


Article

# DBSCAN Clustering Algorithms for Non-Uniform Density Data and Its Application in Urban Rail Passenger Aggregation Distribution

Xiaolu Li <sup>1</sup>, Peng Zhang <sup>2</sup> and Guangyu Zhu <sup>1,\*</sup>

<sup>1</sup> School of Traffic and Transportation, Beijing Jiaotong University, Beijing 100044, China; lixiaolu@bjtu.edu.cn

<sup>2</sup> Transport Planning and Research Institute, Ministry of Transport, Beijing 100028, China; zhangpeng@tpri.org.cn

\* Correspondence: gyzhu@bjtu.edu.cn

Received: 28 August 2019; Accepted: 27 September 2019; Published: 29 September 2019



**Abstract:** With the emergence of all kinds of location services applications, massive location data are collected in real time. A hierarchical fast density clustering algorithm, DBSCAN(density based spatial clustering of applications with noise) algorithm based on Gauss mixture model, is proposed to detect clusters and noises of arbitrary shape in location data. First, the gaussian mixture model is used to fit the probability distribution of the dataset to determine different density levels; then, based on the DBSCAN algorithm, the subdatasets with different density levels are locally clustered, and at the same time, the appropriate seeds are selected to complete the cluster expansion; finally, the subdatasets clustering results are merged. The method validates the clustering effect of the proposed algorithm in terms of clustering accuracy, different noise intensity and time efficiency on the test data of public data sets. The experimental results show that the clustering effect of the proposed algorithm is better than traditional DBSCAN. In addition, the passenger flow data of the night peak period of the actual site is used to identify the uneven distribution of passengers in the station. The result of passenger cluster identification is beneficial to the optimization of service facilities, passenger organization and guidance, abnormal passenger flow evacuation.

**Keywords:** clustering analysis; heterogeneity data; gaussian mixture model; location data; expectation-maximization

---

## 1. Introduction

With the widespread popularity of GPS, smart phones and other location-aware devices, various kinds of location services and mobile social network applications continue to emerge, thus accumulating a large number of geographic location data, such as user check-in data, vehicle GPS trajectory, micro-blog, Twitter and other massive location data [1]. Applications based on these location data have great application value in areas such as traffic management and control [2], recommendation systems [3], advertising [4], public facility location and assessment, abnormal population identification and evacuation [5]. Therefore, how to effectively and quickly mine meaningful potential patterns becomes the main challenge in processing location data.

The location data records the time and space trajectory information of the user, and embodies the change of the position of the moving object, such as the movement of the person, the movement of the vehicle, and the like [6]. The location data in the city mainly comes from traffic vehicle trajectory data [7], mobile communication location data, mobile social network check-in data [8], mobile media location data, e-commerce logistics location information, and AFC(Automatic Fare Collection) location records [9]. The analysis of massive location data can quantitatively describe and estimate people's social activity characteristics, find people's behavior patterns under different time and space granularities, gain insight into the overall movement trend of the group, and identify the routes and regions that people are interested in [10–12]. Therefore, through the mining of location data can help people understand and discover the great value implied in location data. For example, literature [13] discover relevant interest point patterns by mining users' GPS trajectories, and then find popular tourist routes; literature [14] also mine personal historical location data to realize user's friend recommendation and scenic spot recommendation.

In the application of location data analysis and processing, clustering technology is often used to pre-process and analyze location data to discover spatial or temporal clustering patterns in location data [15,16]. There are many kinds of spatial clustering analysis algorithms, the specific selection depends on the type of data, clustering purposes and other aspects. At present, the commonly used spatial clustering algorithms can be roughly divided into the following five categories: partition-based clustering algorithm, hierarchical clustering algorithm, density-based clustering algorithm, grid-based clustering algorithm and model-based clustering algorithm [17]. Because of the massive location data, arbitrary spatial distribution, fast update speed and large hybrid, density-based DBSCAN algorithm can find clusters of arbitrary shape, and can deal with noise. It is often used in pattern discovery of large location data [18]. However, DBSCAN algorithm has some shortcomings, such as low processing speed when the amount of data is large, in addition, because the density distribution of location data is not uniform, it is difficult for DBSCAN algorithm to be applied in practice, so scholars have improved it in many aspects.

In order to solve the problem of parameter selection and global parameters of DBSCAN, OPTICS(Ordering Points to Identify the Clustering Structure) algorithm is proposed in the literature [19], according to the concepts of core distance and accessible distance, ranks the objects in the data set and outputs an ordered list of objects. The information contained in the list of objects can be used to extract clustering, that is, classify the objects. Compared with the traditional clustering algorithm, the OPTICS algorithm has the greatest advantage that it does not depend on the input parameters. However, the algorithm does not produce clustering results visually, only by sorting the objects, and finally gives an extended cluster sorting structure interactively. DMDBSCAN algorithm [20] is similar to VDBSCAN (Varied Density DBSCAN) algorithm [21], which is an improved clustering algorithm for non-uniform data sets. By sorting and visualizing the parameter  $Dist_k$  values of the objects in the data set, can observe the key points of drastic changes in the  $Dist_k$  sorting graph to divide the non-uniform data set into density levels. However, the two algorithms can not automatically segment and extract the density hierarchy of data sets. The algorithm SA-DBSCAN(Self-adaptive DBSCAN) [22] fits the parameter  $Dist_k$  curve by inverse Gauss function according to the statistical characteristics of the data set, and automatically determines the global density parameters, which greatly reduces the dependence of DBSCAN on parameters. However, when clustering data sets with non-uniform density, the clustering effect of SA-DBSCAN algorithm will be greatly disturbed. DBSCAN algorithm has a high I/O overhead when the amount of data is large, which leads to low clustering efficiency. Many scholars have proposed corresponding improved algorithms to improve its efficiency. In order to extend to large-scale data clustering, a common method is to prune the search space of neighborhood queries in DBSCAN using spatial indexing technology (such as R-Tree) to improve clustering efficiency, but the efficiency of optimization algorithm still depends on data size [23]. A density clustering algorithm DBCURE-MR based on MapReduce is proposed in the literature [24].

Although the method achieves speedup of more than ten times or tens of times, it needs the support of dedicated processors or clusters of dedicated servers.

In this paper, a fast hierarchical clustering algorithm based on Gaussian mixture model is proposed for the rapid clustering problem of location data with uneven density distribution. The Gaussian mixture model based on DBSCAN algorithm can realize the density aggregation of position data in general computing platform. First, the Gaussian mixture model is used to fit the probability distribution of the dataset to determine different density levels; then, based on the DBSCAN algorithm, the sub-datasets with different density levels are locally clustered, and at the same time, the appropriate seeds are selected to complete the cluster expansion; finally, the sub-datasets clustering results are merged. In the method clustering effect analysis, multi-angle experimental evaluations were performed on multiple public test data and large-scale location data (from Microsoft T-Driver project) to verify the clustering effect and performance of the proposed algorithm. The method is applied to the identification of the distribution of passenger flow clusters by using the peak time data of the actual urban rail station.

The structure of this paper is as follows: Section 2 describes the core idea and detailed process of the improved DBSCAN algorithm. In Section 3, multi-angle (accuracy, noise impact, time efficiency) experimental evaluations were carried out on multiple public test data and large-scale location data (Microsoft T-Driver project) to verify the clustering effect of the proposed algorithm. In Section 4, the improved algorithm is applied to the recognition of passenger gathering characteristics in urban rail transit stations. Section 5 concludes the paper.

## 2. Improved DBSCAN Algorithm for Heterogeneous Data

For the location data set with uneven density distribution, the basic idea of the algorithm is as follows: firstly, Gauss mixture model is used to obtain the density and distribution characteristics of location data set, so as to realize the hierarchical processing of non-uniform density data set; then, a fast expansion processing method of data cluster is designed to improve DBSCAN algorithm and realize clustering analysis of hierarchical data sets. The core idea of the algorithm is as follows: (1) hierarchical processing for data sets with uneven density distribution; (2) fast cluster expansion processing.

### 2.1. Symbolic Description

For the convenience of understanding the method, the relevant parameters are shown in Table 1.

**Table 1.** Main symbol description.

Symbol	Notation	Symbol	Notation
$\theta$	The set of all parameters of the gauss mixture model	$\alpha_i$	Mixture ratio of gauss components
$\mu_i$	The mean vector of the i-th gauss component	$\Sigma_i$	Covariance matrix of the i-th gauss component
$\omega$	Latent variable	$P(\theta)$	Prior probability density of parameters
$Eps$	Data point neighborhood range radius	$MinPts$	Minimum Neighbor Point Number Threshold
$N_{Eps}(p)$	Neighbor point set of point $p$ in $Eps$ neighborhood	$Dist_k$	$k$ nearest neighbor distance

## 2.2. Density Layering Based on Gauss Mixture Model

The distribution characteristics of sample points in data set are fitted by a linear combination of a certain number of Gauss distributions [25–27]. The probability density function of each Gauss distribution is obtained by the expectation maximization algorithm (EM), and the probability density function is calculated for different sample points, to measure the density distribution of the sample point. The sample points with similar probability density function values are divided into the same density layer, so as to achieve stratified processing of uneven density data sets.

In the above process, the estimation of the unknown parameters in the Gaussian mixture model is the core of the dataset hierarchy. The author uses EM algorithm to solve the model parameters.

The form of the probability density function of the Gaussian mixture model [28,29] used in this paper is shown in Equations (1) and (2).

$$P(X|\theta) = \sum_{i=1}^m \alpha_i p_i(x|\theta_i) \quad (1)$$

$$p(x|\theta_i) = (2\pi)^{-d/2} |\Sigma_i|^{-1/2} \exp(-\frac{1}{2}(x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i)) \quad (2)$$

Among them:  $\theta = (\theta_i, \alpha_i)^T$ ,  $\theta_i = (\mu_i^T, \Sigma_i)^T$ ,  $\mu_i$  is the expectation of the  $i$ -th Gaussian distribution,  $\Sigma_i$  is the  $i$ -th gaussian distributed covariance matrix,  $m$  is the order of the Gaussian mixture model,  $d$  is the dimension of the vector.

The specific steps for estimating the parameters of the Gaussian mixture model using the EM algorithm are as follows, through the cycle E-Step, M-Step until the final result converges.

(E-Step) Calculate the value of the implicit variable from the value of the parameter estimated in the previous step:

$$\omega_{ij}^{(t+1)} = p(j|x_i, \theta^{(t)}) = \frac{\alpha_j^{(t)} p(x_i|\theta_j^{(t)})}{\sum_{k=1}^m \alpha_k^{(t)} p(x_i|\theta_k^{(t)})} = \frac{|\Sigma_j^{(t)}|^{-1/2} \exp(-\frac{1}{2}(x_i - \mu_j^{(t)})^T (\Sigma_j^{(t)})^{-1} (x_i - \mu_j^{(t)}))}{\sum_{k=1}^m |\Sigma_k^{(t)}|^{-1/2} \exp(-\frac{1}{2}(x_i - \mu_k^{(t)})^T (\Sigma_k^{(t)})^{-1} (x_i - \mu_k^{(t)}))} \quad (3)$$

(M-Step) Estimating the unknown parameter value according to the value of the implied variable:

$$\hat{\alpha}_j^{(t+1)} = \frac{1}{n} \sum_{i=1}^n \omega_{ij}^{(t+1)} \quad (4)$$

$$\hat{\mu}_j^{(t+1)} = \frac{\sum_{i=1}^n \omega_{ij}^{(t+1)} x_i}{n \hat{\alpha}_j^{(t+1)}} \quad (5)$$

$$\hat{\Sigma}_j^{(t+1)} = \frac{1}{n \hat{\alpha}_j^{(t+1)}} \sum_{i=1}^n \omega_{ij}^{(t+1)} x_i (x_i - \hat{\mu}_j^{(t+1)}) (x_i - \hat{\mu}_j^{(t+1)})^T \quad (6)$$

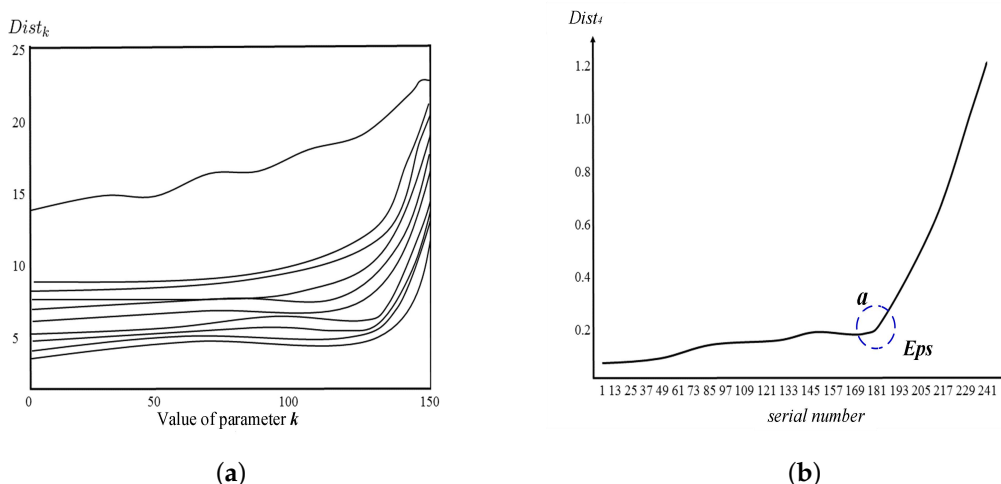
When using the EM algorithm to solve Gaussian mixture model parameters, E-Step first uses the  $\alpha_j^{(0)}$ ,  $\mu_j^{(0)}$ ,  $\Sigma_j^{(0)}$  parameter values obtained by initializing M-Step to estimate the value of the implicit class variable  $\omega_{ij}^{(t+1)}$ ; then, the  $\omega_{ij}^{(t+1)}$  value is substituted into each formula of M-Step. The value of each  $\alpha_j^{(t+1)}$ ,  $\mu_j^{(t+1)}$ ,  $\Sigma_j^{(t+1)}$  parameter is calculated; when the maximum likelihood maximum value is obtained, the  $\omega_{ij}^{(t+1)}$  value needs to be recalculated, and iterates until the parameter estimation value obtained twice in succession satisfies the convergence condition of the algorithm and the algorithm terminates.

### 2.3. Parameter Determination of Data Sets with Different Density Levels

The dataset layered by the Gaussian mixture model eliminates the influence of uneven distribution of dataset density on the clustering effect. The traditional DBSCAN algorithm can be used to perform local clustering on the data set and obtain the appropriate  $MinPts$  and  $Eps$  parameter values.

In the traditional DBSCAN algorithm, the threshold parameter  $MinPts = 4$  is usually set empirically; the value of the neighborhood radius  $Eps$  parameter is determined by traversing the distance between each sample point and all objects in the data set. That is, in the data set of a certain density layer, the  $k$ -th nearest neighbor distance  $Dist_k$  is calculated for any sample point, and the distance values are sorted in ascending order. When the threshold parameter  $MinPts$  value corresponds to multiple  $k$  values, it can be obtained as shown in Figure 1a. The distribution effect of  $Dist_k$  is shown; Figure 1b shows the distribution of  $Dist_4$  for each sample point when the threshold parameter  $MinPts = 4$ .

It can be seen from Figure 1 that the parallel distribution features of the density distribution curve of the dataset after dividing the hierarchy are obvious. In addition, most curves will have a turning point. Figure 1 can reveal the density distribution characteristics of the dataset to some extent. In general, noise points and clusters in the dataset should have large density differences. That is, there is a certain threshold point in the  $Dist_k$  graph. The points in the dataset can be divided into two parts with large differences in density. This point is shown in Figure 1b, which is a sudden change point of  $Dist_k$  curve from slow to steep  $a$ . The characteristic of the mutation point is that the  $k$  nearest neighbor distance  $Dist_k$  changes smaller before the point of mutation, and the  $k$  nearest neighbor distance  $Dist_k$  of each sample point is larger after the mutation point  $k$ . The data point on the right of the  $a$  point is the noise point of the low density distribution. The data point on the left of the  $a$  point is the data point of the high density distribution, and the value  $Dist_4$  of the  $a$  point is  $Eps$  of the density parameter of the cluster, which is the value of the density parameter  $MinPts$ . According to this feature, the  $Dist_k$  value corresponding to this point can be taken as the value of the  $Eps$ , that is, a set of maximum-density connected objects based on density reachability can be obtained. Through this method, the  $Eps$  parameter determination for a uniform density data set has better feasibility and effectiveness, and the processing process is relatively simple and rapid.



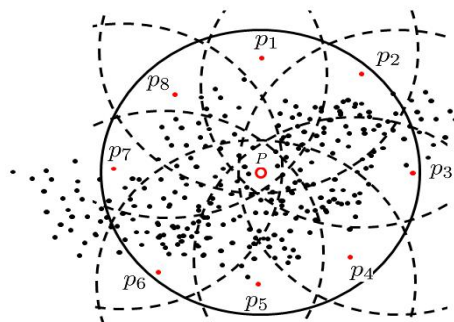
**Figure 1.**  $Dist_k$  distribution of a single density hierarchical dataset (a) Threshold parameter  $MinPts$  value is not fixed,  $Dist_k$  distribution effect; (b) Threshold parameter  $MinPts = 4$ , distribution of sample points  $Dist_k$ .

#### 2.4. Fast Expansion of Clusters

In view of the shortcomings of the traditional DBSCAN algorithm in the efficiency of execution, a Fast-DBSCAN algorithm is proposed. In the process of clustering the hierarchical data sets, all the density reachable objects in the neighborhood are classified as the same cluster, and then a certain number of representative points are selected as the seed objects to extend the cluster. When the number of seed objects is small, the algorithm will lose the object in the cluster expansion process. In this paper, 8 position points are taken as reference to select the representative points as seed objects when the two or more cluster extensions are extended, and there are usually  $2^n$  quadrants and  $3^n - 1$  reference points for  $N$  dimensional space, so the number of seed objects is up to  $3^n - 1$ . This paper focuses on the expansion of clusters in two-dimensional space, so it is reasonable to select 8 seed objects for arbitrary core objects. The selection method is to take the core object  $p$  as the center, draw the circle with its  $Eps$  radius, first select 4 points at the top, the bottom, the left and the right of the circle, and then take the middle point of the arc between every two points of the above 4 points as the other 4 points, as shown in Figure 2.

In the Algorithm 1, when the first core point of the new class is found, the first batch of representative points is selected as the seed point for class expansion. In the subsequent class expansion round, the new seed is continuously added to the seed point collection seed for subsequent class expansion. This cycle continues until the representative seed is empty. This indicates that the class has been expanded.

In the process ExpandCluster, a process Representative-Seed-Selection is added to select a representative point from the neighborhood of the core point.



**Figure 2.** Cluster extension schematic of core points.

#### 2.5. Improved DBSCAN Algorithm for Heterogeneous Dataset

The input of the DBSCAN algorithm based on the Gaussian mixture model is the data set  $D$ , the order  $m$  of the Gaussian mixture model and the initial value of the parameter, and the threshold parameter  $MinPts$ ; the output is the cluster or cluster obtained by clustering. The flow of the algorithm is shown in Figure 3, the pseudocode of the improved algorithm is shown in Algorithm 2.

---

**Algorithm 1** ExpandCluster (SetofPoints, Eps, MinPts, RepresentativeMinPts)

---

**Input:** Data set  $D(\text{Setofpoints})$ ,  $Eps$ ,  $MinPts$ ,  $RepresentativeMinPts$ **Output:** Result of Cluster expansion

```

1: for i=1 to Set of points-size do Point=Setofpoints.get(i)
2:   if Point.ClusterID=Unclassified then
3:     if ExpandCluster(SetofPoints, Eps, MinPts, RepresentativeMinPts) then
4:       ClusterID=nextID(ClusterID)
5:     end if
6:   end if
7: end for
8: candidate.seeds=SetofPoints.query(Point,Eps)
9: if candidate.seeds<MinPts then
10:  SetofPoints.changeClusterID(Point,Noise);
11: return False;
12: else
13:  SetofPoints.changeClusterID(candidate.seeds,ClusterID);
14:  while Representative.Seeds ≠ ∅ do
15:    current p=Representative.Seeds();
16:    Result=SetofPoints.query(current p,Eps);
17:    if Result.size ≥ MinPts then
18:      Rrepresentative.Seeds.Select (Result, RepresentativeMinPts, current p);
19:      for each point p in Rrepresentative.Seeds do
20:        if p. ClusterID= Unclassified then
21:          Add point p to the set Rrepresentative.Seeds();
22:        end if
23:      end for
24:    else
25:      for each point p in Result do
26:        if p. ClusterID= Unclassified or Noise then
27:          SetofPoints.changeClusterID(p,ClusterID);
28:        end if
29:      end for
30:    end if
31:  end while
32:  Rrepresentative-Seeds.delete(current point p)
33: end if
34: return True.

```

---

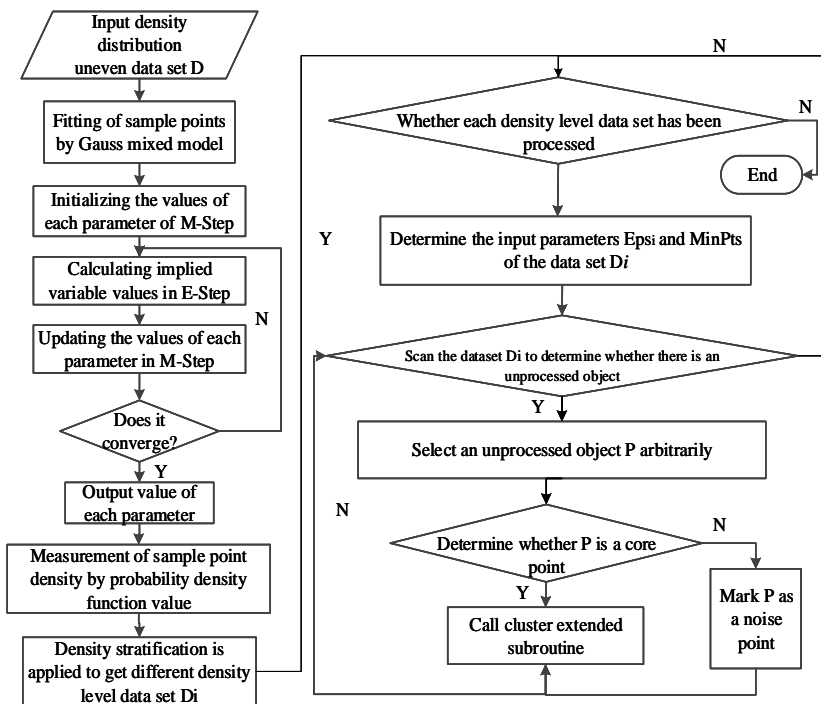


Figure 3. Flow chart of the improved algorithm.

Step 1: Determine the data set  $D$ , initialize the object, and mark the sample point as uncategorized.

Step 2: The Gauss mixture model is used to fit the sample points in the dataset  $D$ , and the EM model is used to solve the model parameters.

Step 3: The Gauss mixed model is represented as a linear combination of multiple Gauss distributions. Based on the value of the probability density function of each sample point, the data set of the density distribution is layered, and the data sets of different density levels,  $\{C_1, C_2, \dots, C_n\}$  are obtained.

Step 4: According to the data set  $C_i (1 \leq i \leq n)$  of different density levels, the threshold parameter  $MinPts = 4$  is set to determine the value of  $Epsi_i (1 \leq i \leq n)$  parameters, and the local clustering of the data set  $C_i$  is used in turn by using the traditional DBSCAN algorithm.

Step 5: Randomly select an unprocessed object  $p$  in the data set  $C_i$  to find the total number of objects whose  $Epsi_i$  and  $MinPts$  density can reach, and determine the relationship between the number and the size of the  $MinPts$ . If the value is greater than or equal to the value of  $MinPts$ , consider  $p$  as the core. In this case, all the reachable objects in the  $Epsi$  neighborhood of  $p$  are grouped into a cluster. Step 6 is used to expand the cluster until no cluster can be added to the cluster. This clustering of the core object  $p$  is completed. If less than  $MinPts$ ,  $p$  is marked as noise point, after the object  $p$  is processed, Step 5 is repeated until there is no unprocessed point in the data set  $C_i$ . At this time, the data set of the density level has been clustered and transferred to the next density-level datasets are analyzed.

Step 6: Select  $k$  representative objects as seeds in the cluster to join  $Seeds$ . Determine whether the seed object  $s_i$  in the  $Seeds$  is the core point. If it is not the core point, remove it directly from the  $Seeds$ . If it is the core point, then add the  $s_i$  neighbors. The unclassified objects in the domain join the cluster, and continue to select  $k'$  representative objects as seeds in the  $s_i$  neighborhood to join the  $Seeds'$ . Then the seed objects in the  $Seeds'$  are traversed, and uncategorized seeds are added to the  $Seeds$  as new seed objects.  $s_i$  is now processed and removed from  $Seeds$ . The above process is repeated until all the seed objects in the  $Seeds$  are



processed. At this time, the cluster expansion of the core point  $p$  ends, and the cluster expansion process of the next core point is transferred.

Step 7: When all the density level data sets  $C_i$  are clustered, the clustering results of each data set can be merged to output the final clustering results of the whole data set  $D$ .

### 3. Numerical Experiments and Effect Evaluation

In this section, the simulation experiment process includes two parts: the first part is to test the clustering accuracy by using open UCI data sets; the second part is to test the clustering accuracy by using classical simulation data sets, and make a comparative analysis of the clustering effect quality of the two algorithms.

#### 3.1. Analysis of Clustering Effects for Public Datasets

This section evaluates the clustering effect and efficiency of the improved DBSCAN, and compares it with DBSCAN. The data used in the experiment are shown in Table 2. Iris-t7.10k are six data sets including clusters of arbitrary shape, density, dimension and quantity. Four of these data sets are shown in Figure 4. Aggregation includes seven clusters of non-Gaussian distribution. DS1 consists of five clusters with different densities and a large amount of noise is doped in the data set. t4.8k is composed of seven density clusters of different shapes, nested each other and doped with a large amount of noise. t7.10k consists of nine clusters with different densities and nested among clusters. Taxi1 and Taxi2 data are from the T-Drive project of Microsoft Asia Research Institute. T-Drive collected GPS data of 10,357 taxis in Beijing within a week. Taxi 1 extracted the location data of 5000 taxis within the Fifth Ring Road in Beijing for 8 h. Taxi 2 extracts 8-h location data for 10,000 taxis.

**Table 2.** Test data sets.

Data Sets	Data Size	Data Dimension	Category	Data Sources
Iris	150	4	3	<a href="http://archive.ics.uci.edu/ml/datasets/">http://archive.ics.uci.edu/ml/datasets/</a>
Wine	178	13	3	<a href="http://archive.ics.uci.edu/ml/datasets/">http://archive.ics.uci.edu/ml/datasets/</a>
Aggregation	788	11	2	Artificial data [30]
DS1	8000	2	5	Artificial data [31]
t4.8k	8000	2	7	<a href="http://glaros.dtc.umn.edu/gk/home/">http://glaros.dtc.umn.edu/gk/home/</a>
t7.10k	10,000	2	9	<a href="http://glaros.dtc.umn.edu/gk/home/">http://glaros.dtc.umn.edu/gk/home/</a>
Taxi 1	1,194,976	2	Unknown	<a href="https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/">https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/</a>
Taxi 2	2,148,225	2	Unknown	<a href="https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/">https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/</a>

In order to validate the effectiveness of the proposed method, the clustering results of DBSCAN and improved DBSCAN methods were tested on 6 benchmark datasets such as Iris-t7.10k. The evaluation method was compared with DBSCAN clustering results. In order to verify the efficiency of the algorithm, the CPU time and memory consumption of the proposed method and the comparison algorithm are measured on two large-scale location data sets, Taxi1 Taxi2. The efficiency of the algorithm and the sensitivity to input parameters are evaluated.

**Algorithm 2** DBSCAN algorithm based on Gauss mixture model.**Input:** Data set  $D$ , Gaussian mixture model order  $m$ , Threshold parameter  $MinPts$ **Output:** Clustering result  $C = C_1, C_2, \dots, C_i$ ;

```

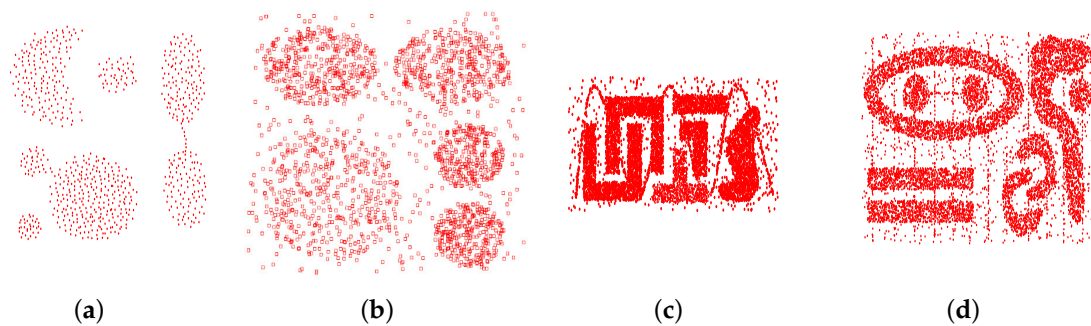
1:  $C = \emptyset$ , marked all objects in data set  $D$  as unprocessed;
2: Phase 1 Hierarchical processing of data sets with uneven density distribution
3: Initialize the Gaussian mixture clustering model parameters  $m$ ;
4: repeat
5:   for  $n = 1, 2, \dots, i$  do
6:     Calculate the value of the implicit variable from (3);
7:     for  $m = 1, 2, \dots, k$  do
8:       Calculate the mean vector using (4);
9:       Calculate the covariance matrix with (5);
10:      Calculate the mixing factor with (6);
11:      Update parameter  $\theta = \{\alpha_j^t, \mu_j^t, \Sigma_j^t\}$ 
12:    end for
13:  end for
14: until
15:  $\left| \frac{\log p(x|\theta^t) - \log p(x|\theta^{(t+1)})}{\log p(x|\theta^t)} \right| \leq e (e = 0.001)$ .
16:
17:
18: Get datasets  $D_i$  of different density levels of the data set  $D$  according to the Gaussian distribution
    parameter obtained by the formula (3)–(6);
19: Phase 2 Clustering processing of data sets of each layer
20: for datasets  $D_i (i = 1, 2, \dots, n)$  do
21:   Set threshold parameters  $MinPts = 4$ , the  $Eps_i (i = 1, 2, \dots, n)$  values of each density level are
    sequenced from small to large, and the traditional DBSCAN algorithm is used to locally cluster the
    data set  $D_i$ .
22:   for each object  $p$  in data set  $D_i$  do
23:     if  $p$  has been classified as a cluster or marked as noise then
24:       continue;
25:     else
26:       find objects  $p$  that are reachable to  $Eps$  and  $MinPts$ ;
27:       if  $N_{Eps}(p)$  (the  $Eps$  neighborhood of object  $p$ ) contains fewer objects than  $MinPts$  then
28:         mark object  $p$  is a boundary point or a noise point;
29:       else
30:          $p = Representative.Seeds()$ ;
31:         ExpandCluster(Set of Points,  $Eps$ ,  $MinPts$ ,  $RepresentativeMinPts$ );
32:         for All objects  $q$  in  $N_{Eps}(p)$  that have not been processed do
33:           Check its  $Eps$  neighborhood  $N_{Eps}(q)$ . If  $N_{Eps}(q)$  contains at least  $MinPts$  objects, add
            objects in  $N_{Eps}(q)$  that are not classified into any cluster to cluster  $C$ ;
34:         end for
35:       end if
36:     end if
37:   end for
38: end for
39: return Cluster division  $C = C_1, C_2, \dots, C_i$ .

```

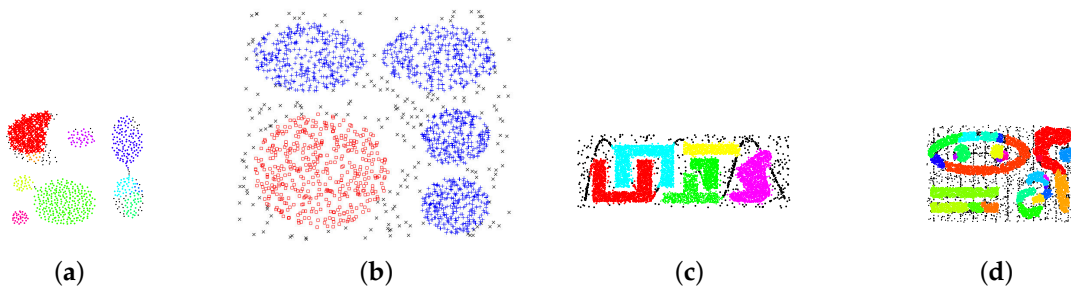
### 3.2. Effect Evaluation

The clustering results on the benchmark data set are shown in Figure 5. The improved DBSCAN and DBSCAN have the same MinPts parameters on the same data set, which are 10, 4, 20, 4 respectively.

In order to distinguish easily, different colors and shapes are used to represent different clusters. It can be seen from the graph that the improved DBSCAN can find the same cluster structure as DBSCAN and recognize noise points. This is because the improved DBSCAN follows density correlation on the basis of density stratification and belongs to accurate density clustering method. Traditional DBSCAN is not easy. Identifying uneven density distribution data, such as DS1 data sets, is difficult to separate right cluster; on t4.8k, it is also difficult to separate green adjacent cluster regions; on t7.1k data sets, it is easy to recognize the surrounding location points as noise when trying to distinguish the nested cluster in the upper left corner; to distinguish different clusters, it is necessary to set a smaller *Eps* value, but it is not easy to distinguish between them. At the same time, a cluster will be divided into several parts or generate more noise or as shown in the data sets Aggregation and t7.1k.

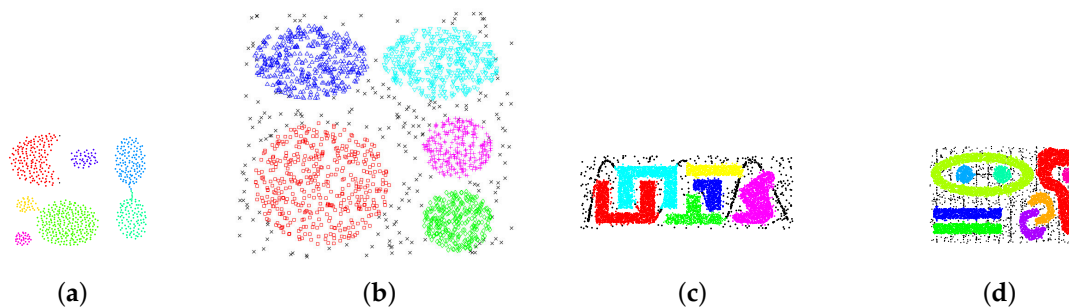


**Figure 4.** Four data sets used in experiments. (a) Aggregation. (b) DS1. (c) t4.8k. (d) t7.10k.



**Figure 5.** Clustering effect of DBSCAN on four data sets. (a) Aggregation. (b) DS1. (c) t4.8k. (d) t7.10k.

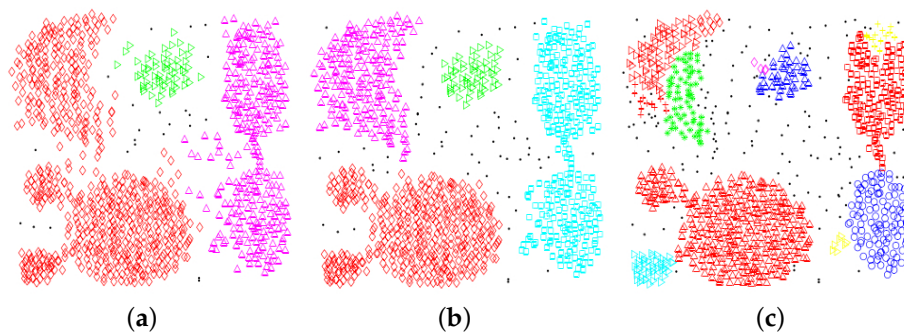
It can be seen from Figure 6 that the DBSCAN algorithm cannot find a suitable parameter *Eps* in the process of clustering datasets with uneven density distribution so that each cluster can be correctly identified. If the value of the *Eps* parameter is large, only clusters of low-density sample points can be separated, and higher-density sample points can be merged into the same cluster. Cluster separation of high-density distribution sample points cannot be completed; if the value of the *Eps* parameter is lower Small, clusters of higher-density sample points can be isolated, while lower-density sample points are separated into multiple sub-classes and generate a large number of noise points. It can be seen that the DBSCAN algorithm can not accurately complete the effective clustering of density distribution uneven data sets.



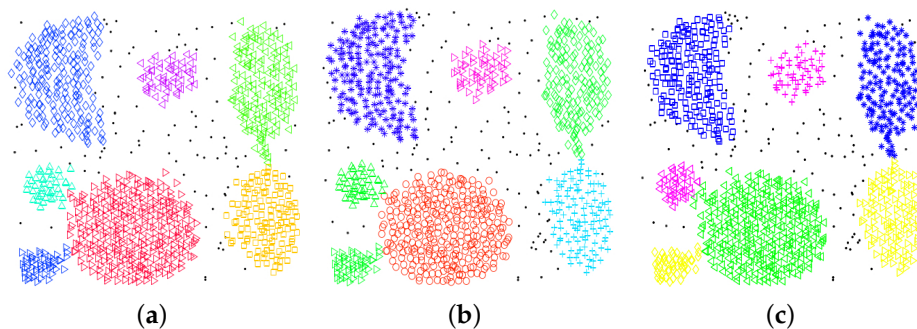
**Figure 6.** Clustering effect of improved DBSCAN on four data sets. (a) Aggregation. (b) DS1. (c) t4.8k. (d) t7.10k.

### 3.2.1. Effect of Noise Intensity on Clustering Effect

The effect of improved DBSCAN and traditional DBSCAN algorithm on noise recognition is tested on Aggregation data. 100, 150, 200 randomly distributed noise points were added to Aggregation data respectively. In Figures 7 and 8, the clustering results of improved DBSCAN and traditional DBSCAN algorithm with different intensity noise are presented respectively. The improved DBSCAN can identify the noise data by adjusting the parameters, and can obtain clustering results similar to those before adding noise. The traditional DBSCAN algorithm is difficult to distinguish the connected clusters from the original data. On the noise data, it is easy to connect the similar clusters into one cluster through the noise points, and it is also difficult to identify the noise data. Although some noises can be recognized by reducing *Eps* parameters, meaningful clustering results can not be obtained.



**Figure 7.** Influence of noise intensity on DBSCAN clustering effect. (a) noise = 100. (b) noise = 150. (c) noise = 200.



**Figure 8.** Influence of noise intensity on improved DBSCAN clustering effect. (a) noise = 100. (b) noise = 150. (c) noise = 200.

### 3.2.2. Evaluation Results of Clustering Effect

For the evaluation of clustering results, selecting a single clustering evaluation index sometimes can not explain the effectiveness of the clustering algorithm very well, so in the experiment, several evaluation indexes are used to verify the effectiveness of the algorithm, which makes the conclusion of this experiment more convincing. In the experiment, Accuracy, Fowlkes-Mallows, Jaccard Coefficient (JC index) and Precision are used as the evaluation indexes of clustering results. The mathematical formulas are as follows [32]:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

$$FM = \sqrt{Precision \times Recall} = \sqrt{\frac{TP}{TP + FP} \times \frac{TP}{TP + FN}} \tag{8}$$

$$Jaccard = \frac{TP}{TP + TN + FP} \tag{9}$$

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

where *TP* represents true positive, the set of common pairs of objects in both the sample cluster and clustering result cluster; *TN* represents true negative, the set of objects are neither in the sample cluster nor in the clustering result cluster; *FP* represents false positive, the set of objects in the result cluster but not in sample cluster; *FN* represents false negative, the number of pairs of objects in the sample cluster but not in result cluster.

The range of values of these clustering evaluation indicators is between 0 and 1. The larger their values are, the better the clustering effect of the improved clustering algorithm will be. If the value of the evaluation index is 1, it shows that the clustering effect of the clustering algorithm is exactly the same as the label in the real data set. In the process of experiment, the evaluation indexes of the four clustering algorithms need to be considered and the effectiveness of the algorithms should be evaluated comprehensively. The comparison results are shown in Table 3.

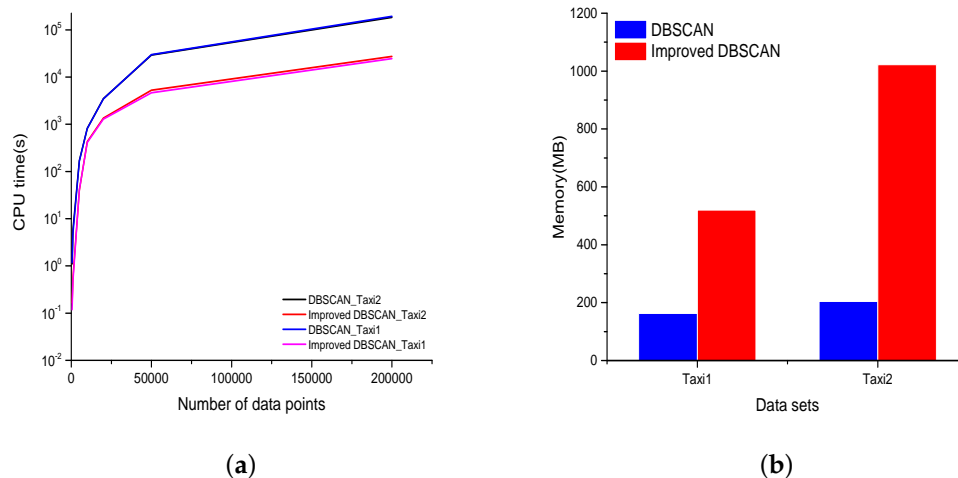
**Table 3.** Comparisons of clustering results in data sets.

Data Sets	DBSCAN				Improved DBSCAN			
	Accuracy	FM	Jaccard	Precision	Accuracy	FM	Jaccard	Precision
Iris	0.6467	0.6396	0.5951	0.5951	0.8733	0.8651	0.7190	0.8384
Wine	0.6238	0.6014	0.4527	0.5268	0.7917	0.7701	0.5329	0.5768
Aggregation	0.7112	0.7004	0.6367	0.7037	0.8922	0.8696	0.7109	0.8246
DS1	0.5856	0.5798	0.3616	0.3826	0.8108	0.8019	0.3642	0.3966
t4.8k	0.5456	0.5310	0.3383	0.3715	0.7001	0.6833	0.3553	0.3879
t7.10k	0.3971	0.3506	0.2297	0.2384	0.8334	0.8367	0.6121	0.6606

From the clustering results, it can be seen that the improved algorithm has better clustering effect than the traditional DBSCAN algorithm in data sets with uneven density distribution. In addition, in order to better verify the efficiency of the improved algorithm, the time complexity of clustering method is verified by the traffic location data of the T-Drive project of Microsoft Research Institute.

### 3.2.3. Clustering Time Comparison

This section tests the clustering efficiency and spatial memory consumption of the improved DBSCAN algorithm on location big data and compares it with traditional DBSCAN. Each algorithm sets the same parameters on the same data set. The parameters on each data set are  $Eps = 0.005$  and  $MinPts = 20$ . The CPU time trend graph consumed by each algorithm is shown in Figure 9a, and the memory consumption is shown in Figure 9b.



**Figure 9.** Comparisons of Clustering Performance of Microsoft Taxi Location Data Set: (a) CPU Time (Log); (b) Memory.

#### 4. Cluster Analysis of Passenger Position Data in Urban Rail Stations

In this section, the simulation shows that the passenger flow position data has uneven distribution characteristics, and then the method validity is verified by using the peak time data of the actual platform.

##### 4.1. Uneven Distribution of Passenger Position Data

Cross-interference between passenger streamlines will lead to uneven density distribution of passenger groups in the station. On the basis of field investigation and simulation by Anylogic, the location distribution map of passenger groups at different times on the platform can be obtained. The simulation model of Anylogic in this paper is mainly divided into four modules: pedestrian behavior logic module, train logic module, Guo Gongzhuang platform physical environment module. In the pedestrian behavior logic module, pedestrian behavior logic module is constructed from two aspects: inbound passenger flow and outbound passenger flow. Part of the passenger flow goes directly to the platform for waiting, while the other part leaves the platform through stairway. When the train arrives and the waiting passengers finish boarding, the passenger flow disappears. The train logic module mainly simulates the process of subway train entering and leaving the station through Rail Librai. The module functions include the generation and disappearance of the train, the alighting of passengers and the boarding of waiting passengers after the arrival of the train. The physical environment module of the station includes the key facilities such as the wall, obstacles and stairs of the station. It is constructed with space marking tools in the pedestrian database. The parameters in the simulation are derived from the field investigation of Guogongzhuang Station on 23 July 2019. During peak hours, the departure interval of adjacent trains is about 150 s, and the average time of train stopover is 55 s. The ratio of passenger flow from left and right entrance to opposite platform is about 8.7:1.3 and 8.4:1.6. It can be seen that there is a more obvious non-uniform density distribution characteristic, as shown in Figure 10.

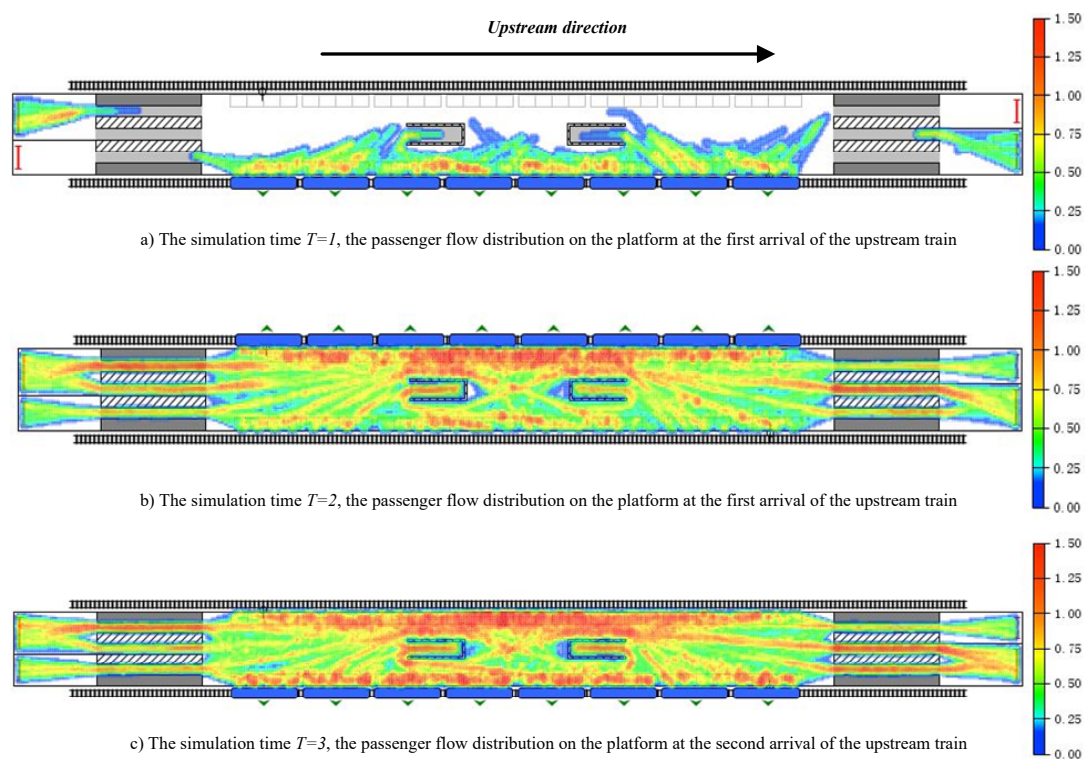


Figure 10. Location Distribution Map of Passenger Flow on Urban Rail Platform at Different Times.

In order to further verify the practicability of the proposed algorithm, the actual data of an urban rail transit station in Beijing was used to perform clustering analysis using the traditional DBSCAN algorithm and the proposed algorithm. The data of 18:00–18:30 in the evening peak period of Guogongzhuang subway station on 18 July 2016 is used as experimental data to investigate the performance and accuracy of the clustering algorithm. The data sample has a total of 50,000 sampling points. Table 4 shows the information of each attribute data of some sampling points. Each sample point consists of attributes such as ID, time, longitude, and latitude. The data source is the passenger position data that is continuously uploaded to the database server in a fixed period (10 s) during the test of the WIFI positioning system in the transfer station.

**Table 4.** Format of pre-processed WiFi positioning data.

Number	User_ID	TimeStamp	X	Y
1	0017	20160718180110	117.78	8.76
2	0059	20160718180130	86.40	5.78
3	0068	20160718180150	23.20	8.48
4	0094	20160718180210	102.84	8.84
		.....		
49,500	0397	20160718182920	72.90	3.68
49,510	0415	20160718183000	113.09	7.89

The first column in the table is the data number, the second column is the mobile device (or passenger) number, the third column shows the time of data acquisition, and the fourth and fifth columns record the geographical coordinates of the passenger at that time.

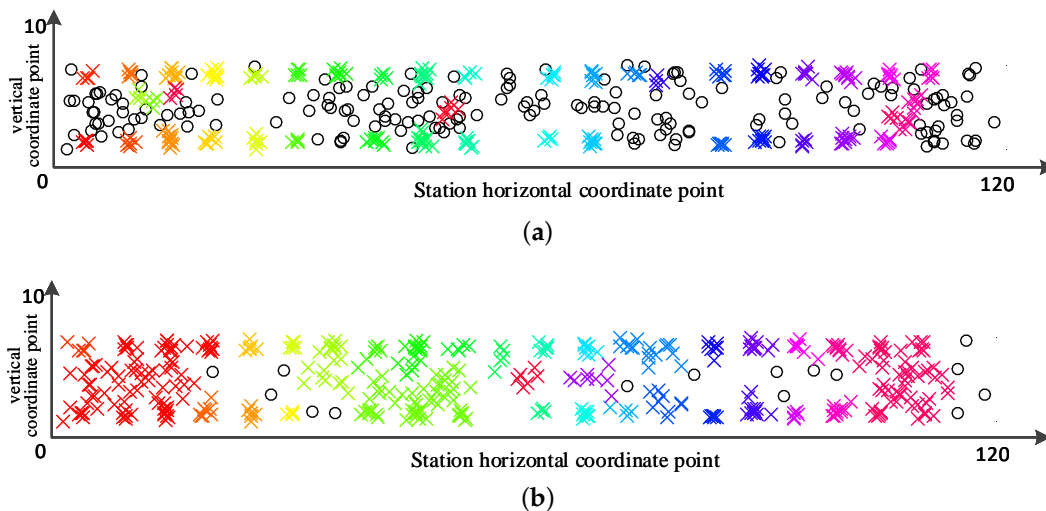
#### 4.2. Cluster Analysis of Passenger Position Data Based on Traditional DBSCAN Algorithm

The traditional DBSCAN algorithm is used to perform cluster analysis on the above-mentioned passenger position data. The threshold parameter *Minpts* is set to 4, and different clustering effects can be obtained by selecting different *Eps* values. The *x* of different colors in the figure represents different categories. *o* represents the noise point.

When *MinPts* = 4, *Eps* = 1.0, the clustering effect of the algorithm is shown in Figure 11a. Only the cluster analysis of high-density passenger position data can be achieved, but for low-density passenger position data, it is basically impossible to complete. Clustering results in a large number of passenger position data being treated as noise points, and it is not possible to present the aggregation and distribution characteristics of passenger mass transfer behavior.

When *MinPts* = 4, *Eps* = 1.5, the clustering effect of the algorithm is shown in Figure 11b, which can satisfy the cluster analysis of low-density distribution of passenger position data. However, because the value of the neighborhood parameter *Eps* is too large, it cannot correctly identifying the clustering and distribution characteristics of passenger clusters in similar areas results in the high density distribution of passenger location data being clustered into the same category.





**Figure 11.** DBSCAN algorithm for passenger location data clustering effect chart (a)  $MinPts = 4, Eps = 1.0$ ; (b)  $MinPts = 4, Eps = 1.5$ .

It can be seen that using the traditional DBSCAN algorithm, cluster analysis is performed on the passenger position data with uneven density distribution. If the global  $Eps$  parameter value is selected by referring to the high-density passenger position data, the low-density passenger position data cannot be clustered, and a large number of noise points are generated. If the global  $Eps$  parameter value is selected by referring to the low-density passenger position data, the high-density passenger position data will be merged into the same cluster, which affects the accuracy of the clustering result.

### 4.3. Cluster Analysis of Passenger Position Data Based on Improved DBSCAN Algorithm

Using the DBSCAN algorithm based on the Gaussian mixture model proposed in this paper, the clustering test is conducted for the above dataset. The Gaussian mixture model was used to fit the test passenger position data set, and the model order was set to  $m = 4$ . The probability density function formula was obtained from Equations (1) and (2) to fit the distribution of the passenger position data set on the platform at the moment. feature. In the fitting process, the EM algorithm parameters given in Table 1 are used to solve the process, and the unknown parameter  $\theta_i = \{\alpha_i, \mu_i, \Sigma_i\}$  ( $1 \leq i \leq 4$ ) in the Gaussian mixture model is estimated. The estimated results of each parameter are shown in Table 5.

**Table 5.** Gaussian mixture model parameter estimation results.

Gaussian Branch	Mixing Ratio	Mean (x,y)	Covariance Matrix [2*2 Order]
1	0.3003	(13.4299, 5.4690)	[25.7538 -10.7629; -10.7629 14.6453]
2	0.1864	(46.9046, 3.7905)	[25.7538 0.1623; 0.1623 24.6453]
3	0.2322	(73.2017, 6.4068)	[16.8556 -0.8974; -0.8974 17.0216]
4	0.2811	(106.4897, 4.7309)	[30.2539 15.4563; 15.4563 16.1026]

The location of the passenger's location determines the range of the mean value of the single Gauss distribution to a certain extent, and the point of the passenger location data is fitted to the corresponding single Gauss distribution, while the remaining test passenger location data is fitted to the most suitable single Gauss distribution.

Figure 12 shows the iterative process that the parameters of the Gauss model are gradually convergent and stable.

By setting the appropriate threshold and dividing the data set, we can get the hierarchical result for the dataset, as shown in Figure 13.

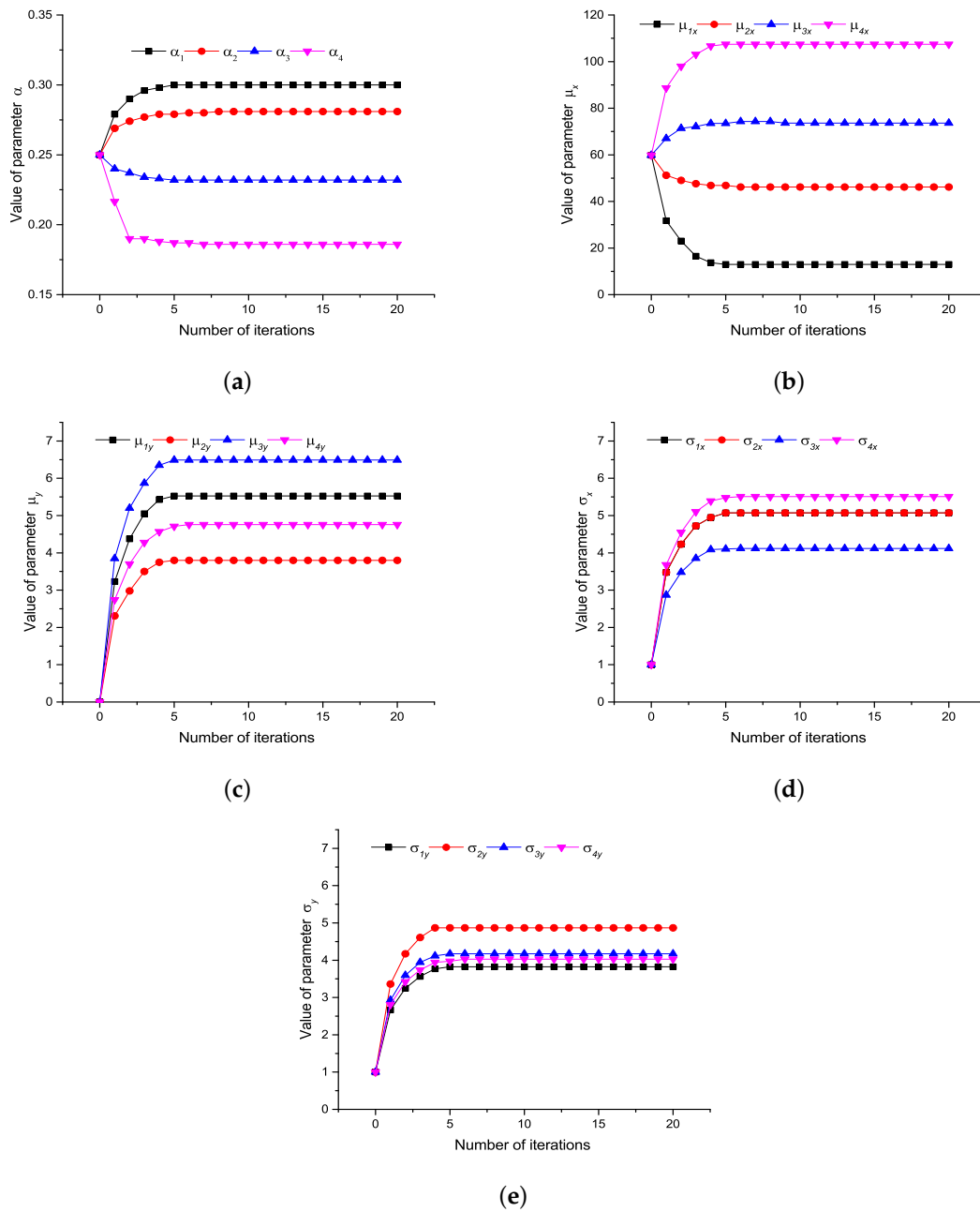
According to the parameter determination method of the single density hierarchical data set, the clustering parameters of the first density data of the data concentration are  $Minpts = 4, Eps = 1.0$ , and the clustering parameters of the second density hierarchical data are  $Minpts = 4, Eps = 1.5$ . According to the order from small to large, the clustering is carried out in order, the higher density level of passenger location data is first clustered, and then the data of the lower density level are clustered, and the cluster is labeled as  $C_{ij}$  (the  $j$  cluster of the density level is  $D_i$ ) so as to ensure that the sample points completing the clustering will not be repeated again. When the data of all density levels are clustered, the data points that are not clustered are labeled as noise points. With the local clustering results of different density level data, the aggregation and distribution characteristics of the passenger group behavior on the station platform can be obtained, as shown in Figure 14.

Compared with the clustering results shown in Figure 11a,b, the improved algorithm has a better clustering effect for high density distribution and low density distribution of passenger location data. At the same time, a large number of passenger location data are avoided to be mishandled as noise points, thus overcoming the defects of the traditional algorithm for the failure of the passenger cluster in the similar areas.

#### 4.4. Passenger Location Aggregation Visualization

In order to intuitively describe the differences in the number, size, and degree of density between different passenger clusters, visualization of the heat map is used to present the aggregation and distribution characteristics of the passenger groups so as to facilitate the operator's organization of the passengers and the evacuation of abnormal passenger flows. By establishing a reasonable mapping relationship between the passenger cluster characteristics and the visualization object, the discrete parameters of the passenger cluster are converted into a continuous form, and the passenger position data distribution characteristics are transmitted using the color change.

First, the passenger cluster space coordinates are projected to the interface coordinates, and the average of the coordinates of each passenger location in the cluster is worth the centroid of the map. The appropriate icon size is set by the envelope area method, so that it can truly reflect the range of the passenger cluster activity, and the passenger cluster is characterized by setting the color coverage range. In order to describe the difference between the density of the different passenger clusters and standardize the maximum density of the density, the degree of the aggregation of the passenger cluster is expressed by the linear gradient color, so as to ensure the fast and accurate identification of the passenger cluster with different distribution characteristics. The aggregation and distribution characteristics of the tested passenger groups at a certain time on the platform are visualized, as shown in Figure 15.



**Figure 12.** Parameter convergence process of Gauss mixed model (a)  $\alpha$  convergence process; (b)  $\mu_x$  convergence process; (c)  $\mu_y$  convergence process; (d)  $\Sigma_x$  convergence process; (e)  $\Sigma_y$  convergence process.

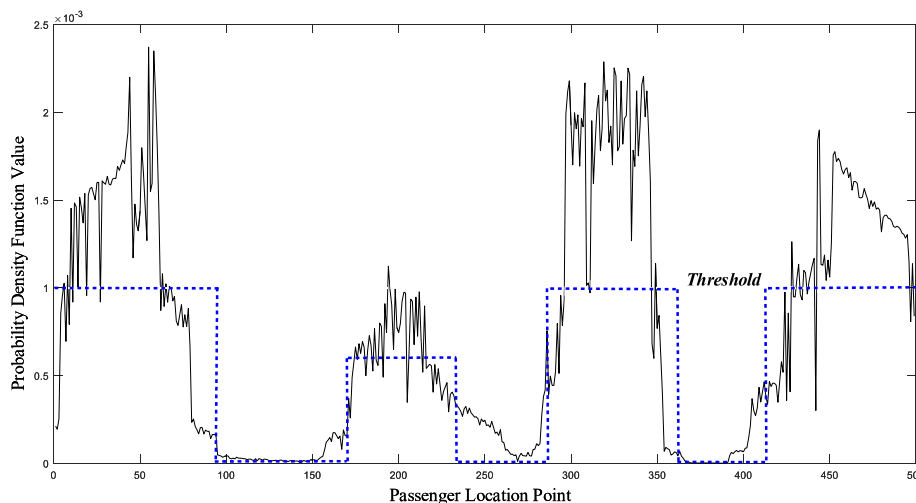


Figure 13. Schematic diagram of passenger location data density level division.

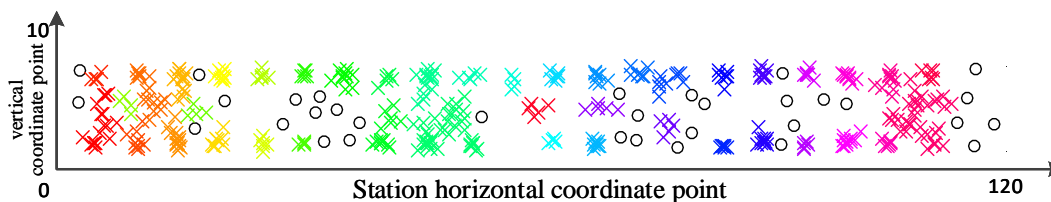


Figure 14. Improved DBSCAN algorithm for passenger location data clustering ( $MinPts = 4; Eps1 = 1.0, Eps2 = 1.5$ ).

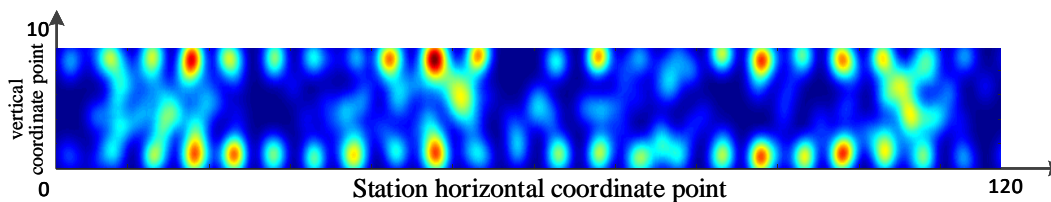


Figure 15. Passenger location aggregation visualization.

In Figure 15, the degree of aggregation of passengers is gradually reduced by red, yellow, and green. The red area is the area where passengers gather most densely. The blue area shows that there is basically no aggregation of passengers. Therefore, according to the visualization of the passenger’s location aggregation analysis results, the basic situation of the aggregation and distribution of the passenger group behavior can be effectively identified.

### 5. Conclusions

The purpose of this study is to cluster and analyze the location data with inhomogeneous density distribution. The Gaussian distributed probability density function is used to implement the dataset delamination, which eliminates the effect of uneven density distribution on the clustering effect. The rapid expansion of clusters improves the inefficiency of algorithm execution. The numerical experiments show the effectiveness of the algorithm by using the open data set and the influence of clustering accuracy, clustering time efficiency and noise intensity on clustering results. This method is applied to the recognition of passenger clusters at urban rail transit stations and verifies the practicability of the algorithm.

Through accurate clustering analysis of passenger location data in the station, not only can the aggregation of conventional areas be clearly presented, but also real-time passenger hot spots can be found in the station, so that the operators can organize and guide the behavior of passenger groups. In addition, when an emergency occurs in a station, the aggregation and distribution characteristics of passenger group behavior at the current moment can be used to continuously update and display the changes of passenger group behavior during evacuation process. Station operators can communicate in real time through information sharing, and adopt targeted measures such as manual and broadcast guidance to make reasonable arrangements. The number and location of guides can effectively adjust the passenger route selection scheme and improve the passenger evacuation speed.

The method in this paper needs to set the order parameter of the Gaussian mixture model artificially. In the follow-up study, researchers can explore whether information can be obtained from the distribution of data sets to determine the order of the model. In addition, when clustering location data, researchers can further consider extracting trajectory characteristics and mining the internal rules of trajectory data. So as to provide support for emergency decision-making.

**Author Contributions:** Data curation, P.Z.; Funding acquisition, G.Z.; Investigation, P.Z.; Methodology, X.L.; Software, X.L.; Supervision, G.Z.; Validation, X.L.; Writing—original draft, X.L.

**Funding:** This work is supported by the National Science Foundation of China (No. 61872037), the Fundamental Research Funds for the Central Universities (No. 2019YJS103).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Shirخورshidi, A.S.; Aghabozorgi, S.; Wah, T.Y.; Herawan, T. Big Data Clustering: A Review. In Proceedings of the International Conference on Computational Science and Its Applications-ICCSA 2014, Guimaraes, Portugal, 30 June–3 July 2014.
2. Wang, Y.; Qin, K.; Chen, Y.; Zhao, P. Detecting anomalous trajectories and behavior patterns using hierarchical clustering from taxi gps data. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 25. [[CrossRef](#)]
3. Xu, X.; Zhou, J.; Yu, L.; Xu, Z.; Zhao, X. Taxi-rs: Taxi-hunting recommendation system based on taxi gps data. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 1716–1727. [[CrossRef](#)]
4. Kai, L.; Du, T.C. Building a targeted mobile advertising system for location-based services. *Decis. Support Syst.* **2012**, *54*, 1–8.
5. Garcia-Rubio, C.; Redondo, R.P.; Campo, C.; Vilas, A.F. Using entropy of social media location data for the detection of crowd dynamics anomalies. *Electronics* **2018**, *7*, 380. [[CrossRef](#)]
6. Chepuri, A.; Joshi, S.; Arkatkar, S.; Joshi, G.; Bhaskar, A. Development of new reliability measure for bus routes using trajectory data. *Transp. Lett. Int. J. Transp. Res.* **2019**, 1–12. [[CrossRef](#)]
7. Wang, H.; Huang, H.; Ni, X.; Zeng, W. Revealing Spatial-Temporal Characteristics and Patterns of Urban Travel: A Large-Scale Analysis and Visualization Study with Taxi GPS Data. *ISPRS Int. J. Geo-Inf.* **2019**, *8*, 257. [[CrossRef](#)]
8. Lin, L.; Lei, Y.; Haihong, Z.; Rongrong, D.; Torres, C.C. Explorative analysis of Wuhan intra-urban human mobility using social media check-in data. *PLoS ONE* **2015**, *10*, e0135286.
9. Zou, Q.; Yao, X.; Zhao, P.; Wei, H.; Ren, H. Detecting home location and trip purposes for cardholders by mining smart card transaction data in Beijing subway. *Transportation* **2018**, *45*, 919–944. [[CrossRef](#)]
10. Hasan, U.; Satish, V. Urban activity pattern classification using topic models from online geo-location data. *Transp. Res. Part C Emerg. Technol.* **2014**, *44*, 363–381. [[CrossRef](#)]
11. Samiul, H.; Ukkusuri, S.V.; Zi-Ke, Z. Location contexts of user check-ins to model urban geo life-style patterns. *PLoS ONE* **2015**, *10*, e0124819.
12. Montoliu, R.; Blom, J.; Gatica-Perez, D. Discovering places of interest in everyday life from smartphone data. *Multimed. Tools Appl.* **2013**, *62*, 179–207. [[CrossRef](#)]

13. Zheng, Y.; Zhang, L.; Xie, X.; Ma, W.Y. Mining Interesting Locations and Travel Sequences from GPS Trajectories. In Proceedings of the 18th International Conference on World Wide Web (WWW 2009), Madrid, Spain, 20–24 April 2009.
14. Zheng, Y.; Zhang, L.; Ma, Z.; Xie, X.; Ma, W.Y. Recommending friends and locations based on individual location history. *ACM Trans. Web* **2011**, *5*, 5. [[CrossRef](#)]
15. Guo, C.; Liu, J.N.; Fang, Y.; Luo, M.; Cui, J.S. Value extraction and collaborative mining methods for location big data. *J. Softw.* **2014**, *25*, 713–730.
16. Shekhar, S.; Jiang, Z.; Ali, R.; Eftelioglu, E.; Tang, X.; Gunturi, V.; Zhou, X. Spatiotemporal data mining: A computational perspective. *ISPRS Int. J. Geo-Inf.* **2015**, *4*, 2306–2338. [[CrossRef](#)]
17. Aldstadt, J. Spatial Clustering. In *Handbook of Applied Spatial Analysis*; Springer-Verlag: Berlin/Heidelberg, Germany, 2010.
18. Mao, Y.; Zhong, H.; Qi, H.; Ping, P.; Li, X. An adaptive trajectory clustering method based on grid and density in mobile pattern analysis. *Sensors* **2017**, *17*, 2013. [[CrossRef](#)]
19. Ankerst, M.; Breunig, M.M.; Kriegel, H.P.; Sander, J. OPTICS: Ordering Points to Identify the Clustering Structure. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Philadelphia, PA, USA, 1–3 June 1999; Volume 28, pp. 49–60.
20. Elbatta, M.T.; Bolbol, R.M.; Ashour, W.M. A vibration method for discovering density varied clusters. *Isrn Artif. Intell.* **2012**, *2012*, 723516. [[CrossRef](#)]
21. Liu, P.; Zhou, D.; Wu, N. VDBSCAN: Varied Density Based Spatial Clustering of Applications with Noise. In Proceedings of the 2007 International Conference on Service Systems and Service Management, Chengdu, China, 9–11 June 2007; pp. 1–4.
22. Xia, L.N.; Jing, J.W. SA-DBSCAN: A self-adaptive density-based clustering algorithm. *J. Grad. Sch. Chin. Acad. Sci.* **2009**, *26*, 530–538.
23. Chen, Y.; Tang, S.; Bouguila, N.; Wang, C.; Du, J.; Li, H. A fast clustering algorithm based on pruning unnecessary distance computations in DBSCAN for high-dimensional data. *Pattern Recognit.* **2018**, *83*, 375–387. [[CrossRef](#)]
24. Kim, Y.; Shim, K.; Kim, M.S.; Lee, J.S. DBCURE-MR: An efficient density-based clustering algorithm for large data using MapReduce. *Inf. Syst.* **2014**, *42*, 15–35. [[CrossRef](#)]
25. Liu, Z. An efficient and scalable density-based clustering algorithm for normalize data. *Procedia Comput. Sci.* **2015**, *92*, 136–141.
26. Ros, F.; Guillaume, S. Dides: A fast and effective sampling for clustering algorithm. *Knowl. Inf. Syst.* **2016**, *50*, 543–568. [[CrossRef](#)]
27. Ghanavati, M.; Wong, R.K.; Fong, S.; Gholamian, M.R. Extending the grenade explosion approach for effective clustering. In Proceedings of the IEEE Tenth International Conference on Digital Information Management, Porto, Portugal, 19–21 September 2016; pp. 28–35.
28. Yang, M.S.; Lai, C.Y.; Lin, C.Y. A robust em clustering algorithm for gaussian mixture models. *Pattern Recognit.* **2012**, *45*, 3950–3961. [[CrossRef](#)]
29. Lei, Y.; Yang, T.; Chan, A.B. Density-preserving hierarchical EM algorithm: Simplifying Gaussian mixture models for approximate inference. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 1323–1337.
30. Gionis, A.; Mannila, H.; Tsaparas, P. Clustering aggregation. *ACM Trans. Knowl. Discov. Data* **2007**, *1*, 1–30. [[CrossRef](#)]
31. Karypis, G.; Han, E.H.; Kumar, V. Chameleon: Hierarchical clustering using dynamic modeling. *Computer* **1999**, *32*, 68–75. [[CrossRef](#)]
32. Brun, M.; Sima, C.; Hua, J.; Lowey, J.; Carroll, B.; Suh, E.; Dougherty, E.R. Model-based evaluation of clustering validation measures. *Pattern Recognit.* **2007**, *40*, 807–824. [[CrossRef](#)]

