Local Peaks-Based Clustering Algorithm in Symmetric Neighborhood Graph

ZHI LIU, CHUNRONG WU, QINGLAN PENG, JIA LEE, AND YUNNI XIA, (Senior Member, IEEE)

1School of Architecture and Urban Planning, Chongqing University, Chongqing 400044, China
2College of Computer Science, Chongqing University, Chongqing 400044, China
3Chongqing Key Laboratory of Software Theory and Technology, Chongqing University, Chongqing 400044, China

Corresponding author: Chunrong Wu (crwu@cqu.edu.cn)

This work was supported in part by the National Key Research and Development Program of China “Digital Simulation and Evaluation Model for Reconstruction of Town and Village Settlements Space” under Grant 2018YFD1100300.

ABSTRACT Density-based clustering methods have achieved many applications in data mining, whereas most of them still likely suffer poor performances on data sets with extremely uneven distributions, like the manifold or ring data. The paper proposes a novel method for clustering with local peaks in the symmetric neighborhood. Local peaks are points with maximum densities at the local level. During the searching of local peaks, all data, except those outliers, can be easily divided into a number of small clusters in accordance with the local peaks in each point’s neighborhood. Especially, a graph-based scheme is adopted here to merge similar clusters based on their similarity in the symmetric neighborhood graph, followed by assigning each outlier to the closest cluster. A variety of artificial, real data sets and a real building data set have been tested for clustering by the proposed method and compared against other popular density-based methods and other algorithms.

INDEX TERMS Data mining, clustering, density-based clustering, symmetric neighborhood, local peaks.

I. INTRODUCTION

Clustering is an indispensable and fundamental method for data mining, which attempts to classify data objects into categories or clusters on the basis of their similarity. Clustering has found tremendous applications in many fields, such as business intelligence, pattern recognition and cloud computing [1]. Up to now, various algorithms have been proposed, including partitioning methods [2], [3], density-based clustering [4]–[6], spectral clustering [7], [8], hierarchical clustering [9], [10], and distribution-based methods [11].

Partitioning methods firstly partition a data set into \( k \) clusters and then use an iterative control strategy to optimize an objective function. k-means [12] and k-medoids [13] are major representatives, however, these methods can not identify non-spherical clusters. Hierarchical clustering can correctly cluster non-spherical data sets. The top-down method and down-top method are two kinds of hierarchical clustering. The top-down method regards data sets as a cluster, then decides the cluster into the given number of groups according to a rule. However, the down-top method is just the opposite.

Chameleon [9] is a representative algorithm, which integrates top-down and down-top methods. Chameleon constructs the \( k \)-nearest neighbor graph of the data set and divides the graph into a large number of smaller subgraphs through a graph-based partition algorithm. Especially, each subgraph represents an initial cluster. Finally, Chameleon repeatedly merges initial clusters based on the similarity of clusters. However, Chameleon is susceptible to outliers.

Density-based clustering aims at detecting clusters of arbitrary shapes and spotting outliers, inspired by the observation that the centers of each cluster in any data set usually have a higher density than other points around them. A well-known density-based method is the DBSCAN [4], which measures cluster centers in terms of local density. The local density of a point is computed by the number of points in a radius, whereby a point with a certain population density may become a (local) core. Besides, DBSCAN merges two initial clusters if they are density reachable. For the sake of achieving a parameter-free clustering technique, \( k \)-NN-DBSCAN [5] introduces \( k \)-nearest neighbors graph into DBSCAN where \( k > 0 \). Another typical density-based method is the density peak clustering (DPC [14]) algorithm, which uses local density and distance from points of higher density to measure...
whether a point is a cluster center. After DPC locates cluster centers through a decision graph constructed from the preceding two measures, each of the rest points is assigned to the same cluster as its nearest neighbor with higher density. Due to the difficulty in determining a proper cut-off distance that is crucial to cluster center selection, several researchers [15]–[21] adopted k-nearest neighbors (kNN), mutual k-nearest neighbors (MkNN) and natural neighbors (NN) to estimate the local densities of each point. Some studies also consider residual-error [22], [23] and gravitation [24] to improve the algorithm. However, both DBSCAN, DPC and some improved algorithms may not successfully cluster data sets with various densities and overlapping levels.

To efficiently discover clusters with different densities, we recently proposed an improved algorithm based on DPC named DPC-SNR [25]. The algorithm uses reverse k-nearest neighbors to calculating the local density of each point, find the global peak obtained through local density and distance from points of higher density, and cluster from the global peak. Finally, there are several big clusters and some tiny clusters including outliers. Then the tiny clusters are assigned to big ones according to the number of public edges. However, DPC-SNR may not identify small clusters with less than k points except for outliers and sometimes does not obtain the real number of clusters. Moreover, DPC-SNR does not perform well on data sets with different types.

Therefore, to solve the problem that the existing density-based algorithm like DBSCAN and DPC is not friendly to the manifold data with different densities, we propose a new algorithm based on local peaks in symmetric neighborhood graph, called LP-SNG. Considering that using reverse k-nearest neighbors to compute the local density of each point can increase the difference between the core point and the normal point proven in DPC-SNR, DPC-SNG also adopts this way. Moreover, symmetric neighborhood is also used in LP-SNG, which can strengthen the gap between clusters. We obtain the local peaks that are points with a local maximum in the symmetric neighborhood. Moreover, points in the symmetric neighborhood of each local peak belong to the same cluster with the local peak. After that, we will get many small clusters, not including outliers. Then, we obtain the clusters by continuous merging clusters which contain the maximum similarity in the symmetric neighborhood graph. Finally, the rest points are assigned to the same clusters as their nearest points. The experimental results on synthetic and real data sets and a building data set show that the proposed algorithm is more effective. The main contribution of this article is that we propose a new density-based algorithm to fit manifold data with different densities and we consider a new perspective namely local peaks in symmetric neighbors rather than global peaks in DPC and DPC-SNR.

The rest of this paper is organized as follows. Section II outlines the local density and symmetric neighborhood relationship. Section III describes our new clustering algorithm. Section IV performs experiments on a number of synthetic and real data sets, and analyzes the efficiency of our clustering method. This paper finishes with conclusions in Section V.

II. RELATED WORKS

In this section, we will review the process of DPC and introduce these studies which arouse interests in DPC and introduce the symmetric neighborhood relationship.

A. LOCAL DENSITY

Local density often appears in density-based clustering, especially DBSCAN [4] and DPC [14]. DBSCAN and DPC think that the local density is related to the number of neighbors in a certain range. DPC uses the cutoff distance to measure the local density of each point, and the formula shows as followed:

$$\rho_i = \sum_j \chi(\text{dist}(i, j) - d_c)$$

(1)

where $\chi(a) = 1$ if $a < 0$, otherwise $\chi(a) = 0$, dist(i, j) denotes Euclidean distance between point i and point j. DPC-KNN [21] also uses the formula to obtain the local density.

Rodriguez and Laio also use Gaussian kernel function to present $\rho_i$:

$$\rho_i = \sum_j \exp\left(-\frac{\text{dist}^2(i, j)}{d_c^2}\right)$$

(2)

where the point j is eligible when dist(i, j) is less than $d_\epsilon$. $d_c$ is the only influence parameter in two formulas above.

However, we find that DPC performs badly on manifold data sets with different densities. If there are two manifold data with different densities, we find that two cluster centers are still located in manifold data with high density as the value of $d_c$ changes.

There are also some other formulas about local density. Some studies do not change the form in Eq.(2), and introduce nearest neighbors as a scope to limit the local density, the function of which just like $d_c$, in Eq.(2). ADPC-kNN [19] uses kNN to compute $\rho_i$, while $d_c$ is a constant and $d_c = \mu + \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (d_i - \mu)^2}$, $\mu$ is the mean value of $d_i$ of all points, and $d_i = \max_{j \in \text{kNN}(i)} \text{dist}(i, j)$. Similarly, CDP [20] introduces the mutual k-nearest neighbors to get the local density $\rho_i$, however, $d_c$ here is a parameter. For FkNN-DPC [16], they also use kNN to compute the local density, and they change the distance squared becomes distance once with no parameter $d_c$. Especially, DPC-kNN-PCA [17] considers a different form to obtain local density using kNN, which shows as follows:

$$\rho_i = \exp\left(-\frac{1}{p} \sum_{j \in \text{kNN}(i)} \text{dist}^2(i, j)\right)$$

(3)

where $p$ is computed as a percent of the number of data sets.

Some studies think that the local density is the reciprocal of the average distance over a range. Therefore, STClu [15] also
uses \( kNN \) as the range to obtain \( \rho_i \), which defines as follows:

\[
\rho_i = \frac{k}{\sum_{j \in kNN(i)} \text{dist}(i,j)}
\]  

where \( k \) is an input parameter, \( kNN(i) \) the number of \( kNN \) of point \( i \). NaNDP [18] uses the similar formula to get the local density, while they use natural neighbors to obtain the appropriate value of \( k \).

Compared these formulas of \( \rho_i \), we can see that the region of influence is different. Moreover, we think that the differences between core points and non-core points should be increased. Therefore, we choose reverse \( kNN \) as the region of influence, which means the value degree from others. Local peaks should be surrounded by dense points so that the value degree of local peaks will be higher. The traditional kernel density estimation methods may be affected by the setting of the parameters and cannot work well for some complex data sets, so that Yan et al. [26] used a novel potential-based method to estimate density values.

B. SYMMETRIC NEIGHBORHOOD RELATIONSHIP

\( kNN \) is initially used for classification, which arouses the interests of many researchers [27]. Then, \( kNN \) is applied to clustering [15], [16], [28], obtained by the Euclidean distance. Inspired by \( kNN \), there are many studies about other nearest neighbors, such as natural neighbors, mutual \( k \)-nearest neighbors [20], [33] and shared nearest neighbors [29].

\( kNN \) and reverse \( kNN \) are symmetric neighborhood relationship [30], which is the main idea of mutual \( k \)-nearest neighbors and nature neighbors. Let \( D \) be a database, \( i \) and \( j \) be some objects in \( D \), and \( k \) be a positive integer. \( \text{dist}(i,j) \) denotes the Euclidean distance between object \( i \) and \( j \).

The \( k \)-distance of \( i \), denoted as \( k_{\text{dist}}(i) \), is the distance \( \text{dist}(i,o) \) between point \( i \) and point \( o \) in \( D \), which shows as:

- For at least \( k \) objects \( o' \in D \) it holds that \( \text{dist}(i,o') \leq \text{dist}(i,o) \), and
- For at most \( (k-1) \) objects \( o' \in D \) it holds that \( \text{dist}(i,o') < \text{dist}(i,o) \)

If point \( j \) meets \( \text{dist}(i,j) \leq k_{\text{dist}}(i) \), then call \( j \) as one of the \( kNN \) of \( i \). A set of points \( J \) which contains finite points \( j \) forms the \( kNN \) of \( i \), denoted as \( kNN(i) \). The definition of \( kNN(i) \) is:

\[
kNN(i) = \{ J \in D | \text{dist}(i,J) \leq k_{\text{dist}}(i) \}
\]  

Likewise, point \( i \) is regarded as the reverse \( kNN \) of \( j \), and a set of points \( I \) which contains finite points \( i \) composes the reverse \( kNN \), denoted as \( RkNN(j) \), which can be defined as:

\[
RkNN(i) = \{ j | j \in D, i \in kNN(j) \}
\]  

The results of the intersection between the \( kNN \) and the reverse \( kNN \) are used to estimate the density distribution around \( i \), which is called as the symmetric neighborhood of \( i \), denoted as \( SNk(i) \). \( SNk(i) \) means that two people are true friends only when they agree with each other, which shows as follows:

\[
SNk(i) = \{ o | o \in D, o \in (kNN(i) \cap RkNN(i)) \}
\]
III. CLUSTERING ALGORITHM BASED ON SYMMETRIC NEIGHBORHOOD GRAPH

This section will present the details of the proposed clustering algorithm and analyze its complexity.

This paper presents a new density-based clustering algorithm using symmetric neighborhood relationship. Here is the basic idea: firstly, find the symmetric neighborhood of each point; then calculate the local density of each point using reverse $k$NN, and find these representatives for each point to obtain the local peaks that are points with maximum local density in symmetric neighborhood. Especially, orange stars denote representatives and orange lines show the scope of each representative. Moreover, the rest points are assigned to the same cluster as their nearest local peak belongs to so that the initial clusters are obtained. Finally, continually merge clusters with the maximum similarity in the symmetric neighborhood graph. Noting that outliers should not belong to these initial clusters and the algorithm will end the merging process in advance if the value of similarity between clusters is zero. After that, each outlier is assigned to the cluster its nearest neighbor belongs to so that the initial clusters are obtained.

We find that more than a pair of clusters at a time has the maximum similarity in Algorithm 1, and we should merge these clusters at the same time. However, there may be the same clusters between each pair of clusters, which will belong to the same clusters. For example, $(c_1, c_3)$ and $(c_1, c_5)$ are two pairs of clusters with the maximum similarity, then $c_1$, $c_3$ and $c_5$ will be combined into one cluster. Moreover, $c_2$, $c_4$ and $c_6$ belong to the same cluster if $(c_2, c_4)$ and $(c_6, c_4)$ have the maximum similarity at the same time.

The similarity between clusters is based on the weight of edges and the number of edges. The weight of edge $v_{ij}$ between point $i$ and point $j$ is calculated as:

$$w(v_{ij}) = \frac{1}{1 + \text{dist}(i, j)} \quad (8)$$

And then, the formula of the similarity between cluster $c_a$ and cluster $c_b$ shows as follows:

$$\text{Sim}(c_a, c_b) = \sum_{v_{ij} \in V_{c_a,c_b}} w(v_{ij}) \quad (9)$$

where $V_{c_a,c_b}$ means the number of edges between cluster $c_a$ and cluster $c_b$ in symmetric neighborhood graph.
The way to search local peaks is inspired by [10], after obtaining the local density of each point, we will find the local peaks in the symmetric neighborhood. If the local density of point \( i \) is the maximum among the symmetric neighborhood of point \( j \), then point \( i \) is the representative of point \( j \) and other points in the symmetric neighborhood of point \( j \), which is called as \( \text{Rep}(j) = i \). Especially, there may be two or more representatives of a point at the same time. Therefore, if \( \text{Rep}(j) = i \) and \( \text{Rep}(j) = o \) at the same time and \( \text{dist}(j, i) > \text{dist}(j, o) \), then \( \text{Rep}(j) = o \). Moreover, if \( \text{Rep}(j) = i \) and \( \text{Rep}(j) = o \), then \( \text{Rep}(j) = o \). Supposing that the preceding two conditions do not exist simultaneously. If \( \text{Rep}(j) = j \), we call point \( j \) as a local peak. Local peaks are points with maximum local density in the symmetric neighborhood, which are also the initial cluster centers. Each point \( i \) which meets \( \text{Rep}(i) = o \) and \( \text{Rep}(o) = o \) belongs to the same cluster as point \( o \). How to get the local peaks is shown in Algorithm 2.

**Algorithm 2 Getting Local Peaks**

**Input:** The symmetric neighborhood of points, \( \text{SN}_k \); The distance between two points, \( \text{dist} \); The local density of points, \( \rho \); the flag of outliers, \( \text{Outlier} \); The data set, \( D \);

**Output:** The set of initial clusters, \( C \leftarrow c_1, c_2, \ldots, c_m \);

1. Initializing: \( \text{Rep}[i] \leftarrow 0 \), \( \text{localPeaks}[i] \leftarrow 0 \), \( \text{Num} \leftarrow 0 \);
2. for each point \( i \) in descending order of local density do
   3. Find point \( y \) with the maximum local density in \( \text{SN}_k[i] \);
   4. for each point \( j \in \text{SN}_k[i] \) do
      5. if \( \text{Rep}[j] == 0 \) then
         6. \( \text{Rep}[j] \leftarrow y \);
      end if
      7. if \( \text{Rep}[j] \neq y \) then
         8. \( q \leftarrow \text{Rep}[j] \);
         9. if \( \text{dist}[j, q] > \text{dist}[j, y] \) then
            10. \( \text{Rep}[j] \leftarrow y \);
         end if
      end if
      11. for each point \( o \in D \) do
         12. if \( \text{Rep}[o] == j \) then
            13. \( \text{Rep}[o] \leftarrow y \);
         end if
      end for
   end for
   14. Assign each point \( a \) excluding outliers to the cluster which its representative \( \text{Rep}[a] \) belongs to, and get the initial clusters \( c_i \);

**Table 1. Data Characteristics of artificial data sets.**

| Dataset   | Instances | Attributes | Clusters | from |
|-----------|-----------|------------|----------|------|
| Data set 1| 3031      | 2          | 9        | [34] |
| Data set 2| 1000      | 2          | 4        |      |
| Data set 3| 999       | 2          | 3        |      |
| Data set 4| 588       | 2          | 13       | [36] |
| Data set 5| 400       | 2          | 3        |      |
| Data set 6| 373       | 2          | 2        | [35] |
| Data set 7| 770       | 2          | 6        |      |
| Data set 8| 266       | 2          | 3        | [37] |

**B. THE TIME COMPLEXITY ANALYSIS**

The proposed algorithm mainly contains three steps: obtain the symmetric neighborhood, partition the data set into initial clusters by searching local peaks in symmetric neighborhood and merge these initial clusters. The first step contains getting
the distance between two points \((O(N^2))\), searching \(k\)NN and reverse \(k\)NN of each point \((O(N \log N))\) and computing SN \((O(N^2))\). When getting the symmetric neighborhood, it requires computing the local density of each point and dividing the data set by searching local peaks, thus the time complexity is \(O(lN)\) and \(O(lN^2)\) respectively, especially \(l\) is the number of points in symmetric neighborhood of point \(i\). Assuming the number of initial clusters is \(N_c (N_c \ll N)\). Since the similarity between two clusters is stored in a matrix beforehand, the time complexity of computing the similarity is \(O(l^2 N_c^2)\). If there are only one pair points every time, the time complexity of merging these clusters in each iteration is \(O(l^2 N_c^3)\), which is the worse situation. Actually, we find that there are more than one pair of points \((N_p \text{ pairs})\) with maximum similarity every time, the time complexity of merging these clusters in each iteration is \(O(l^2 N_c^3 N_p^3)\) and the number of iterations is less than the difference value between \(N_c\) and the true number of clusters. Therefore, the overall time complexity of the proposed algorithm is \(O(N^2)\).

### IV. EXPERIMENTS

To demonstrate the efficiency of the LP-SNG, we compare the proposed method with DPC-SNR, DBSCAN, DPC, AP [31] and k-means algorithms on synthetic and real data sets including building data. The code of DPC algorithm was provided by its authors and DBSCAN, AP and k-means were implemented with MATLAB R2017b.

In this paper, we use these parameters mentioned in their papers to conduct these experiments, including the number of nearest neighbors \(k\) in DPC-SNR, percent \(p\) in DPC, damping coefficient \(\lambda\), iteration invariant number \(\text{convits}\) and maximum number of iterations \(\text{maxits}\) in AP and the distance \(Eps\) and the number of points in the current distance \(\text{MinPts}\) in DBSCAN. Especially, the DPC implemented by the authors in [39] uses the distance value corresponding to \(p\) percent of the data set in the descending order of the distance value as \(d_c\). Therefore, parameter \(p\) is considered rather than

### TABLE 2. Comparison of two benchmarks for six clustering algorithms on artificial data sets.

| Algorithm | Par | Val | CI | Acc | NMI | Algorithm | Par | Val | CI | Acc | NMI |
|-----------|-----|-----|----|-----|-----|-----------|-----|-----|----|-----|-----|
| Data set 1 | | | | | | | | | | | |
| LP-SNG | \(k\) | 8 | 9 | 1.0000 | 1.0000 | LP-SNG | \(k\) | 5 | 4 | 1.0000 | 1.0000 |
| DPC-SNR | \(k\) | 15 | 8 | 0.9637 | 0.9692 | DPC-SNR | \(k\) | 6 | 4 | 1.0000 | 1.0000 |
| DPC | \(p\) | 10.5 | 9 | 0.7351 | 0.7835 | DPC | \(p\) | 1.3 | 4 | 0.4180 | 0.2362 |
| DBSCAN | \(Eps/\text{MinPts}\) | 16.09 | 9 | 0.9993 | 0.9995 | DBSCAN | \(Eps/\text{MinPts}\) | 0.013/4 | 3 | 1.0000 | 1.0000 |
| AP | \(\lambda/\text{convits}/\text{maxits}\) | 0.95/528 | 9 | 0.5401 | 0.7012 | AP | \(\lambda/\text{convits}/\text{maxits}\) | 0.65/2129 | 17 | 0.0800 | 0.0889 |
| k-means | | 9 | 0.4886 | 0.6685 | k-means | | 4 | 0.4160 | 0.5282 | |

### TABLE 3. Data characteristics of real data sets.

| Dataset | Instances | Attributes | Clusters | From |
|---------|-----------|------------|----------|------|
| Wisconsin_PBC | 198 | 33 | 2 | [41] |
| Sonar | 208 | 60 | 2 | [41] |
| Glass | 214 | 9 | 6 | [41] |
| Haberman_S | 306 | 3 | 2 | [41] |
| Liver_D | 345 | 6 | 2 | [41] |
| Dermatology | 366 | 34 | 6 | [41] |
| Yeast | 1484 | 8 | 10 | [41] |
| Chess | 3196 | 36 | 2 | [41] |
| Spectr_H | 267 | 22 | 2 | [41] |
| Contraceptive_MC | 1473 | 9 | 3 | [41] |
| Hayes_R | 160 | 5 | 3 | [41] |
| Spectr_II | 267 | 22 | 2 | [41] |
| Wisconsin_DBC | 569 | 30 | 2 | [41] |
| Car_E | 1728 | 6 | 4 | [41] |
| Ionosphere | 351 | 34 | 2 | [41] |
| Ecoli | 336 | 7 | 8 | [41] |
TABLE 4. Comparison of two benchmarks for six clustering algorithms on real data sets.

| Algorithm          | Par | Val | Cl | Acc   | NMI  | Algorithm          | Par | Val | Cl | Acc   | NMI  |
|--------------------|-----|-----|----|-------|------|--------------------|-----|-----|----|-------|------|
| Wisconsin_PBC      |     |     |    |       |      |                    |     |     |    |       |      |
| LP-SNG             | k   | 31  | 2  | 0.5756 | 0.0379 | LP-SNG             | k   | 8  | 2  | 0.5673 | 0.0826 |
| DPC-SNR            | k   | 11  | 2  | 0.6414 | 0.0153 | DPC-SNR            | k   | 3  | 3  | 0.4904 | 0.0478 |
| DPC                | p   | 3.8 | 2  | 0.4899 | 0.0216 | DPC                | p   | 5.5 | 2  | 0.4808 | 0.0027 |
| DBSCAN             | Eps/Mpts | 100/3 | 2  | 0.5601 | 0.0187 | DBSCAN             | Eps/Mpts | 1.24 | 2  | 0.4904 | 0.0499 |
| AP                 | lam/conv/maize | 0.9/25/65 | 12 | 0.1465 | 0.0568 | AP                 | lam/conv/maize | 0.9/5/18 | 2  | 0.5673 | 0.0118 |
| k-means            | k   | 2   | 0.5364 | 0.0219 | k-means | k   | 2   | 0.5825 | 0.0124 |
| Glass              |     |     |    |       |      |                    |     |     |    |       |      |
| Liver_D            | k   | 6   | 6  | 0.4673 | 0.4642 | Liver_D            | k   | 3  | 2  | 0.5786 | 0.0196 |
| Yeast              | k   | 8   | 3  | 0.5449 | 0.0112 | Yeast              | k   | 10 | 7  | 0.5164 | 0.5066 |
| Spectral_H         | k   | 4.0 | 2  | 0.4317 | 0.0116 | Spectral_H         | k   | 2  | 2  | 0.5251 | 0.1834 |
| Hayes_R            | k   | 3   | 10 | 0.3187 | 0.1142 | Hayes_R            | k   | 12 | 3  | 0.5191 | 0.0168 |
| DPC-D             | k   | 8   | 4  | 0.5427 | 0.1565 | DPC-D             | k   | 17 | 2  | 0.5316 | 0.0345 |
| DPC                | p   | 3.8 | 10 | 0.1071 | 0.0186 | DPC                | p   | 6.4 | 2  | 0.5044 | 0.0043 |
| DBSCAN             | Eps/Mpts | 0.06/6 | 11 | 0.0856 | 0.0659 | DBSCAN             | Eps/Mpts | 1.71/11 | 2  | 0.4916 | 0.0113 |
| AP                 | lam/conv/maize | 0.9/8/23 | 12 | 0.1819 | 0.2501 | AP                 | lam/conv/maize | 0.9/5/17 | 4  | 0.5352 | 0.0232 |
| k-means            | k   | 10  | 0.5070 | 0.0014 | k-means | k   | 6   | 0.2678 | 0.2115 |
| Spectral_M         | k   | 24  | 10 | 0.7828 | 0.0147 | Spectral_M         | k   | 19 | 3  | 0.4277 | 0.0155 |
| DPC-D             | k   | 9   | 2  | 0.7191 | 0.0010 | DPC-D             | k   | 8  | 4  | 0.3897 | 0.0306 |
| DPC                | p   | 3.5 | 10 | 0.0592 | 0.0000 | DPC                | p   | 3.3 | 3  | 0.3170 | 0.0168 |
| DBSCAN             | Eps/Mpts | 20/5 | 2  | 0.0472 | 0.0397 | DBSCAN             | Eps/Mpts | 1.9/8 | 3  | 0.3130 | 0.0252 |
| AP                 | lam/conv/maize | 0.9/5/23 | 12 | 0.1819 | 0.2501 | AP                 | lam/conv/maize | 0.7/4/35 | 2  | 0.5044 | 0.0601 |
| k-means            | k   | 10  | 0.5070 | 0.0014 | k-means | k   | 6   | 0.2678 | 0.2115 |
| Wisconsin_DBC      |     |     |    |       |      |                    |     |     |    |       |      |
| LP-SNG             | k   | 45  | 2  | 0.8981 | 0.5143 | LP-SNG             | k   | 5  | 4  | 0.7014 | 0.0408 |
| DPC-SNR            | k   | 11  | 3  | 0.4535 | 0.1918 | DPC-SNR            | k   | 7  | 3  | 0.4682 | 0.1589 |
| DPC                | p   | 2.2 | 2  | 0.7735 | 0.3170 | DPC                | p   | 3.8 | 4  | 0.4462 | 0.1520 |
| DBSCAN             | Eps/Mpts | 100/3 | 2  | 0.6061 | 0.0187 | DBSCAN             | Eps/Mpts | 1.2/5 | 3  | 0.3009 | 0.1816 |
| AP                 | lam/conv/maize | 0.9/4/35 | 2  | 0.6732 | 0.1342 | AP                 | lam/conv/maize | 0.8/4/28 | 24 | 0.0579 | 0.1816 |
| k-means            | k   | 2   | 0.8541 | 0.0472 | k-means | k   | 4   | 0.2708 | 0.1440 |
| Ionosphere         |     |     |    |       |      |                    |     |     |    |       |      |
| LP-SNG             | k   | 47  | 2  | 0.6555 | 0.0669 | LP-SNG             | k   | 5  | 8  | 0.6467 | 0.6187 |
| DPC-SNR            | k   | 12  | 2  | 0.5536 | 0.1073 | DPC-SNR            | k   | 7  | 8  | 0.3869 | 0.4628 |
| DPC                | p   | 1.2 | 2  | 0.4872 | 0.0706 | DPC                | p   | 2.8 | 8  | 0.3244 | 0.4292 |
| DBSCAN             | Eps/Mpts | 0.3/4 | 26 | 0.0712 | 0.1011 | DBSCAN             | Eps/Mpts | 0.1/6 | 7  | 0.3571 | 0.3856 |
| AP                 | lam/conv/maize | 0.9/3/50 | 26 | 0.1053 | 0.2400 | AP                 | lam/conv/maize | 0.7/6/15 | 17 | 0.1458 | 0.5930 |
| k-means            | k   | 2   | 0.7094 | 0.1230 | k-means | k   | 8   | 0.5179 | 0.5807 |

$d_{c}$ for simplicity in our paper. For k-means algorithm, there is no parameter and k is the number of clusters. We did multiple experiments and got the best result of each method.

Two indexes including clustering accuracy (Acc) [10], [17] and Normalized Mutual Information (NMI) [10], [32] are used to evaluate the clustering performance on these experiments, which are positive indexes. And the formula of Acc shows as follows:

$$\text{Acc} = \frac{1}{N} \sum_{i=1}^{N} \delta(r_i, \text{map}(s_i))$$  \hspace{1cm} (11)$$

where $r_i$ is the real cluster label, $s_i$ the serial number obtained by clustering. If $a = b, \delta(a, b) = 1$; otherwise, $\delta(a, b) = 0$. The larger value of Acc means the better clustering performance of the algorithm.

The formula of NMI shows as follows:

$$\text{NMI}(X, Y) = \frac{\text{MI}(X, Y)}{\sqrt{H(X) \cdot H(Y)}}$$  \hspace{1cm} (12)$$

where $\text{MI}(X, Y)$ is the mutual information between two random variables $X$ and $Y$, $H(Z)$ the entropy of random variables $Z$. When the value of NMI is bigger, clustering performance is better.
A. CLUSTERING ON ARTIFICIAL DATA SETS

We choose eight artificial data sets to demonstrate the efficiency of LP-SNG, the details of which are shown in Table 1. Especially, Data set 1 and Data set 4 consist of several irregular shapes. Data set 2 contains 4 ring data with the same density, which is also in Data set 3 with two spherical data in different densities. While Data set 7 has a ring data with different densities and a spherical data with the same density. The remaining data sets are dominated by manifold data with different densities. The Acc and NMI scores about these four algorithms are shown in Table 2, and the running times are also listed in Table 5. Noting that nothing is done about outliers in DBSCAN.

From Table 5, we can see that LP-SNG always has a higher running time, because the process of calculating the similarity between clusters need to take up three-quarters of total time. Moreover, more the number of local peaks means more iterations, which will greatly increase the running time.

The clustering results of the proposed algorithm can be seen in Fig. 3. Symmetric neighbors can weaken the connection between clusters in [38], and LP-SNG divides the data set into many small initial clusters with the help of local peaks in symmetric neighbors, which makes LP-SNG perform well on manifold and ring data. Moreover, LP-SNG removes outliers temporarily to make the boundary of clusters clearer. Table 2 shows that LP-SNG can get higher scores of Acc and NMI, which means LP-SNG is robust to outliers and more powerful than other clustering algorithms on data sets with extremely uneven distributions and irregular shapes.

DPC-SNR also performs well on manifold and ring data with different densities and overlapping levels in Fig. 4, while DPC-SNR can not identify tiny clusters on Data set 7 because DPC-SNR thinks a tiny cluster is initially seen as an individual, however later must attach itself to the large cluster. Though DPC-SNR can recognize clusters on Data set 4, DPC-SNR mistakes two clusters that are close together as a whole for Data set 1.

For the DPC algorithm, we adopt the Gaussian function to obtain the local density. The results in Fig. 5 show that DPC cannot select the right cluster centers on manifold and ring data with extremely uneven distributions and ring data with even distributions. Meanwhile, DPC is noneffective on the data set with different shapes, and even DPC identifies two close clusters as a cluster.

Seeing from Table 2, DBSCAN can have a certain capacity to cluster ring data and data with different shapes. Especially, DBSCAN thinks of clusters with three points as outliers so that DBSCAN only identifies two clusters on Data set 6, just like DPC-SNR. However, DBSCAN performs badly on spherical and manifold data with different densities and overlapping levels and DBSCAN even can not completely identify the number of clusters on spherical data in Fig. 6. Unfortunately, DBSCAN even identifies a manifold cluster as four clusters with a few outliers on Data set 4.

AP performs badly on manifold and ring data with different densities and overlapping levels and data with different shapes. Moreover, AP cannot recognize clusters on Data set 7 because AP thinks a small cluster as an individual, however later must attach itself to the large cluster. Though AP can recognize clusters on Data set 4, AP mistakes two close clusters as a whole for Data set 1.
shapes in Fig. 7, which cannot identify true clusters on ring data with the same density. Furthermore, AP often identifies the wrong number of clusters. Fortunately, AP can correctly identify some spherical clusters.

Though the running time of k-means algorithm is the least, k-means algorithm is also not suitable for manifold and ring data in Fig. 8. Since the number of clusters in k-means algorithm is given, there is no problem with the incorrect number of clusters. This algorithm randomly specifies the cluster center, while the location of the cluster center largely determines the clustering accuracy. Along with the principle of nearby assignment, a cluster will think of points that should belong to other clusters as its own, especially on data set 6.

### B. CLUSTERING ON REAL DATA SETS

To further demonstrate the efficiency of the proposed algorithm, we compare our algorithm with the five algorithms mentioned above on several benchmark real data sets from UCI [41], which are often used in clustering or classification. The detailed information about these data sets is shown in Table. 3, which are preprocessed to better serve the clustering algorithm.
The comparison of Acc and NMI scores are listed in Table. 4. The best results are shown in bold. The results show that the Acc and NMI scores of LP-SNG are higher than AP, DBSCAN, DPC and k-means algorithms on most data sets. Moreover, LP-SNG can obtain almost fifteen percent of improvement on Wisconsin_PBC, Haberman_S, Dermatology, Spect_H, Wisconsin_DBC, Car_E and Ecoli data sets in terms of Acc. Especially, LP-SNG performs well on Wisconsin_PBC, Liver_D, Dermatology, Wisconsin_DBC and Ecoli data sets in terms of Acc and NMI. DBSCAN can get the best result on Haberman_S and Hayes_R in terms of NMI and AP gets the best value of Acc on Glass and Sonar, NMI on Spect_H. Moreover, k-means performs well on Ionosphere data set. However, sometimes DPC-SNR, DBSCAN and AP can not identify the true number of clusters.

Moreover, the running time of each algorithm can be seen in Table. 6. When the size of the data set is small, there is not much difference in the running time between LP-SNG and DPC-SNR. However, when the size is big, the running time of LP-SNG is bigger than that of DPC-SNR. Because LP-SNG need to spend more time repeatedly calculating the similarity between clusters and finding the clusters with max similarity.
C. CLUSTERING ON BUILDING DATA SETS

We collect data from 10 communities in Chongqing, China, each with a concentration of buildings. These data include the number of building floors, the ground floor area, the building perimeter, the building height, the building X coordinate and the building Y coordinate. Especially, the center of the building is regarded as a point to obtain the coordinates of the building. The overview of these buildings is shown in Fig. 9 and the data set can be seen in [42]. The clustering results and running time based on six algorithms are shown in Table 7. LP-SNG can get higher values than other five algorithms. Especially, only LP-SNG, DPC and k-means can get the true number of clusters.

D. STUDYING ON DIFFERENT VALUES OF k

We study the sensitivity of parameter $k$ on artificial data sets and the results are shown in Fig. 10. We find that within some interval of $k$, the change of $k$ has no effect on the results. Moreover, for data set 6, the interval is big, while it is small for data set 7. Especially, complex data sets have only one optimal $k$ value, such as data set 1. We also do several experiments on different data sets with different values of $k$, the results of which can be roughly divided into two categories, as shown in Fig. 11. One has the constant number of clusters in Fig. 11(b), and the other has the number of clusters that fluctuate and then tend to stay the same.
This paper proposed a new method for clustering, called LP-SNG, which introduces reverse \( k \)-nearest neighbors to Cluster Analysis. The global \( k \)-means clustering algorithm,” *Pattern Recognit.*, vol. 36, no. 2, pp. 451–461, Feb. 2003.

3. R. Ng and J. Han. “CLARANS: A method for clustering objects for spatial data mining,” *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 5, pp. 1003–1016, Sep. 2002.

4. 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, no. 34, pp. 226–231, 1996.

5. A. Sharma and A. Sharma. “KNN–DBSCAN: Using \( k \)-nearest neighbor information for parameter-free density based clustering,” in *Proc. Int. Conf. Intell. Comput.*, *Instrum. Control Technol. (ICICICT)*, Jul. 2017, pp. 787–792.

6. Y. Lv, T. Ma, M. Tang, J. Cao, Y. Tian, A. Al-Dhelaan, and M. Al-Rodhaan, “An efficient and scalable density-based clustering algorithm for datasets with complex structures,” *Neurocomputing*, vol. 171, pp. 9–22, Jan. 2016.

7. P. Yang, Q. Zhu, and B. Huang. “Spectral clustering with density sensitive similarity function,” *Knowl.-Based Syst.*, vol. 24, no. 5, pp. 621–628, Jul. 2011.

8. H. Chang and D.-Y. Yeung, “Robust path-based spectral clustering,” *Pattern Recognit.*, vol. 41, no. 1, pp. 191–203, Jan. 2008.

9. G. Karypis, E.-H. Han, and V. Kumar, “Chameleon: Hierarchical clustering using dynamic modeling,” *Computer*, vol. 32, no. 8, pp. 68–75, 1999.

10. D. Cheng, Q. Zhu, J. Huang, Q. Wu, and L. Yang, “A novel cluster validity index based on local cores,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 985–999, Apr. 2019.

11. G. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, vol. 382, Hoboken, NJ, USA: Wiley, 2007.

12. J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*. Amsterdam, The Netherlands: Elsevier, 2012.

13. L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. New York, NY, USA: Wiley, 2005.

14. A. Rodriguez and A. Laio, “Clustering by fast search and find of density peaks,” *Science*, vol. 344, no. 6191, pp. 1492–1496, Jun. 2014.

15. G. Wang and Q. Song, “Automatic clustering via outward statistical testing on density metrics,” *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 8, pp. 1971–1985, Aug. 2016.

16. 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.

17. 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.

18. D. Cheng, Q. Zhu, J. Huang, and L. Yang, “Natural neighbor-based clustering algorithm with density peaks,” in *Proc. Int. Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 92–98.

19. L. Yaohui, M. Zhengming, and Y. Fang, “Adaptive density peak clustering based on \( k \)-nearest neighbors with aggregating strategy,” *Knowl-Based Syst.*, vol. 133, pp. 208–220, Oct. 2017.

20. Z. Li and Y. Tang. “Comparative density peaks clustering,” *Expert Syst. Appl.*, vol. 95, pp. 236–247, Apr. 2018.

21. 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,” *Phys. A. Stat. Mech. Appl.*, vol. 523, pp. 702–713, Jun. 2019.

22. M. Parmar, D. Wang, X. Zhang, A.-H. Tan, C. Miao, J. Jiang, and Y. Zhou, “REDPC: A residual error-based density peak clustering algorithm,” *Neurocomputing*, vol. 348, pp. 82–96, Jul. 2019.

23. M. D. Parmar, W. Pang, D. Hao, J. Jiang, W. Liu, L. Wang, and Y. Zhou, “REDPC: A feasible residual error-based density peak clustering algorithm with the fragment merging strategy,” *IEEE Access*, vol. 7, pp. 89789–89804, 2019.
[24] J. Jiang, Y. Chen, D. Hao, and K. Li, “DPC–LG: Density peaks clustering based on logistic distribution and gravitation,” Phys. A, Stat. Mech. Appl., vol. 514, pp. 25–35, Jan. 2019.

[25] C. Wu, J. Lee, T. Isokawa, J. Yao, and Y. Xia, “Efficient clustering method based on density peaks with symmetric neighborhood relationship,” IEEE Access, vol. 7, pp. 60684–60696, 2019.

[26] H. Yan, Y. Lu, and L. Li, “A potential-based density estimation method for clustering using decision graph,” in Intelligent Data Engineering and Automated Learning IC IDEAL (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2017, pp. 73–82.

[27] W. Wu, J. Liu, H. Rong, H. Wang, and M. Xian, “Efficient k-nearest neighbor classification over semantically secure hybrid encrypted cloud database,” IEEE Access, vol. 6, pp. 41771–41784, 2018.

[28] Y. Qin, Z. L. Yu, C.-D. Wang, Z. Gu, and Y. Li, “A novel clustering method based on hybrid K-nearest-neighbor graph,” Pattern Recognit., vol. 74, pp. 1–14, Feb. 2018.

[29] L. Ertöz, M. Steinbach, and V. Kumar, “Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data,” in Proc. SIAM Int. Conf. Data Mining, May 2003, pp. 47–58.

[30] W. Jin, A. K. H. Tung, J. Han, and W. Wang, “Ranking outliers using symmetric neighborhood relationship,” in Proc. Pacif. Asia Conf. Knowl. Discovery Data Mining. Berlin, Germany: Springer, 2006, pp. 577–593.

[31] B. J. Frey and D. Dueck, “Clustering by passing messages between data points,” Science, vol. 315, no. 5814, pp. 972–976, Feb. 2007.

[32] S.-O. Abbasi, S. Nejatian, H. Parvin, V. Rezaie, and K. Bagherifard, “Clustering ensemble selection considering quality and diversity,” Artif. Intell. Rev., vol. 52, no. 2, pp. 1311–1340, Aug. 2019.

[33] Z. Hu and R. Bhatnagar, “Clustering algorithm based on mutual K-nearest neighbor relationships,” Statist. Anal. Data Mining, vol. 5, no. 2, pp. 100–113, Apr. 2012.

[34] S. Salvador and P. Chan, “Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms,” in Proc. 16th IEEE Int. Conf. Tools Artif. Intell., Feb. 2005.

[35] A. K. Jain and M. H. C. Law, “Data clustering: A user’s dilemma,” in Pattern Recognition and Machine Intelligence (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2005, pp. 1–10.

[36] K. Faceli, T. C. Sakata, M. C. De Souto, and A. C. De Carvalho, “Partitions selection strategy for set of clustering solutions,” Neurocomputing, vol. 73, nos. 16–18, pp. 2809–2819, Oct. 2010.

[37] L. Zelnik-Manor and P. Perona, “Self-tuning spectral clustering,” in Proc. Adv. Neural Inf. Process. Syst., 2005, pp. 1601–1608.

[38] M. A. Abbas and A. A. Shoukry, “CMUNE: A clustering using mutual nearest neighbors algorithm,” in Proc. 11th Int. Conf. Inf. Sci., Signal Process. Appl. (ISSPA), Jul. 2012, pp. 1192–1197.

[39] A. Laio. Clustering by Fast Search-and-Find of Density Peaks. Accessed: Oct. 24, 2019. [Online]. Available: https://people.sissa.it/~laio/Research/Res_clustering.php

[40] P. Fränti and S. Sieranoja, “K-means properties on six clustering benchmarks,” Neurocomputing, vol. 74, pp. 1311–1340, Aug. 2019.

[41] Z. Liu et al.: Local Peaks-Based Clustering Algorithm in Symmetric Neighborhood Graph

ZH LIU received the B.S. degree in urban planning from the Heilongjiang University of Science and Technology, Harbin, China, in 2003, and the M.S. degree in building technology from Chongqing University, Chongqing, China, in 2009, where he is currently pursuing the Ph.D. degree. His research interests include urban big data collection, calculation, and application.

CHURONG WU received the B.S. degree in software engineering from Xinjiang University, Urumchi, China, in 2016, and the M.S. degree in software engineering from Zhejiang University, Hangzhou, China, in 2018. She is currently pursuing the Ph.D. degree with Chongqing University, Chongqing, China. Her research interests include data mining and intelligent computing.

QINGLAN PENG received the B.S. degree in software engineering from Xinjiang University, Urumchi, China, in 2016, and the M.S. degree in software engineering from Zhejiang University, Hangzhou, China, in 2018. He is currently pursuing the Ph.D. degree with Chongqing University, Chongqing, China. His research interests include cloud computing and service computing.

JIA LEE received the B.E, M.E., and Ph.D. degrees from Hiroshima University, Hiroshima, Japan, in 1996, 1998, and 2001, respectively. He is currently a Professor with the College of Computer Science, Chongqing University, China. His research interests include cellular automata, swarm intelligence, and asynchronous circuits.

YUNNI XIA (Senior Member, IEEE) received the B.S. degree in computer science from Chongqing University, China, in 2003, and the Ph.D. degree in computer science from Peking University, China, in 2008. Since 2008, he has been an Associate Professor with the School of Computer Science, Chongqing University. He has authored or coauthored over 50 research publications. His research interests are in Petri nets, software quality, performance evaluation, and cloud computing system dependability.

YUNNI XIA (Senior Member, IEEE) received the B.S. degree in computer science from Chongqing University, Chongqing, China, in 2003, and the Ph.D. degree in computer science from Peking University, China, in 2008. Since 2008, he has been an Associate Professor with the School of Computer Science, Chongqing University. He has authored or coauthored over 50 research publications. His research interests are in Petri nets, software quality, performance evaluation, and cloud computing system dependability.