Expert Systems

WILEY

Expert Systems

ORIGINAL ARTICLE

An Improved Grid Clustering Algorithm for Geographic Data Mining

Honglei He

School of Information Engineering, Lianyungang Technical College, Lianyungang, China

Correspondence: Honglei He (hhl@lygtc.edu.cn)

Received: 16 August 2024 | **Revised:** 21 February 2025 | **Accepted:** 16 March 2025

Keywords: georeferenced clustering | grid analysis | grid clustering | hybrid clustering | improved algorithms

ABSTRACT

Grid clustering is a classical clustering algorithm with the advantage of lower time complexity, which is suitable for the analysis of large geographic data. However, it is sensitive to the grid division parameter *M* and density threshold *R*, and the clustering accuracy is poor. The article proposes a hybrid clustering algorithm HCA-BGP based on grid and division. the algorithm first uses grid clustering to obtain the core part of the class family, and then uses the division-based method to obtain the edge part of the class family. Through experiments on simulated datasets and real geographic datasets, it is proved to have better results than the existing grid clustering as well as some other classical algorithms. In terms of clustering accuracy, compared with the classical grid clustering algorithm Clique, the clustering F-value of this paper's algorithm is improved by 20.3% on dataset S1, 81.8% on dataset R15, and 7.6% on average on the eight geographic datasets. In terms of the sensitivity of parameters *M* and *R*, compared with Clique, the variance of the clustered F-value of this paper's algorithm is reduced by 89.3% on dataset S1; the variance of the clustered ARI is reduced by 99.9% on the real geographic dataset Data8. Compared to another grid-based clustering algorithm, GDB, HCA-BGP also demonstrates significant advantages.

1 | Introduction

Clustering, as a significant branch of unsupervised learning, encompasses diverse algorithm types with wide-ranging applications. Each clustering method possesses unique characteristics and demonstrates distinct advantages. Grid clustering algorithms, characterised by lower time complexity, have shown strong potential in handling massive datasets. Furthermore, these algorithms exhibit insensitivity to incremental data and variations in input order, capable of detecting clusters of arbitrary shapes. They maintain good scalability regardless of dataset size and demonstrate excellent noise resistance when confronted with noisy data. Therefore, grid clustering algorithms are commonly used for mining analysis of geographic datasets.

However, traditional grid clustering methods are not without limitations. Sensitivity to grid partition parameters *M* and dense grid threshold *R* often results in varied clustering outcomes due to parameter setting differences. Setting *M* too small may overlook some clusters, while setting it too large might split a cohesive cluster into multiple parts. Moreover, using a fixed grid for data partitioning can lead to misclassification of boundary region data points into low-density grids, subsequently diminishing clustering accuracy.

Therefore, many scholars have proposed enhancements to classical grid clustering algorithms.

One approach involves improvements to grid partitioning methods. For example, Starczewski Artur introduced a method based on the kdist function to automatically determine the number of grid cells, where the kdist function calculates the distance between each data point and its k-th nearest neighbour (Starczewski et al. 2021). Gui et al. (2020) proposed

a multiscale grid clustering method that detects hierarchical spatial patterns by integrating granularity analysis and visual cognition. Ma et al. (2024) presented a batch incremental CLIQUE clustering algorithm based on critical grids to adapt to changes in cluster structures. Zheng and Cao (2019) proposed an adaptive grid partitioning method to simplify the setting of initial parameters in grid clustering. Wang et al. (2020) proposed a local grid dynamic clustering algorithm that uses dynamic grid partitioning to avoid deletion errors at cluster boundaries. Cai et al. (2023) developed a distributed clustering algorithm with adaptive grid partitioning. Lin et al. (2018) proposed an improved grid clustering algorithm that employs boundary correction and sliding grid methods to avoid jagged clustering boundaries.

Another approach combines grid clustering with density analysis. For instance, Bureva et al. (2021) introduced a technique combining subspace grid clustering with density-based analysis, dividing the data space into a finite number of grid cells to detect similar groups and define clusters. Xu (2021) used CLIQUE grids to eliminate noise and determine initial cluster centres based on regional density, integrating the Fast Search and Find of Density Peaks (CFSFDP) method to reduce the influence of grid density errors on cluster centre selection. Zheng and Cao (2019) designed a clustering algorithm that combines density-based and grid-based methods. Guo et al. (2023) proposed a composite grid clustering algorithm based on density and balance. Sun et al. (2022) developed an improved clustering algorithm that integrates grid partitioning and DBSCAN. Dai et al. (2022) introduced a grid-based local adaptive DBSCAN clustering algorithm. Li et al. (2020) proposed an adaptive grid clustering algorithm with adaptive density threshold selection to improve cluster boundary accuracy.

Some other methods use additional algorithms to optimise grid clustering. Lin et al. (2018) proposed a method combining subspace clustering and ensemble learning. Oghadam and Ahmadi (2024) applied the classic CLIQUE algorithm to partition the data space into multiple parts and utilised an improved deer algorithm (RDA) for searching. Zou et al. (2023) presented a method combining grid clustering with Gaussian distribution for data detection. Chen et al. (2023) proposed a parallel gravity clustering algorithm (PGCGP) based on grid partitioning. Zhang et al. (2023) proposed a grid clustering method based on localised differential privacy.

These studies address some shortcomings of classical grid clustering algorithms but remain sensitive to initial parameters, requiring multiple input parameters. Some of the proposed algorithms are intricate and have higher time complexities, limiting their applicability to specific types of data while aiming to enhance clustering accuracy.

This paper proposes a hybrid clustering algorithm based on grid and partitioning techniques, requiring only one input parameter to operate effectively. The algorithm automatically determines the number of grid divisions M and density threshold R based on the distribution of the dataset. Each grid can independently split, making it insensitive to parameters M and R and suitable for complex data shapes.

Contributions of this paper:

1. Hybrid Clustering Algorithm Based on Grid and Partitioning: The core of each cluster is obtained using grid clustering, while the edges are identified using partitioning clustering. This approach combines the speed and accuracy advantages of both grid clustering and partitioning clustering.
2. Automatic Determination of Grid Division M and Density Threshold R: The algorithm proposes a method to automatically determine these parameters based on dataset characteristics.
3. Grid Splitting Method: If the number of samples within a grid exceeds the density threshold R but is unevenly distributed, the grid is split. This splitting process is recursively applied to sub-grids. This method ensures that the grid clustering remains insensitive to parameters M and R and is applicable to datasets with complex distributions.

The rest of the paper is organised as follows. In Section 2, the classical grid clustering algorithm and the division-based clustering algorithm are introduced. In Section 3, the main idea of the algorithm proposed in this paper is explained. In Section 4, the algorithm proposed in this paper is experimented on and compared with other existing clustering algorithms on different datasets. The experimental datasets include simulated datasets and real geo-referenced datasets. In Section 5, the effect of grid parameters M and R on the clustering results is discussed. Section 6 contains the conclusions.

2 | Related Work

2.1 | Classical Grid Clustering Algorithm

The classical grid clustering algorithm CLIQUE uniformly divides each dimension of the data space into M intervals, thereby forming a total of M^T grid cells for a T-dimensional data space.

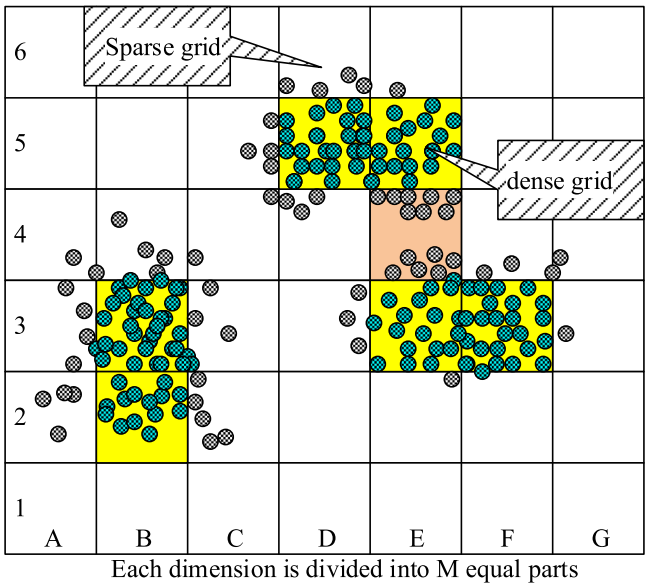


FIGURE 1 | Key concepts in classical grid clustering.

Subsequently, the dataset to be analysed is mapped into these grids. A key concept in the algorithm is the density threshold R. If the number of data points within a grid cell exceeds this threshold, the grid is termed a dense grid; otherwise, if the number of points is less than or equal to R, it is termed a sparse grid, as shown in Figure 1.

The following are key definitions involved in the CLIQUE algorithm:

Definition 1. Dense Grid: A grid cell is defined as dense if the number of data points it contains exceeds the predefined density threshold R.

Definition 2. Sparse Grid: A grid cell is classified as sparse if the number of data points inside it equals or is less than the density threshold R.

Based on these definitions, the CLIQUE algorithm further identifies and combines all adjacent dense grids to form clusters. Specifically, two grid cells are considered adjacent if they share at least one edge or face in multi-dimensional space.

Definition 3. Adjacent Grids: Two grid cells are considered adjacent if they share at least one boundary (edge or face) in multi-dimensional space. For example, grids E3 and F3 are adjacent in a specific dimension, thus forming an adjacent grid pair.

Thus, all dense grids and their adjacent grids collectively form a complete cluster, where all contained data points are considered part of the same cluster. Through these definitions and operations, the CLIQUE algorithm systematically identifies dense regions in the dataset and effectively partitions the data into multiple clusters by analysing the adjacency relationships between grids.

The classical CLIQUE algorithm has several main drawbacks: Firstly, data in sparse grids are pruned, affecting clustering accuracy and potentially causing data that should be classified to be discarded. Secondly, the algorithm is sensitive to the number of grid divisions M and the density threshold R; a too small R value may merge clusters, while a too large R value may lead to cluster fragmentation. Additionally, CLIQUE does not consider clusters that may form from non-adjacent grids. It assumes that data within the same grid belong to the same cluster, disregarding situations where data in the same grid, such as grid E4, may belong to different clusters. These limitations restrict the algorithm's accuracy and applicability.

2.2 | Partition-Based Clustering

Partition-based clustering algorithms are a common data analysis technique used to partition objects in a dataset into different groups or clusters, where objects within each group are similar to each other while objects across different groups are dissimilar. The goal of such algorithms is to automatically discover underlying patterns and structures in data without requiring prior knowledge.

Partition-based clustering algorithms typically involve the following steps:

Initialization: Select initial cluster centres or the number of clusters. The initial choice of cluster centres significantly impacts the algorithm's performance.

Assignment: Assign each object in the dataset to the cluster whose centre is closest to it, forming initial clusters.

Update: Recalculate the centre position for each cluster, usually as the mean of all objects in the cluster.

Iteration: Repeat the assignment and update steps until a stopping condition is met, such as cluster centres no longer changing or reaching a preset number of iterations.

Common partition-based clustering algorithms include K-means and K-medoids algorithms (Abiodun et al. 2023).

The advantages of partition-based clustering algorithms lie in their simplicity, high computational efficiency, and are particularly suitable for large-scale datasets. However, they also have limitations, such as sensitivity to the initial choice of cluster centers (Abubaker and Ashour 2013; He et al. 2022; Mann and Chawla 2023).

3 | Method

3.1 | Design Philosophy of Improved Algorithm

The design concept of the grid and partition-based hybrid clustering algorithm is as follows: First, grid clustering is used to obtain the main parts of the clusters in the original dataset, as shown in Figure 2. Then, a partition-based method is used to assign the remaining data in the original dataset to the clusters obtained in the previous step, as shown in Figure 3.

In the actual clustering process, to accommodate complex data shapes and reduce sensitivity to initial parameters,

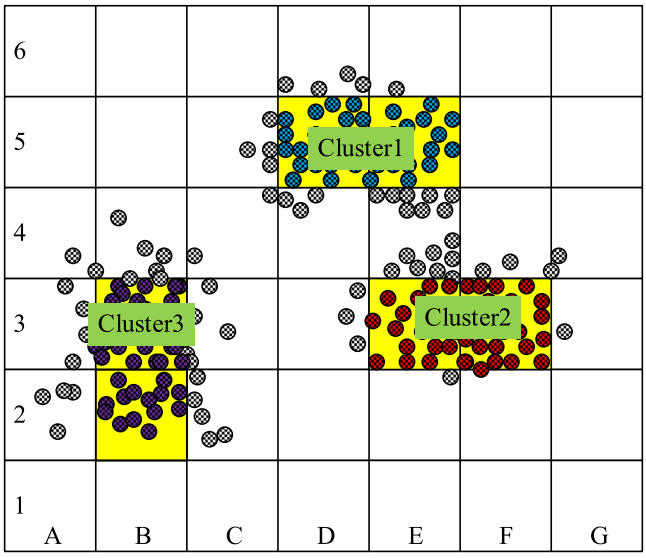


FIGURE 2 | Obtaining the main parts of the clusters using grid clustering.

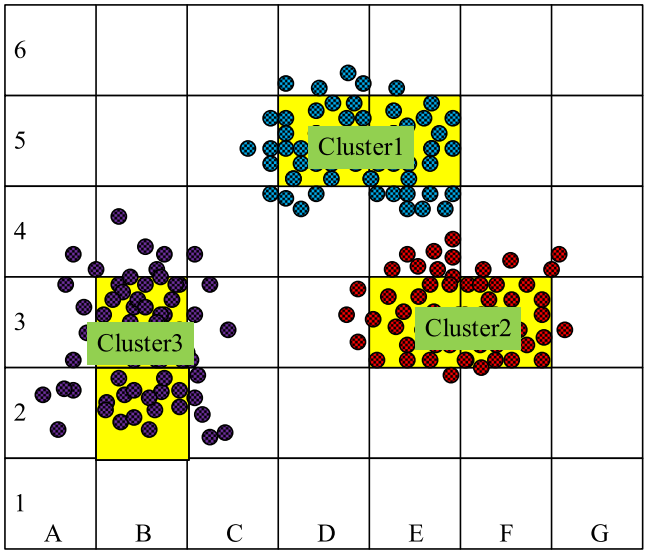


FIGURE 3 | Obtaining the edge data of the clusters using a partition-based method.

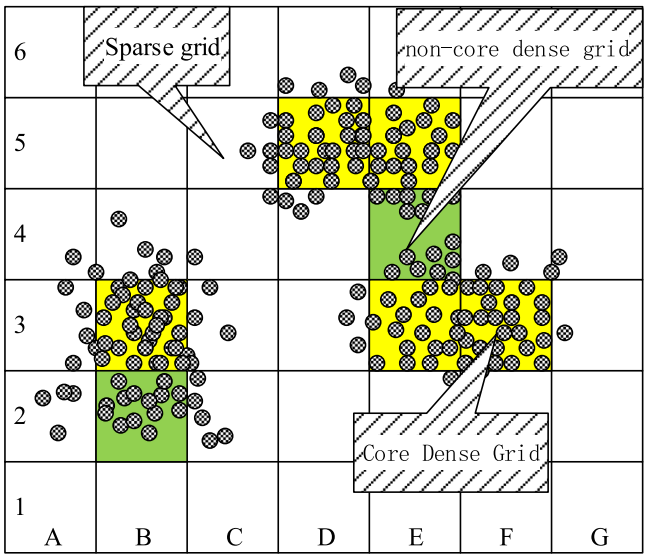


FIGURE 4 | Determine the type of grid.

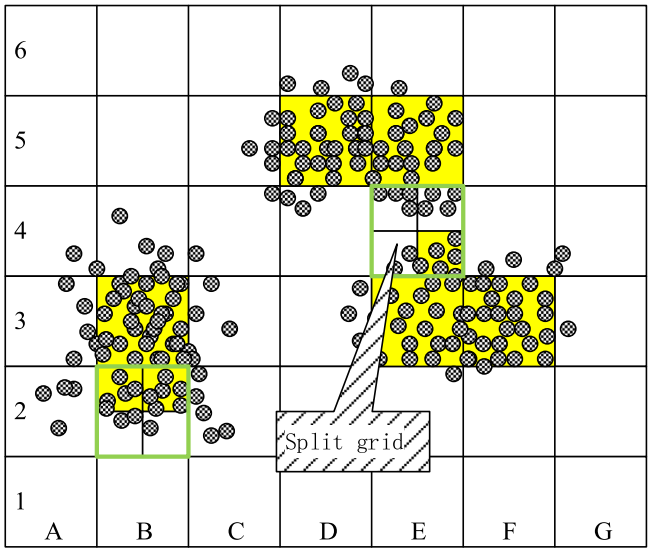


FIGURE 5 | Split the non-core dense grid.

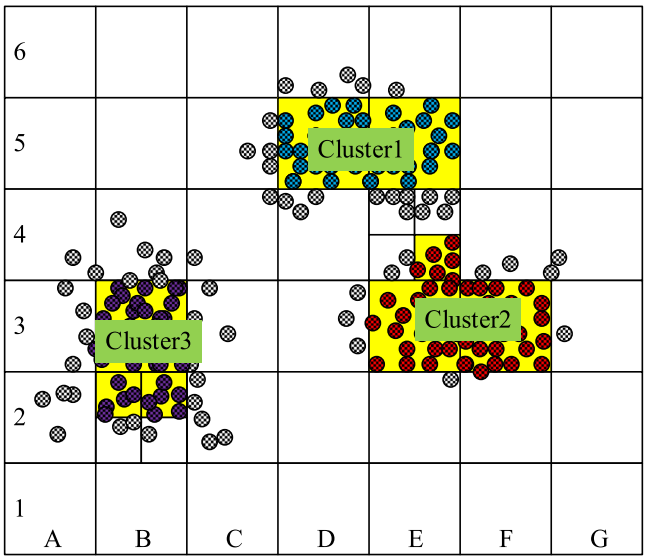


FIGURE 6 | Connected core dense grids form a cluster.

improvements were made to traditional grid clustering and partition clustering algorithms. The specific steps are as follows:

1. Calculate the number of grid divisions M and the density threshold R. The calculation method is provided in Section 3.2.
2. Project the dataset onto the grid.
3. Traverse all the grids and determine the type of each grid (method described in Section 3.2). The grid types include core dense grids, non-core dense grids, sparse grids and empty grids, as shown in Figure 4.

Definition 4. Core Dense Grid. A grid cell containing a number of data points greater than the density threshold R and having uniformly distributed data, as illustrated by the F3 grid in Figure 4. The data distribution is measured using the mean deviation from the grid center and the data dispersion.

Definition 5. Non-Core Dense Grid. A grid cell containing a number of data points greater than the density threshold R but with unevenly distributed data, as illustrated by the E4 grid in Figure 4.

4. If the grid is a non-core dense grid, split the grid as shown in Figure 5. Divide each dimension into two equal parts, forming 2^T sub-grids. Traverse all sub-grids, and if a sub-grid is a non-core dense grid, continue to split it.
5. Form a cluster core from all connected core dense grids, as shown in Figure 6.

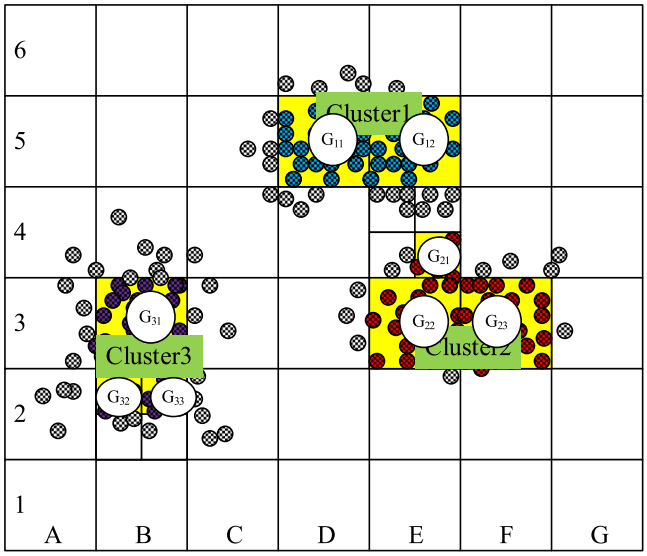


FIGURE 7 | Calculate the centroid of each core dense grid.

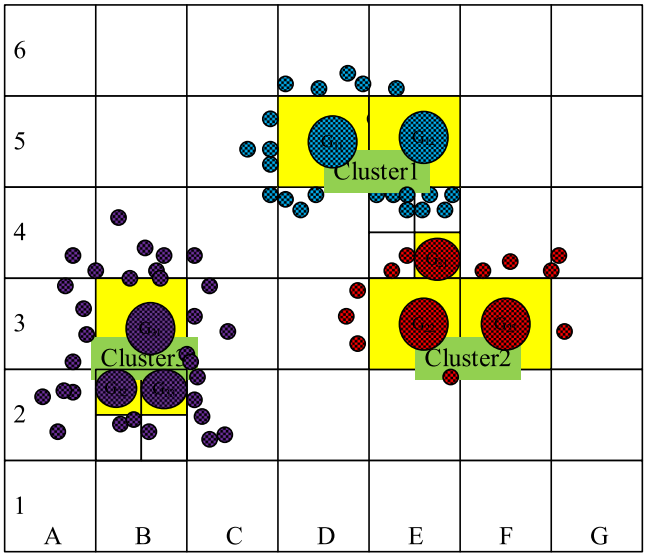


FIGURE 9 | Assign these data points to the nearest core dense grid.

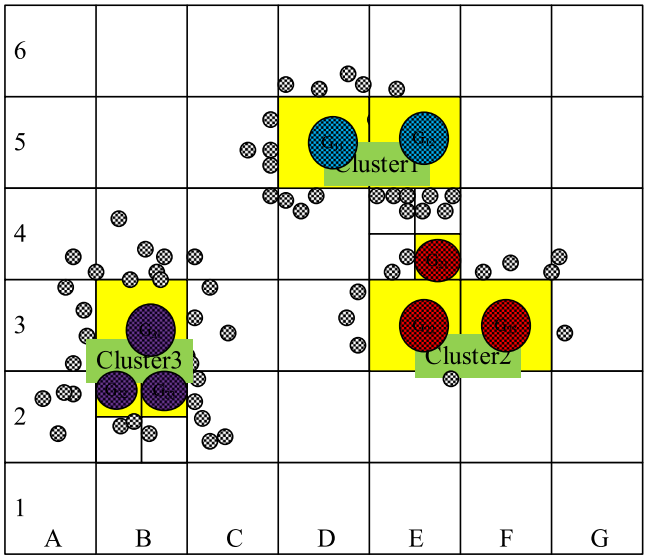


FIGURE 8 | Determine the initial centroids for K-means clustering.

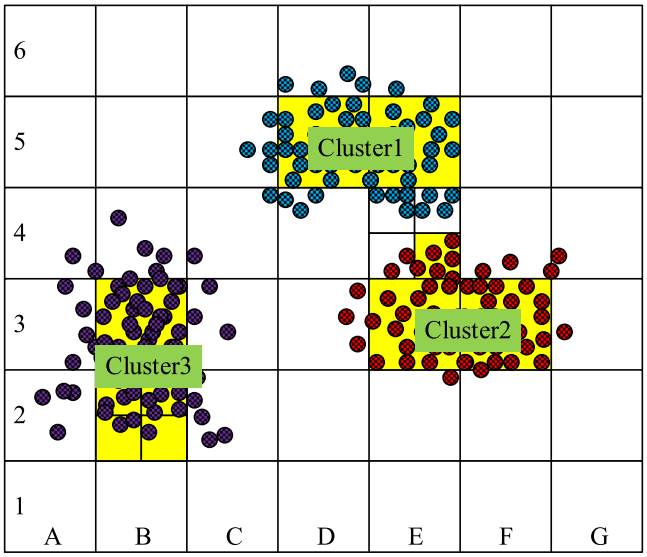


FIGURE 10 | Obtain the initial results of clustering.

6. Calculate the centroid of each core dense grid, as shown in Figure 7.

7. Use these centroids as the initial centroids for K-means clustering to cluster the data within other grids, as shown in Figure 8. Assign these data points to the nearest core dense grid, as shown in Figure 9.

8. Merge the edge data with the core dense grids to obtain the initial results of the hybrid clustering, as shown in Figure 10. Typically, the clustering process ends here.

9. If further reduction in the number of final clusters is needed, use the MPNN (M pairs of nearest neighbours) algorithm to merge clusters. As shown in Figure 11, merge cluster 1 and cluster 2 into a new cluster.

MPNN is a method for calculating adjacency relationships. It evaluates distances between different subsets of data. The calculation method for M is as follows:

Assume the dataset has dimension T, and datasets A and B are uniformly distributed. Each dimension of dataset A has an average of $\sqrt[T]{|\text{dataset A}|}$ samples. Each facet of dataset A has $\left\lfloor \sqrt[T-1]{|\text{dataset A}|} \right\rfloor$ samples. Therefore, there are M samples at the interface between dataset A and dataset B, calculated as shown in Equation (1).

$$M = \min\left(\left\lfloor \sqrt[T-1]{|\text{dataset } A|} \right\rfloor \middle| \left\lfloor \sqrt[T-1]{|\text{dataset } B|} \right\rfloor \right) \quad (1)$$

The calculation steps for MPNN are as follows:

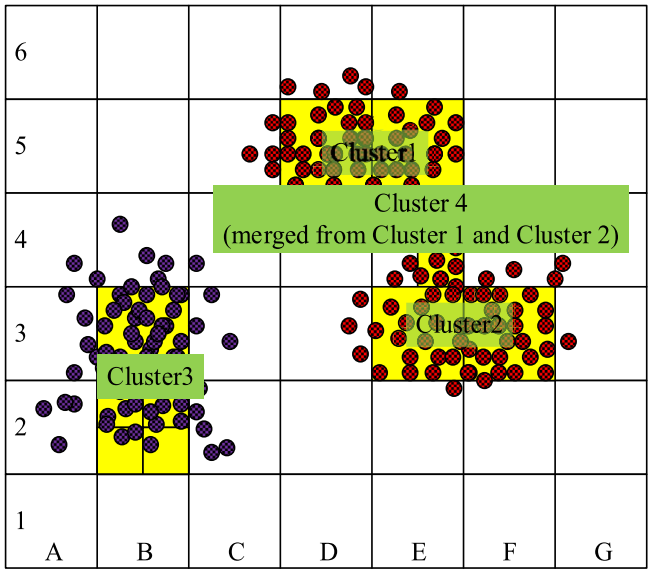


FIGURE 11 | Merge clusters to obtain the final result.

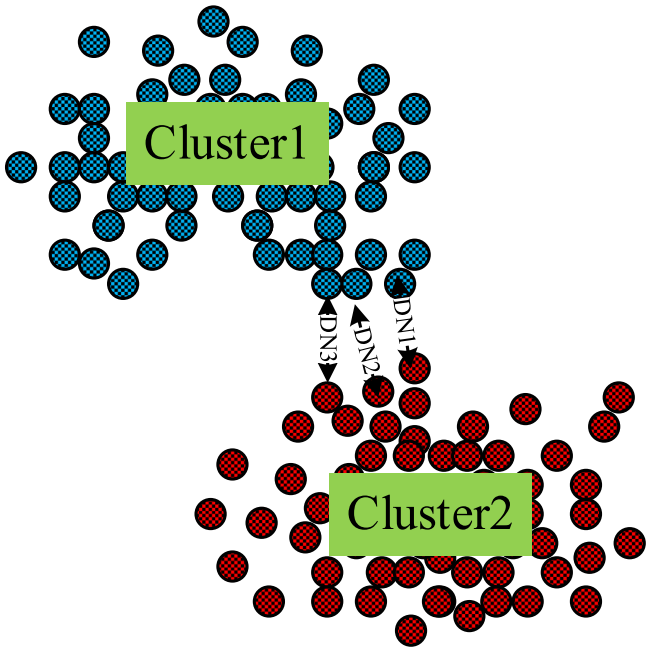


FIGURE 12 | MPNN (*M* pairs of nearest neighbours).

1. Calculate the distance between the two closest samples from two subsets of data, denoted as DN_1.
2. Remove these two closest samples from the subset.
3. Continue to calculate the distance between the two closest samples in the two subsets, denoted as DN_2.
4. Repeat steps 2 and 3 until you obtain the distance of the *M*th closest pair of samples, denoted as DN_m.
5. Calculate the weighted average of DN_1, DN_2, ..., DN_m. As shown in Figure 12.

3.2 | Automatically Calculate the Number of Grid Partitions *M* and the Density Threshold *R*

The calculation process for determining the number of grid partitions M and the density threshold R based on the original dataset DataSet and the final number of clusters K is as follows:

1. Project the dataset into the grid space, dividing each dimension into M partitions initially set to K. Calculate the distribution of data points within each grid, as shown in Table 1. The median density of all non-zero grids is denoted as Mdg(M). From Table 1, it is observed that Mdg(M) decreases as M increases.
2. Assume the minimum cluster of interest, denoted as $U_{\min}$, contains $|U_{\min}|$ samples. In this paper, $|U_{\min}|$ is determined by Equation (2):

$$|U_{\min}| = \frac{|\text{dataSet}|}{10 * k} \tag{2}$$

If there exists a value M_i such that $\text{Mdg}(M_i) < |U_{\min}|$, then set the initial value of M to M_i.

3. If $\text{Mdg}(M_i) < |U_{\min}|$ is not satisfied, increment the value of M by K until the condition is met.
4. The initial value of the density threshold R is determined by Equation (3).

$$R = \frac{\text{Mdg}(M_i)}{2} \tag{3}$$

3.3 | Criteria for Determining Core Dense Grids

Core dense grids constitute the main body of a cluster, judged primarily based on two conditions: the density and uniform distribution of samples within the grid. Density within a grid is determined by whether the number of samples exceeds R. The method to judge if samples are uniformly distributed involves calculating the deviation of the data mean from the grid centre, and measuring data dispersion.

Calculate the centroid of samples within grid j, as shown in Equation (4):

$$C_j = \frac{1}{|M_j|} \sum_{x_i \in M_j} x_i \tag{4}$$

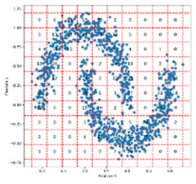
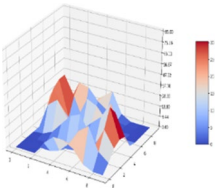
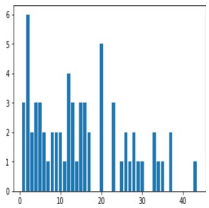
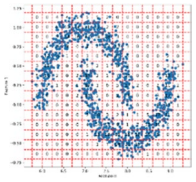
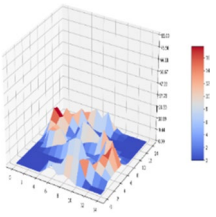
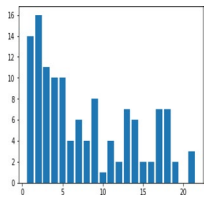
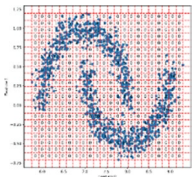
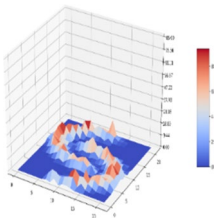
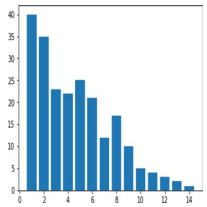
Next, calculate the centre point of grid j, as shown in Equation (5):

$$G_j = \frac{G_j^{MAX} - G_j^{MIN}}{2} \tag{5}$$

Then, compute the ratio of centroid deviation from the centre, as per Equation (6):

$$p_j = \frac{|G_j - C_j|}{G_j^{MAX} - G_j^{MIN}} \tag{6}$$

TABLE 1 | The distribution of grid data density under different values of *M*.

M	Project the data onto a grid	3D heatmap of grid densities	Statistical analysis of grid density
10			
15			
20			

Finally, calculate the data dispersion within grid *j*, as shown in Equation (7):

$$D_j = \frac{STP_{G_j}}{G_j^{MAX} - G_j^{MIN}} \tag{7}$$

In this paper, if $p_j < 0.1$ and $D_j < 0.5$, then grid *j* is considered a core dense grid.

3.4 | Core Algorithm

The core algorithm for improving grid clustering is as follows:

Firstly, evenly partition the sample space in each dimension into *M* parts. In each dimension, check if the number of samples within each grid exceeds the density threshold *R*. If so, the grid is classified as a one-dimensional dense grid unit.

Definition 6. One-dimensional dense grid unit (denseUnits1D^i) in dimension *i* includes attributes such as *i* (dimension), sort (order of the grid unit in that dimension), maxBin (lower boundary), minBin (upper boundary), and binPoints (samples contained within the grid).

Definition 7. N-dimensional dense grid unit (denseUnitsND) is the intersection of one-dimensional dense grid units from N different dimensions, as shown in Equation (8):

$$\text{denseUnitsND} = \bigcap_{i=1}^{n} \text{denseUnits1D}^i \tag{8}$$

Definition 8. N-dimensional core dense grid unit (coreUnitsND) is a dense grid unit (denseUnitsND) where data distribution is uniform across N dimensions.

Definition 9. $\text{distance}_{a,b}^i$ represents the distance between grids a and b in dimension *i*, calculated by Equation (9). Here, maxBin_a^i is the upper boundary line of grid a in dimension *i*, and minBin_a^i is the lower boundary line of grid a in dimension *i*.

$$\text{distance}_{a,b}^i = \left[\frac{\min\left(\left|\text{maxBin}_a^i - \text{maxBin}_b^i\right|, \left|\text{minBin}_a^i - \text{minBin}_b^i\right|\right)}{\min\left(\left|\text{maxBin}_a^i - \text{minBin}_a^i\right|, \left|\text{maxBin}_b^i - \text{minBin}_b^i\right|\right)} \right] \tag{9}$$

Definition 10. distance(a, b)_N represents the distance between *N*-dimensional dense grid units *a* and *b*, which is the sum of distances across all dimensions, as shown in Equation (10):

$$\text{distance(a, b)}_N = \sum_{i=1}^{N} \text{distance}_{a,b}^i \tag{10}$$

This value is used to determine if two dense grid units are adjacent; if the cumulative distance is less than 1, they are considered adjacent.

Definition 11. Grid clustering involves *N*-dimensional core dense grid units with adjacency, forming a cluster, as described in Algorithm 1.

Algorithm 1 denseUnitsToClusters

Input: denseUnitsND,dataset
// 'denseUnitsND' represents all N-dimensional dense grid units, and ' dataset' refers to the original dataset

Output: pred,c
// 'pred' represents the clustering labels for all samples, and 'c' represents the clustering label for dense grids.

```
L←len(denseUnitsND) // The number of dense grids
initialization C[0：L]，C[]← -1// Storing the final cluster numbers for each dense grid.
initialization EK← -1// Cluster ID.
initialization pred [0：len(dataset)]，pred []←-1// Store the clustering labels for all samples
for i:0 to L: // Calculate the clustering labels for all dense grids
    if C[i]== -1 then
        EK←EK+1, C[i]←EK
        initialization Current []//Used to store dense grids that are adjacent to the current dense grid
        for j:0 to L
            if C[j]== -1 and distance(i,j) <=1 then
                C[j]←EK
                add C[i] to Current
            for y in Current
                for s:0 to L
                    if C[s]== -1 and distance(i,j) <=1 then
                        C[s]←EK
                        add C[s] to Current
for i:0 to EK // Assign clustering labels to all samples in the dataset
    for j:0 to L
        if C[j]==i then
            pred[C[j]. binPoints] ←i
Return pred,c.
```

The flowchart of the proposed algorithm is shown in Figure 13.

3.5 | Mathematical Justification

As shown in Figure 14, assume the dataset is normalised, with each dimension having a length of 1. Each dimension is divided into M equal parts, so the initial length of each grid is $1/M$. After the dataset is projected onto the grid, the type of each grid is determined. If a grid is a core dense grid, no further processing is required (grids B3, D5, E3, E5 and F3 in the figure). If a grid is a non-core dense grid, it will be further split (grids B2 and E4 in the figure), and the type of the resulting sub-grids will be determined. Some sub-grids turn out to be core dense grids (sub-grids B2_1, B2_2 and E4_1 in the figure). Core dense grids are represented with yellow shading.

The core dense grids form a set: {B3, D5, E3, E5, F3, B2_1, B2_2, E4_1}, and the adjacency relationships between dense grids are then determined.

If two core dense grids a and b satisfy the equation distance$(a,b) \leq 1$, then a and b are considered adjacent.

From Equations (9) and (10), it follows that a and b must have $N-1$ identical dimensions in N-dimensional space, and the difference in the remaining dimension cannot exceed the length of one grid.

For example, to determine the adjacency relationship between grids B3 and D5:

$$\text{distance}(\text{B3}, \text{D5})_2 = \text{distance}^{0}_{\text{B3,D5}} + \text{distance}^{1.}_{\text{B3,D5}}$$

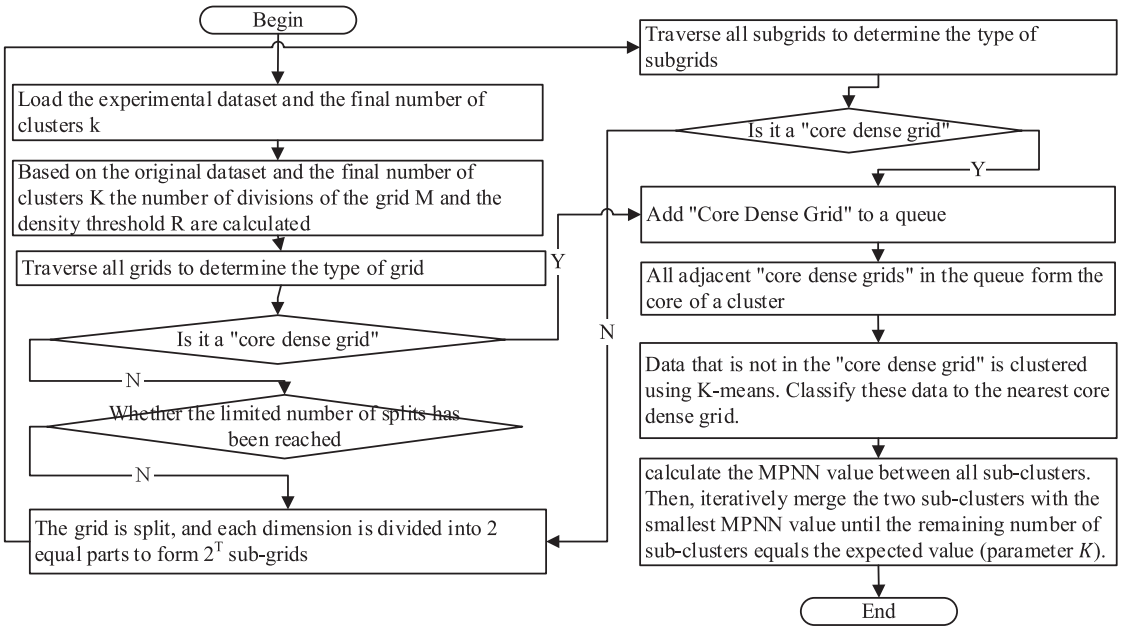


FIGURE 13 | The flowchart of the proposed algorithm.

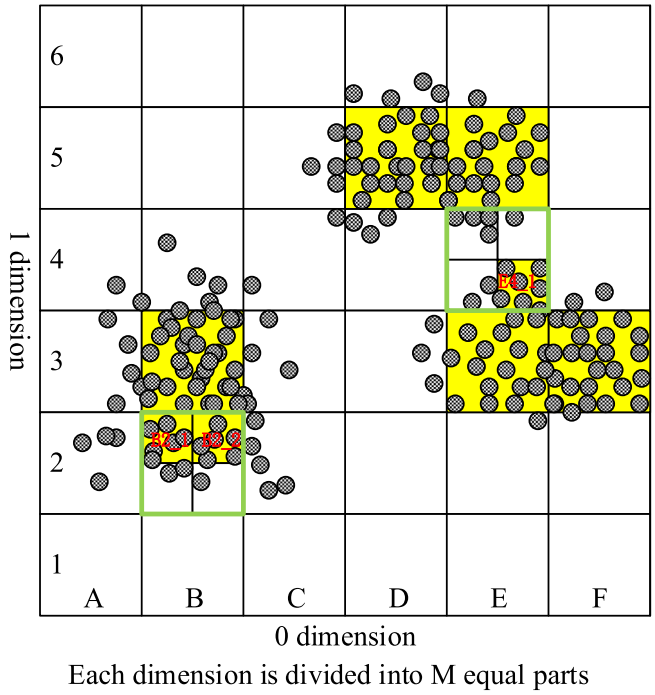


FIGURE 14 | Mathematical justification of the clustering process.

$$= \left[\frac{\min\left(\left|\mathrm{maxBin}^0_{B3} - \mathrm{maxBin}^0_{D5}\right|, \left|\mathrm{minBin}^0_{B3} - \mathrm{minBin}^0_{D5}\right|\right)}{\min\left(\left|\mathrm{maxBin}^0_{B3} - \mathrm{minBin}^0_{B3}\right|, \left|\mathrm{maxBin}^0_{D5} - \mathrm{minBin}^0_{D5}\right|\right)}\right] + \left[\frac{\min\left(\left|\mathrm{maxBin}^1_{B3} - \mathrm{maxBin}^1_{D5}\right|, \left|\mathrm{minBin}^1_{B3} - \mathrm{minBin}^1_{D5}\right|\right)}{\min\left(\left|\mathrm{maxBin}^1_{B3} - \mathrm{minBin}^1_{B3}\right|, \left|\mathrm{maxBin}^1_{D5} - \mathrm{minBin}^1_{D5}\right|\right)}\right]$$

$$= \left[\frac{\min\left(|-2/M|, |-2/M|\right)}{\min\left(|1/M|, |1/M|\right)}\right] + \left[\frac{\min\left(|-3/M|, |-3/M|\right)}{\min\left(|1/M|, |1/M|\right)}\right]$$

$$= 2 + 3 = 5$$

Therefore, B3 and D5 are not adjacent.

The process of determining the adjacency relationship between B3 and B2_1 is as follows:

$$\mathrm{distance}(B3, B2\_1)_2 = \mathrm{distance}^0_{B3,B2\_1} + \mathrm{distance}^1_{B3,B2\_1}$$

$$= \left[\frac{\min\left(\left|\mathrm{maxBin}^0_{B3} - \mathrm{maxBin}^0_{B2\_1}\right|, \left|\mathrm{minBin}^0_{B3} - \mathrm{minBin}^0_{B2\_1}\right|\right)}{\min\left(\left|\mathrm{maxBin}^0_{B3} - \mathrm{minBin}^0_{B3}\right|, \left|\mathrm{maxBin}^0_{B2\_1} - \mathrm{minBin}^0_{B2\_1}\right|\right)}\right] + \left[\frac{\min\left(\left|\mathrm{maxBin}^1_{B3} - \mathrm{maxBin}^1_{B2\_1}\right|, \left|\mathrm{minBin}^1_{B3} - \mathrm{minBin}^1_{B2\_1}\right|\right)}{\min\left(\left|\mathrm{maxBin}^1_{B3} - \mathrm{minBin}^1_{B3}\right|, \left|\mathrm{maxBin}^1_{B2\_1} - \mathrm{minBin}^1_{B2\_1}\right|\right)}\right]$$

$$= \left[\frac{\min\left(|1/2M|, |0|\right)}{\min\left(|1/M|, |1/2M|\right)}\right] + \left[\frac{\min\left(|1/M|, |1/2M|\right)}{\min\left(|1/M|, |1/2M|\right)}\right]$$

$$= 0 + 1 = 1$$

Therefore, B3 and B2_1 are adjacent.

The adjacency relationships between all dense grids form an adjacency matrix. Adjacency is represented by 1, and non-adjacency is represented by 0. As shown in Table 2.

Then, based on the adjacency matrix, the clustering labels for each core dense grid are obtained (see Algorithm 1). {B3, B2_1, B2_2} forms a sub-cluster with clustering label 0. {D5, E5} forms a sub-cluster with clustering label 1, and {E3, F3, E4_1} forms a sub-cluster with clustering label 2.

3.6 | Complexity Analysis

Assume that the size of the dataset is n. The first stage of the algorithm is to obtain the backbone of the class family using grid clustering. First, project the data to the grid with time

complexity $O(n)$. Then, determine the distribution of data within the grid with time complexity $O(n)$. Finally, clustering the core dense grid, the time complexity is $O(g)$; g is the number of core dense grids, which is much smaller than n. So, the first stage time complexity can be simplified to $O(n)$.

The second stage of the algorithm is to obtain the remainder of the class family using a division-based approach. The data within the core dense grid can be represented as $g*R$, so the time complexity is $O(n-g*R)$, which can be simplified to $O(n)$.

The third stage of the algorithm computes the merging between sub-clusters, each of which has n/k data on average. The time complexity is $O(k(n/k)^2)$ since it is necessary to calculate the distance between data points in two neighbouring clusters. So in the worst case, the time complexity of the algorithm is $O(n^2)$.

So the time complexity of the whole algorithm is $O(n^2)$.

4 | Experiment

4.1 | Experimental Environment

The experimental computer configuration includes an Intel Core i3-9100F CPU at 3.6 GHz, 16 GB of RAM, and a 4 TB hard drive. The operating system used is Windows 10, and the algorithms are implemented in Python 3.7.

The algorithms used for comparison in the experiment are Hierarchical Clustering, DBSCAN (Density-Based Spatial Clustering of Applications with Noise), DPC (Density Peak Clustering), K-Means, Clique, GDB(Grid-based DBSCAN) and HDBSCAN.

HDBSCAN—Hierarchical Density-Based Spatial Clustering of Applications with Noise. Performs DBSCAN over varying epsilon values and integrates the result to find a clustering that gives the best stability over epsilon. This allows HDBSCAN to find clusters of varying densities (unlike DBSCAN) and be more robust to parameter selection (Bot et al. 2023).

TABLE 2 | The adjacency relationships between all dense grids form an adjacency matrix.

	B3	D5	E3	E5	F3	B2_1	B2_2	E4_1
B3	1	0	0	0	0	1	1	0
D5	0	1	0	1	0	0	0	0
E3	0	0	1	0	1	0	0	1
E5	0	1	0	1	0	0	0	0
F3	0	0	1	0	1	0	0	0
B2_1	1	0	0	0	0	1	0	0
B2_2	1	0	0	0	0	0	1	0
E4_1	0	0	1	0	0	0	0	1

GDB is a clustering algorithm based on grid and density. It first uses grid techniques to perform an initial partitioning of the dataset, followed by improved DBSCAN for further clustering of the dataset.

The parameter settings for each algorithm are as follows:

1. The proposed hybrid clustering algorithm HCA-BGP (Hybrid Clustering Algorithm Based on Grid and Partitioning) has the parameter of the final number of clusters K.
2. The Clique algorithm's parameters are the grid partition count M and density threshold R.
3. The K-Means algorithm's parameter is the final number of clusters K.
4. Hierarchical Clustering's parameters are the final number of clusters K, with the clustering method selected as 'average'.
5. DBSCAN algorithm's parameters are the truncation distance *eps* and the minimum density *MinPts*.
6. DPC algorithm's parameters are the truncation distance *distPercen*t and the final number of clusters K.
7. The parameter of the HDBSCAN algorithm is min_cluster_size, the number of samples included in the minimum cluster.
8. The GDB algorithm's parameters are the grid partition count M and the minimum density threshold R.

4.2 | Experimental Results and Analysis on Simulated Data Sets

Artificially simulated datasets Aggregation, Flame, R15 and S1 were selected. See Table 3 for the attributes of each dataset.

The evaluation metrics for clustering results of each algorithm are shown in Table 4, including F-score and Adjusted Rand Index (ARI). For K-Means, results are averaged over 100 runs.

The three-dimensional bar chart corresponding to Table 4 is shown in Figure 15.

From the data in Table 4, we can see that the hybrid clustering algorithm proposed in this paper, HCA-BGP, achieves the best clustering performance indicators on the experimental datasets. Compared to the classical CLIQUE algorithm, the F-score of our algorithm improves by 20.3% on the S1 dataset, by 81.8% on the R15 dataset, and by 23.6% on the Jain dataset. Compared to the

TABLE 3 | Artificial simulation data set used in experiment.

Data set	Number of samples	Number of clusters
Aggregation	700	7
R15	600	15
S1	5000	15
Jain	373	2

GDB algorithm, the *F*-score of our algorithm improves by 19.5% on the S1 dataset, by 33.3% on the R15 dataset, and by 18.3% on the Jain dataset.

Using our algorithm, which combines grid-based partitioning with clustering based on non-dense grid data, achieves robust clustering results. The clustering process for the S1 dataset involves three main steps:

Firstly, the dataset is projected onto a grid, where grids are split based on the distribution of data within them to identify core dense grids. These core dense grids form the central part of each cluster (coloured regions in the figure), as shown in Figure 16.

Next, the remaining data is assigned to the clusters generated in the previous step using partition-based methods, obtaining the peripheral parts of each cluster. Through these two steps, all

TABLE 4 | Comparison of clustering evaluation metrics for artificial datasets across various algorithms.

	Aggregation		Jain		R15		S1	
Algorithm	***F***	**ARI**	***F***	**ARI**	***F***	**ARI**	***F***	**ARI**
HCA-BGP	**0.995**	**0.991**	**0.974**	**0.932**	**0.993**	**0.993**	**0.989**	**0.988**
	$K=7$		$K=2$		$K=15$		$K=15$	
Clique	0.812	0.762	0.788	0.220	0.546	0.773	0.822	0.810
	$M=13, R=5$		$M=10, R=5$		$M=20, R=5$		$M=22, R=40$	
GDB	0.761	0.699	0.823	0.463	0.7446	0.7226	0.827	0.811
	$M=20, R=5$		$M=10, R=6$		$M=10, R=2$		$M=33, R=3$	
K-Means	0.798	0.742	0.808	0.510	0.976	0.984	0.987	0.986
	$K=7$		$K=2$		$K=15$		$K=15$	
Hierarchical	0.849	0.807	0.790	0.514	0.989	0.989	0.984	0.983
	$n=7$		$n=2$		$n=15$		$n=15$	
DBSCAN	0.985	0.981	0.965	0.912	0.939	0.934	0.941	0.937
	eps = 0.12, MinPts = 5		eps = 0.3, MinPts = 1		ps = 0.03, MinPts = 3		eps = 0.09, MinPts = 20	
DPC	0.789	0.731	0.850	0.595	0.978	0.985	0.988	0.987
	$n=7$,distPercent = 3		$n=2$,distPercent = 3		$n=15$,distPercent = 3		$n=15$,distPercent = 3	
HDBSCAN	0.879	0.840	0.960	0.901	0.927	0.922	0.935	0.931
	min_cluster_size = 6		min_cluster_size = 5		min_cluster_size = 6		min_cluster_size = 9	

Note: Bold values represent the optimal value.

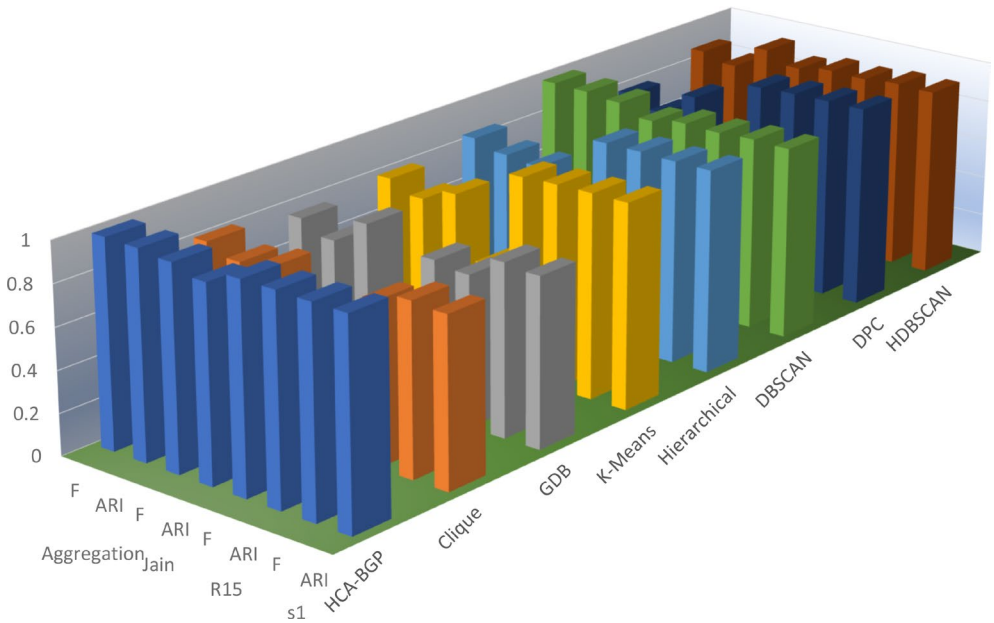


FIGURE 15 | Comparison of clustering evaluation metrics for artificial datasets across various algorithms.

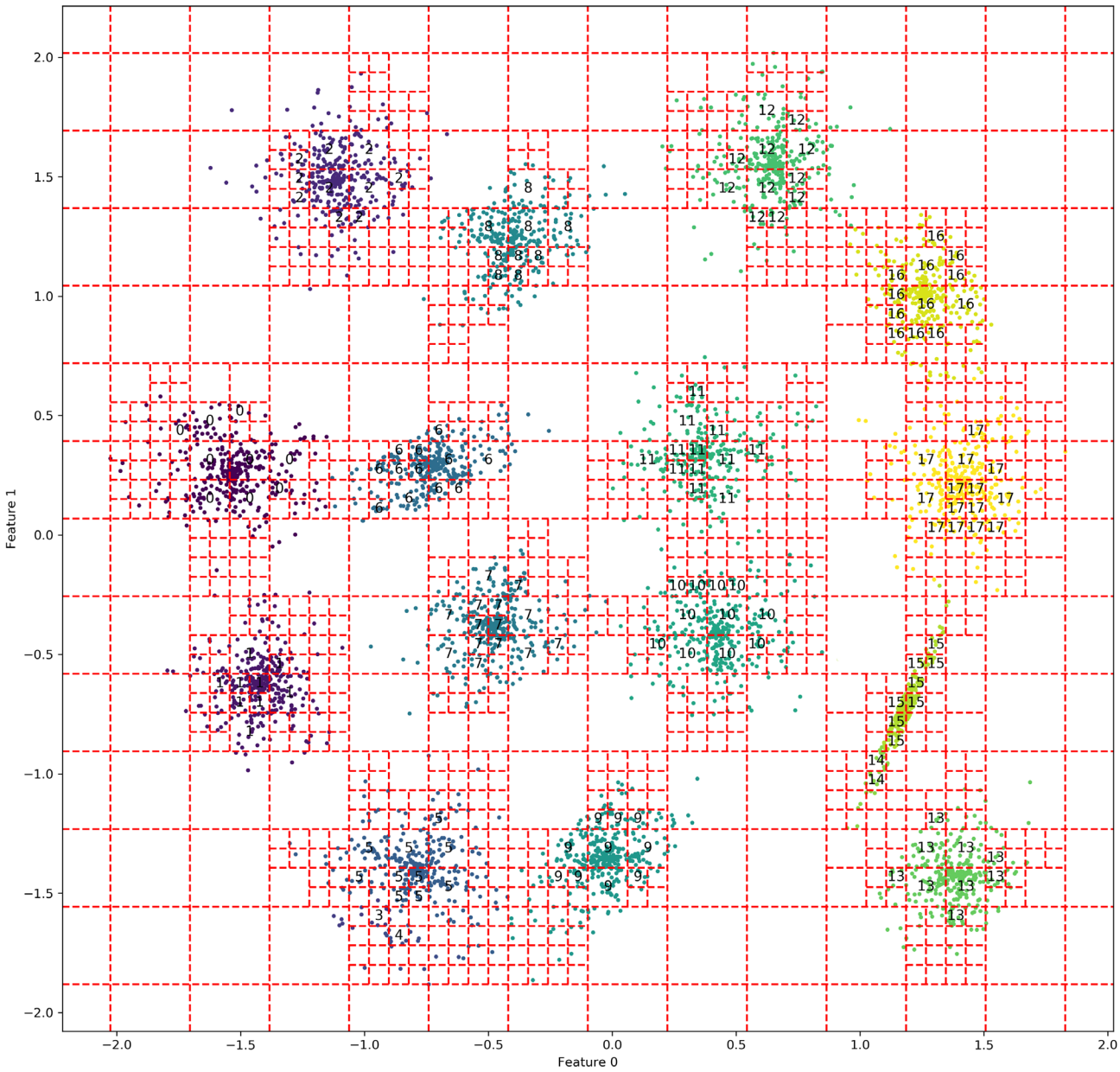


FIGURE 16 | The clustering process of our algorithm for the S1 dataset, Step 1—obtaining core dense grids, where core dense grids form the central components of each cluster.

data points receive cluster labels. Subsequently, the adjacency relationships between all clusters are computed using the M pairs of nearest neighbours (MPNN), as shown in Figure 17.

Finally, based on the number of clusters K and the adjacency relationships between clusters (MPNN), clusters are merged to obtain the final clustering results, as shown in Figure 18.

4.3 | Experimental Results and Analysis on Real Data

The real datasets used include urbanGB, IRIS, Abalone, Segmentation and Wine, provided by UCI.

The urbanGB dataset contains the geographic coordinates of traffic accident locations in various cities across England, covering data from a total of 497 cities. For experimental comparison purposes, eight datasets were selected from these 497 cities, each containing data from adjacent multiple cities, as shown in Figure 19. In the figures, each bubble represents the municipal coordinates of a city, and coloured dots represent the geographical coordinates where traffic accidents occurred in each city.

First, the geographical latitude and longitude coordinates from each dataset were converted into coordinate data (Maddah et al. 2021). Subsequently, each algorithm was used for clustering. The clustering results for Data1, Data2, Data3 and Data4 are shown in Figure 20.

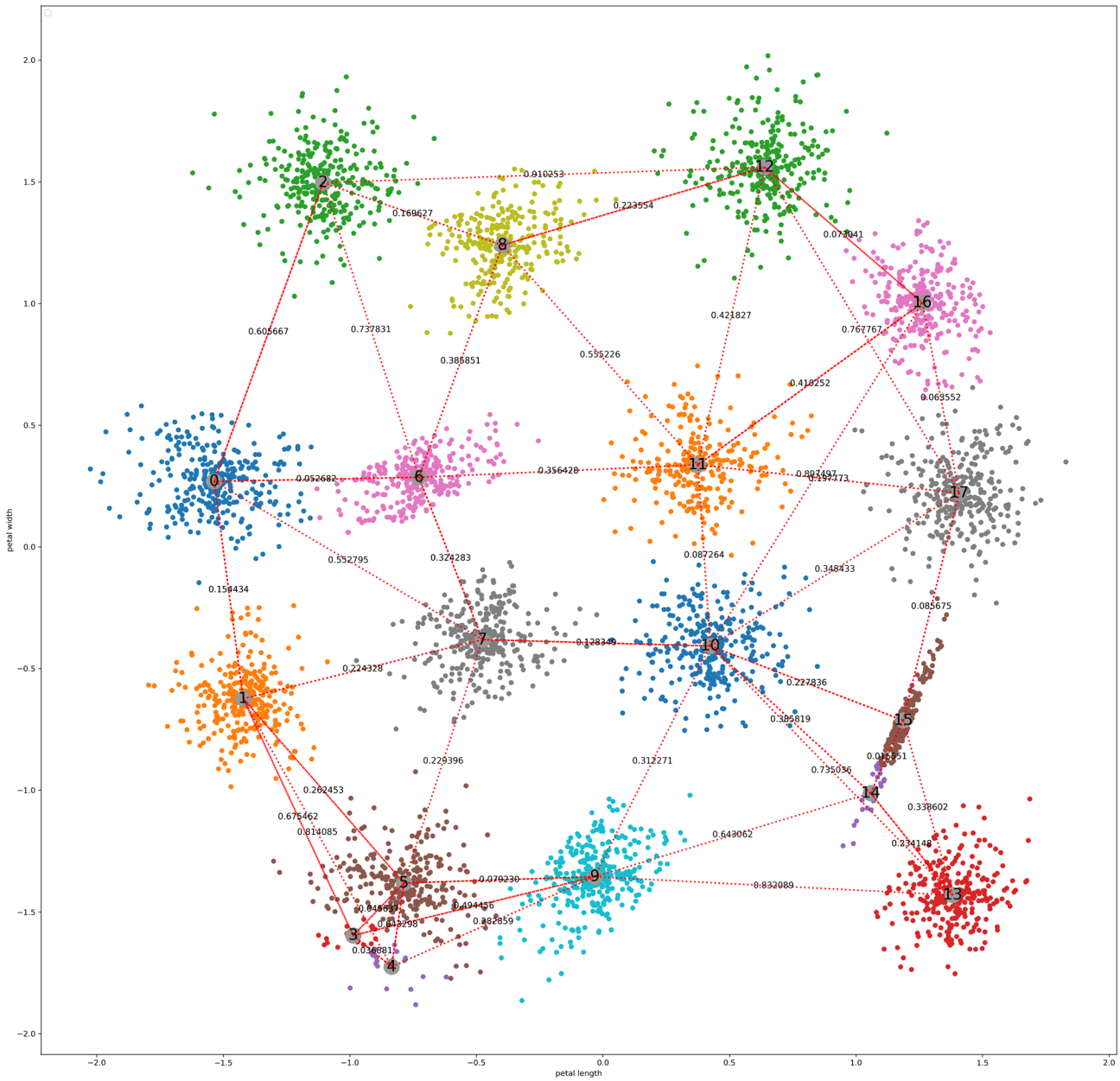


FIGURE 17 | The clustering process of our algorithm for the S1 dataset, Step 2—obtaining the peripheral parts of each cluster using partition-based methods, followed by computing the adjacency relationships between all clusters.

As shown in Figure 20, the clustering results of the HCA-BGP algorithm are more accurate and can better handle datasets with complex shapes, such as those in data1 and data2.

The clustering results for Data5, Data6, Data7 and Data8 are shown in Figure 21.

As shown in Figure 21, for datasets with relatively simple shapes, the HCA-BGP algorithm achieves the highest clustering accuracy. Other grid-based algorithms, CLIQUE and GDB, produce errors due to the density threshold, which lowers their clustering accuracy.

The evaluation metrics for the final clustering results of all eight datasets are shown in Table 5. The evaluation metrics include: Adjusted Rand Index (ARI), Normalised Mutual Information (NMI), *F*-score and V-Measure. For K-Means, the results are averaged over 100 runs.

The 3D bar chart corresponding to Table 5 is shown in Figures 22 and 23.

From Table 5, it can be seen that our algorithm has better clustering accuracy and stability, and can adapt well to data of various shapes.

Across the 8 datasets, comparing the average values of our algorithm with the other 6 algorithms, ARI increased by 94.3%, NMI increased by 19.7%, *F*-score increased by 25.1%, and V-Measure increased by 19.6%. Compared to the classic

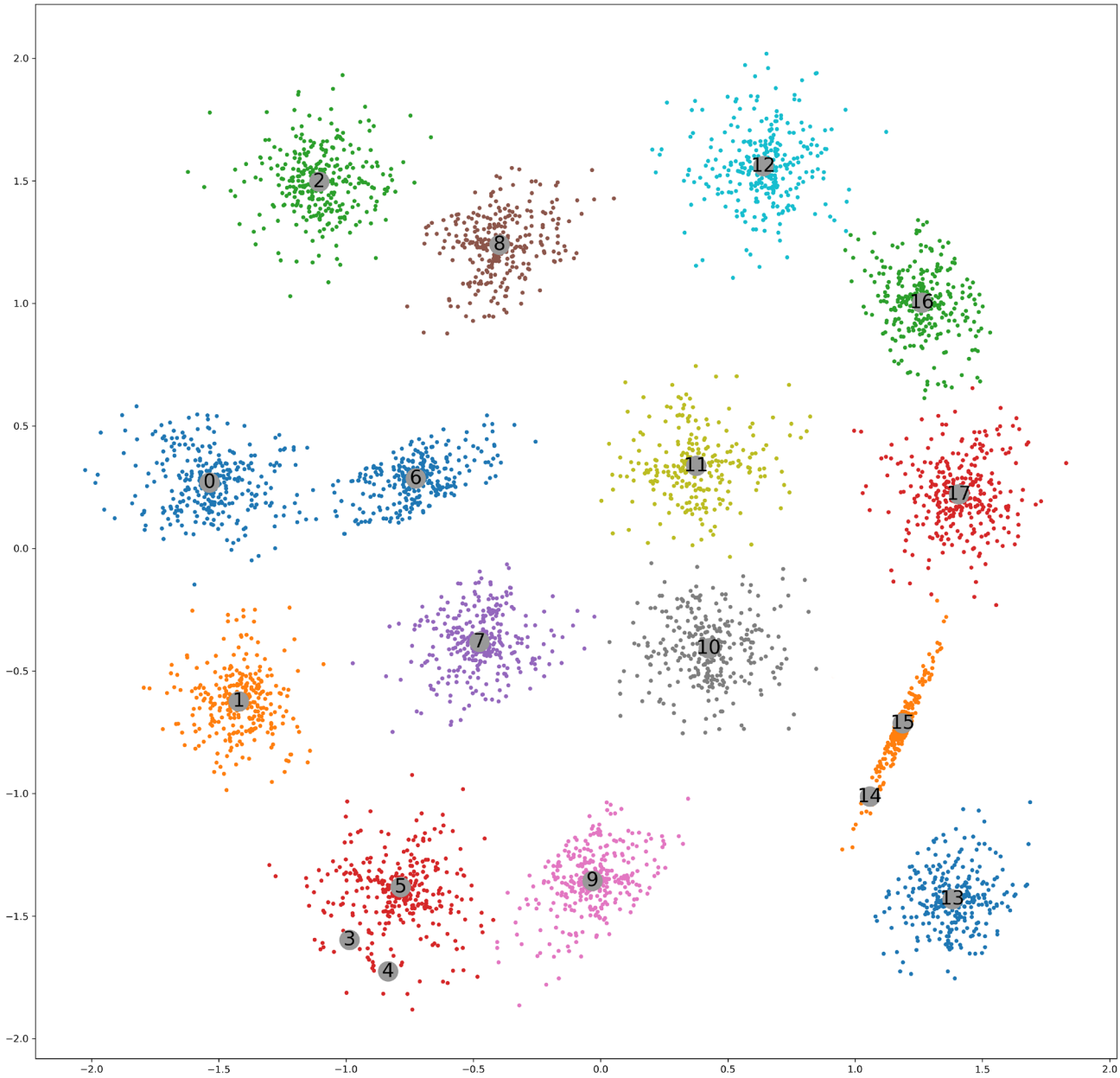


FIGURE 18 | The clustering process of our algorithm for the S1 dataset, Step 3—merging clusters based on the number of clusters K and the adjacency relationships between clusters to obtain the final clustering results.

CLIQUE algorithm, our algorithm improved ARI by 32.5%, NMI by 22.6%, *F*-score by 7.6%, and V-Measure by 22.4%. Compared to the GDB algorithm, our algorithm improved ARI by25.8%, NMI by 14.1%, *F*-score by 7.9%, and V-Measure by 14.1%.

The attributes of the IRIS, Abalone, Segmentation and Wine datasets are shown in Table 6.

The clustering results of these four datasets using seven algorithms are shown in Table 7.

The 3D bar chart corresponding to Table 7 is shown in Figure 24.

As shown in Table 6 and Figure 24, the proposed algorithm performs well on the four datasets with different dimensions (IRIS, Abalone, Segmentation and Wine) compared to the other six algorithms, especially when compared to the grid-based clustering algorithms CLIQUE and GDB.

Across the four datasets, comparing the average values of our algorithm with the other six algorithms, ARI increased by 30.2%, NMI increased by 18.3%, *F*-score increased by 14.5%, and V-Measure increased by 19.2%. Compared to the classic CLIQUE algorithm, our algorithm improved ARI by 37.8%, NMI by 28.1%, *F*-score by 24.3%, and V-Measure by 30.2%. Compared to the GDB algorithm, our algorithm improved ARI by 32.8%, NMI by 23.6%, *F*-score by 16.8%, and V-Measure by 23.7%.

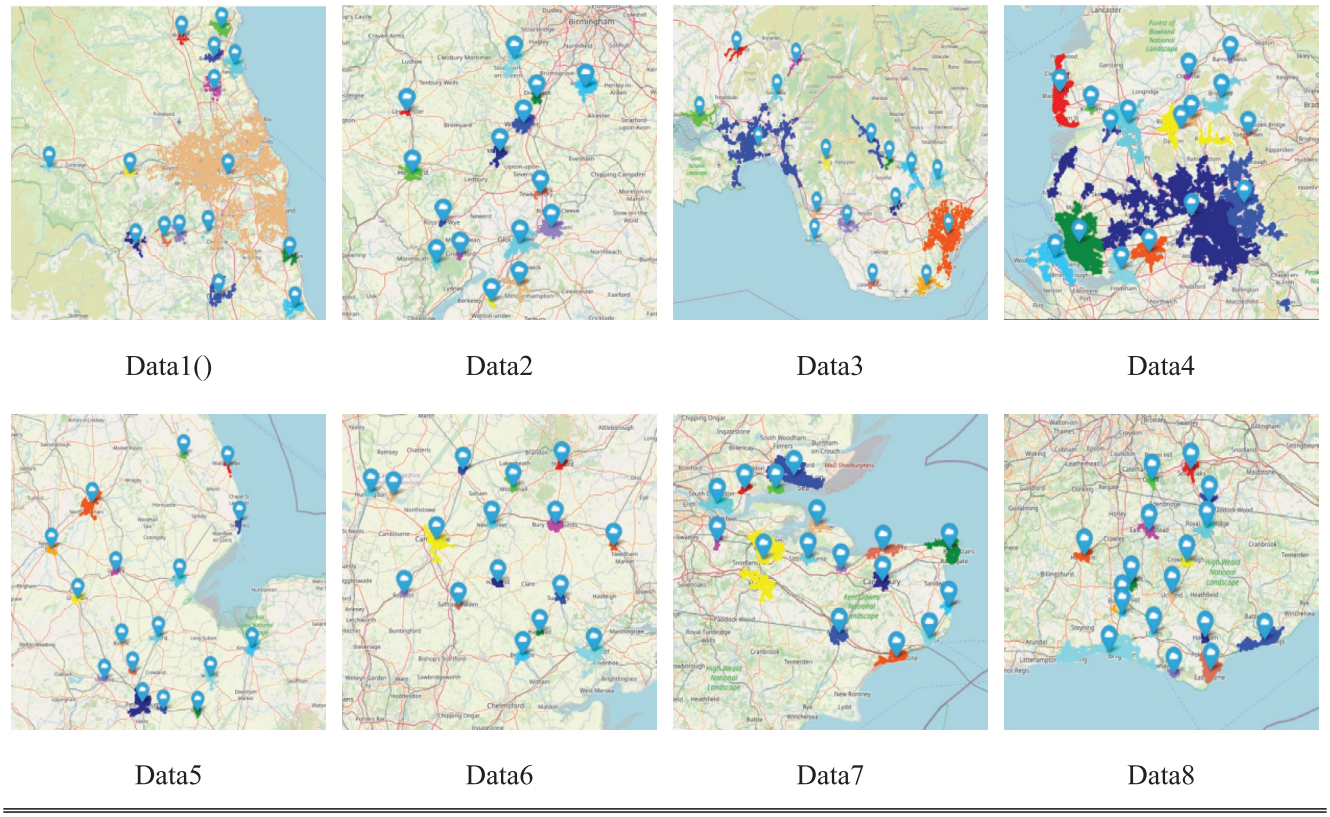


FIGURE 19 | Eight experimental datasets generated from the urbanGB provided by UCI.

5 | Discussion—The Impact of Parameters *M* and *R* on Clustering Results

The proposed Hybrid Clustering Algorithm Based on Grid and Partitioning (HCA-BGP) can automatically calculate the number of grid partitions (M) and the density threshold (R). Because it can automatically split grids based on data distribution, it is less sensitive to parameters M and R compared to the classic grid clustering algorithm CLIQUE, and it has better robustness.

Figure 25 shows the clustering results of the S1 dataset for HCA-BGP, CLIQUE and GDB under different M and R parameters. Table 8 shows the F-score of the clustering results.

As shown in Figure 25, regardless of how M and R vary, the clustering results of the HCA-BGP algorithm remain good and stable. However, the clustering results of the CLIQUE and GDB algorithms vary significantly.

The 3D bar chart corresponding to Table 8 is shown in Figure 26.

As shown in Table 8 and Figure 26, The average F-score of the proposed algorithm is 0.9676, compared to 0.38 for the CLIQUE algorithm and 0.399 for the GDB algorithm. This indicates that the proposed algorithm achieves significantly higher clustering accuracy than both the CLIQUE and GDB algorithms.

The variance of the F-score for the proposed algorithm is 0.000861351, compared to 0.013785878 for the CLIQUE algorithm and 0.011749296 for the GDB algorithm. The variance of the proposed algorithm is 6.2% of that of CLIQUE and 7.3% of that of GDB, demonstrating its superior stability.

Statistical Test Results for Clustering Metrics on the Simulated Dataset 1:

T-test results between the proposed algorithm HCA-BGP and the CLIQUE algorithm:

t-value = 25.68, p-value = 3.71 e-22.

T-test results between the proposed algorithm HCA-BGP and the GDB algorithm:

t-value = 26.79, p-value = 5.70 e-23.

These results indicate significant differences in the clustering metrics between the HCA-BGP algorithm and the other two algorithms.

F-test results between HCA-BGP and CLIQUE:

F-value between the two algorithms = 586.25, F-value across different R and M combinations = 0.77.

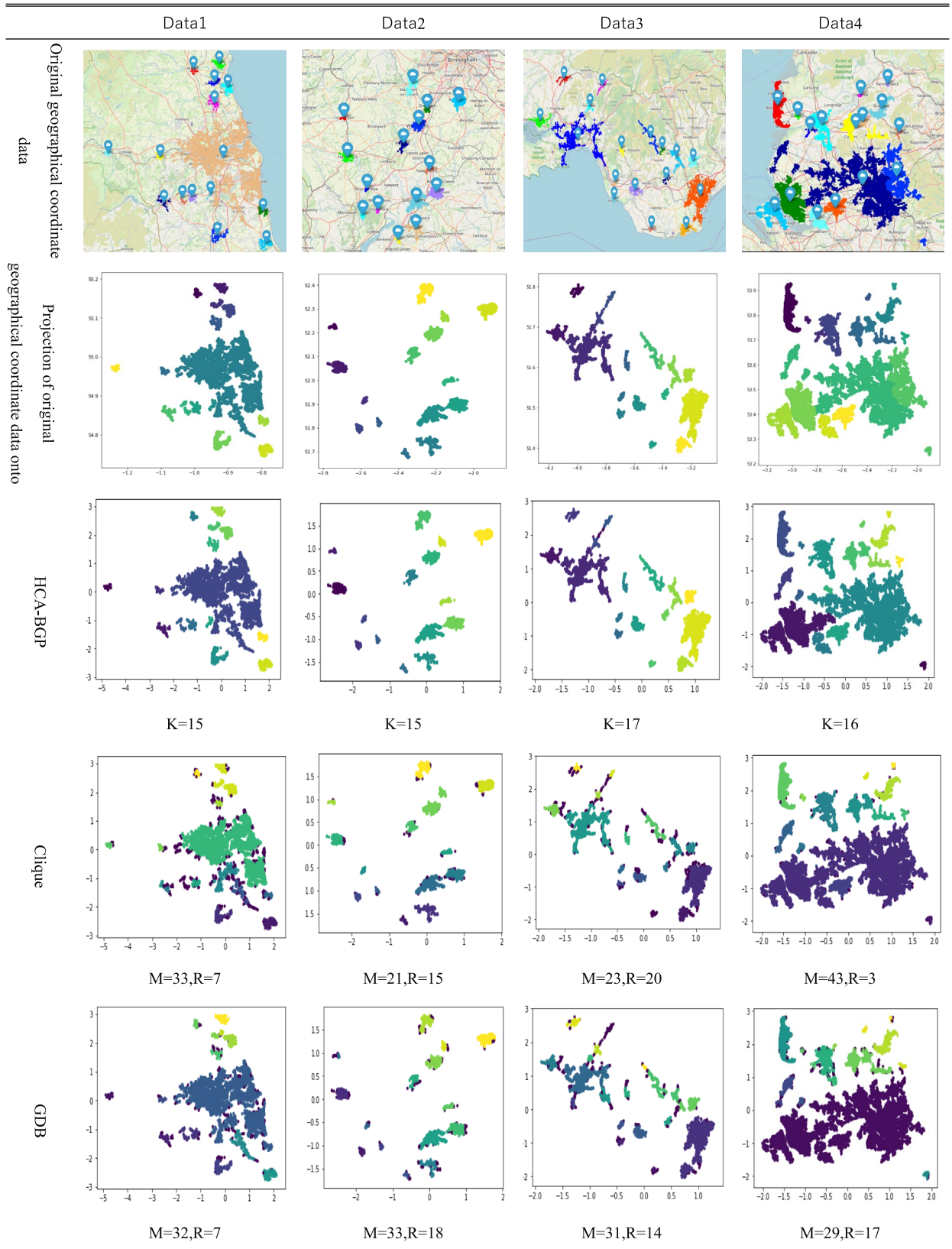


FIGURE 20 | Legend on next page.

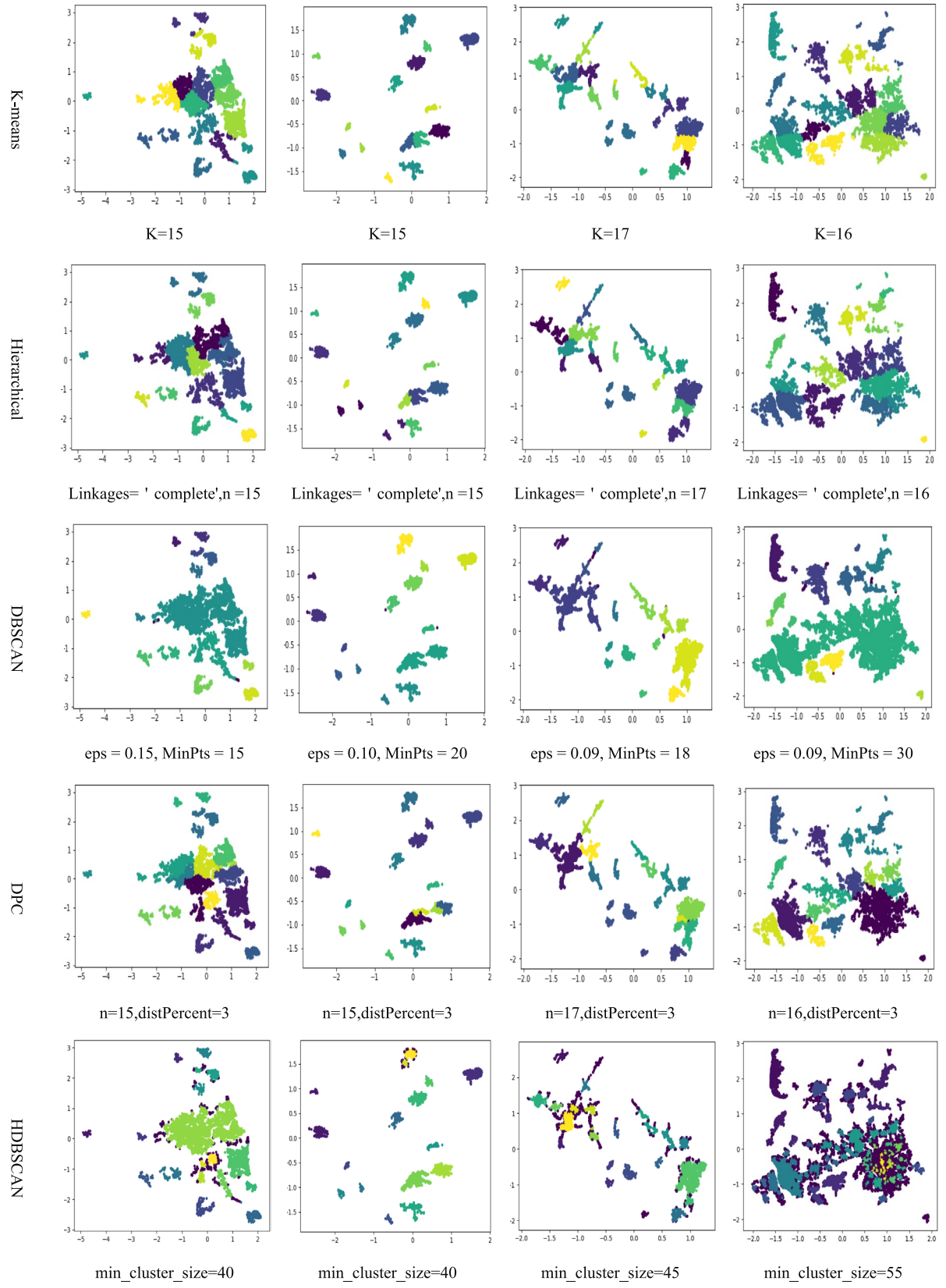


FIGURE 20 | Clustering results of various algorithms for Data1, Data2, Data3 and Data4.

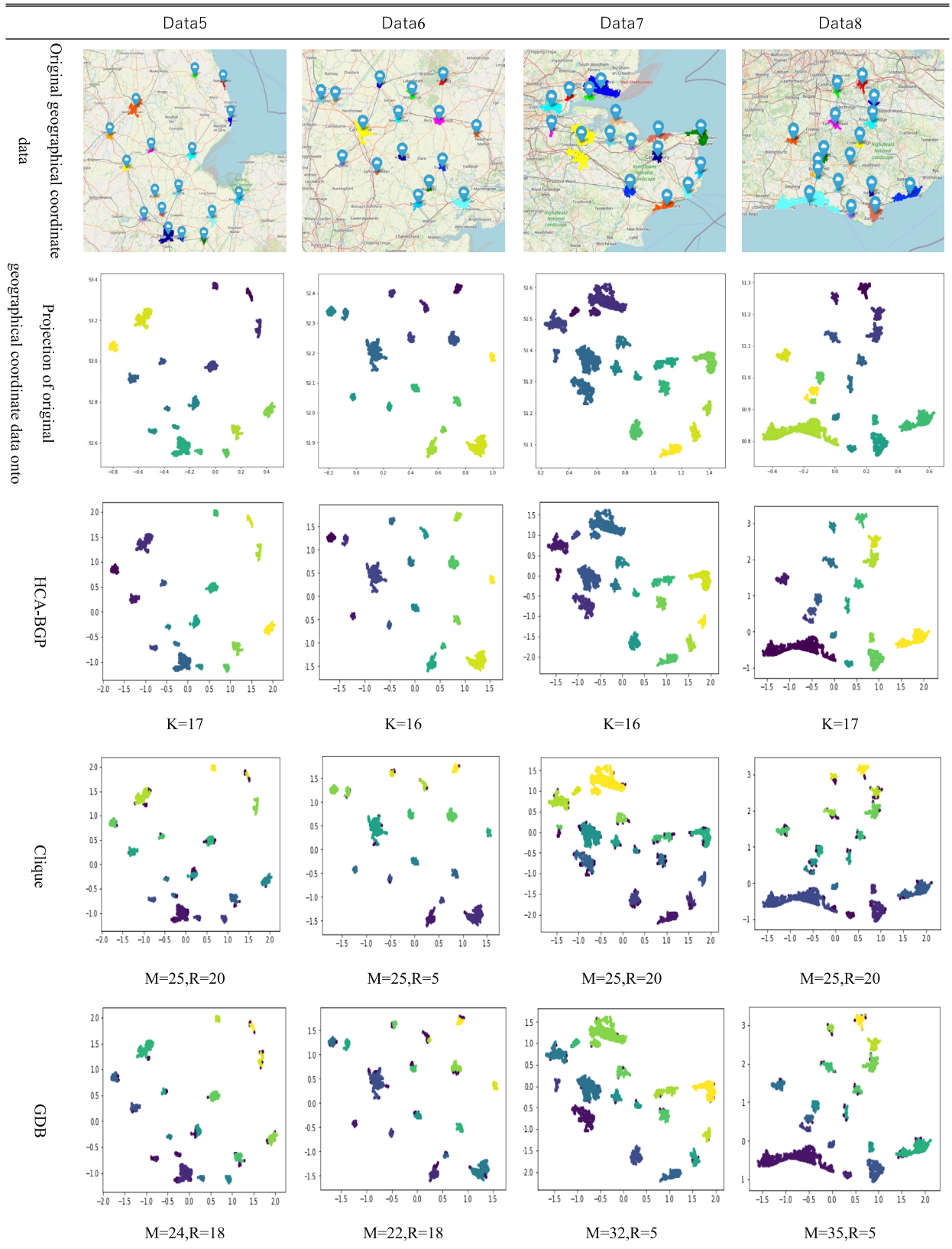


FIGURE 21 | Legend on next page.

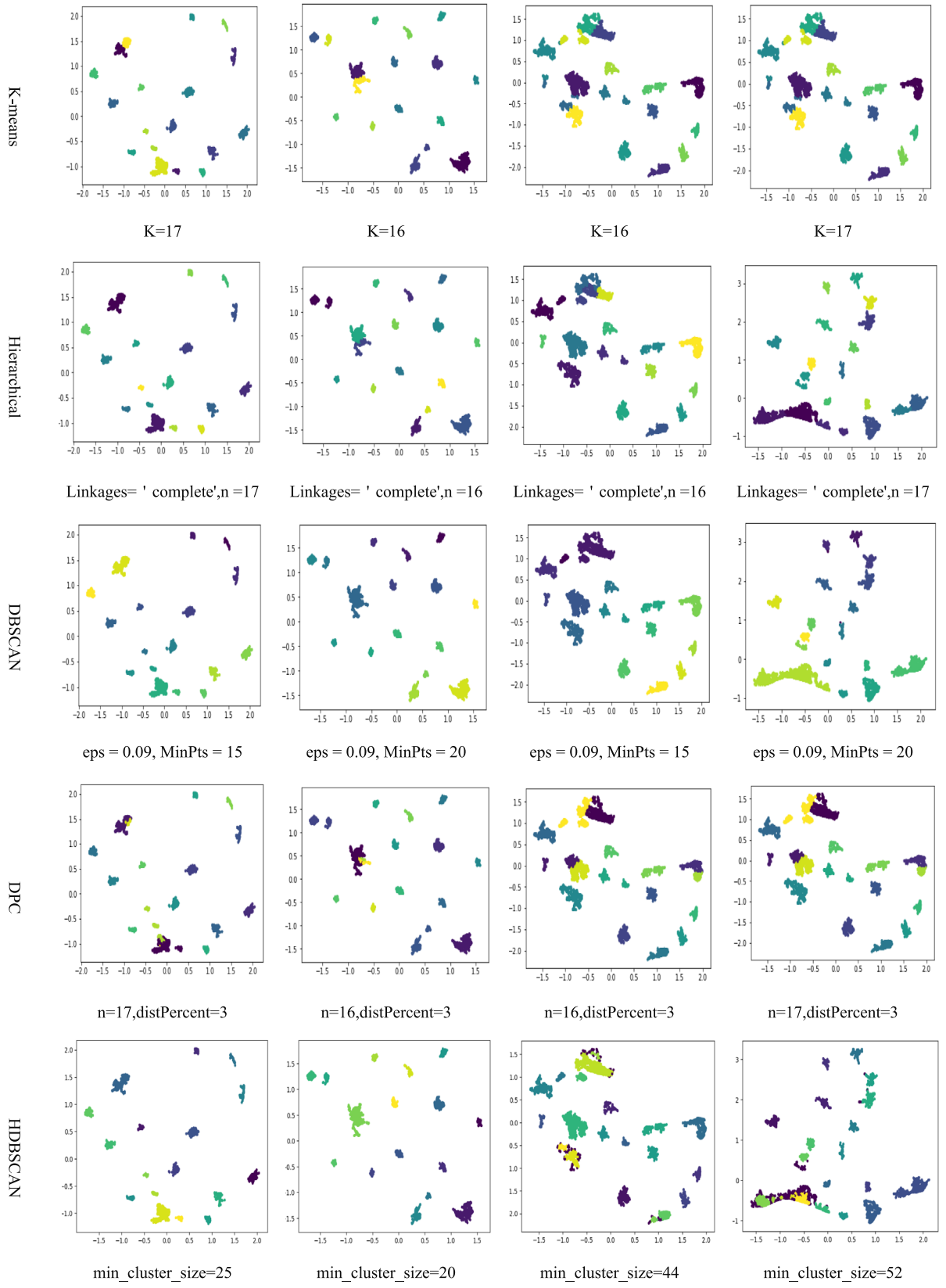


FIGURE 21 | Clustering results of various algorithms for Data5, Data6, Data7 and Data8.

TABLE 5 | Clustering evaluation metrics for various algorithms on 8 real geographical datasets.

Datasets		HCA-BGP	clique	GDB	K-Means	Hierarchical	DBSCAN	DPC	HDBSCAN
Data1	ARI	**1.000**	0.5193	0.8146	0.0838	0.111	0.7911	0.0773	0.4405
	NMI	**1.000**	0.5208	0.8137	0.424	0.4735	0.8674	0.4223	0.6576
	F	**1.000**	0.894	0.9421	0.3739	0.4222	0.9305	0.3705	0.7573
	V-Measure	**1.000**	0.5243	0.8159	0.4279	0.4772	0.8688	0.4262	0.6615
Data2	ARI	**1.000**	0.9571	0.9442	0.8762	0.8645	0.7242	0.8583	0.9719
	NMI	**1.000**	0.9433	0.9234	0.9628	0.9532	0.9319	0.9413	0.9755
	F	**1.000**	0.9629	0.9517	0.896	0.8864	0.7903	0.8788	0.9757
	V-Measure	**1.000**	0.944	0.9243	0.9631	0.9537	0.9327	0.9419	0.9758
Data3	ARI	**0.9971**	0.9244	0.7989	0.4884	0.5012	0.9917	0.5559	0.6537
	NMI	**0.9861**	0.859	0.8066	0.7498	0.7565	0.9735	0.7873	0.7621
	F	**0.9978**	0.9438	0.8503	0.6198	0.6275	0.9939	0.6728	0.742
	V-Measure	**0.9863**	0.8602	0.8085	0.7516	0.7586	0.9737	0.7891	0.765
Data4	ARI	**0.6142**	0.2564	0.2503	0.3532	0.3220	0.3436	0.5078	0.2685
	NMI	**0.7889**	0.5435	0.5451	0.7321	0.6923	0.6736	0.7742	0.6083
	F	**0.7257**	0.5691	0.5660	0.5043	0.4687	0.6111	0.6187	0.4082
	V-Measure	**0.7893**	0.5443	0.5458	0.7335	0.6939	0.6741	0.7755	0.6102
Data5	ARI	**1.000**	0.9625	0.9691	0.9358	1.000	1.000	0.7952	1.000
	NMI	**1.000**	0.9513	0.9509	0.9709	1.000	1.000	0.9241	1.000
	F	**1.000**	0.9683	0.9738	0.9463	1.000	1.000	0.8273	1.000
	V-Measure	**1.000**	0.9519	0.9515	0.9712	1.000	1.000	0.9249	1.000
Data6	ARI	**1.000**	0.9656	0.9631	0.834	0.8694	1.000	0.8693	1.000
	NMI	**1.000**	0.9618	0.9388	0.9478	0.8694	1.000	0.9367	1.000
	F	**1.000**	0.9723	0.9703	0.8697	0.8694	1.000	0.8958	1.000
	V-Measure	**1.000**	0.9624	0.9397	0.9485	0.9502	1.000	0.9376	1.000
Data7	ARI	**0.8246**	0.7988	0.8199	0.6865	0.6427	0.8246	0.6442	0.7984
	NMI	**0.9445**	0.9016	0.9336	0.8931	0.8858	0.9445	0.8747	0.9187
	F	**0.8516**	0.8299	0.8477	0.7479	0.7084	0.8516	0.7101	0.8366
	V-Measure	**0.9448**	0.9022	0.9339	0.8937	0.8864	0.9448	0.8754	0.9193
Data8	ARI	**0.9997**	0.9504	0.9938	0.6865	0.6473	0.9997	0.6442	0.4669
	NMI	**0.9977**	0.9134	0.9744	0.8931	0.8744	0.9959	0.8747	0.8149
	F	**0.9998**	0.9654	0.9957	0.7479	0.7536	0.9998	0.7101	0.6191
	V-Measure	**0.9977**	0.9144	0.9746	0.8937	0.7536	0.9959	0.8754	0.8169

Note: Bold values represent the optimal value.

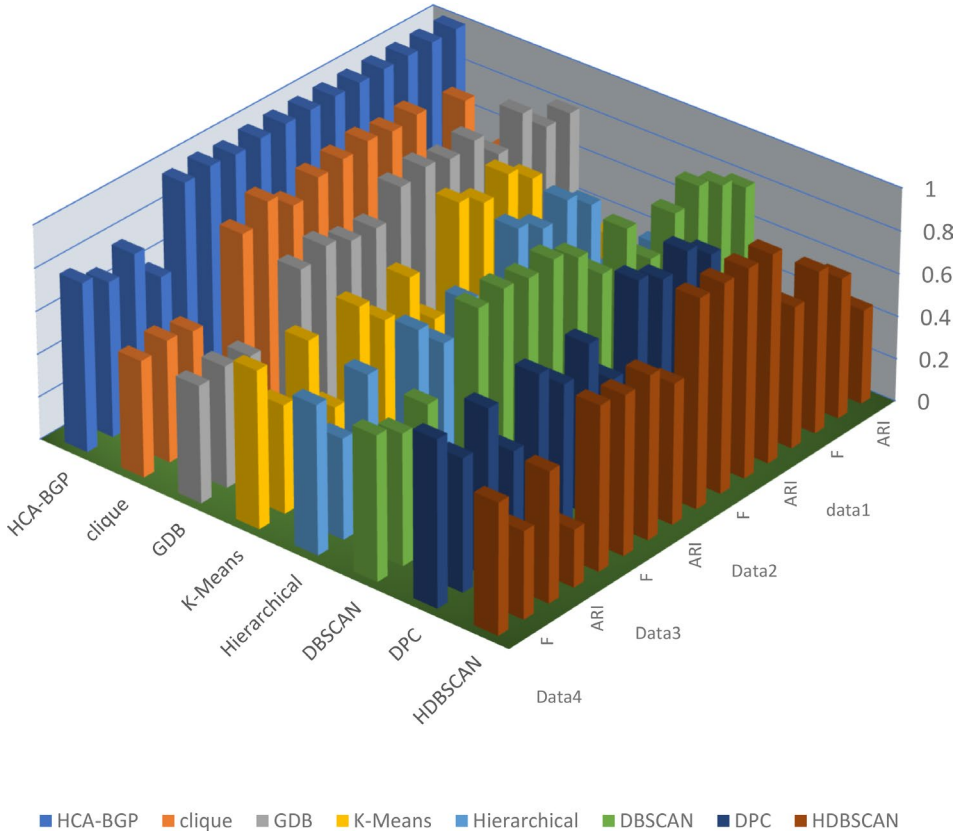


FIGURE 22 | Comparison of clustering metrics for various algorithms on real geographical datasets Data1, Data2, Data3 and Data4.

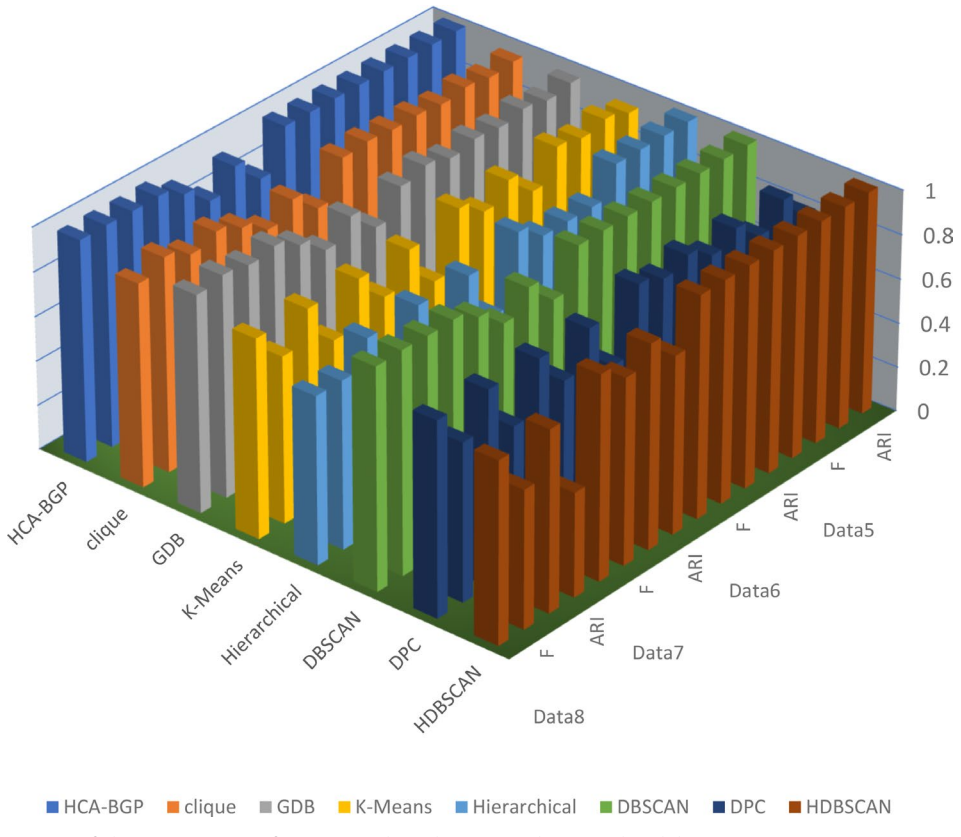


FIGURE 23 | Comparison of clustering metrics for various algorithms on real geographical datasets Data5, Data6, Data7 and Data8.

F-test results between HCA-BGP and GDB:

F-value between the two algorithms = 644.20, *F*-value across different *R* and *M* combinations = 0.79.

TABLE 6 | Attributes of the IRIS, Abalone, Segmentation and Wine Datasets.

Data set	Number of samples	Dimension	Number of clusters
IRIS	150	4	3
Abalone	4177	8	3
Segmentation	210	19	7
Wine	178	13	3

These results show that the primary source of variance is the algorithm itself, with *R* and *M* being secondary sources.

Figure 27 shows the clustering results of the Data8 dataset for HCA-BGP, CLIQUE and GDB under different *M* and *R* parameters. Table 9 shows the ARI values of the clustering results.

As shown in Figure 27, regardless of how *M* and *R* vary, the clustering results of the HCA-BGP algorithm remain good and stable. However, for the CLIQUE and GDB algorithms, when *M* is very small, different clusters may be incorrectly merged, and when *R* is very large, clustering accuracy is lost.

The 3D bar chart corresponding to Table 9 is shown in Figure 28.

As shown in Table 9 and Figure 28, the average ARI of the HCA-BGP algorithm is 0.9990, the average ARI of the CLIQUE

TABLE 7 | Clustering results of comparison algorithms on the IRIS, Abalone, Segmentation and Wine Datasets.

Datasets		HCA-BGP	Clique	GDB	K-Means	Hierarchical	DBSCAN	DPC	HDBSCAN
IRIS	ARI	**0.799**	0.5248	0.725	0.7302	0.7591	0.540	0.6274	0.5681
	NMI	**0.8454**	0.5629	0.7202	0.7551	0.8032	0.6573	0.7319	0.7315
	F	**0.8849**	0.6788	0.8104	0.8208	0.8407	0.7328	0.7629	0.7714
	V-Measure	**0.848**	0.5707	0.7249	0.7581	0.8056	0.6654	0.7354	0.7336
		$K=3$	$M=7$, $R=1$	$M=5$, $R=3$	$K=3$	Linkages = 'complete' $n=3$	eps = 0.09, MinPts = 20	$n=3$, distPercent = 2	min_cluster_size = 6
Abalone	ARI	0.1427	0.0644	0.0148	0.1331	0.107	0.1223	**0.1506**	0.1192
	NMI	0.1420	0.0744	0.046	0.1289	**0.1578**	0.1163	0.1307	0.1444
	F	**0.5296**	0.4156	0.4777	0.461	0.423	0.4849	0.4574	0.523
	V-Measure	**0.1815**	0.0777	0.0465	0.1293	0.1581	0.1166	0.1311	0.1448
		$K=3$	$M=6$, $R=2$	$M=4$, $R=10$	$K=3$	Linkages = 'complete' $n=3$	eps = 0.5, MinPts = 10	$n=3$, distPercent = 2	min_cluster_size = 15
Segmentation	ARI	**0.4903**	0.3782	0.2971	0.4361	0.3527	0.2161	0.4553	0.2159
	NMI	0.5928	0.5142	0.4805	**0.5967**	0.5133	0.4504	0.6076	0.4838
	F	**0.5661**	0.4716	0.4326	0.5554	0.4797	0.4125	0.56	0.4363
	V-Measure	0.6126	0.5421	0.5174	**0.6171**	0.5375	0.4819	0.6282	0.5048
		$K=7$	$M=7$, $R=1$	$M=3$, $R=2$	$K=7$	Linkages = 'ward' $n=7$	eps = 1.5, MinPts = 3	$n=7$, distPercent = 2	min_cluster_size = 4
Wine	ARI	**0.9168**	0.6431	0.5297	0.8975	0.7899	0.3823	0.7839	0.3768
	NMI	**0.899**	0.712	0.5865	0.8746	0.7842	0.4927	0.8054	0.4607
	F	0.928	0.7021	0.6258	**0.9319**	0.8602	0.6468	0.8569	0.6003
	V-Measure	**0.8996**	0.6853	0.5949	0.8759	0.7864	0.5027	0.8074	0.4665
		$K=3$	$M=7$, $R=1$	$M=2$, $R=3$	$K=3$	Linkages = 'ward' $n=3$	eps = 3.9, MinPts = 2	$n=3$, distPercent = 2	min_cluster_size = 3

Note: Bold values represent the optimal value.

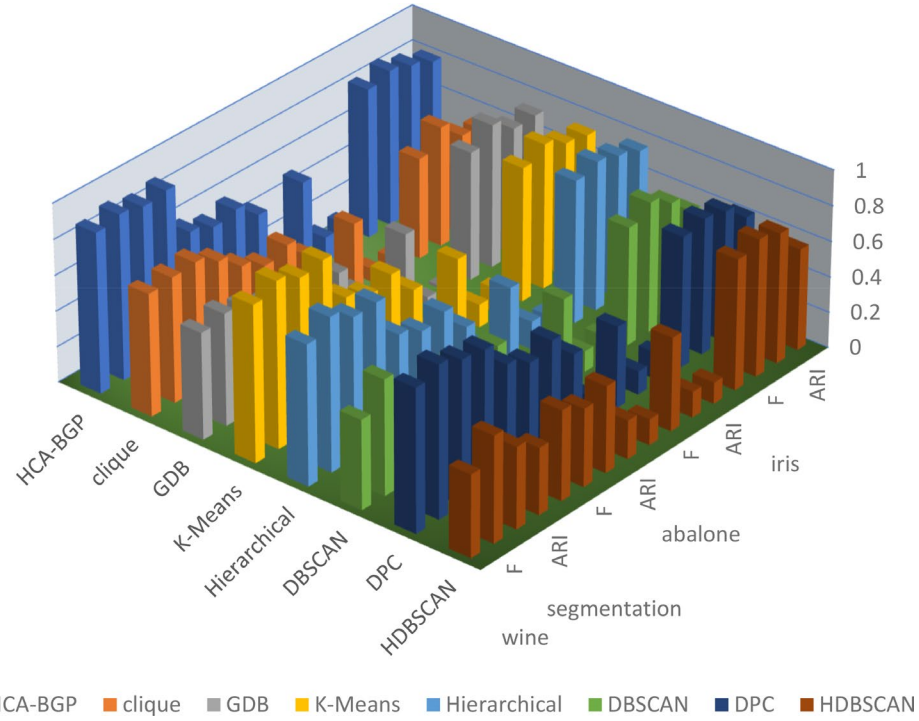


FIGURE 24 | Clustering results of comparison algorithms on the IRIS, Abalone, Segmentation and Wine Datasets.

algorithm is 0.8225, and the average ARI of the GDB algorithm is 0.889, indicating that the HCA-BGP algorithm outperforms both the CLIQUE and GDB algorithms in clustering performance.

The ARI variance of the HCA-BGP algorithm is 4.3948E-06, the ARI variance of the CLIQUE algorithm is 0.0376, and the ARI variance of the GDB algorithm is 0.0156. The ARI variance of the HCA-BGP algorithm is 0.01% of that of the CLIQUE algorithm and 0.03% of that of the GDB algorithm. Therefore, compared to the CLIQUE and GDB algorithms, the HCA-BGP algorithm is less sensitive to the parameters M and R.

Statistical Test Results for Clustering Metrics on the Geographic Dataset Data8:

T-test results between the proposed algorithm HCA-BGP and the CLIQUE algorithm:

t-value = 5.75, p-value = 1.12 e-06.

T-test results between the proposed algorithm HCA-BGP and the GDB algorithm:

t-value = 5.56, p-value = 2.06 e-06.

These results indicate significant differences in the clustering ARI values between the HCA-BGP algorithm and the other two algorithms.

F-test results between HCA-BGP and CLIQUE:

F-value between the two algorithms = 33.39, F-value across different R and M combinations = 1.01.

F-test results between HCA-BGP and GDB:

F-value between the two algorithms = 31.37, F-value across different R and M combinations = 1.02.

These results again confirm that the primary source of variance is the algorithm itself, while R and M are secondary sources.

Analysis of the Reasons for HCA-BGP's Insensitivity to Parameters M and R:

If the values of M and R are both small ($M = 12$, $R = 10$), the classical grid clustering algorithm CLIQUE tends to group multiple clusters into one, leading to poor clustering results, as shown in Figure 29. In contrast, HCA-BGP first determines whether a grid is a core dense grid, and if it is not, it splits the grid. The clustering process is shown in Table 10.

If the values of M and R are both large ($M = 40$, $R = 40$), CLIQUE may divide data that should belong to the same cluster into multiple clusters, resulting in suboptimal clustering performance, as illustrated in Figure 30. HCA-BGP, however, merges the generated subclusters, and the clustering process is detailed in Table 10.

As shown in Table 8, the grid splitting and sub-cluster merging in the HCA-BGP algorithm are the main reasons for its insensitivity to the parameters M and R.

6 | Conclusion

Grid clustering is known for its good time complexity and ability to adapt to datasets of arbitrary shapes, making it a popular choice for geographic data analysis. However, it is sensitive to

FIGURE 25 | Legend on next page.

FIGURE 25 | Clustering results of HCA-BGP, CLIQUE and GDB on S1 with different M and R values.

TABLE 8 | Clustering *F*-scores of HCA-BGP, CLIQUE and GDB on S1 with different *M* and *R* values.

M		*R* = 40	*R* = 48	*R* = 56	*R* = 64
10	HCA-BGP	0.9291	0.9896	0.9892	0.9888
	Clique	0.6098	0.5952	0.5658	0.4674
	GDB	0.5061	0.5420	0.5219	0.5661
12	HCA-BGP	0.9888	0.9892	0.9868	0.9888
	Clique	0.5084	0.4806	0.4671	0.4299
	GDB	0.5925	0.5360	0.4762	0.4417
14	HCA-BGP	0.9269	0.9275	0.9242	0.9235
	Clique	0.477	0.4144	0.3822	0.3446
	GDB	0.5176	0.4851	0.4389	0.3947
16	HCA-BGP	0.9249	0.9287	0.9865	0.9872
	Clique	0.5363	0.3944	0.3163	0.2881
	GDB	0.4852	0.4061	0.3715	0.3149
18	HCA-BGP	0.9242	0.9233	0.9856	0.9852
	Clique	0.3974	0.3203	0.2800	0.2742
	GDB	0.3790	0.3393	0.3096	0.2947
20	HCA-BGP	0.9876	0.9856	0.9864	0.9868
	Clique	0.3333	0.2751	0.2568	0.2376
	GDB	0.3210	0.2908	0.2717	0.2502
22	HCA-BGP	0.9832	0.9892	0.9888	0.9872
	Clique	0.2737	0.2425	0.2369	0.2355
	GDB	0.3255	0.2986	0.2510	0.2456

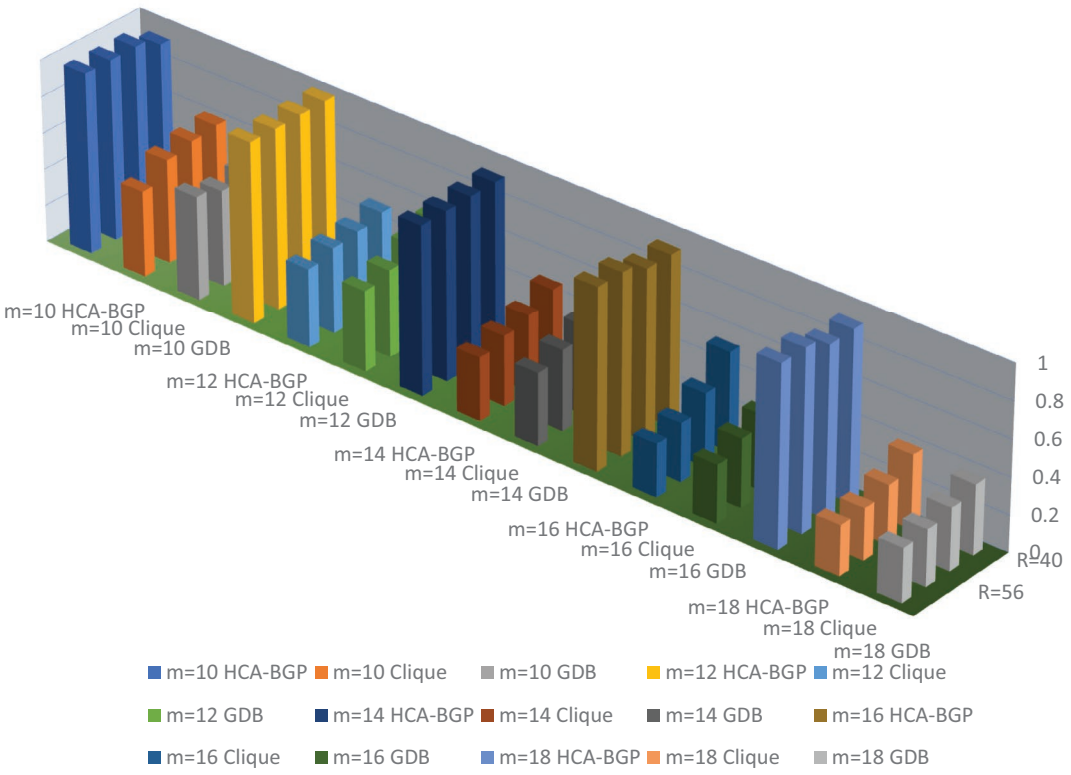


FIGURE 26 | The impact of parameters *M* and *R* on the clustering *F*-scores of HCA-BGP, CLIQUE and GDB for dataset S1.

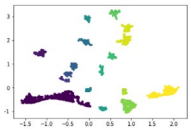
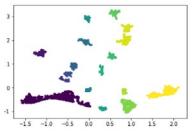
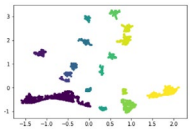
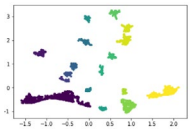
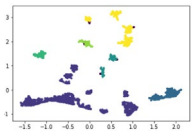
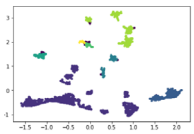
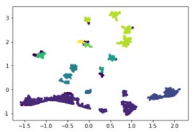
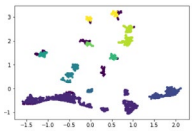
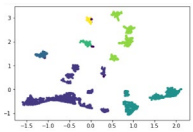
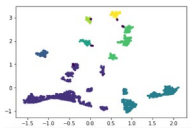
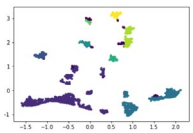
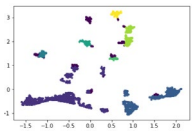
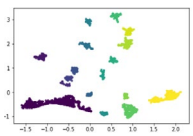
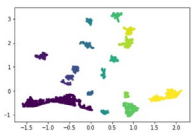
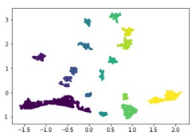
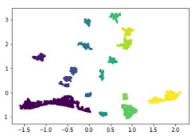
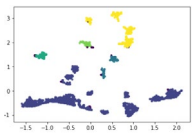
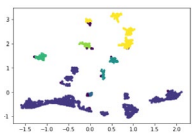
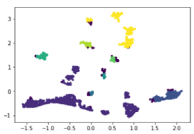
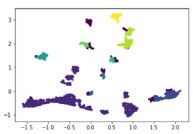
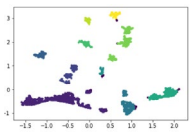
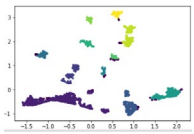
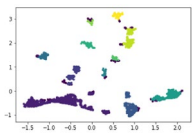
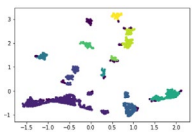
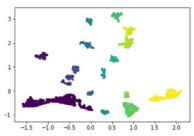
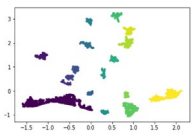
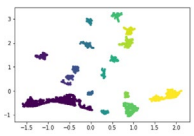
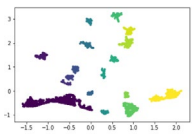
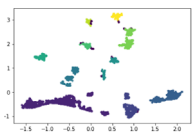
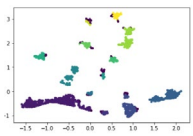
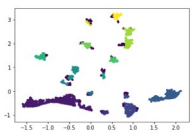
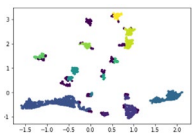
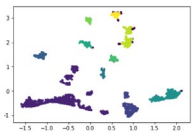
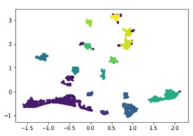
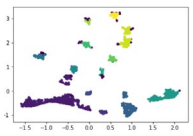
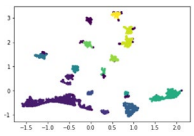
M		R=10	R=20	R=30	R=40
10	HCA-BGP				
	Clique				
	GDB				
12	HCA-BGP				
	Clique				
	GDB				
14	HCA-BGP				
	Clique				
	GDB				

FIGURE 27 | Legend on next page.

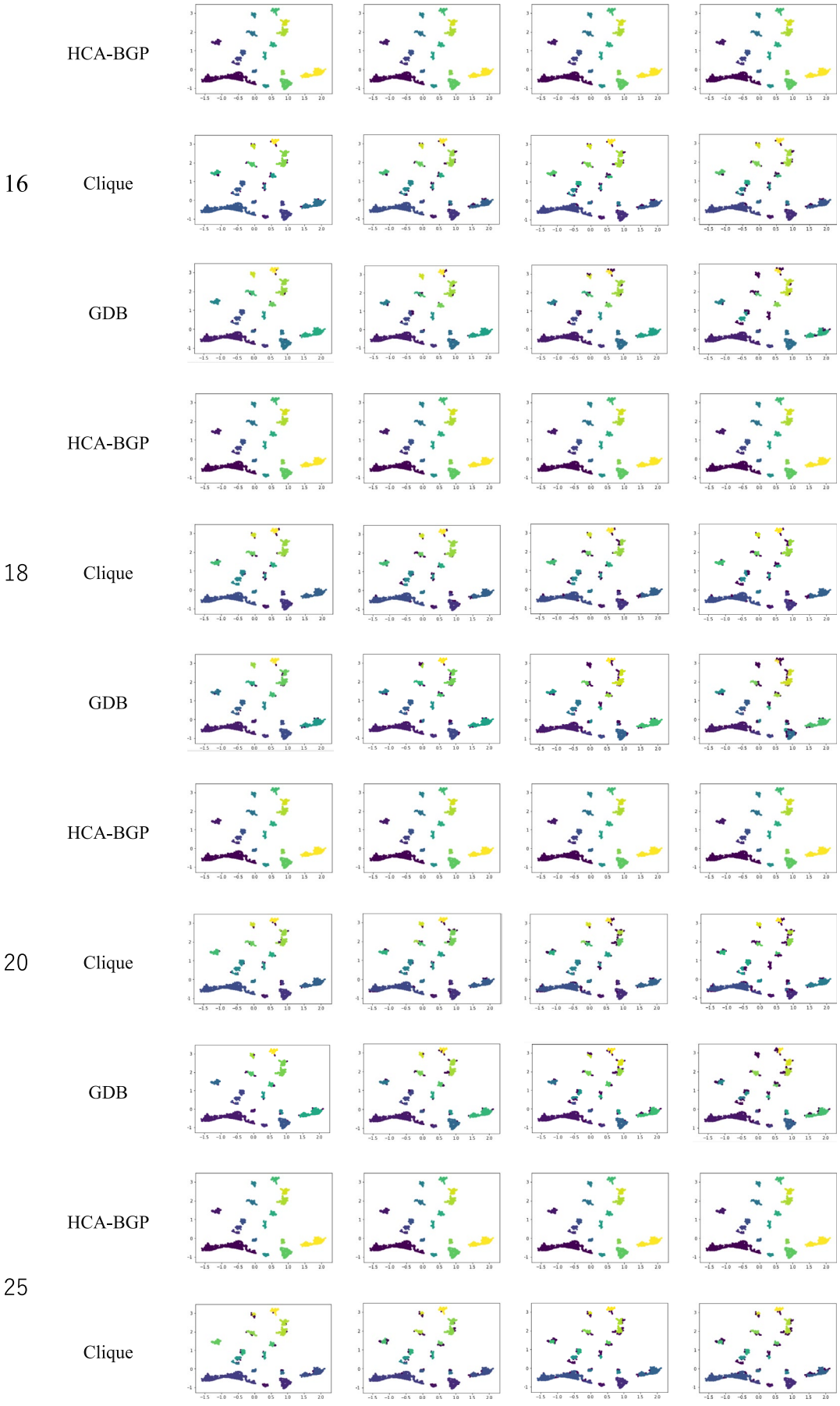


FIGURE 27 | Legend on next page.

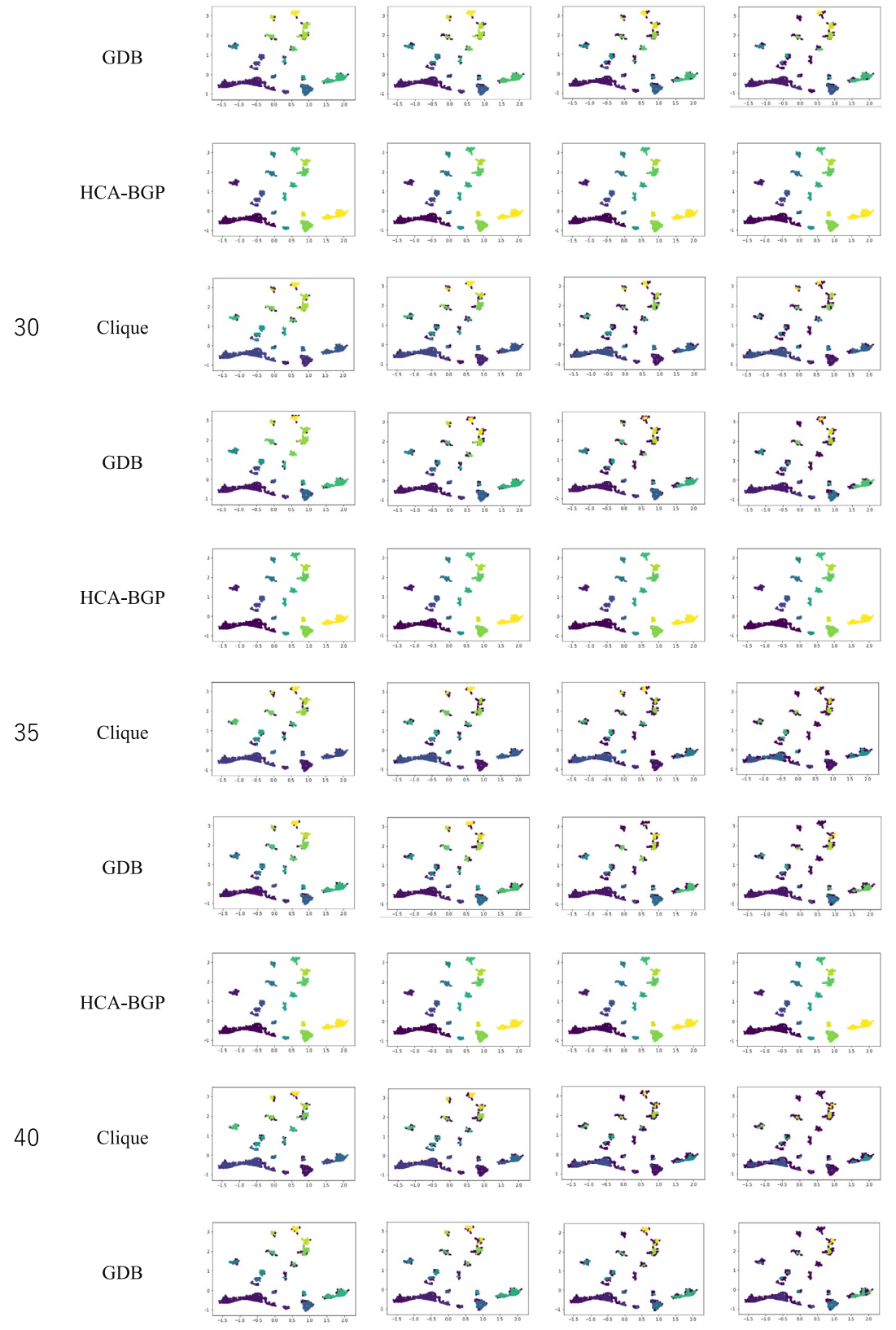


FIGURE 27 | Clustering results of HCA-BGP, CLIQUE and GDB on Data8 with different *M* and *R* values.

TABLE 9 | Clustering ARI of HCA-BGP, CLIQUE and GDB on Data8 with different *M* and *R* values.

M		*R* = 10	*R* = 20	*R* = 30	*R* = 40
10	HCA-BGP	0.9928	0.9928	0.9928	0.9928
	Clique	0.5612	0.5606	0.6628	0.6692
	GDB	0.7401	0.7497	0.7520	0.7702
12	HCA-BGP	0.9997	0.9997	0.9997	0.9997
	Clique	0.3068	0.5794	0.5837	0.5832
	GDB	0.9239	0.9208	0.9204	0.9201
14	HCA-BGP	0.9997	0.9997	0.9997	0.9997
	Clique	0.9253	0.9788	0.9763	0.9587
	GDB	0.8298	0.9315	0.9305	0.9354
16	HCA-BGP	0.9997	0.9997	0.9997	0.9997
	Clique	0.8604	0.9584	0.975	0.975
	GDB	0.9810	0.9802	0.9792	0.9690
18	HCA-BGP	0.9997	0.9997	0.9997	0.9997
	Clique	0.9834	0.9691	0.9575	0.9552
	GDB	0.9511	0.9704	0.9035	0.8911
20	HCA-BGP	0.9997	0.9997	0.9997	0.9997
	Clique	0.984	0.9662	0.8822	0.8742
	GDB	0.9603	0.9695	0.9240	0.9137
25	HCA-BGP	0.9997	0.9997	0.9997	0.9997
	Clique	0.9882	0.9782	0.9027	0.8954
	GDB	0.9872	0.9749	0.8834	0.8569
30	HCA-BGP	0.9997	0.9997	0.9997	0.9997
	Clique	0.9821	0.9055	0.865	0.835
	GDB	0.9835	0.9674	0.8536	0.8251
35	HCA-BGP	0.9997	0.9997	0.9997	0.9997
	Clique	0.9819	0.9171	0.8598	0.4625
	GDB	0.9826	0.9042	0.8520	0.8033
40	HCA-BGP	0.9997	0.9997	0.9997	0.9997
	Clique	0.905	0.8498	0.4846	0.4036
	GDB	0.9768	0.6150	0.5380	0.4710

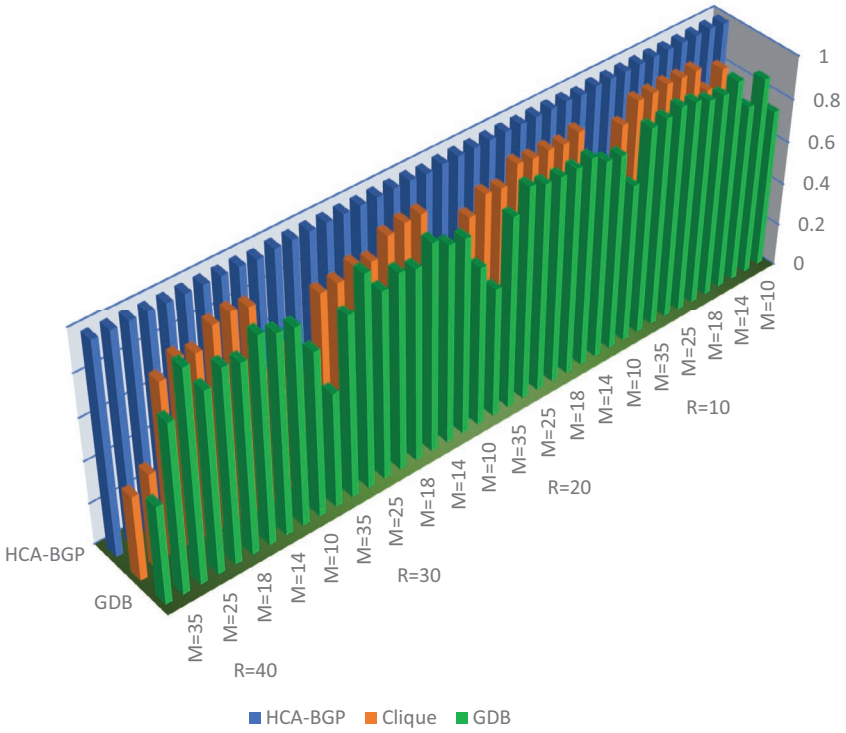


FIGURE 28 | The impact of parameters M and R on the clustering ARI of HCA-BGP, CLIQUE and GDB for dataset Data8.

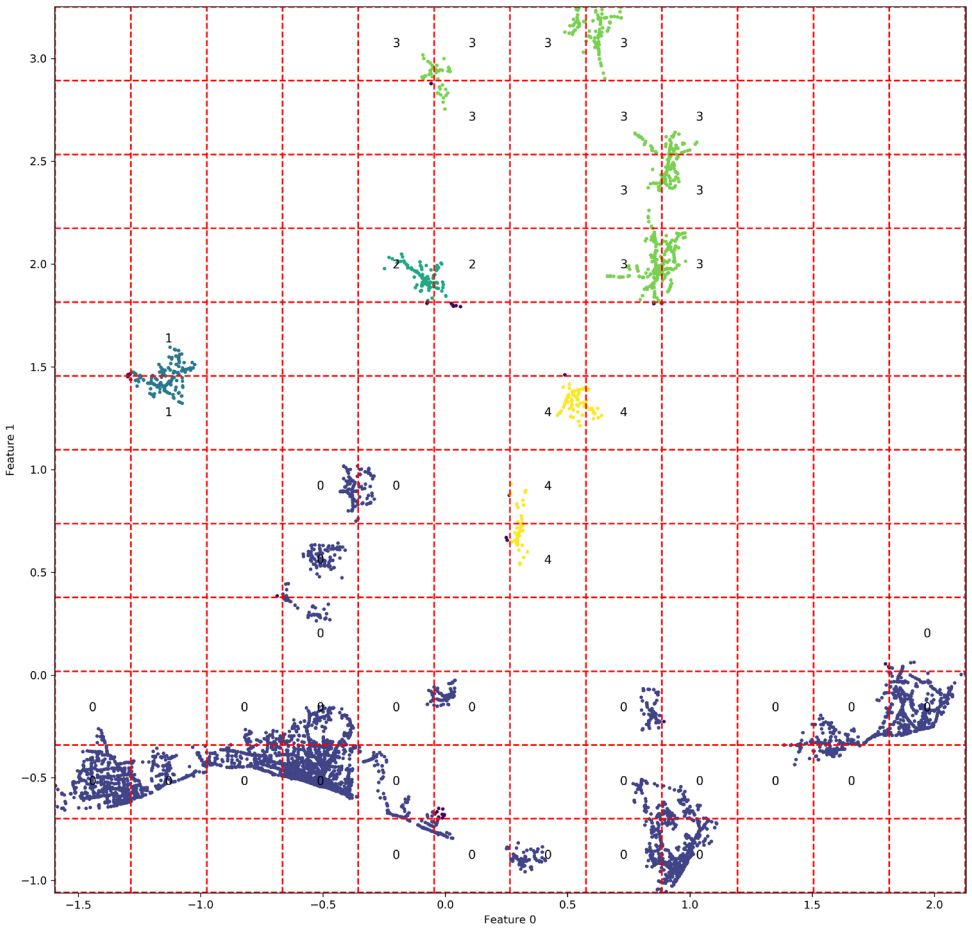
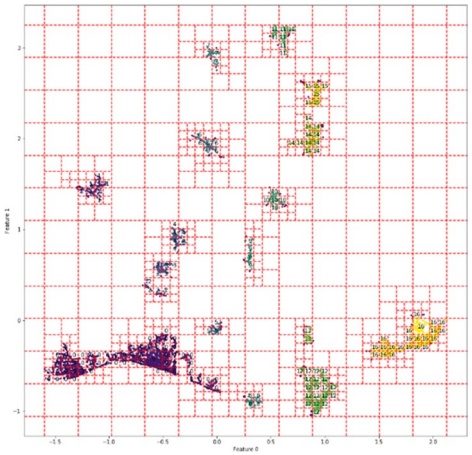
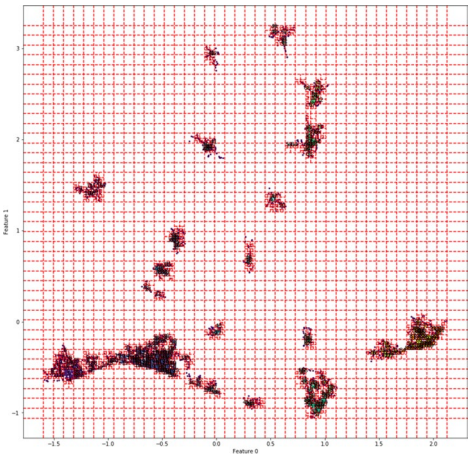
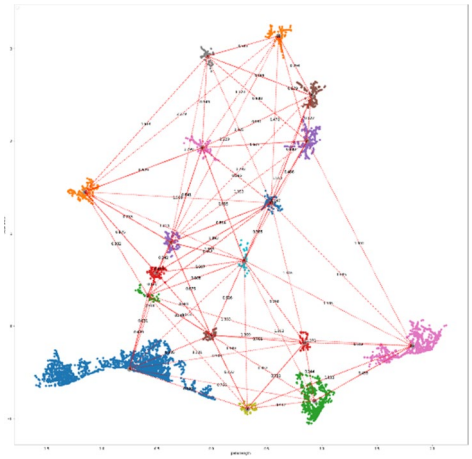
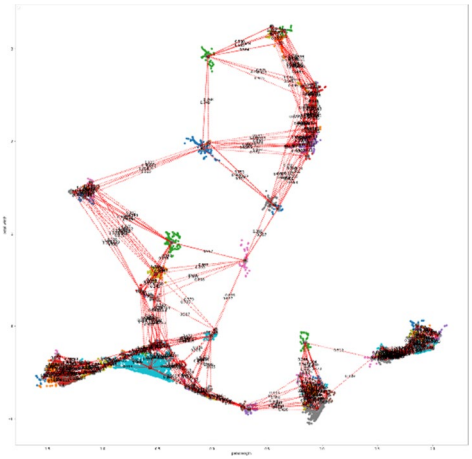
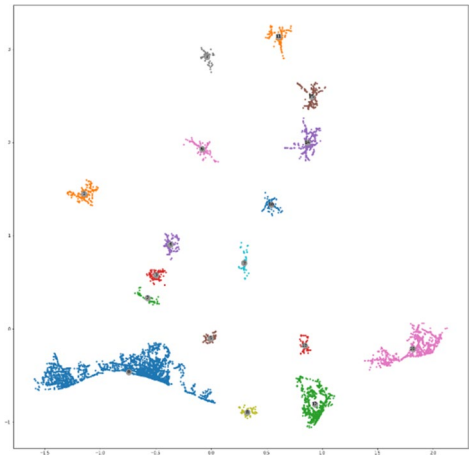
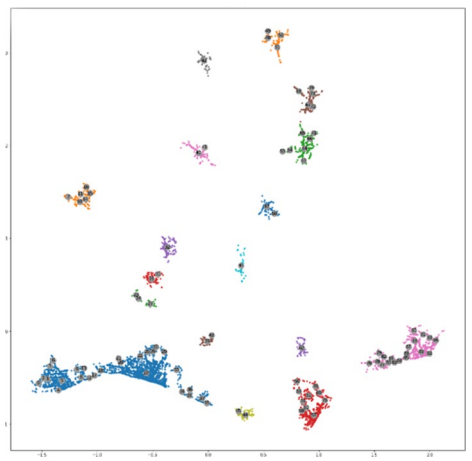


FIGURE 29 | Clustering of the geographic dataset Data8 using the CLIQUE algorithm when $M=12$ and $R=10$, where multiple clusters are grouped into a single cluster.

TABLE 10 | Clustering process of geographic dataset Data8 using HCA-BGP under different M and R parameters.

	$M=12, R=10$	$M=40, R=40$
Automatically split the grid, obtain core dense grids, and form the backbone of the clusters.		
Calculate the peripheral data of each cluster, then determine the adjacency relationships between neighbouring clusters.		
Merge clusters according to the target number of clusters k to obtain the final result.		

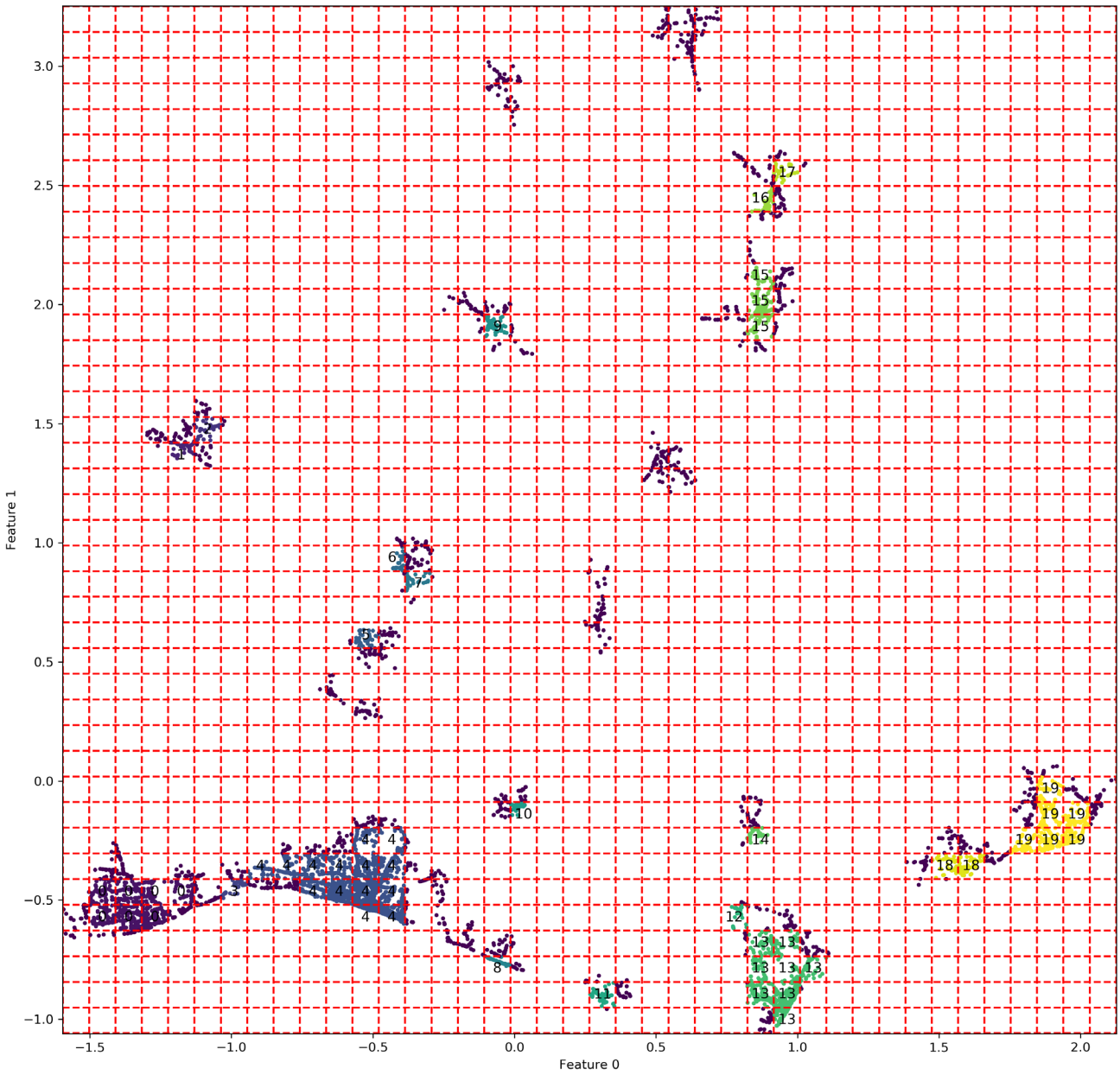


FIGURE 30 | Clustering of the geographic dataset Data8 using the CLIQUE algorithm when $M = 40$ and $R = 40$, where data from a single cluster is divided into multiple clusters.

the grid division number M and the density threshold R, which can lead to poor clustering results.

This paper proposes an improved grid clustering algorithm for geographic data mining, which requires only a single parameter, the final number of clusters k, to operate effectively.

The algorithm first uses improved grid clustering to obtain the core parts of each cluster, and then uses partition clustering to obtain the edge parts. This algorithm combines the advantages of both grid and partition clustering: it is fast and accurate.

The algorithm can automatically determine the grid division number M and the density threshold R based on the distribution of the dataset. Additionally, each grid can be independently split, making the algorithm insensitive to M and R and suitable for datasets with complex distributions.

Experiments on both synthetic and real geographic datasets verify the effectiveness of the proposed algorithm. The clustering results are superior to those of six other clustering algorithms. In particular, it shows significant advantages compared to the classic grid clustering algorithms CLIQUE and GDB, as well as the classic partition clustering algorithm K-means.

Conflicts of Interest

The author declares no conflicts of interest.

Data Availability Statement

The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

Abiodun, M., A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming. 2023. "K-Means Clustering Algorithms: A Comprehensive Review, Variants Analysis, and Advances in the Era of Big Data." *Information Sciences* 622: 178–210.

Abubaker, M., and W. Ashour. 2013. "Efficient Data Clustering Algorithms: Improvements Over K-Means." *International Journal of Intelligent Systems and Applications* 5, no. 3: 37–49.

Bot, D. M., J. Peeters, J. Liesenborgs, and J. Aerts. 2023. "FLASC: A Flare-Sensitive Clustering Algorithm: Extending HDBSCAN for Detecting Branches in Clusters." *Arxiv* 2311.15887.

Bureva, V., V. Traneva, D. Zoteva, and S. Tranev. 2021. "Generalized Net Model Simulation of Cluster Analysis Using CLIQUE: Clustering in Quest." In *Advances in High Performance Computing. HPC 2019. Studies in Computational Intelligence*, edited by I. Dimov and S. Fidanova, vol. 902. Springer. https://doi.org/10.1007/978-3-030-55347-0_5.

Cai, L., H. Wang, J. Zhou, and J. Liu. 2023. "Improved Distributed Clustering Algorithm for Adaptive Mesh Partitioning." *Journal of Chinese Computer Systems* 44, no. 4: 731–736.

Chen, L., F. Chen, Z. Liu, M. Lv, T. He, and S. Zhang. 2023. "Parallel Gravitational Clustering Based on Grid Partitioning for Large-Scale Data." *Applied Intelligence* 53, no. 3: 2506–2526. https://doi.org/10.1007/s10489-022-03661-7.

Dai, S., X. Liu, Z. Lai, and Z. Ren. 2022. "Gridded Local Adaptive DBSCAN Clustering Algorithm." *Journal of Chongqing University of Posts and Telecommunications (Natural Science)* 34, no. 2: 250–257.

Gui, Z., D. Peng, H. Wu, and X. Long. 2020. "MSGC: Multi-Scale Grid Clustering by Fusing Analytical Granularity and Visual Cognition for Detecting Hierarchical Spatial Patterns." *Future Generation Computer Systems* 112: 1038–1056. https://doi.org/10.1016/j.future.2020.06.053.

Guo, L., D. Li, and Z. Cai. 2023. "A Composite Grid Clustering Algorithm Based on Density and Balance Degree." In *Spatial Data and Intelligence. SpatialDI 2023. Lecture Notes in Computer Science*, edited by X. Meng, X. Li, J. Xu, et al., vol. 13887. Springer. https://doi.org/10.1007/978-3-031-32910-4_4.

He, H., Y. He, F. Wang, and W. Zhu. 2022. "Improved K-Means Algorithm for Clustering Non-Spherical Data." *Expert Systems* 39, no. 9: e13062.

Li, S., B. Zhang, J. Du, H. Zhu, and B. Fu. 2020. "Improved CLIQUE Algorithm for Clustering of Residential Hotspots." *Journal of Chinese Computer Systems* 41, no. 1: 61–65.

Lin, P., X. Chen, P. Long, and M. Fu. 2018. "Improved CLIQUE Algorithm and its Parallelization." *Computing Technology and Automation* 37, no. 4: 49–54.

Ma, F., C. Wang, J. Huang, Q. Zhong, and T. Zhang. 2024. "Key Grids Based Batch-Incremental CLIQUE Clustering Algorithm Considering Cluster Structure Changes." *Information Sciences* 660: 120109. https://doi.org/10.1016/j.ins.2024.120109.

Maddah, L., J. M. Arauzo-Carod, and F. A. López. 2021. "Detection of Geographical Clustering: Cultural and Creative Industries in Barcelona." *European Planning Studies* 31, no. 3: 554–575.

Mann, S. K., and S. Chawla. 2023. "A Proposed Hybrid Clustering Algorithm Using K-Means and BIRCH for Cluster Based Cab Recommender System (CBCRS)." *International Journal of Information Technology* 15: 219–227.

Oghadam, P., and A. Ahmadi. 2024. "A Novel Two-Stage Bio-Inspired Method Using Red Deer Algorithm for Data Clustering." *Evolutionary Intelligence* 17: 1819–1836.

Starczewski, A., M. M. Scherer, W. Książek, M. Dębski, and L. Wang. 2021. "A Novel Grid-Based Clustering Algorithm." *Journal of Artificial Intelligence and Soft Computing Research* 11, no. 4: 319–330.

Sun, L., and Y. Liang. 2022. "Improved Clustering Algorithm Fusing Grid Partition and DBSCAN." *Computer Engineering and Applications* 58, no. 14: 73–79.

Wang, W., R. Wang, and F. Zhou. 2020. "Dynamic Clustering Algorithm Based on Local Grid." *Modern Electronics Technique* 43, no. 1: 102–106.

Xu, H. 2021. "An Improved K-means Clustering Algorithm." In *2020 IEEE 5th International Symposium on Smart and Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS), Dortmund, Germany*, 1–5. IEEE.

Zhang, D., W. Ni, S. Zhang, N. Fu, and L. Hou. 2023. "A Local Differential Privacy Based Privacy-Preserving Grid Clustering Method." *Chinese Journal of Computers* 46, no. 2: 422–435.

Zheng, C., and Y. Cao. 2019. "Self-Adaptive Based on Grid Density Clustering Algorithm." *Application Research of Computers* 36, no. 11: 3278–3281 +3309.

Zou, B., K. Yang, X. Kui, et al. 2023. "Anomaly Detection for Streaming Data Based on Grid-Clustering and Gaussian Distribution." *Information Sciences* 638: 118989.