Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 7535575, 11 pages
https://doi.org/10.1155/2022/7535575

Research Article

Label Propagation Clustering Algorithm Based on Adaptive Angle

Hui Du, Manjie Zhang, Zhihe Wang, Qiaofeng Zhai, and Xuyan Cao

The School of Computer Science and Engineering, Northwest Normal University, Lanzhou 730070, China

Correspondence should be addressed to Manjie Zhang; 2020222042@nwnu.edu.cn

Received 13 March 2022; Accepted 21 July 2022; Published 25 August 2022

Academic Editor: Xingsi Xue

The direction-based label propagation clustering (DBC) algorithm needs to set the number of neighbors (k) and the angle value (*degree*), which are highly sensitive. Moreover, DBC algorithm is not suitable for datasets with uneven neighbor density distribution. To overcome above problems, we propose an improved DBC algorithm based on adaptive angle and label redistribution (ALR-DBC). The ALR-DBC algorithm no longer input parameter *degree*, but dynamically adjusts the deviation angle through the concept of high-low density region to determine the receiving range. This flexible receiving range is no longer affected by the uneven distribution of neighbor density. Finally, those points that do not meet the expectations of the main direction are redistributed. Experiments show that the ALR-DBC algorithm performs better than DBC algorithm in most artificial datasets and real datasets. It is also superior to the classical algorithms listed. It also has good experimental results when applied to wireless sensor data annotation.

1. Introduction

The frontier development of computer science has focused on data mining and artificial intelligence in recent years. Cluster analysis is the most classical research in the unsupervised direction of machine learning. It is widely used in image analysis [1], knowledge discovery [2], medicine [3], pattern recognition [4], and other fields. In the case of unknown sample information, the points are divided by some similarity measurement method so that the similarity of points within clusters is high while that of points between clusters is low [5]. Many classical algorithms show good results on different datasets [6].

The basic idea of hierarchical clustering is bottom-up and merging layer by layer [7]. The clustering process starts with each point being a separate class, and then, the two classes with the highest similarity are merged and iteratively repeated. The similarity is mainly measured by distance. The closer the distance is, the higher the similarity is. The advantage of this approach is that it does not input the number of clusters and hierarchical relationship between classes can be discovered. The disadvantage is high time complexity and low efficiency [8]. K-means is an early classic and widely used algorithm in the field of data mining [9]. Compared with hierarchical clustering, it has a much lower time complexity. It has high scalability and compressibility when dealing with big datasets. It is highly dependent on the selection of the initial centroid, so it often takes several iterations to achieve better clustering results. And K-means algorithm cannot cluster nonspherical datasets [10]. Compared with K-means clustering method, affinity propagation clustering algorithm (AP) is more robust and accurate [11]. It starts with initializing two preset matrices and then iteratively updates them. The "responsibility" $r(p, z)$ represents the suitability of point z as the clustering center of point p. The "availability" $a(p, z)$ represents the suitability of point p to select point z as its cluster center in the current round. These two matrices are interrelated and determine the final clustering result. When both are large, it means that point z has strong competitiveness and is more likely to be selected as the clustering center. AP algorithm has good performance and efficiency. Different from the clustering centers in other algorithms, the exemplar (center of clustering) in AP algorithm is an exact data point in the original data. And it is started by inputting the similarity matrix, so the data are allowed to be asymmetric and the sum of squares of error is low. However, the complexity of AP algorithm is high and the running time is long when the data is large [12].

In order to solve the clustering problem of irregular shapes, density-based spatial clustering of applications with noise algorithm (DBSCAN) was proposed [13]. It uses density reachability and density connection to cluster by establishing the definition of a core point, boundary point, and noise point [14]. DBSCAN algorithm can achieve adaptive clustering. There is no need to give the parameter of expected cluster number, and the advantage of insensitivity to noise points is conducive to clustering. The disadvantage is that the parameter sensitivity is high, and small parameter changes may lead to large differences in the results. And the DBSCAN algorithm must specify a density threshold to remove noise points below this density threshold. Based on the above analysis, the clustering by fast search and find of density peaks (DPC) [15] is put forward on the premise that the two assumptions are true: the cluster center is surrounded by points with lower density than it, and the distance between these points and the cluster center is closest compared with other cluster centers. And it has a relatively far distance from the point where density is higher than itself. Only meeting these conditions at the same time can it be possible to become the clustering center. Its disadvantage is that it needs to calculate the distance between all points. The DPC algorithm also does not cluster well those sample sets with multidensity peaks [16].

In order to better cluster samples with uneven density distribution, scholars continue to explore. The reference [17] uses two parameters. The parameter k is used to determine the receiving direction, and the parameter *degree* is used to define the receiving range that can deviate from the receiving direction. When the density distribution is not uniform, the clustering effect of DBC algorithm will not be greatly affected. After a large number of experiments, it is known that if the first parameter k is not appropriate, the second parameter *degree* will be adjusted many times to achieve the desired clustering effect. After determining the k-nearest neighbor value on some datasets, the value of degree can only change in a small range; otherwise, the expected value cannot be achieved. Therefore, the DBC algorithm is improved. We reduce the parameter sensitivity by selecting the method of dynamically adjusting the deviation angle to determine the receiving range of each point and using the DBC algorithm to cluster the points. It can be seen from the evaluation index of the experimental part that the improved algorithm holds higher NMI and ARI on most of the tested datasets. In this paper, the new improved algorithm is named ALR-DBC algorithm.

Clustering also has outstanding performance in practical applications, such as applying it to the Internet of Things. The Internet of Things came into being with the vigorous development of the information technology industry. It connects the object with the network through the information sensing device according to the agreement. As the core of the Internet of Things, the perception layer can not only sense signals and identify objects but also has the function of processing and controlling. The wireless sensor network is an important part of perception layer. It realizes the data collection, processing, and transmission and sends the information to the network owner. To enhance the communication between sensor networks, it is essential to establish semantic connections between sensor ontologies in this field. Literature [18] proposed a new sensor ontology integration technology, which utilizes the debate mechanism to extract sensor ontology alignment, greatly improving the effectiveness of the whole wireless sensor network. It enhances the communication ability between wireless sensors and improves the performance of the whole network. At the same time, we also read the relevant literature on matching ontology [19, 20] to better understand the important role and help of these methods in this field. Inspired by this, this paper applies ALR-DBC algorithm to wireless sensor networks and conducts performance comparison experiments.

2. Related Work

In this chapter, we will explain the label propagation algorithm (LP) [21] and the direction-based label propagation algorithm (DBC). The basic idea of these two algorithms is clustering through the similarity between the sample points. The DBC algorithm is an improvement of LP algorithm by adding parameters of angle value. Through the description and comparison of these two algorithms, it is more helpful to understand the improved DBC algorithm.

2.1. LP Algorithm. It is assumed that each point can find k neighbors closest to it, which is the basic assumption of LP algorithm. These neighbors all have unique class tags. Then, the update of the cluster label is determined by the neighbor's label. Among the labels to which the neighbor belongs, the label that occupies the largest number is the new cluster label of the point. Repeat the process; when the label of all points do not change, the iteration ends [22]. In the LP algorithm, the number of iterations needs to be set to avoid being overcalculated affecting the final result. In the running process of the algorithm, there is no need to calculate any clustering index, nor to input the number of clustering. The disadvantage of LP algorithm is that the randomness of the sequence in the iterative process will lead to the same initial label setting, but the clustering results are very different. It may also be affected by the k nearest neighbors set, the maximum values of neighbor labels of some points are the same, so random selection is adopted to update the cluster labels. Based on this, scholars also put forward improvement strategies, such as introducing potential function [23] and LeaderRank value [24] to increase the weight of nodes or edges. In this way, the centroid effect can be preliminarily determined and the classification accuracy can be improved. The label entropy attribute [25] can also be added so that it can be sorted according to the label entropy of each point and avoid the random influence in the iteration process.

2.2. DBC Algorithm. DBC is a direction-based cluster label propagation clustering algorithm. It need not enter the number of classes that will eventually be generated and can cluster any shape of clusters with stable results. Distance and direction are the two basic physical metrics, which are helpful for clustering. The major difference between LP and DBC algorithm is that the DBC algorithm considers

the orientation relationship between sample points, while the LP algorithm considers the relationship between the numbers of labels to which neighboring points belong. The DBC introduces the second parameter *degree* and finds the direction and receiving range with the greatest density in each point's neighborhood. There is no feedback or update during tag propagation, so the selection of receiving range is very important. The general process of the algorithm is to find the receiver of each point according to the parameters set and select the point with the largest number of receivers as the starting point of this round. After it is assigned an initial label, the clustering starts and the points in the receiver list are classified into one category. These points continue to pass their labels to their respective receivers as new senders until a round of clustering ends without new senders. Next, the remaining unlabeled points continue the next round of clustering until all points have class labels, and the clustering is over. We assume that point P is an arbitrary sample point in the dataset. The mathematical concept shows that point multiplication reflects the "similarity" of two vectors. The more similar the two vectors are, the greater the point multiplication is. Therefore, the formula of receiving direction of point P is shown in

$$\vec{v} = \arg \max_{\vec{v}_i \in V} \sum_{\vec{v}_j \in V : \vec{v}_i \cdot \vec{v}_j \geq 0.5} \vec{v}_i \cdot \vec{v}_j. \quad (1)$$

The set V stores all the neighbor vectors of point P. $\vec{v}$ is the most densely distributed direction of the neighbors of point P.

After determining the receiving direction, the DBC algorithm also sets the maximum deviation angle to determine the receiving range of cluster labels. It refers to the maximum angle that a neighbor vector can deviate from the receiving direction, and points within this angle range can pass the cluster label to point P. The DBC algorithm also defines the concepts of sender and receiver. After the parameters are determined, each point can obtain its sender and receiver according to its receiving direction and maximum deviation angle. Then, the label propagation of DBC algorithm begins. Algorithm 1 describes the DBC algorithm.

We default that each time the tag number is updated and a new pass is started. The SenderList keeps storing the sample that currently has no tags and has the most receivers. ID represents the number of samples. NewSenderList is a list of new senders. Line 21 of Algorithm 1 is to find the number of samples that are not currently labeled.

```
1:  Num = Dataset.size
2:  ID = 0
3:  //Traverse SenderList to get its recipients. Assign
    //labels to unlabelled recipients and store them
    //in NewSenderList. Until all points have class
    //labels
4:  while Num > 0 do
5:    ID = ID + 1
6:    while SenderList.size > 0 do
7:      NewSenderList = []
8:      for i = 0; i < SenderList.size; i + + do
9:        Receivers = getReceivers(SenderList[i])
10:       for j = 0; j < Receivers.size; j + + do
11:         Receiver = receivers[j]
12:         if Receiver.label == 0 then
13:           Receiver.label = ID
14:           NewSenderList.append(receiver)
15:         end if
16:       end for
17:     end for
18:     SenderList.clear()
19:     SenderList = NewSenderList.copy()
20:   end while
21:   Num = NumberOfUnassignedPoints(dataset)
22: end while
```

Algorithm 1: Label transfer process of DBC.

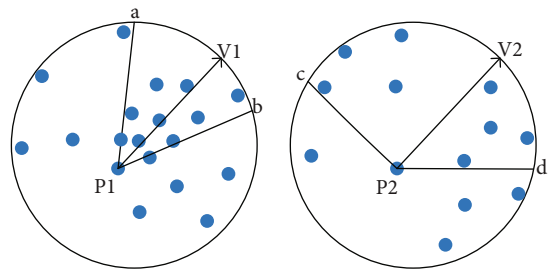
3. ALR-DBC Algorithm

3.1. Basic Idea. In describing the DBC algorithm, we mention that it introduces the concept of direction and angle as a reference condition based on the basic method of cluster label transfer. After determining the range of label transmission, the clustering process will begin. We consider that the number of neighbors and the angle value are global parameters. In fact, the neighbor density around each point is unevenly distributed. If the receiving range of each point can be dynamically changed according to its neighbor density, the accuracy of clustering results can be improved. We use Figure 1 to understand this hypothesis.

In Figure 1(a), the receiving range is all samples within the angle formed by $P1a$ and $P1b$. In Figure 1(b), the receiving range is all samples within the angle formed by $P2c$ and $P2d$. These points within the receiving range of point P can pass their labels to point P. It is concluded that when the distribution of neighbor points on both sides of the main direction is dense, the receiving range should be reduced, and when the distribution of neighbor points on both sides is sparse, the receiving range should be enlarged.

3.2. Adaptive Angle. In the previous section, we mentioned that DBC algorithm sets the *angle* as a global parameter. For different datasets, it can also achieve the ideal clustering effect after adjusting the parameter many times. However, there are also problems in the experimental process. The two parameters k and *degree* will affect each other. If k at the beginning is too large or too small for the current test dataset, it may require multiple attempts to achieve the expectation when setting the second parameter *degree*. So it will increase the uncertainty of clustering time. For this shortcoming, we introduce the concepts of high-density and low-density regions to explain the ALR-DBC algorithm.

In the sample set with class labels, we can find the obvious rule that the cluster distribution divided into one category is relatively dense, which can be represented by high-density regions. There will be sparsely distributed sample points between the two clusters, that is, the low-density region we want to refer to. In this way, we define the



(a) Dense main direction (b) Sparse main direction

Figure 1: The receiving range for the dynamic selection of $P1$ and $P2$.

distribution states of high-low-high and divide clusters. As shown in Figure 2, our goal is to find a continuous high-density area as a receiving area for point P.

The next question we need to think about is how to find the criteria for distinguishing high- and low-density regions. From the above, it can be seen that the main receiving direction is easily determined by formula (1). Then, based on this direction, it is feasible to use the ratio definition method to divide regions. When the condition of $A/B \geq M$ is satisfied, it is included in the receiving range of this point.

$$A = \sum_{\vec{v}_i \cdot \vec{v}_j \geq 0.5} \vec{v}_i \cdot \vec{v}_j. \tag{2}$$

All neighbor vectors of point P are stored in list V. As shown in formula (2), we traverse each vector V_i in list V, taking the dot product of each vector with the rest of its neighbors and summing them up. So each neighbor vector will get the corresponding A value. The value 0.5 in the constraint condition is consistent with the DBC algorithm. B is the value A of vector PV_1 in Figure 2. Because each point is receiving in a unique direction, the value B is fixed after locking a point P. M is a receiving threshold we set artificially.

We analyzed the scanning process according to the discriminant conditions. The idea of ALR-DBC algorithm is to scan the neighbor vectors from the main direction vector of point P to both sides and judge them in turn. The points that meet the receiving threshold on both sides of the main direction are put into the receiving range list of point P. Because the density distribution of each dataset is different, the advantage is that we will judge whether the direction currently scanned meets the receiving condition, rather than using fixed angle to define the range. Here is the dynamic adjustment process of the algorithm. The final receiving range includes all high-density direction vectors without crossing the low-density region. The specific scan is shown in Figures 3 and 4, showing the neighbor distribution of point P and the dynamic change process of one side of the main direction.

To help understand how the ALR-DBC algorithm dynamically selects the receiving angle of each point, a partial scanning process is drawn. As shown in Figure 4, point P finally finds a receiving boundary on the clockwise side that is a vector from p to c. Judge from vector a with the

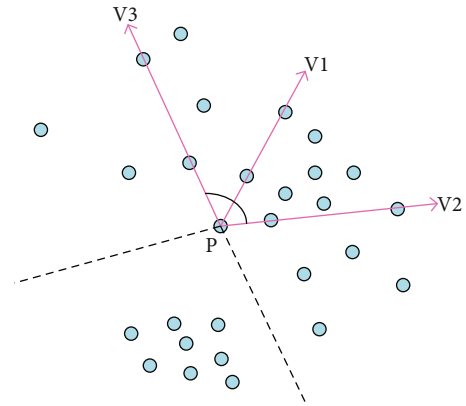


Figure 2: Cluster partition at high and low density.

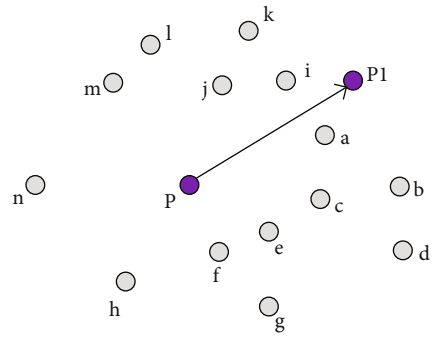


Figure 3: The main receiving direction and neighbor distribution of point P.

minimum angle away from the main direction. Since the vector meets the conditions that $A/B \geq M$, it is added to the receiving list of point P. Then continue scanning to point i in Figure 4(b) with the second smaller angle from the main direction. Repeat the above discrimination steps. Until the unqualified vector is found as shown in Figure 4(d), the vector is saved and recorded as the boundary vector of one side (assumed as the X side) of point P.

The focus of the next algorithm is to find the boundary vector on the other side of point P. We continue to scan the neighbor vector outward and judge it by inequality. If the condition is met and the vector is not on the side of X, it is included in the receiving range of point P; otherwise, it is not included and continues to scan. That is because if

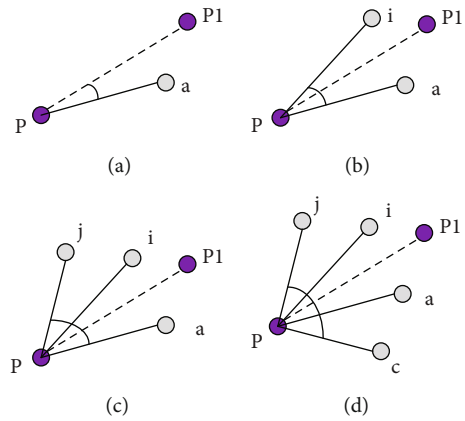


Figure 4: The process diagram of point P finding the receiving boundary on one side of the main direction.

this vector is on X side, it is already outside the reception boundary. Although it meets the density required by the receiving condition, it does not meet the continuous high-density area mentioned above. Then, when the scanned neighbor vector does not meet the density required by the receiving condition, if it is not on the side of X, we find the boundary vector on the other side of point P (assumed as the Y side), that is, the low-density region after the continuous high-density region. So far, we have set the receiving range of point P.

3.3. Label Redistribution. In describing the idea of the DBC algorithm, we mention that it can determine an edge of the receiving range by finding the receiving direction at point P. The other side is determined by the angle value, which allows neighbor vectors within a certain deviation angle to pass their class labels to point P. When the program runs, the ideal clustering results can be achieved by adjusting the two parameters. However, we need to make it clear that point P should best be grouped with its principal direction vector. In the original DBC algorithm, when a point becomes a new cluster label sender, it will immediately pass its class label to all its receivers. Assuming that the receiver's receiving direction label is different from the tag, the final clustering effect will be worse than expected. We illustrate this by using a spiral dataset.

As shown in Figure 5, it is the clustering diagram of the spiral dataset when only the receiving angle is dynamically adjusted. We observed that the points were divided into three categories and were spirally distributed. At the top of the figure, there is a blue sample point F next to the green sample point. Obviously, the sample point is wrongly classified at this time, and it should actually be classified as a green class. The reason for this result is that the receiving range of a blue point in the dataset includes F. At the beginning of each new round of clustering, the transfer of clustering labels starts from the unlabeled samples with the largest number of receivers. It can be found from Figure 5 that the closer to the center, the closer the spiral is, that is, the higher the density of points is. Therefore, the sample points classified as the blue class have labels earlier than the green class. Then, F is given the wrong class label in this round. When F has a

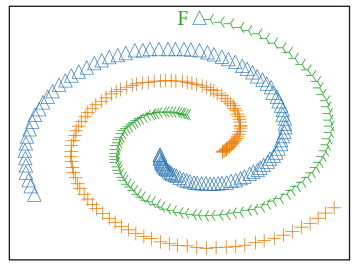


Figure 5: Clustering graph with only improved receiving angle ($k = 8$).

label, the subsequent clustering process is no longer assigned, considering only the data points that are not currently labeled. We consider whether we can change the clustering order so that the points of the green class get the class label first. Although this ensures the correct classification of F, there are few receivers of this part of the points, which will force the points that should be classified into one category to be classified into multiple categories because they cannot be reached in a round of label transmission. Therefore, we adopt another improvement idea to solve this problem. After all points have class labels, all points are traversed to determine whether the current label is consistent with the label in the receiving direction. If not, the cluster label on the receiving direction is redistributed to that point. This is a search lookup and redistribution process. The class label of sample points is most likely to be consistent with its receiving direction, because the receiving direction represents the most densely distributed region of neighbors. These regions are also most likely to be clustered eventually.

3.4. Algorithm Description. Through the determination of the above adaptive angle, the receiving range of each point is obtained, and then, the DBC algorithm is used for clustering. After the first clustering, judge whether there are samples that have not been assigned labels. If so, start the second round of clustering. From the remaining samples without labels, continue to select the most current recipients as a new starting point to start a new round of label delivery. Continue the iteration until all points have class tags. To

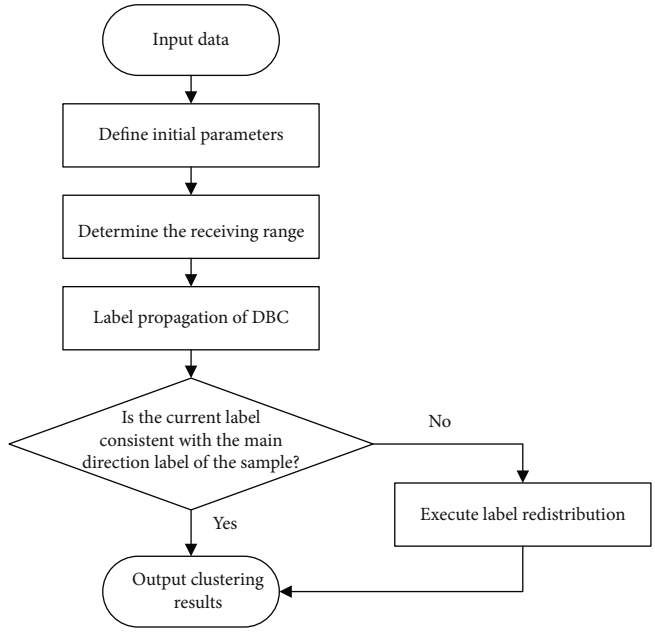


Figure 6: Flowchart of ALR-DBC.

have a clearer explanation, we provide the flowchart of the whole process as shown in Figure 6.

Algorithm 2 describes the process of determining the receiving angle of data point i. VecList stores all neighbor vectors of data point i. sList is the value computed by the neighbor vector of data point i according to formula (2). The dictionary a is arranged in descending order to represent the points that are scanned outward from the principal direction vector according to the increment of the angle value. MaxV takes the maximum value in sList. The Vector list stores the receiving direction vector of each point. M is the acceptance threshold mentioned in Section 3.2. The range list stores the sender within the current deviation range of point i.

In the fifth line, we determine the receiving direction of data point i. Lines 6 to 9 compute the dot product of the received direction vector of data point i and all its neighbor vectors and store it in dictionary a. Lines 12 to 19 determine the receiving range of one side. Lines 20 to 28 determine the receiving range on the other side.

Then, we use the DBC algorithm for clustering. The obtained clustering results are stored in the Result list. At this point, we begin to redistribute cluster labels. The existing label of each point is compared with the label of its receiving direction. If the labels do not match, the existing label for that point is modified.

3.5. Parameter Selection. The first parameter to be set in the algorithm is k. It is the number of k neighbors of the sample point that are closest to it. In general, k is between 5 and 30, and some datasets need larger k. At this time, k is between 30 and 60. M is a low-sensitivity parameter. When M takes an increasing value, it makes the receiving angle smaller, and then, the final number of clusters becomes larger. In this paper, a large number of experiments show that the clustering effect for most datasets is ideal when M is between 0.6 and 0.7. It is easier to determine than the angle parameter of DBC algorithm.

4. Experimental Result

This chapter evaluates the clustering effect of ALR-DBC algorithm. Comparison algorithms are DBC algorithm, FCM algorithm, and DBSCAN algorithm. Since FCM algorithm is a popular fuzzy clustering algorithm, DBSCAN algorithm is a classical density-based clustering algorithm. Therefore, as a comparison, it can prove the superiority of ALR-DBC algorithm. Test datasets include artificial datasets and real datasets. Artificial datasets are Flame, Threecircles, Twomoons, Aggregation, Lsun, and Hard [17]. Real datasets include Iris [26], Dermatology [27], Balance, Vote, and Vowel. The introduction to these artificial datasets is listed in Table 1.

The experimental environment is AMD Ryzen 5 4600 H @ 3.00 GHz. The memory is 16 GB. The programming environment is Python 3.8 and the compiler is PyCharm.

Normalized interactive information (NMI) [28], adjusted Rand index (ARI) [29], and Homogeneity are selected as the evaluation indexes of clustering.

4.1. Artificial Dataset. In Table 1, Flame is a dataset with overlapping regions, which can test whether the improved algorithm can have good clustering results for such datasets. Threecircles is a typical representative of the nonconvex dataset, and our algorithm can cluster well. The Hard dataset is characterized by large density differences between clusters. Aggregation and Twomoons datasets are composed of irregular clusters. The Lsun dataset is composed of clusters with uneven density distribution. In the original DBC algorithm, through the continuous adjustment of two parameters, we

```
//All points in the sample set are traversed to
//dynamically determine their receiving range
for i from 1 to the maximum number of dataset do
    Compute sList
    MaxV = Max(sList)
    l = Max(sList).label
    Vector[i] = VecList[l]
    for j = 0; j < VecList.size; j + + do
        a1.append(Vector[i] · VecList[j])
        a = a1.sorted(reverse = true)
    end for
    for each u ∈ a do
        if sList[u]/MaxV > M then
            Range[i].append(u)
        else
            Boundary = VecList[u]
            R = u and exit this cycle
        end if
    end for
    for n from R to the remaining points do
        D = sList[n]/MaxV
        if D/MaxV > M and n not in the side of boundary then
            Range[i].append(n)
        else if D/MaxV < M and n not in the side of boundary then
            Break
        else
            Continue
        end if
    end for
end for
```

Algorithm 2: Receiving angle.

Table 1: Artificial datasets for testing.

Name	Instances	Clusters
Flame	240	2
Threecircles	299	3
Twomoons	1502	2
Aggregation	788	7
Lsun	400	3
Hard	1501	3

can obtain better clustering results of these six datasets. Figure 7 shows the clustering effect of the ALR-DBC algorithm when only one parameter is adjusted.

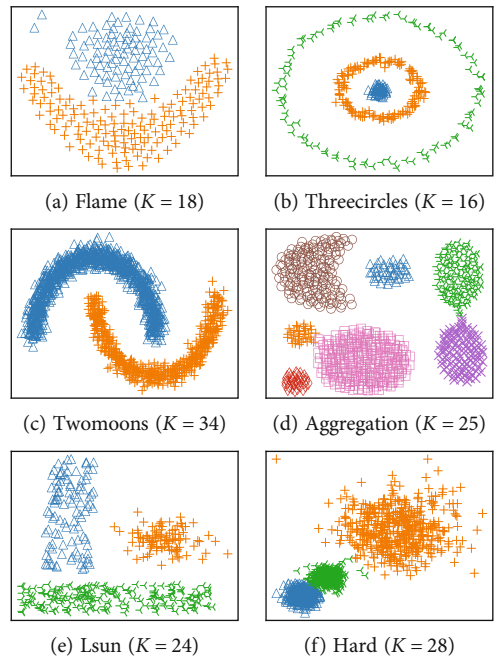
When the DBC algorithm executes the parameter set in Table 2, the optimal NMI values can be achieved. On these artificial datasets, the ALR-DBC algorithm achieves the best NMI of DBC algorithm under the condition that M is 0.6. For some datasets, even a small range of fluctuations in the angular parameters of the DBC algorithm can affect the final clustering results. We can analyze it in Table 3.

In Table 3, we use the Compound dataset as the experimental subject. The NMI changes are observed by adjusting the parameters. In the DBC algorithm, when k is 9 and *degree* is 90, NMI can reach 0.8812. However, when the angle value is 86 degrees, it can be seen that the NMI value is reduced to 0.8170. The number of points classified wrong increases, and their density distribution is uniform. At the same angle value, the decrease of k from 9 to 8 also leads to a decrease in NMI. Since the dataset is two-dimensional, the distance to observe these points is also very close. It is precisely because the change of angle value leads to the deviation of experimental results.

To better illustrate the influence of parameters in the ALR-DBC algorithm, we performed a parameter sensitivity test on the Compound dataset. NMI was taken as the evaluation index, as shown in Table 4. We can find that when M gradually increases from 0.60, NMI does not fluctuate significantly.

4.2. Real Dataset. The ALR-DBC algorithm shows good clustering effect on the artificial dataset in the previous section. In this section, we use real high-dimensional datasets to further test its clustering effect and compare with other algorithms. In the experiment, NMI, ARI, and Homogeneity are used to evaluate the performance. Their ranges are all $[0, 1]$. The larger the value in this interval is, the closer the clustering result is to the real label, and the better the effect is. The UCI dataset used in the experiment is shown in Table 5.

For different algorithms, it is necessary to set its iterative method according to the characteristics of parameters to find the best evaluation index. For FCM algorithm, input the number of clusters and run the program many times.



(a) Flame ($K = 18$) (b) Threecircles ($K = 16$)

(c) Twomoons ($K = 34$) (d) Aggregation ($K = 25$)

(e) Lsun ($K = 24$) (f) Hard ($K = 28$)

Figure 7: Clustering results obtained by ALR-DBC on six datasets (the experimental parameter k values for each dataset are given, and M is 0.6 for all).

Table 2: The best experimental results of DBC algorithm.

Name	k	*Degree*	NMI
Flame	18	35	0.9269
Threecircles	16	60	1.0
Twomoons	34	80	1.0
Aggregation	20	60	0.9956

Table 3: Experimental results of DBC on Compound.

k	*Degree*	NMI
9	90	0.8812
9	86	0.8170
8	90	0.8285
10	90	0.8768
10	80	0.8051

Table 4: Experimental results of ALR-DBC on Compound.

k	M	NMI
9	0.60	0.8977
9	0.62	0.8977
9	0.64	0.8977
9	0.66	0.8475
9	0.68	0.8475

When the clustering results do not reach better within at least 20 times, it proves that we have obtained the desired experimental results. For DBSCAN, we set a reasonable iteration range for two parameters e and MinPts, respectively.

Table 5: UCI datasets for testing.

Name	Instances	Attributes	Clusters
Iris	150	4	3
Dermatology	358	34	6
Balance	625	4	3
Vote	435	16	2
Vowel	990	13	11
Landsat	2000	36	6
Ecoli	336	7	8
WDBC	569	30	2

In our experiment, e is traversed from 0.01 to 1, and MinPts traverses from 1 to 30. For the improved algorithm ALR-DBC, the adjustment range of parameter k is 3 to 50. Then, we get three groups of evaluation index of each algorithm as shown in Table 6.

In general, the clustering effect of the DBC algorithm and ALR-DBC algorithm in these eight sets of data is better. The ALR-DBC algorithm uses the advantage of adaptive angle to divide the receiving range of sample points under different distributions. It works best in Iris, Dermatology, and Balance datasets. The two evaluation indexes of ALR-DBC algorithm and DBC algorithm are consistent and optimal on Vote dataset and close to the optimal clustering effect on Landsat dataset. Compared with the DBC algorithm, the ALR-DBC algorithm performs better in all indicators on five datasets. The FCM algorithm performs better on WDBC dataset because it has better adaptability for datasets with large density difference and overlapping between clusters. This is the advantage that other comparison algorithms do not have.

Table 6: Comparison of clustering effects on the UCI datasets.

Dataset	Criteria	DBSCAN	FCM	DBC	ALR-DBC
Iris	NMI	0.7336	0.7433	0.8705	0.8980
	ARI	0.5681	0.7287	0.8857	0.9221
	Homogeneity	0.5793	0.7404	0.8696	0.8980
Dermatology	NMI	0.6205	0.7644	0.8102	0.8486
	ARI	0.4151	0.6866	0.7307	0.7871
	Homogeneity	0.6484	0.6878	0.7859	0.7879
Balance	NMI	0.0178	0.0093	0.2082	0.2104
	ARI	0.0200	0.0075	0.0424	0.0440
	Homogeneity	0.0101	0.0083	0.5125	0.5181
Vote	NMI	0.3977	0.4547	0.4942	0.4942
	ARI	0.4480	0.5232	0.5709	0.5709
	Homogeneity	0.5034	0.4633	0.5030	0.5030
Vowel	NMI	0.5317	0.5420	0.5576	0.5693
	ARI	0.4170	0.4004	0.4221	0.4579
	Homogeneity	0.4902	0.5503	0.5242	0.5199
Landsat	NMI	0.5768	0.6054	0.6415	0.6365
	ARI	0.4153	0.5216	0.5966	0.6306
	Homogeneity	0.5026	0.6091	0.6667	0.6395
Ecoli	NMI	0.6161	0.5448	0.6550	0.6571
	ARI	0.1459	0.3620	0.7009	0.7165
	Homogeneity	0.6285	0.6431	0.5558	0.5580
WDBC	NMI	0.3790	0.6151	0.5203	0.5258
	ARI	0.4661	0.7304	0.5189	0.5883
	Homogeneity	0.3868	0.6080	0.6528	0.6656

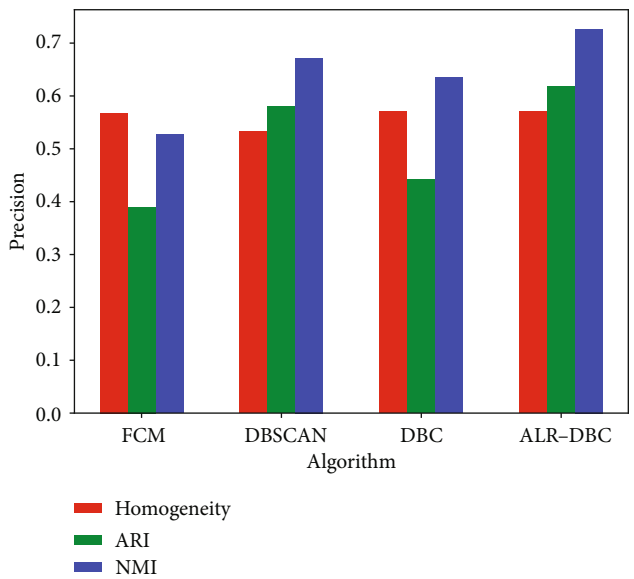


Figure 8: Performance comparison of different algorithms.

4.3. Application. Our improved algorithm can also be applied to wireless sensor data annotation in IoT. In this section, a set of high-risk behavior monitoring data of elderly volunteers in clinical activities is used to verify the effectiveness of ALR-DBC algorithm in practical application. The perception layer of the IoT is covered by various sensors and sensor gateways. Their function is to identify objects and collect information. The test data we used were provided by the research in literature [30] and classified by clinical activity status. Through ALR-DBC algorithm, we obtain the class labels of activity data in all monitoring time periods and compare the evaluation indicators with other algorithms mentioned above, as shown in Figure 8. Experiments show that our method has a good application effect in wireless sensor data annotation of the IoT.

5. Conclusions

In this paper, we reduce the number of parameters of the DBC algorithm through the strategy of adaptive angle, and the problem of misclassification caused by the order of scanning points in the clustering process is solved by the method of redistribution of cluster labels. In the experimental process, we found that it can also well separate clusters with large density difference and nonuniformity. It shows good

clustering effect on artificial datasets. In some UCI datasets, it can surpass the clustering effect of the original algorithm. We apply the improved algorithm to wireless sensor data annotation. Good application effect can be obtained through experiments. For those datasets with more overlapping regions between different clusters, although the evaluation metrics have been improved, they still cannot achieve the desired results. In future research, we intend to improve the ALR-DBC algorithm by combining the knowledge of depth measurement learning. We will also discuss its prospects in the field of IoT to make it more effective.

Data Availability

In this paper, the experiments using real datasets are available at url = "http://archive.ics.uci.edu/ml". References have been marked at corresponding positions in the article.

Conflicts of Interest

The authors declare that there is no conflict of interest in publishing this article.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (61962054).

References

[1] A. Rebiai, B. B. Seghir, H. Hemmami et al., "Clustering and discernment of Algerian bee pollen using an image analysis system," *Algerian Journal of Chemical Engineering*, vol. 1, no. 2, pp. 41–48, 2021.

[2] K. Vantas and E. Sidiropoulos, "Knowledge discovery using clustering analysis of rainfall timeseries," in *EGU General Assembly Conference Abstracts*, pp. 21–14758, Vienna, Austria, 2021.

[3] N. Han, S. Qiao, G. Yuan, P. Huang, D. Liu, and K. Yue, "A novel Chinese herbal medicine clustering algorithm via artificial bee colony optimization," *Artificial Intelligence in Medicine*, vol. 101, article 101760, 2019.

[4] A. Caggiano, F. Napolitano, and R. Teti, "Hierarchical cluster analysis for pattern recognition of process conditions in die sinking edm process monitoring," *Procedia CIRP*, vol. 99, no. 2, pp. 514–519, 2021.

[5] M. Du, S. Ding, and H. Jia, "Study on density peaks clustering based on *k*-nearest neighbors and principal component analysis," *Knowledge-Based Systems*, vol. 99, pp. 135–145, 2016.

[6] F. A. Ferdous, "A conceptual review on different data clustering algorithms and a proposed insight into their applicability in the context of COVID-19," *Journal of Advances in Technology and Engineering Research*, vol. 6, no. 2, pp. 58–68, 2020.

[7] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[8] A. Dogan and D. Birant, "*K*-centroid link: a novel hierarchical clustering linkage method," *Applied Intelligence*, vol. 52, no. 5, pp. 5537–5560, 2022.

[9] J. A. Hartiganm and M. A. Wong, "Algorithm as 136: a *k* -means clustering algorithm," *Journal of the Royal Statistical Society*, vol. 28, no. 1, pp. 100–108, 1979.

[10] L. Y. Tseng and S. B. Yang, "A genetic clustering algorithm for data with non-spherical-shape clusters," *Pattern Recognition*, vol. 33, no. 7, pp. 1251–1259, 2000.

[11] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[12] Q. Zhang and X. Chen, "Agglomerative hierarchical clustering based on affinity propagation algorithm," in *2010 Third International Symposium on Knowledge Acquisition and Modeling*, pp. 250–253, Wuhan, 2010.

[13] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A densitybased algorithm for discovering clusters in large spatial databases with noise," *AAAI Press*, vol. 96, no. 34, pp. 226–231, 1996.

[14] Y. Chen, L. Zhou, N. Bouguila, C. Wang, Y. Chen, and J. du, "BLOCK-DBSCAN: fast clustering for large scale data," *Pattern Recognition*, vol. 109, article 107624, 2021.

[15] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[16] Y. Li, W. J. Zhou, and H. K. Wang, "F-DPC: fuzzy neighborhood-based density peak algorithm," *IEEE Access*, vol. 8, pp. 165963–165972, 2020.

[17] N. Xiao, K. Li, X. Zhou, and K. Li, "A novel clustering algorithm based on directional propagation of cluster labels," in *2019 International Joint Conference on Neural Networks*, pp. 1–8, Budapest, Hungary, 2019.

[18] X. S. Xue, X. J. Wu, C. Jiang, G. Mao, and H. Zhu, "Integrating sensor ontologies with global and local alignment extractions-financial big data based on Internet of things and wireless network communication," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 8944618, 12 pages, 2021.

[19] X. S. Xue and Q. H. Huang, "Generative adversarial learning for optimizing ontology alignment," *Expert Systems*, vol. 39, pp. 1–12, 2022.

[20] X. S. Xue and J. Zhang, "Matching large-scale biomedical ontologies with central concept based partitioning algorithm and adaptive compact evolutionary algorithm," *Applied Soft Computing*, vol. 106, article 107343, 2021.

[21] J. G. Liang, X. F. Zhou, Y. Sha, P. Liu, L. Guo, and S. Bai, "Unsupervised clustering strategy based on label propagation," in *IEEE International Conference on Data Mining Workshops*, pp. 788–794, Dallas, TX, USA, 2013.

[22] X. J. Zhu, *Semi-Supervised Learning with Graphs*, Carnegie Mellon University ProQuest Dissertations Publishing, 2005.

[23] J. X. Shi and L. X. Zhang, "Label propagation algorithm based on potential function for community detection," *Journal of Computer Applications*, vol. 34, no. 3, p. 738, 2014.

[24] M. Y. Shi, Y. Zhou, and Y. Xing, "Community detection by label propagation with leaderrank method," *Journal of Computer Applications*, vol. 35, no. 2, p. 448, 2015.

[25] N. Y. Chen, Y. Liu, H. Q. Chen, and J. Cheng, "Detecting communities in social networks using label propagation with information entropy," *Physica A: Statistical Mechanics and its Applications*, vol. 471, pp. 788–798, 2017.

[26] P. Zhong and M. Fukushima, "Regularized nonsmooth newton method for multi-class support vector machines," *Optimization Methods and Software*, vol. 22, no. 1, pp. 225–236, 2007.

[27] N. Rajkumar and P. Jaganathan, "A new RBF kernel based learning method applied to multiclass dermatology diseases classification," in *2013 IEEE Conference on Information & Communication Technologies*, Thuckalay, India, 2013.

[28] T. P. Q. Nguyen and R. J. Kuo, "Partition-and-merge based fuzzy genetic clustering algorithm for categorical data," *Applied Soft Computing*, vol. 75, 2018.

[29] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance," *The Journal of Machine Learning Research*, vol. 11, pp. 2837–2854, 2010.

[30] A. P. Sample, R. S. Torres, D. C. Ranasinghe, Q. Shi, and A. P. Sample, "Sensor enabled wearable rfid technology for mitigating the risk of falls near beds," in *IEEE International Conference on RFID*, Orlando, FL, USA, 2013.