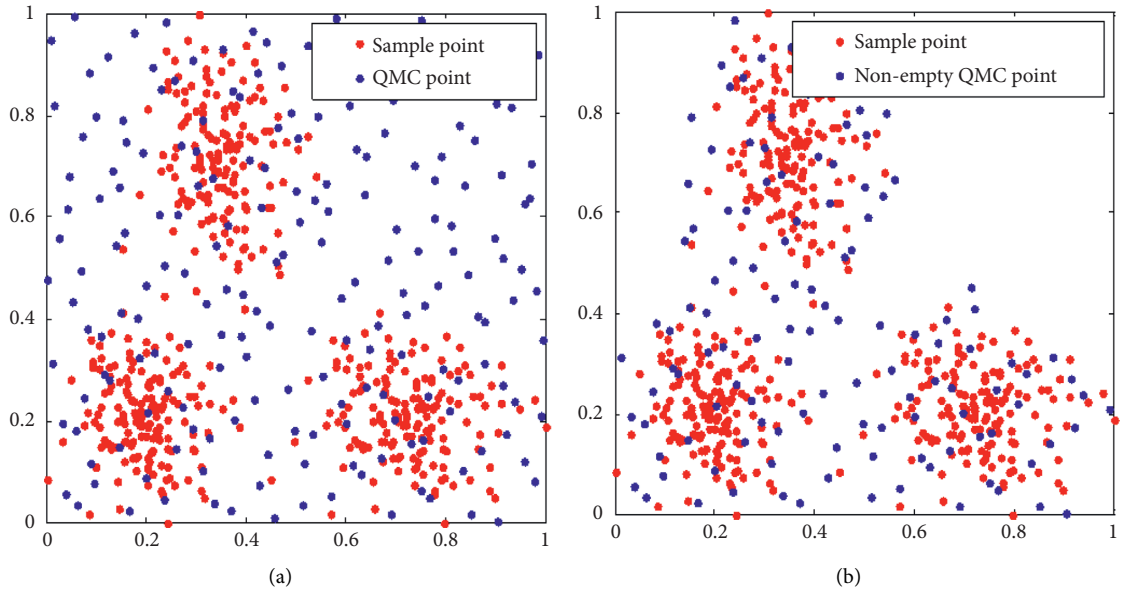


FIGURE 3: Sample points and circular data units. (a) Sample points and circular data units. (b) Sample points and nonempty circular units.

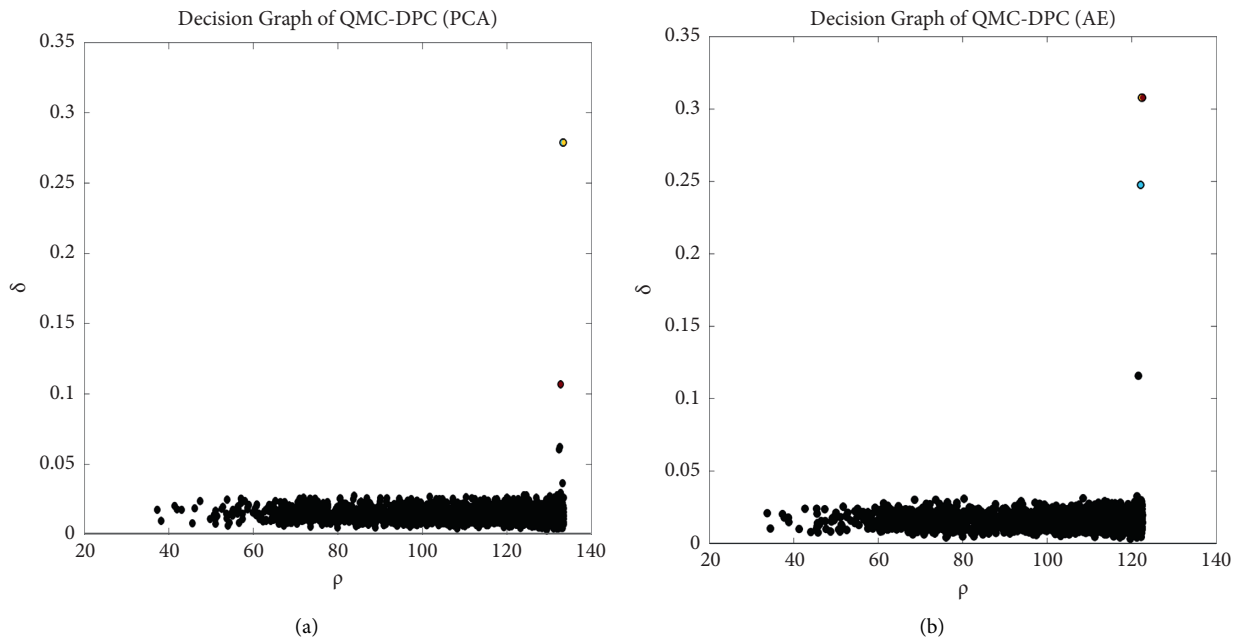


FIGURE 4: Continued.

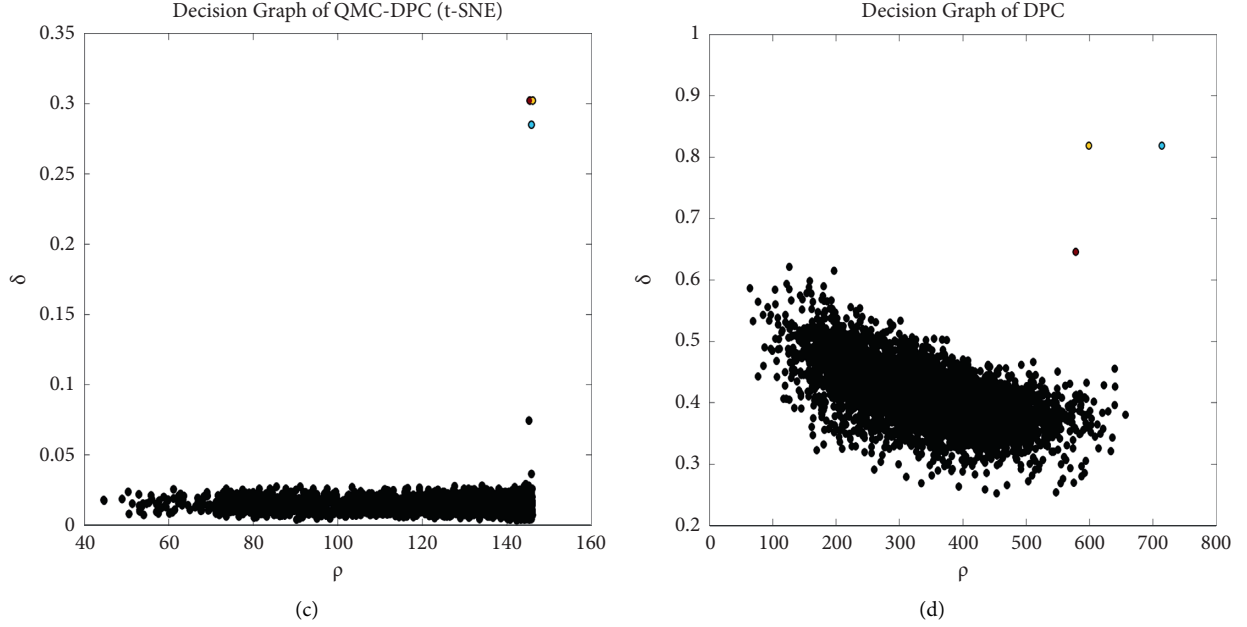


FIGURE 4: The decision graph on waveform. (a) QMC-DPC (PCA). (b) QMC-DPC (AE). (c) QMC-DPC (t-SNE). (d) DPC.

Carlo points is enlarged. Finally, according to the nearest distance principle, we propose a two-stage classification strategy:

- (i) The density peaks are selected as the class centers, and the remaining nonempty Quasi-Monte Carlo points are assigned to the nearest density peak. The first step obtains the clustering results of all nonempty Quasi-Monte Carlo points.
- (ii) The data points of  $X'$  are assigned to the nearest nonempty Quasi-Monte Carlo point. As the feature mapping is unique, the classification result of  $X$  is equivalent to the classification result of  $X'$ . The second step obtains the final clustering results of all data points of  $X$ .

After the above discussion, QMC-DPC is depicted in Algorithm 1 and the whole process is shown in Figure 5.

**3.3. Algorithm Complexity Analysis.** The key of DPC is to draw the decision graph based on  $\rho_i$  and  $\delta_i$ . Our work retains the idea of choosing cluster centers, but QMC-DPC only calculates  $\rho_i$  and  $\delta_i$  for  $N$  nonempty Quasi-Monte Carlo points after the screening, making the computational complexity far less than DPC.

For the data set  $X'$ , the DPC takes the space complexity of  $O(n^2)$  to store the distance matrix. The space complexity of QMC-DPC mainly includes:  $O(N_0)$  is required to generate Quasi-Monte Carlo points,  $O(N)$  is required to retain nonempty Quasi-Monte Carlo points, and  $O(N^2)$  is required to store the distance matrix of nonempty Quasi-Monte Carlo point pairs. Therefore, the spatial complexity of QMC-DPC is  $O(N_0 + N + N^2)$ . When  $n$  is large, there is  $O(N_0 + N + N^2) < O(n^2)$  in general. However, when  $n$  is relatively small, the space complexity of QMC-DPC

becomes larger due to generating  $N_0$  Quasi-Monte Carlo points.

When calculating  $\rho_i$  and  $\delta_i$ , DPC needs to calculate the distance matrix with the time complexity of  $O(n^2)$ . After selecting the cluster centers, the time complexity of classifying data points is also  $(n \times k)$ . Therefore, the time complexity of DPC algorithm is  $O(n^2) + nk$ . The time complexity of QMC-DPC mainly includes  $O(n \times N_0)$  to calculate the unit density of Quasi-Monte Carlo points,  $O(N^2)$  is required to calculate the  $\rho_i$  and  $\delta_i$  of nonempty Quasi-Monte Carlo points, and  $O(N \times k)$  is required to classify the nonempty Quasi-Monte Carlo points and  $O(n \times N)$  when classifying data points. Therefore, the time complexity of QMC-DPC algorithm is  $O(n \times N + N^2)$ . In general, there are always  $N \ll N_0$  and  $N_0 \ll n$ , making the time complexity of the QMC-DPC less than that of the DPC. However, when  $n$  is relatively small, the time cost of QMC-DPC is more than that of DPC. In the experiment, we will further prove that even with the addition of the feature reduction module, the proposed algorithm still has time superiority.

## 4. Experiment and Analysis

**4.1. Experimental Setup.** To verify the performance of QMC-DPC, the proposed method is compared with related clustering algorithms, including DPC-KNN-PCA [17], SNN-DPC [18], DLORE-DP [16], DPC [5], AP [4], DBSCAN [3], and K-means [2]. The nearest neighbor number is set to 4 in SNN-DPC. The ratio of low-density points in DLORE-DP is set to 0.2. For DBSCAN, the parameter Min Pts is set to 3 and  $\epsilon$  is empty. K-means needs to specify the number of classes in advance. The data sets adopted in this section include two major categories: unlabeled data sets and labeled data sets. The details of these

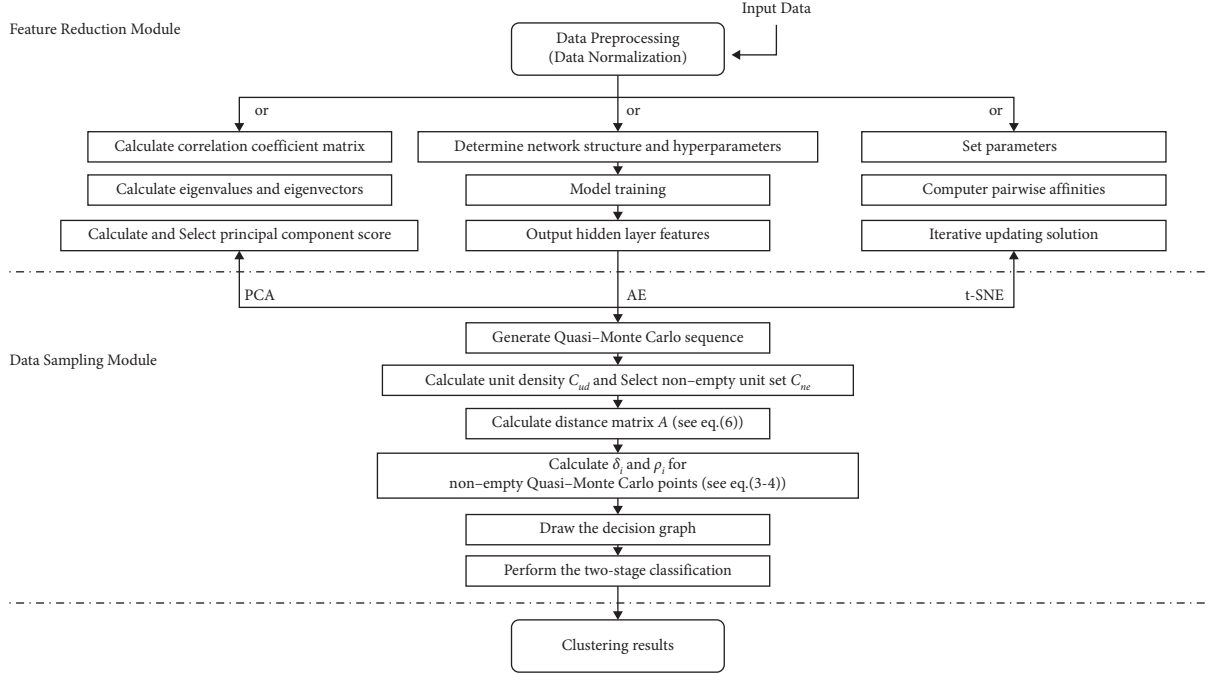


FIGURE 5: The work flow of the overall QMC-DPC.

**Input:**The Data set:  $X = \{x_1, x_2, \dots, x_n\}$ **Output:**Clustering results  $Y$ **Steps:**

- (1) Perform feature reduction on  $X$  to obtain low-dimensional feature data  $X'$ ;
- (2) Generate  $N_0$  Quasi-Monte Carlo points and determine circular data unit  $C$  on  $X'$ ;
- (3) Count the density for each circular data unit  $C$  and generate  $C_{ne}$  and  $C_e$ ;
- (4) Calculate the matrix  $A$  based on  $C_{ne}$  and remove the zero elements in  $A$ . Sort the remaining elements and determine the intercept  $d_c$ ;
- (5) Calculate the  $\delta_i$  and  $\rho_i$  for each nonempty Quasi-Monte Carlo points by equations (3) and (4);
- (6) Draw the decision graph to select cluster centers and determine the number of  $k$ ;
- (7) According to the principle of nearest distance, assign the remaining nonempty Quasi-Monte Carlo Points;
- (8) Assign the data points to the class of the nearest nonempty Quasi-Monte Carlo Points;
- (9) Return the clustering results  $Y$ .

ALGORITHM 1: QMC-DPC.

data sets are listed in Table 1. In labeled data sets, all data sets are UCI data sets. In unlabeled data sets, Flame, Aggregation, and S2 are Synthetic data sets. KDD is a biological data set, which is used to verify the superiority of our proposed algorithm on large-scale and high-dimensional feature data sets.

Four evaluation criteria are adopted to evaluate the model performance on labeled data sets, i.e, the Accuracy (Acc) and F-measure (F), Normalized mutual information (NMI), and Adjusted rand index (ARI). These evaluation criteria are described as follows: Assume that  $X = \{x_1, x_2, \dots, x_n\}$  is the data set.  $Y = \{y_1, y_2, \dots, y_n\}$  and  $\tilde{Y} = \{\tilde{y}_1, \tilde{y}_2, \dots, \tilde{y}_n\}$  represent the real labels and the predicted labels, respectively. Acc is denoted as

$$\text{Acc} = \sum_{i=1}^n \frac{\delta(y_i, \text{map}(\tilde{y}_i))}{n}, \quad (7)$$

where  $\text{map}(\cdot)$  is a permutation mapping function, which uses Hungarian algorithm to match the predicted labels with the real labels.

The F-measure is a harmonic mean of precision (prec) and recall (rec). Prec is the ratio between the number of correct positive results and the number of all positive results returned by the classifier. Rec is the ratio between the number of correct positive results and the number of all data that should have been identified as positive.  $U_i$  is the set of the number of all data that should have been classified as positive.  $V_j$  is the set of the number of all positive results



TABLE 1: The unlabeled data sets and labeled data sets.

Data set	Samples	Attributes	Categories	Source
<i>Unlabeled data sets</i>				
Flame	240	2	3	[29]
Aggregation	788	2	7	[31]
S2	5000	2	15	[32]
KDD	145 751	74	2000	[33]
<i>Labeled data sets</i>				
Iris	150	4	3	[30]
Wine	178	13	3	[30]
Segment	2310	18	7	[30]
Waveform	5000	21	3	[30]
Wpbc	198	33	2	[30]
Breast	277	9	2	[30]
Landsat	2000	36	9	[30]
Pima	768	8	2	[30]
Zoo	101	16	7	[30]

identified by the classifier. Prec, Rec, and F-measure are defined by the following equations:

$$\text{prec}(U_i, V_j) = \frac{|U_i \cap V_j|}{|V_j|}, \quad (8)$$

$$\text{rec}(U_i, V_j) = \frac{|U_i \cap V_j|}{|U_i|},$$

$$F(U_i, V_j) = \frac{(\beta^2 + 1)\text{prec} \cdot \text{rec}}{\beta^2\text{prec} + \text{rec}}, \quad (9)$$

where  $\beta$  is a nonnegative real number that is set to 1. For the  $U_i$  divided by each real label, the nearest one in  $V_j$  is selected as its  $F$  value:

$$F(U_i) = \max_{1 \leq j \leq C} F(U_i, V_j). \quad (10)$$

Then, we use the weighted average of  $F(U_i)$  to get the final  $F$  value:

$$F = \sum_{i=1}^R \omega_i \cdot F(U_i), \omega_i = \frac{|U_i|}{n}. \quad (11)$$

The Normalized Mutual Information (NMI) measures the information that the predicted labels  $\tilde{Y}$  share with the ground truth  $Y$ . NMI is defined as the following equation:

$$\text{NMI} = \frac{I(\tilde{Y}, Y)}{\sqrt{H(\tilde{Y})H(Y)}}, \quad (12)$$

where  $I(\tilde{Y}, Y)$  is the mutual information between clustering result and ground truth.  $H(\tilde{Y})$  and  $H(Y)$  denote the entropy of clustering result and ground truth, respectively.

The Adjusted Rand Index (ARI) is the extension of Rand Index (RI). ARI is defined as the following equation:

$$\text{ARI} = \frac{\text{RI} - E[\text{RI}]}{\max(\text{RI}) - E[\text{RI}]}, \quad (13)$$

where  $\text{RI} = (uv + \overline{u\overline{v}})/(uv + \overline{u\overline{v}} + \overline{u}v + u\overline{v})$ ,  $uv$  denotes the data pairs which are in the same class in  $U$  and in the same class in  $V$ ,  $\overline{u\overline{v}}$  denotes the data pairs which are in different classes in  $U$  and in different classes in  $V$ .  $\overline{u}v$  denotes the data pairs which are in different classes in  $U$  and in the same class in  $V$ .  $u\overline{v}$  denotes the data pairs which are in the same class in  $U$  and in different classes in  $V$ . The value of ARI is in the range  $[-1, 1]$ . The upper bound of these evaluation criterions is 1. The larger these criterions are, the better the clustering results are.

In the feature reduction module, some parameters are set in advance. For t-SNE, the learning rate is 500, the number of perplexity is 30, and the number of epochs is 800. For AE, the learning rate is 0.01, optimizer is Adam, and the number of epochs is 300.

**4.2. Experimental Results on Labeled Data Sets.** In this section, 9 UCI data sets in Table 1 are used to verify the performance of QMC-DPC. All data are normalized to between  $[0, 1]$ . To avoid extreme cases, each algorithm runs 10 times and records the average results. The values of evaluation criteria are shown in Table 2 and the best values are highlighted in bold. The relevant parameters of the QMC-DPC are recorded in Table 3.

As shown in Table 2, our proposed algorithm is superior to other algorithms on the whole. Acc indicates the ratio of the number of correct predicted samples to the number of total samples. In terms of Acc, QMC-DPC achieves the highest performance on all data sets except Waveform and Landsat. In particular, QMC-DPC is 33.6% and 34.3% higher than DPC on Zoo and Pima, respectively. F-measure indicates the matching degree between the predicted labels and the true labels of the data set, which is the weighted harmonic mean of precision and recall. In terms of F-measure, QMC-DPC achieves the highest performance on nearly half the data set. NMI quantifies the similarity between the predicted labels and the true labels, which measures the robustness of the algorithm. In terms of NMI, QMC-DPC achieves the highest performance on all data sets except Landsat, Pima, and Zoo. In particular, QMC-DPC is

TABLE 2: The comparison of all algorithms in terms of Acc, F, NMI, and ARI on UCI data sets.

Algorithm	Acc	F	NMI	ARI	Acc	F	NMI	ARI	Acc	F	NMI	ARI
	Iris				Wine				Segment			
QMC-DPC (PCA)	0.860	0.760	0.677	0.742	0.528	0.648	0.523	0.345	<b>0.681</b>	<b>0.611</b>	<b>0.751</b>	0.539
QMC-DPC (AE)	0.883	0.785	<b>0.787</b>	<b>0.771</b>	0.516	0.611	<b>0.588</b>	0.406	0.616	0.578	0.612	<b>0.562</b>
QMC-DPC (t-SNE)	<b>0.906</b>	0.795	0.730	0.762	<b>0.588</b>	<b>0.661</b>	0.517	<b>0.671</b>	0.610	0.525	0.529	0.439
DPC-KNN-PCA	0.046	0.626	0.620	0.467	0.418	0.562	0.479	0.234	0.643	0.377	0.482	0.539
DLORE-DP	0.790	<b>0.798</b>	0.777	0.744	0.537	0.586	0.578	0.178	0.631	0.475	0.549	0.540
SNN-DPC	0.600	0.740	0.689	0.557	0.442	0.568	0.526	-0.005	0.617	0.374	0.016	0.429
DPC	0.886	0.785	0.783	0.719	0.465	0.610	0.486	0.353	0.617	0.534	0.701	0.539
DBSCAN	0.680	0.775	0.733	0.568	0.331	0.577	0.009	-0.002	0.442	0.377	0.431	0.227
AP	0.807	0.769	0.618	0.376	0.428	0.460	0.526	0.268	0.670	0.328	0.586	0.502
K-means	0.667	0.699	0.593	0.514	0.361	0.589	0.478	0.384	0.643	0.572	0.625	0.493
	Wpbc				Waveform				Breast			
QMC-DPC (PCA)	0.715	0.699	0.036	<b>0.042</b>	0.522	0.498	0.207	0.155	0.472	0.644	<b>0.073</b>	0.017
QMC-DPC (AE)	<b>0.779</b>	0.573	<b>0.053</b>	0.022	0.495	0.513	0.368	0.263	0.505	0.639	0.062	0.017
QMC-DPC (t-SNE)	0.681	0.672	0.027	0.022	0.472	<b>0.630</b>	<b>0.431</b>	<b>0.437</b>	<b>0.523</b>	0.654	0.068	0.014
DPC-KNN-PCA	0.762	<b>0.797</b>	0.015	0.012	0.506	0.561	0.260	0.247	0.285	<b>0.764</b>	0.027	<b>0.020</b>
DLORE-DP	0.673	0.650	0.020	-0.061	<b>0.595</b>	0.587	0.382	0.333	0.303	0.741	0.052	0.004
SNN-DPC	0.752	0.786	0.019	-0.013	0.375	0.530	0.068	0.016	0.213	0.685	0.071	0.016
DPC	0.642	0.691	0.015	-0.007	0.222	0.546	0.217	0.216	0.516	0.541	0.000	0.004
DBSCAN	0.752	0.786	0.009	-0.013	0.331	0.577	0.000	0.025	0.353	0.438	0.059	0.015
AP	0.415	0.221	0.025	0.005	0.213	0.105	0.231	0.016	0.086	0.178	0.059	0.012
K-means	0.550	0.567	0.029	0.003	0.314	0.508	0.374	0.259	0.491	0.538	0.041	0.003
	Landsat				Pima				Zoo			
QMC-DPC (PCA)	0.630	0.551	0.492	0.421	<b>0.690</b>	0.735	0.061	<b>0.124</b>	0.693	0.792	0.811	<b>0.808</b>
QMC-DPC (AE)	0.618	0.503	0.539	0.382	0.634	<b>0.758</b>	0.004	0.022	0.684	0.789	0.796	0.796
QMC-DPC (t-SNE)	0.612	0.517	0.458	0.370	0.390	0.687	0.007	0.027	<b>0.712</b>	0.813	0.822	0.803
DPC-KNN-PCA	0.617	0.456	0.381	0.141	0.347	0.738	0.004	0.002	0.589	0.786	0.689	0.731
DLORE-DP	0.635	0.546	0.540	0.358	0.352	0.709	0.002	0.011	0.029	0.739	0.690	0.644
SNN-DPC	0.517	0.463	0.267	0.089	0.651	0.736	0.000	0.001	0.049	<b>0.856</b>	<b>0.831</b>	0.713
DPC	<b>0.636</b>	0.573	0.547	0.411	0.347	0.738	0.004	0.002	0.376	0.352	0.348	0.076
DBSCAN	0.506	0.430	0.000	0.000	0.343	0.733	0.002	0.005	0.693	0.792	0.761	0.691
AP	0.104	0.327	0.504	0.187	0.010	0.159	<b>0.079</b>	0.017	0.267	0.758	0.792	0.573
K-means	0.565	<b>0.647</b>	<b>0.548</b>	<b>0.469</b>	0.622	0.559	0.028	0.054	0.674	0.712	0.743	0.633

The bold values mean the best results.

TABLE 3: Parameter settings on all data sets.

Data sets	Iris	Wine	Segment	Wpbc	Waveform	Breast	Landsat	Pima	Zoo	KDD	Flame	Aggregation	S2
Feature dimension	2	2	2	2	4	2	4	2	2	3	2	2	2
$r$	0.35	0.55	0.45	0.35	0.45	0.25	0.40	0.20	0.45	0.35	0.40	0.40	0.35
$N_0$	200	200	2000	1500	2500	700	2000	1700	180	5000	400	1000	2500

21.4% higher than DPC on Waveform. ARI is used to measure the degree of coincidence of the two data distributions. In terms of ARI, QMC-DPC achieves the highest performance on all data sets except Breast and Landsat. The ARI value of QMC-DPC is 73.2% higher than DPC on Zoo. In addition, the evaluation criterion values of QMC-DPC (PCA), QMC-DPC (AE), and QMC-DPC (t-SNE) are similar, and the model performance is better than that of DPC on the whole. The above results indicate that the combination of the feature reduction module and the feature sampling module can improve the model performance.

**4.3. Experimental Results of Unlabeled Data Sets.** Since there are no real labels for the unlabeled data sets, the evaluation criteria Acc, F-measure, NMI, and ARI cannot be applied to

the unlabeled data sets. To compare the performance on the unlabeled data sets, the evaluation criteria Silhouette Coefficient (SC) and Calinski-Harabasz (CH) are defined. For SC, we first calculate the silhouette coefficient for each data point  $i$ :

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \quad (14)$$

where  $a(i)$  is average dissimilarity between data point  $i$  and other data points in the same class,  $b(i)$  is the minimum value of the average dissimilarity between data point  $i$  and other categories. Next, we obtain the silhouette coefficient for data set based on  $s(i)$ :

$$SC = \frac{1}{n} \sum_{i=1}^n s(i), \quad (15)$$

where  $n$  is the number of all data points. The value of SC is in the range  $[-1, 1]$ . The larger the SC value is, the better the clustering result is.

CH is defined as follows:

$$\text{CH} = \frac{\text{trace } B / (k - 1)}{\text{trace } W / (k - n)}, \quad (16)$$

where  $\text{trace } B = \sum_{i=1}^n n_i \|u_i - u\|^2$ ,  $n_i$  is the number of data points in class  $i$ ,  $u_i$  is the average of data points in class  $i$ , and  $u$  is the average of all data points.  $\text{trace } W = \sum_{j=1}^k \sum_{i=1}^n \|x_i - u_j\|^2$ ,  $k$  is the cluster numbers. The larger the CH value is, the better the clustering result is.

In this section, three synthetic data sets and KDD are selected to verify the performance of QMC-DPC. Flame, Aggregation, and S2 are the classical synthetic data sets. KDD is a large-scale data set with high-dimensional features. Table 4 shows the SC and CH of all algorithms on unlabeled data sets. The best values are highlighted in bold. The relevant parameters of the QMC-DPC are recorded in Table 3.

As shown in Table 4, our proposed method obtains the best clustering results on the whole, especially QMC-DPC (AE). The “—” in Table 4 indicates that the algorithm cannot execute because it exceeds the virtual memory. For the SC, QMC-DPC (t-SNE) is higher than DPC on Flame. And, DPC obtains the same results as our proposed method on Aggregation and S2. DPC-KNN-PCA also obtains the same results as our proposed method on S2. In general, QMC-DPC (AE) and QMC-DPC (t-SNE) achieve the better performance than QMC-DPC (PCA) except KDD. Limited by the t-SNE method, QMC-DPC (t-SNE) fails to perform clustering on KDD. In Section 4.6, we will make further comprehensive analysis. In addition, we visualized the classification results on synthetic data sets. Figure 6 shows the classification results on Aggregation and S2.

**4.4. Experimental Results of Running Time.** In this subsection, we further verify that our proposed method can effectively reduce the computational overhead. We select data sets with more than 2000 data points and record the running time in Table 5.

As shown in Table 5, compared with DPC, SNN-DPC, and AP, QMC-DPC achieves the best performance in terms of running time. QMC-DPC is at least 34.47%, 61.80%, 25.59%, and 50.85% lower than DPC on Segment, Waveform, Landsat, and s2, respectively. Generally speaking, the larger the data size, the more the time saved. For KDD, QMC-DPC (PCA) and QMC-DPC (AE) obtain the results, while QMC-DPC (t-SNE) will exceed memory. This is limited by the t-SNE method. In addition, DPC, SNN-DPC, and AP also exceed memory. This further confirms the effectiveness of our method. How to select QMC-DPC (PCA), QMC-DPC (AE), and QMC-DPC (t-SNE) will be discussed in Section 4.6. In addition, it can be seen that the running time of the QMC-DPC (PCA) and QMC-DPC (AE) is close. However, the computational overhead of QMC-DPC (t-SNE) is higher than that of QMC-DPC (PCA) and QMC-DPC (AE). The reason is that t-SNE requires a huge computational overhead, while auto-encoder only has a

shallow structure and does not contain a large number of training parameters. Furthermore, we compare the time complexity of our proposed method with that of the baselines methods. The results are recorded in Table 6. In this part, we set the number of data points to  $n$ , the number of cluster categories to  $k$ , the number of neighbors to  $m$ , the number of iteration to  $t$ , and the number of selected Quasi-Monte Carlo points to  $N$ . Although the time complexity of QMC-DPC is square,  $N$  is much smaller than  $n$  in practice. Hence, the time overhead of QMC-DPC will be significantly reduced and the conclusion can also be proved in Table 5.

**4.5. Experimental Results of Sensitivity Analysis.** In this section, we conduct parameter sensitivity analysis from multiple aspects, such as how feature dimensions affect model performance and running time. Specifically, we first calculate Acc, F, NMI, and ARI on UCI data sets where the feature dimension is in the range [16, 24]. The final results are recorded in Tables 7–10, respectively.

From Tables 7 to 10, it can be seen that the performance of the model will decrease slightly with the increase of dimension on the whole. This is limited by the loss of information caused by the sampling strategy as the dimension increases. As the dimensions increase, data distribution will become more complex. To address this issue, there are two methods to reduce the information loss caused by sampling: (1) increase the number of Quasi-Monte Carlo points and (2) appropriately increase the radius  $r$  of the circular data unit. If we adopt the first method, the time complexity of QMC-DPC in generating and storing Quasi-Monte Carlo points is  $O(N_0)$ , which increases the time and space overhead as the number of Quasi-Monte Carlo points increases. If the second method is adopted, the selection of radius  $r$  is very important. When  $r$  is too large and contains the entire data set, the QMC-DPC does not perform sampling operation. Since the main purpose of this paper is to reduce the time overhead of DPC, we give priority to the second method.

In addition, we further study the impact of feature dimension on model performance and running time, where the feature dimension is extended to [2, 9]. In this part, KDD is selected and the results are shown in Figure 7. As shown in Figure 7, QMC-DPC (AE) and QMC-DPC (PCA) achieve high performance in terms of SC. On the contrary, QMC-DPC (AE) and QMC-DPC (PCA) have a poor value in terms of CH when the feature dimension is 7. However, the value of CH increases heavily when the feature dimension is 9. The reason is that when we generate more Quasi-Monte Carlo points to execute sampling strategy, the corresponding running time also increases to a great extent. The relevant parameters on KDD are recorded in Table 11.

**4.6. Algorithm Summary.** Based on the above experiments, we have a comprehensive discussion on QMC-DPC. Specifically, as shown in Tables 2 and 4, it can be found that QMC-DPC achieves the best performance on the whole. On the UCI data sets, QMC-DPC (PCA), QMC-DPC (AE), and QMC-DPC (t-SNE) obtain the highest values of 8, 7, and 9

TABLE 4: The comparison of all algorithms in terms of SC and CH on unlabeled data sets.

Algorithm	Flame		Aggregation		S2		KDD	
	SC	CH	SC	CH	SC	CH	SC	CH
QMC-DPC (PCA)	0.379	153.672	0.461	1032.876	0.452	15 011.749	0.941 5	37 569.031 7
QMC-DPC (AE)	0.303	129.028	<b>0.493</b>	<b>1202.934</b>	<b>0.710</b>	<b>19 744.353</b>	<b>0.944 9</b>	<b>45 484.132 6</b>
QMC-DPC (t-SNE)	<b>0.409</b>	190.938	<b>0.493</b>	<b>1202.934</b>	0.704	19 149.117	—	—
DPC-KNN-PCA	0.403	178.217	0.281	329.073	<b>0.710</b>	<b>19 744.353</b>	—	—
DLORE-DP	0.323	108.244	0.463	883.713	0.611	17 689.904	—	—
SNN-DPC	0.132	37.009	0.363	851.953	0.002	1124.993	—	—
DPC	0.347	133.615	<b>0.493</b>	<b>1202.934</b>	<b>0.710</b>	<b>19 744.353</b>	—	—
DBSCAN	0.295	6.894	0.318	606.285	0.529	3178.635	0.229	18 701.700
AP	0.334	<b>247.017</b>	0.421	660.422	0.496	17 747.519	—	—
K-means	0.245	80.406	0.425	967.394	0.539	7293.533	0.694	13 042.686

The bold values mean the best results.

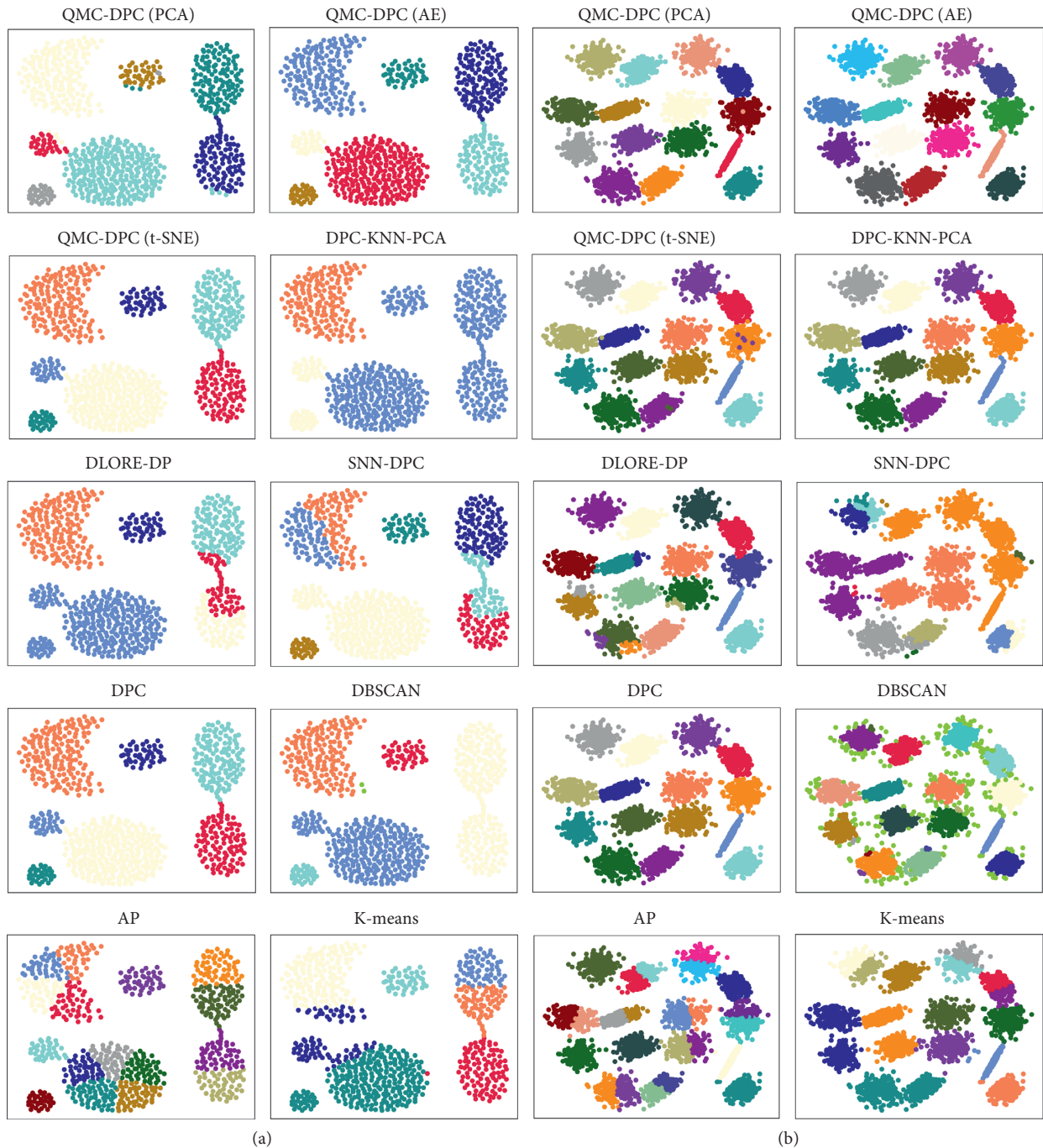


FIGURE 6: The visualization of classification results on aggregation (a) and S2 (b).

TABLE 5: The comparison of running time on Segment, Waveform, Landsat, and S2 (unit: seconds).

Algorithm	Data sets				
	Segment	Waveform	Landsat	S2	KDD
QMC-DPC (PCA)	17.737 4	<b>75.160 1</b>	<b>28.773 8</b>	81.083 1	<b>341.018 2</b>
QMC-DPC (AE)	<b>17.563 4</b>	82.874 1	30.359 8	<b>80.144 5</b>	368.837 1
QMC-DPC (t-SNE)	26.369 4	115.627 1	43.406 8	124.259 4	—
DPC	40.241 1	302.698 3	58.336 1	252.817 5	—
SNN-DPC	279.264 7	1305.837 4	208.851 8	1208.926 8	—
AP	261.218 3	903.003 6	144.370 3	819.570 3	—

The bold values mean the best results.

TABLE 6: The comparison of time complexity.

Algorithm	Time complexity
QMC-DPC	$O(n \times N + N^2)$
DPC-KNN-PCA	$O(n^2)$
DLORE-DP	$O(n \log n)$
SNN-DPC	$O((m + k)n^2)$
DPC	$O(n^2)$
DBSCAN	$O(n \log n) \sim O(n^2)$
AP	$O(n^3)$
K-means	$O(nkt)$

TABLE 7: The comparison of Acc on UCI data sets.

Data sets	Dimension	Algorithm		
		QMC-DPC (PCA)	QMC-DPC (AE)	QMC-DPC (t-SNE)
Iris	2	0.860	0.883	0.906
	3	0.860	0.813	0.913
	4	0.843	0.839	0.829
Wine	2	0.528	0.516	0.588
	3	0.562	0.556	0.589
	4	0.562	0.556	0.519
Segment	2	0.681	0.616	0.610
	3	0.544	0.602	0.519
	4	0.614	0.615	0.631
Wpbc	2	0.715	0.779	0.681
	3	0.684	0.515	0.541
	4	0.723	0.530	0.589
Waveform	2	0.565	0.561	0.479
	3	0.386	0.333	0.380
	4	0.522	0.495	0.472
Breast	2	0.472	0.505	0.523
	3	0.523	0.469	0.610
	4	0.527	0.568	0.419
Landsat	2	0.613	0.696	0.601
	3	0.580	0.624	0.582
	4	0.630	0.618	0.612
Pima	2	0.690	0.634	0.390
	3	0.653	0.554	0.652
	4	0.635	0.677	0.638
Zoo	2	0.693	0.684	0.712
	3	0.701	0.653	0.704
	4	0.687	0.621	0.695

times, respectively. On the unlabeled data sets, QMC-DPC (PCA), QMC-DPC (AE), and QMC-DPC (t-SNE) obtain the highest values of 0, 6, and 3 times, respectively. Obviously,

QMC-DPC combined with nonlinear feature reduction methods achieves better performance, on the whole. In terms of running time, it is obvious that our proposed

TABLE 8: The comparison of F on UCI data sets.

Data sets	Dimension	Algorithm		
		QMC-DPC (PCA)	QMC-DPC (AE)	QMC-DPC (t-SNE)
Iris	2	0.760	0.785	0.895
	3	0.783	0.709	0.947
	4	0.756	0.742	0.595
Wine	2	0.648	0.611	0.661
	3	0.614	0.610	0.500
	4	0.587	0.645	0.578
Segment	2	0.611	0.578	0.525
	3	0.423	0.301	0.412
	4	0.347	0.323	0.356
Wpbc	2	0.699	0.573	0.672
	3	0.632	0.564	0.576
	4	0.733	0.562	0.569
Waveform	2	0.595	0.534	0.479
	3	0.515	0.504	0.523
	4	0.498	0.513	0.630
Breast	2	0.644	0.639	0.654
	3	0.540	0.546	0.639
	4	0.539	0.578	0.598
Landsat	2	0.529	0.611	0.543
	3	0.475	0.591	0.553
	4	0.551	0.503	0.517
Pima	2	0.735	0.758	0.687
	3	0.712	0.711	0.626
	4	0.587	0.713	0.599
Zoo	2	0.792	0.789	0.813
	3	0.801	0.774	0.763
	4	0.765	0.752	0.698

TABLE 9: The comparison of NMI on UCI data sets.

Data sets	Dimension	Algorithm		
		QMC-DPC (PCA)	QMC-DPC (AE)	QMC-DPC (t-SNE)
Iris	2	0.677	0.787	0.771
	3	0.783	0.709	0.947
	4	0.709	0.703	0.362
Wine	2	0.523	0.558	0.517
	3	0.437	0.435	0.248
	4	0.377	0.470	0.384
Segment	2	0.751	0.612	0.529
	3	0.321	0.296	0.468
	4	0.013	0.324	0.343
Wpbc	2	0.036	0.053	0.027
	3	0.018	0.012	0.023
	4	0.733	0.562	0.569
Waveform	2	0.426	0.396	0.323
	3	0.515	0.504	0.523
	4	0.207	0.368	0.431
Breast	2	0.073	0.062	0.068
	3	0.049	0.018	0.058
	4	0.021	0.018	0.039
Landsat	2	0.531	0.590	0.555
	3	0.430	0.598	0.605
	4	0.492	0.539	0.458
Pima	2	0.061	0.004	0.007
	3	0.002	0.001	0.002
	4	0.035	0.040	0.005
Zoo	2	0.811	0.796	0.822
	3	0.784	0.735	0.786
	4	0.762	0.713	0.722

TABLE 10: The comparison of ARI on UCI data sets.

Data sets	Dimension	Algorithm		
		QMC-DPC (PCA)	QMC-DPC (AE)	QMC-DPC (t-SNE)
Iris	2	0.742	0.771	0.762
	3	0.670	0.529	0.922
	4	0.560	0.563	0.266
Wine	2	0.345	0.406	0.671
	3	0.318	0.292	0.149
	4	0.288	0.380	0.353
Segment	2	0.539	0.562	0.439
	3	0.103	0.129	0.295
	4	0.002	0.129	0.197
Wpbc	2	0.042	0.022	0.022
	3	-0.040	-0.008	0.028
	4	0.030	-0.000	-0.018
Waveform	2	0.300	0.301	0.336
	3	0.267	0.255	0.273
	4	0.155	0.263	0.437
Breast	2	0.073	0.062	0.068
	3	-0.001	-0.004	-0.003
	4	-0.000	-0.001	-0.022
Landsat	2	0.428	0.528	0.451
	3	0.285	0.488	0.461
	4	0.421	0.382	0.370
Pima	2	0.124	0.022	0.027
	3	0.010	0.008	0.064
	4	0.076	0.059	0.024
Zoo	2	0.808	0.796	0.803
	3	0.774	0.722	0.785
	4	0.765	0.684	0.752

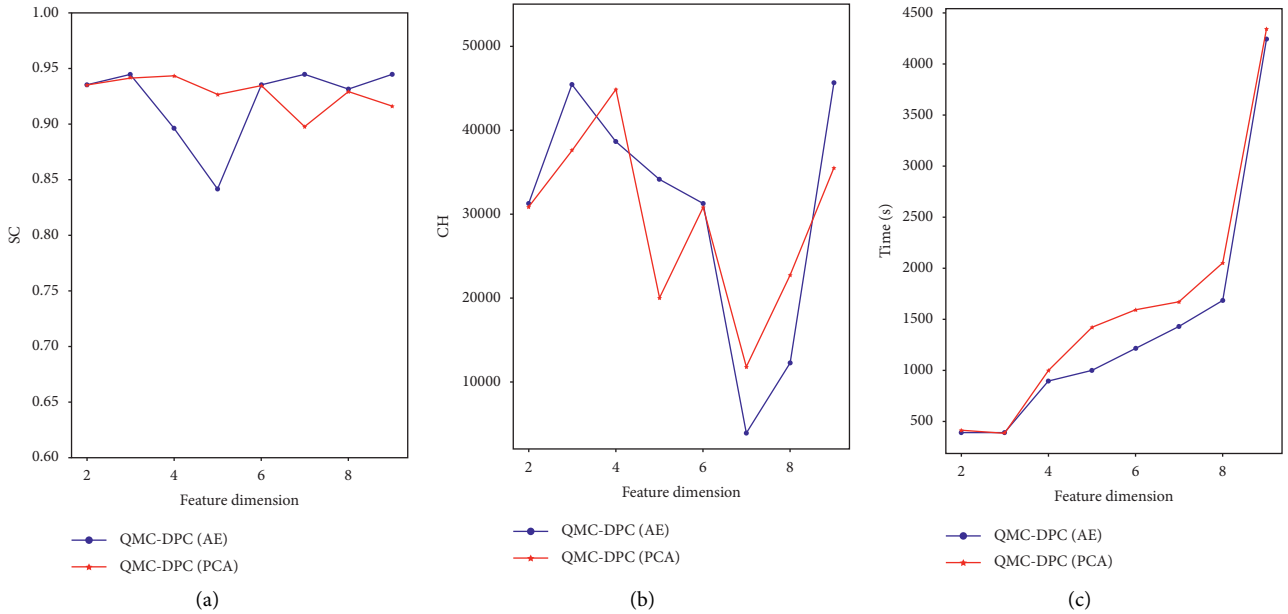


FIGURE 7: Clustering results of KDD under different feature dimensions. (a) The value of SC on KDD. (b) The value of CH on KDD. (c) The value of time on KDD.

method has superior performance. Especially when dealing with a large-scale data set, such as KDD, QMC-DPC achieves good performance, while most other baselines cannot

be executed due to being out of memory. This further verifies the effectiveness of our method. From Section 4.5, we can find that the feature dimension has an impact on the model

TABLE 11: Parameter settings on KDD.

Feature dimension	2	3	4	5	6	7	8	9
$r$	0.35	0.55	0.55	0.65	0.55	0.55	0.65	0.65
$N_0$	3000	3000	5000	6000	8800	9000	12000	25000

performance and various evaluation criteria are affected differently. In general, the model performance will decrease as the feature dimension increases. This is due to the loss of information caused by sampling. In Section 4.5, we propose two methods to overcome this problem, including generating more Quasi-Monte Carlo points and increasing the radius  $r$ . The purpose of both methods is to expand the sampling area. For our proposed method, we make a trade-off between running time and model performance, which generates fewer Quasi-Monte Carlo points and sets fewer iterations for t-SNE and AE. The above operations will reduce the running time while reducing the model performance. In particular, we also increase the radius  $r$  to reduce the information loss.

We summarize the following views on QMC-DPC:

- (i) In general, we choose QMC-DPC combined with nonlinear feature reduction methods, such as QMC-DPC (AE) and QMC-DPC (t-SNE). When dealing with a large-scale data set, we prefer QMC-DPC (AE).
- (ii) To reduce information loss, we give priority to expanding the radius  $r$ . Secondly, we consider adding Quasi-Monte Carlo points.

In addition, there are still exploration directions for our proposed algorithm in the future, which are summarized as follows:

- (i) How to select feature dimensions is a heuristic work. In future work, we hope to build a multi-layer auto-encoder and construct the loss function based on hidden layer features. We aim to design the automatic encoder as a multi-tasks neural network.
- (ii) We hope to propose a more comprehensive sampling method to reduce the loss of information. We can take the sample point itself as the center for sampling, and then filter out the data samples in the sparse area. Finally, we need to have a strategy for the classification of outliers.
- (iii) We hope to propose a more comprehensive sampling method to reduce the loss of information. We can take the sample point itself as the center for sampling, and then filter out the data samples in the sparse area. Finally, we need to have a strategy for the classification of outliers.

## 5. Conclusion

In this paper, a new density peaks clustering algorithm with high computational efficiency is proposed. The original feature space is compressed by different feature reduction methods. We sample the reduced feature space based on the super-uniformly distributed sequence generated by the

Quasi-Monte Carlo method. Our work can effectively overcome the high computation overhead of DPC while improving the model performance. Theoretically, the time complexity can be reduced from  $o(n^2)$  to  $o(Nn)$ , where  $N \ll n$ . The experimental results show that QMC-DPC improves the model performance of the DPC while greatly reducing the time overhead with the increase of data set size.

## Data Availability

The data used to support the findings of this study were supplied by <https://archive.ics.uci.edu/ml/index.php>.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This work was supported by the Major Research Project of National Natural Science Foundation of China (No. 91948303).

## References

- [1] M. Gupta and P. Chandra, "A comprehensive survey of data mining," *International Journal of Information Technology*, vol. 12, pp. 1243–1257, 2020.
- [2] J. Macqueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, California, Berkeley, January 1967.
- [3] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *KDD*, vol. 96, pp. 226–231, 1996.
- [4] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science (New York, N.Y.)*, vol. 315, no. 5814, pp. 972–976, 2007.
- [5] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.
- [6] S. Gong and Y. Zhang, "EDDPC: An efficient distributed density peaks clustering algorithm," *Journal of Computer Research and Development*, vol. 53, no. 6, pp. 1400–1409, 2016.
- [7] Bo Wu and B. M. Wilamowski, "A fast density and grid based clustering method for data with arbitrary shapes and noise," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 1620–1628, 2016.
- [8] J. Jiang, Y. Chen, X. Meng, L. Wang, and K. Li, "A novel density peaks clustering algorithm based on k nearest neighbors for improving assignment process," *Physica A: Statistical Mechanics and Its Applications*, vol. 523, pp. 702–713, 2019.
- [9] J. Xie, H. Gao, W. Xie, X. Liu, P. W. Grant, and Grant, "Robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors," *Information Sciences*, vol. 354, pp. 19–40, 2016.
- [10] M. Jafarzadegan, F. Safi-Esfahani, and Z. Beheshti, "Combining hierarchical clustering approaches using the PCA method," *Expert Systems with Applications*, vol. 137, pp. 1–10, 2019.
- [11] E. G. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.



- [12] L. Van der Maaten and G. Hinton, "Visualizing Data Using T-SNE," *Journal of Machine Learning Research*, vol. 9, no. 11, 2008.
- [13] J. K. Chow, Z. Su, J. Wu, P. S. Tan, X. Mao, and Y. H. Wang, "Anomaly detection of defects on concrete structures with the convolutional autoencoder," *Advanced Engineering Informatics*, vol. 45, Article ID 101105, 2020.
- [14] B. Melit Devassy and S. George, "Dimensionality reduction and visualisation of hyperspectral ink data using t-SNE," *Forensic Science International*, vol. 311, Article ID 110194, 2020.
- [15] J. Warmenhoven, N. Bargary, D. Liebl et al., "PCA of waveforms and functional PCA: a primer for biomechanics," *Journal of Biomechanics*, vol. 116, Article ID 110106, 2021.
- [16] D. Cheng, S. Zhang, and J. Huang, "Dense members of local cores-based density peaks clustering algorithm," *Knowledge-Based Systems*, vol. 193, Article ID 105454, 2020.
- [17] M. Du, S. Ding, and H. Jia, "Study on density peaks clustering based on k-nearest neighbors and principal component analysis," *Knowledge-Based Systems*, vol. 99, pp. 135–145, 2016.
- [18] R. Liu, H. Wang, and X. Yu, "Shared-nearest-neighbor-based clustering by fast search and find of density peaks," *Information Sciences*, vol. 450, pp. 200–226, 2018.
- [19] M. Parmar, Di Wang, Ah-H. Tan, C. Miao, J. Jiang, and Y. Zhou, "A novel density peak clustering algorithm based on squared residual error," in *Proceedings of the 2017 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, pp. 43–48, Shenzhen, China, December 2017.
- [20] M. Parmar, D. Wang, X. Zhang et al., "REDPC: a residual error-based density peak clustering algorithm," *Neurocomputing*, vol. 348, pp. 82–96, 2019.
- [21] M. D. Parmar, W. Pang, D. Hao et al., "FREDPC: a feasible residual error-based density peak clustering algorithm with the fragment merging strategy," *IEEE Access*, vol. 7, pp. 89789–89804, 2019.
- [22] L. Wang, W. Zhou, H. Wang, M. Parmar, and X. Han, "A novel density peaks clustering halo node assignment method based on K-nearest neighbor theory," *IEEE Access*, vol. 7, Article ID 174380, 2019.
- [23] S. Kumar, M. Mohri, and A. Talwalkar, "Sampling methods for the Nyström method," *Journal of Machine Learning Research*, vol. 13, no. 1, pp. 981–1006, 2012.
- [24] J. Dick and M. Feischl, "A quasi-Monte Carlo data compression algorithm for machine learning," *Journal of Complexity*, vol. 67, no. 2021, Article ID 101587, 2021.
- [25] W. Zhang, Y. Guo, J. Zhou, H. Jiang, and R. Wang, "A novel kernel clustering with quasi-Monte Carlo random feature map," in *Proceedings of the 2020 7th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS)*, pp. 854–857, Guangzhou, China, November 2020.
- [26] J. H. Halton, "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," *Numerische Mathematik*, vol. 2, no. 1, pp. 84–90, 1960.
- [27] H. Faure, "Discr pance de suites associ es   un syst me de num ration (en dimension s)," *Acta Arithmetica*, vol. 41, no. 4, pp. 337–351, 1982.
- [28] H. Niederreiter, "Point sets and sequences with small discrepancy," *Monatshefte f r Mathematik*, vol. 104, no. 4, pp. 273–337, 1987.
- [29] L. Fu and E. Medico, "FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data," *BMC Bioinformatics*, vol. 8, no. 1, p. 3, 2007.
- [30] K. Bache and M. Lichman, *UCI Machine Learning Repository*, vol. 28, School of Information and Computer Science, University of California, Irvine, CA, USA, 2013, <http://archive.ics.uci.edu/ml>.
- [31] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering aggregation," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, p. 4, 2007.
- [32] P. Fr nti and O. Virtajoki, "Iterative shrinking method for clustering problems," *Pattern Recognition*, vol. 39, no. 5, pp. 761–775, 2006.
- [33] Y. Zhang, S. Chen, and Y. Ge, "Efficient distributed density peaks for clustering large data sets in MapReduce," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 12, pp. 3218–3230, 2016.