Probability Density Grid-based Online Clustering for Uncertain Data Streams

Haitao He, Lijuan Chen, Jiadong Ren, Wenyan Guo

College of Information Science and Engineering, Yanshan University
Qinhuangdao, P.R.China, 066004
haitao@ysu.edu.cn, ljchenysu@qq.com, jdren@ysu.edu.cn, gwylscx@163.com

Abstract

Most existing stream clustering algorithms adopt the online component and offline component. The disadvantage of two-phase algorithms is that they can not generate the final clusters online and the accurate clustering results need to be got through the offline analysis. Furthermore, the clustering algorithms for uncertain data streams are incompetent to find clusters of arbitrary shapes according to the varieties of uncertain data streams. To address this issue, this paper proposes a novel algorithm PDG-OCUStream, Probability Density Grid-based Online Clustering for Uncertain Data Streams, in which the summary information of uncertain data streams is stored in the probability density grid with relative statistical values. By setting the probability density threshold, clustering quality can be effectively controlled, and probability density grid structure is easy to be maintained and updated, so it can improve the efficiency of online clustering. In this paper we also use the count-based sliding window, which reflects the current situation of the uncertain data stream. System resources can be effectively saved by adjusting the step of sliding window. In addition, this paper defines grid probability density similarity to achieve initializing and updating clusters according to merging connected probability density grids, so the algorithm can distinguish between dense regions and sparse regions, and quickly find the clusters in the data distribution in real time. The experimental results show that PDG-OCUStream algorithm has fast online clustering capability while ensuring a good clustering quality.

Keywords: probability density grid, clustering, uncertain data streams

1. Introduction

In recent years, due to the randomness of the data generation, measurement inaccuracies and the incompleteness of data collection, uncertain data is widespread in many applications, such as data cleaning, information integration and sensor networks computing. The uncertain data also appears in data stream, which is called uncertain data stream. Therefore, clustering algorithm on uncertain data streams has been becoming a hot topic.

The existing algorithms for clustering data stream with deterministic data stream, in which each tuple is deterministic. STREAM[1] is a clustering algorithm based on partitioning and clusters are represented by centroid and weight. However, its clustering results may be controlled by outdated data points, because of without considering the evolution of data streams. Aggarwal et al. proposed a framework for clustering evolving data streams CluStream[2], which adopts online component and offline component. The online component periodically stores summary information by the pyramidal time window, while the offline component analyzes the data streams over different horizons by using summary information which is stored in online component. The extended partitioning and hierarchical methods can not handle data streams of arbitrary shapes better, due to adopting the distance measure. DenStream[3] is an extension of DBSCAN, based on the damped window model, which is capable of discovering clusters with arbitrary shape and emphasizes on inspecting noise points. D-Stream[4] is a density and grid based algorithm, using the density of grid structure. It can find clusters of arbitrary shape and handle noise points. HDenStream[5] is proposed for clustering data stream with heterogeneous features and is also a density-based algorithm, which is capable of clustering arbitrary shapes and handling outliers. Ren et al. proposed a grid density and index tree PKs-tree based algorithm PKS-Stream[6], which improves the efficiency of storage and indexing. The above density-
based and grid-based methods adopt the online component and offline component, they can not generate the final clusters online.

Data uncertainty can be divided into attribute uncertainty and tuple uncertainty. The existing deterministic data streams clustering algorithms do not take the uncertainties into account in data streams, so they can not be directly applied to the clustering data streams with uncertainty. In order to clustering uncertain data streams, UMicro[7] is proposed, which defines uncertain micro-cluster feature to store the summary information. In the process of clustering, distance-based similarity is selected as the measure standard. UPStream[8] is a clustering algorithm for uncertain data stream based on projection space to solve the “dimension disaster” of the high dimensional data clustering. Huang et al. proposed GD-CUStreams[9], which defines the uncertainty grid clustering feature to store the summary information of recent uncertain data streams and can obtain the clusters of arbitrary shapes. The above works only consider the attribute uncertainty and do not consider the tuple uncertainty.

The tuple uncertainty can be described by a variety of ways, which is a probability value between [0, 1] in the point probability model[10-11], where the existence of each tuple is uncertain. EMicro[12] is a clustering algorithm for data streams with tuple uncertainty, which considers distance metric and data uncertainty together and two buffers are maintained to reserve normal micro-clusters and potential outlier micro-clusters respectively in memory. Hu et al. proposed PWSStream[13], which uses exponential histogram of cluster feature to store the summary information of the most recently arrived tuples, and outdated information is deleted within a certain guaranteed range of error. However, the above clustering algorithms for data streams with tuple uncertainty can not obtain clusters of arbitrary shapes, and they also can not generate the final clustering results online.

In order to make up the disadvantages of the existing two-phase algorithm for clustering uncertain data streams, which can not generate the final clusters online, this paper proposes online clustering method, where a count-based sliding window is introduced to reflect the current situation of the uncertain data stream. Meanwhile, in order to achieve initializing and updating clusters, the grid probability density similarity is defined. In addition, in order to obtain clusters of arbitrary shapes, this paper adopts the storage structure based on probability density grid structure, and the clustering quality can be effectively controlled by setting probability density threshold.

The rest of the paper is organized as follows. In the next section, we introduce the basic concepts and related symbols. The algorithm and the analysis of the algorithm are proposed in section 3. Section 4 consists of the experimental results and the experimental analysis. Finally, we conclude the paper in section 5.

2. Problem definition

Assume that uncertain data streams consists of a set of d-dimensional uncertain tuples, denoted by \( US = \{(X_1, p_1), (X_2, p_2), \ldots, (X_n, p_n)\} \), where \( (X_i, p_i) = ((x_{i1}, x_{i2}, \ldots, x_{id}), p_i) \). \( X_i \) is the value of the \( i^{th} \) tuple, \( p_i \) is the probability of the \( i^{th} \) tuple, \( 0 \leq p_i \leq 1, 1 \leq i \leq n \).

Given a d-dimensional data space \( S = S_1 \times S_2 \times \cdots \times S_d \), if each dimension of data space \( S_i \) is divided into a number of same intervals, then the grid units consists of all the intervals on all dimensions form a grid structure \( G \) of \( S \). We set the partitioning number to \( k \), for any dimension \( i \in [1, d] \), assume that the intervals of \( S_i \) are divided into \( S_{i,1} \cup S_{i,2} \cup \cdots \cup S_{i,k} \), then these intervals can be labeled by 1-k, which means for any \( i \in [1, d] \), the label of the i-dimensional subinterval \( S_{i,j} (j = l_i) \) is \( l_i \in [1, k] \). Therefore, any grid units of a d-dimensional data space can be uniquely labeled by integer sequence between [1, k]. For any \( (X, p) = ((x^1, x^2, \ldots, x^d), p) \in S \), \( (X, p) \) can be mapped to grid \( g(X) = (l_1, l_2, \ldots, l_d) \), if and only if all \( i \in [1, d] \), \( x^i \in S_{i,j} \), where the interval label of \( S_{i,j} \) is \( l_j \).

Definition 2.1. (Probability Density Grid Structure and Probability Density Grid Unit) Given a d-dimensional data space \( S \) and a probability density grid structure \( PG \) of \( S \) consists of probability density grid units, a probability density grid unit \( g \) is \( g = \langle id, p, E(g), Cen, PDen(g), Type \rangle \).
Where, id = (l_1, l_2, \ldots, l_d) is the labels of g; \( p_c = \sum_{i=1}^{E(g)} p_i \) is the sum of probability for uncertain tuples contained in g; \( E(g) \) is the number of the uncertain tuples contained in g; \( Cen = \sum x_i / E(g)(i=1,2,\ldots;n; j=1,2,\ldots,d) \) is the centroid of g; \( \text{PDen}(g) \) is the probability density of g; Type is used to identify the type of the corresponding probability density grid unit g.

**Definition 2.2.** (Probability Density) For a probability density grid unit g, its probability density is denoted by \( \text{PDen}(g) = \frac{p_c}{E(g)} \), where \( E(g) \) is the number of the uncertain tuples contained in g, \( p_c \) is the sum of probability for uncertain tuples contained in g.

It is easy to see from definition 2.2, we add a new uncertain tuple \((X_{n+1}, p_{n+1})\) to a probability density grid unit \( g = ((X_1, p_1), (X_2, p_2), \ldots, (X_n, p_n)) \) with probability density \( \text{PDen}(g) \), then there are \( n+1 \) uncertain tuples in g this moment, its probability density is
\[
\text{PDen}(g') = \frac{p_c + p_{n+1}}{n+1} = \frac{n \times \text{PDen}(g) + p_{n+1}}{n+1} = \text{PDen}(g) + \frac{p_{n+1} - \text{PDen}(g)}{n+1}. 
\]
Thus we can get the following property:

**Property 1.** When a new uncertain tuple \((x_{n+1}, p_{n+1})\) is added to a probability density grid unit \( g = ((x_1, p_1), (x_2, p_2), \ldots, (x_n, p_n)) \) with probability density \( \text{PDen}(g) \), if \( p_{n+1} > \text{PDen}(g) \), then the probability density of this probability density grid unit will be increased, that is \( \text{PDen}(g') > \text{PDen}(g) \); if \( p_{n+1} \leq \text{PDen}(g) \), then the probability density will be unchanged or reduced, that is \( \text{PDen}(g') \leq \text{PDen}(g) \).

The type of the probability density grid unit g can be judged according to probability density measure threshold \( \text{PDen}_{\text{max}} \) and \( \text{PDen}_{\text{min}} \): (1) if and only if \( \text{PDen}(g) \geq \text{PDen}_{\text{max}} \), then g is dense grid; (2) if and only if \( \text{PDen}(g) \leq \text{PDen}_{\text{min}} \), then g is sparse grid; (3) if and only if \( \text{PDen}_{\text{min}} < \text{PDen}(g) < \text{PDen}_{\text{max}} \), then g is medium probability density grid unit, where probability density measure threshold
\[
\text{PDen}_{\text{max}} = C_{\text{max}} \times len^d \times \frac{P_c}{E(g)} (C_{\text{max}} \geq 1), \quad \text{PDen}_{\text{min}} = C_{\text{min}} \times len^d \times \frac{P_c}{E(g)} (0 < C_{\text{min}} < 1), \quad \text{len} = 1/k \text{ is the length of the grid interval.}
\]

**Definition 2.3.** (Grid Probability Density Similarity) Given any two adjacent dense or medium probability density grids \( g_i \) and \( g_j \), we call \( S_{ij} \) as grid probability density similarity between \( g_i \) and \( g_j \), where \( S_{ij} = (\min(\text{PDen}_{g_i}, \text{PDen}_{g_j}))/\max(\text{PDen}_{g_i}, \text{PDen}_{g_j}) \).

If the value of \( S_{ij} \) is close to 1 and the distance between \( \text{Cen}_{g_i} \) and \( \text{Cen}_{g_j} \) is less than the length of the grid unit, then we can merge \( g_i \) and \( g_j \).

**Lemma 1.** We can merge any two adjacent dense or medium probability density grids \( g_i \) and \( g_j \) into the same cluster, if the value of \( S_{ij} \) is less than \( S(S) \) is the threshold of grid probability density similarity and the value of \( S \) is close to 1) and the distance between \( \text{Cen}_{g_i} \) and \( \text{Cen}_{g_j} \) is less than the length of the grid unit.

Given a d-dimensional data space \( S \) and a probability density grid structure PG, assume that there are two probability density grid units \( g_i = (e_1, e_2, \ldots, e_d) \) and \( g_j = (h_1, h_2, \ldots, h_d) \), if \( g_i \) and \( g_j \) are adjacent in the j-dimension and have the same value in the other dimensions, then \( g_i \) and \( g_j \) are the probability density grid neighbors in the j-dimension.

Obviously, for a d-dimensional probability density grid structure, there are \( 2^d \) grid neighbors in each probability density grid unit. As shown in figure 1 there is a 2-dimensional probability density grid structure A, B, C, D all are the neighbors of the probability density grid g, and L_1, R_1, L_2, R_2 are the neighbors of the probability density grid A, B, C, D.

**Definition 2.4.** (Probability Density Grid Cluster) Given a d-dimensional data space \( S \) and a probability density grid structure PG, assume that \( g = \langle id, p_c, E(g), \text{Cen}, \text{PDen}(g), \text{Type} \rangle \) is a dense probability density grid unit, then we call all connected dense or medium probability density grid unit with g as a probability density grid cluster, if any two adjacent probability density grid unit satisfy
property 2. If the probability density of \( g \) is maximum in this cluster, then we call \( g \) as cluster center for this probability density grid cluster.

As shown in figure 2, all the gray probability density grid units are dense or medium probability density, so all the connected gray probability density grid units form a probability density grid cluster, where \( g \) is the cluster center with maximum density.

Given a \( d \)-dimensional data space \( S \) and a probability density grid structure \( PG \), assume that \( g_1(A,B,C,D) \) and \( g_2(E,F,H) \) are two probability density grid clusters, which cluster center is \( g_1 \) and \( g_2 \) respectively, and the two probability density grid clusters \( g_1 \) and \( g_2 \) are connected by grid unit \( C \), as shown in figure 3. If \( \text{PDen}(C) > \text{PDen}_{\text{max}} \), which means \( C \) is medium probability density grid or dense grid, then \( g_1 \) and \( g_2 \) are the connected probability density grid cluster.

### 3. Probability Density Grid-based Online Clustering for Uncertain Data Streams

Grid-based clustering has the advantage of processing speed, while the method based on probability density considers the uncertainty of the uncertain data streams, and can discover clusters of arbitrary shapes. The proposed algorithm PDG-OCUStream combines the advantages of both approaches and defines the probability density grid structure. PDG-OCUStream maps the uncertain tuples to probability density grids online, and meanwhile it achieves the purpose of clustering uncertain data stream online by clustering probability density grid. PDG-OCUStream updates probability density grid structure along with uncertain data stream arrives and defines the definition of the grid probability density similarity for the preparation for the initializing and updating cluster. In addition, PDG-OCUStream adopts connected probability density grid merger to update cluster, so that there has a higher probability density within probability density grid cluster and has a lower probability density between the different probability density grid clusters, so the algorithm can distinguish between dense regions and sparse regions, and quickly find the clusters in the data distribution in real time.

PDG-OCUStream reads uncertain tuples into sliding window from uncertain data streams continually. It calls \text{Init}_\text{clustering} \text{Algorithm}, when the number of the uncertain tuples within sliding window is \( WN \) (WM is the size of count-based sliding window). Then the sliding window slides forward while a new uncertain tuple arrives, \( \text{STEP}_C \) old uncertain tuples are removed from the tail of window and \( \text{STEP}_C \) new uncertain tuples are read in the head of window. PDG-OCUStream contains the following parameters: \( \text{len}=1/k \) is the length of the grid interval; \( C_{\text{max}} \) and \( C_{\text{min}} \) are set to control the boundary of the dense degree of the probability density grid unit; \( \text{PDen}_{\text{max}} \) and \( \text{PDen}_{\text{min}} \) are the threshold of the probability density; \( \text{STEP}_C \) is the step count of sliding window slides. PDG-OCUStream is as follows.

**Algorithm: PDG-OCUStream**

**Input:** \( US = \{(X_1, p_1), (X_2, p_2), \ldots, (X_n, p_n)\} \), \( \text{len} \), \( C_{\text{max}} \), \( C_{\text{min}} \), \( \text{STEP}_C \)

**Output:** online clustering results

1. \( \text{count}=0; \) \( n_{\text{app}}=0; \)
2. while uncertain data stream is active do
3. read uncertain tuple \( (X_i, p_i) \) into sliding window;
(4) map uncertain tuple \((X_i, p_i)\) to probability density grid \(g(X_i)\) according to tuple’s attribute value;
(5) update the corresponding summary information of probability density grid \(g(X_i)\);
(6) \(\text{count}++\);
(7) if \(\text{count}=\text{WN}\) then call Init_clustering Algorithm
(8) if \(\text{count}>\text{WN}\) then
(9) \(\text{n}_\text{step}++\);
(10) while \(\text{n}_\text{step} = \text{STEPC}\)
(11) call Update_clustering Algorithm;
(12) \(\text{n}_\text{step}=0\);
(13) end while
(14) end while
(15) return online clustering results;

In the circumstance of uncertain data streams, PDG-OCUSStream gets the uncertain tuples from the uncertain data streams in order and then these tuples are mapped to probability density grids according to tuples’ attribute value, and the summary information such as \(\text{PDen}(g)\) will be updated after adding a new uncertain tuple to the probability density grid. It calls Init_clustering Algorithm, when the number of the uncertain tuples within sliding window is \(\text{WN}\). Then the sliding window slides forward while the new uncertain tuples arrives, it calls Update_clustering Algorithm. This algorithm loops until the final clustering results obtained.

Init_clustering Algorithm maps uncertain tuples within sliding window to probability density grid structure firstly, and classifies all probability density grids according to the threshold of the probability density. Then, it labels dense grid as a single cluster center, and merges all the neighbor grids of this dense grid and then merges the clusters connected with this dense grid according to the definition of grid probability density similarity and property 2. Init_clustering Algorithm is as follows.

Algorithm: Init_clustering Algorithm
Input: \(US = \{(X_1, p_1), (X_2, p_2), \ldots, (X_n, p_n)\}, C_{\text{max}}, C_{\text{min}}\)
Output: Init_clustering results
(1) for each \(g(X_i)\) of the \((X_i, p_i)\)
(2) label the type of \(g(X_i)\): dense, sparse, medium density;
(3) if \(g(X_i)\) is dense then label \(g(X_i)\) as a single cluster center \(g_c\);
(4) for each cluster center grid \(g_c\)
(5) merge all the neighbor grids of \(g_c\);
(6) merge the clusters connected with \(g_c\);
(7) return Init_clustering results;

In line 1-4, for any probability density grid unit, algorithm classifies all probability density grids into dense, sparse and medium probability density grid according to the threshold of the probability density. Then we label it as a single cluster center if this grid is dense grid. In line 5-7 algorithm merges all the neighbor grids in dense grids and merges the clusters connected with dense grids for each cluster center grid, if the value of grid probability density similarity between the two adjacent dense or medium probability density grids is less than \(S\) and the distance between the centroids of the two adjacent dense or medium probability density grids is less than the length of grid unit. Finally algorithm returns Init_clustering results over this sliding window.

Update_clustering Algorithm only deals with the new/old uncertain tuples generated by window sliding, it has a small amount of uncertain data, and so it needs less time during updating clusters. This algorithm keeps the grid unit with the largest probability density as cluster center, and adds all dense grids into grid clusters or as a single cluster center. It does not merge sparse grids until these sparse grids translate to medium probability density grid. Update_clustering Algorithm is as follows.

Algorithm: Update_clustering Algorithm
Input: the uncertain tuple sets according to the step times of the window slides
\[X_{\text{new}} = \{(X_1, p_1), (X_2, p_2), \ldots, (X_{\text{step}}, p_{\text{step}})\}\]
\[X_{\text{old}} = \{(X_n, p_n), (X_{n-1}, p_{n-1}), \ldots, (X_{n-\text{step}+1}, p_{n-\text{step}+1})\}\]
Output: clustering results of the current window
(1) for each \(g_{\text{new}}/g_{\text{old}}\) of \((X_{\text{new}}, p_{\text{new}})/(X_{\text{old}}, p_{\text{old}})\)
(2) if $g_{\text{new}}$ is dense grid
(3) (if $g_{\text{new}}$ is not a cluster center, and $\text{PDen}(g_{\text{new}})$> the probability density of the cluster center
(4) then label $g_{\text{new}}$ as a new cluster center, merge probability density grid cluster;
(5) else return;
(6) else if $g_{\text{new}}$ is a medium probability density grid
(7) (if $g_{\text{new}}$ is a single probability density grid, and the probability density neighbor grid of $g_{\text{new}}$ is center of a cluster C
(8) then add $g_{\text{new}}$ to the cluster C;
(9) else return;
(10) } else if $g_{\text{new}}$ is a sparse grid
(11) then return;
(12) }
(13) if $g_{\text{old}}$ is dense grid
(14) (if $g_{\text{old}}$ is a cluster center
(15) then label the dense grid with the largest probability density in original cluster as a new cluster center;
(16) else return;
(17) else if $g_{\text{old}}$ is a medium probability density grid
(18) (if $g_{\text{old}}$ is a cluster center, and no dense grid in this cluster
(19) then dismiss this probability density cluster;
(20) else return;
(21) else if $g_{\text{old}}$ is a sparse grid
(22) (if $g_{\text{old}}$ has a cluster center
(23) then delete $g_{\text{old}}$ from the original cluster;
(24) else return;
(25) )
(26) if the cluster number of the current window<1 then $r$ reduces 0.1;
(27) if the cluster number of the current window>n then $r$ adds 0.1;
(28) end for
(29) return clustering results of the current window;

At first, in line 2-12 algorithm updates the new probability density grids of the new uncertain tuples, which generated from window sliding. If $g_{\text{new}}$ is a dense grid, and it is not a cluster center, and $\text{PDen}(g_{\text{new}})$ is more than the probability density of the cluster center, then algorithm labels $g_{\text{new}}$ as a new cluster center and merges probability density grid cluster; If $g_{\text{new}}$ is a medium probability density grid and it is a single probability density grid, and the probability density neighbor grid of $g_{\text{new}}$ is center of a cluster C, then algorithm adds $g_{\text{new}}$ to the cluster C. If $g_{\text{new}}$ is a sparse grid, algorithm does not merge sparse grids until these grids are transformed into medium probability density grid. In line 13-25 algorithm updates the old probability density grids of the old uncertain tuples, which generated from window sliding. If $g_{\text{old}}$ is a dense grid and it is a cluster center, then algorithm labels the dense grid with the largest probability density in original cluster as a new cluster center. If $g_{\text{old}}$ is a medium probability density grid and it is a cluster center, and there has no dense grid in this cluster, then this probability density cluster is dismissed. If $g_{\text{old}}$ is a sparse grid and it has a cluster center, then $g_{\text{old}}$ is deleted from the original cluster. In line 26-27 this algorithm sets the optimum value of $r$ by judging the cluster number of the current window. $C_{\text{max}}$ and $C_{\text{min}}$ are set to control the boundary of the dense degree of the probability density grid unit. According to the density threshold ratio of the probability density grid unit $r=C_{\text{max}}/C_{\text{min}}$, on the one hand, $r$ can distinguish the probability density sparse regions and dense regions; on the other hand, due to $C_{\text{max}}$ and $C_{\text{min}}$ was preset in the initialization, Update_clustering Algorithm can adjust value of $r$ within the range of [1,10] according to the current clustering result, in order to avoid the influence of artificial setting parameters on clustering process, so that clustering algorithm can be more responsive to the dynamic changes of uncertain data streams and also can capture the hidden clusters in uncertain data stream. Finally, Update_clustering Algorithm returns clustering results of the current window. And thus achieves a fast updating clustering online.

4. Experimental Evaluation
All of our experiments are conducted on a PC with AMD Athlon(tm) X4 640 3.00GHz CPU and 1.75G RAM running Microsoft XP Professional 2002 Service Pack 3 operating system; PDG-OCUStream is implemented in Microsoft Visual Studio.NET 2008 with C++ programming language.

In order to validate the effectiveness of PDG-OCUStream on clustering uncertain data streams, a synthetic uncertain stream dataset DataSet1 is adopted. The distribution of DataSet1 is shown in Figure 4. Experimental parameters are as follows: the size of count-based sliding window WN=10000, \( r = \frac{C_{\text{max}}}{C_{\text{min}}} = 6 \), \( \text{len} = 0.02 \), \( n_{\text{step}} = 10 \).

First, we test the clustering quality on synthetic dataset DataSet1. Due to non-convexity of DataSet1, CluStream can not get a correct result and CluStream can only be suitable for clustering deterministic data streams. However, when CluStream is applied into the environment of uncertain data streams in point probability model, some probability information of uncertain data streams is very easy to lost, which will lead to the lower clustering purity. Nevertheless, PDG-OCUStream uses the probability density grid structure, which can take the probability information of uncertain tuples into account in the process of clustering. In addition, PDG-OCUStream adopts probability density grid and defines grid probability density similarity to merge the adjacent dense or medium probability density grid, and it does not merge sparse grids until these sparse grids are transformed into medium probability density grid, so it is able to discover non-convex clusters and the clustering quality is improved in the process of clustering. Thus, the clustering quality of PDG-OCUStream is higher than CluStream, as shown in Figure 5.

We also test the clustering quality of PDG-OCUStream among different values of density threshold \( r \) within the scope of \([1, 10]\). From Figure 6, we can see that clustering quality is higher when \( r \) is equal to 4 or 6. When the value of \( r \) decreases, more dense and sparse probability density grids are divided, and then the number of medium probability density grid reduces, and the clusters with lower probability density are generated, clustering quality is deteriorated. By contrast, when the value of \( r \) increases, less dense and sparse probability density grids are divided, and then the number of medium probability density grid increases, and the clusters with higher average probability density are generated, quality of clustering results is improved. Therefore, algorithm can achieve satisfactory clustering results by adjusting the value of \( r \) according to the evolution of uncertain data streams in the process of online clustering.

5. Conclusion

In this paper, we propose a novel algorithm PDG-OCUStream, Probability Density Grid-based Online Clustering for Uncertain Data Streams, in which the summary information of uncertain data streams is stored in the probability density grid structure with relative statistical values. Then the clustering quality can be effectively controlled by adjusting the probability density threshold. Meanwhile, in this paper we also use the count-based sliding window, which reflects the current situation of the uncertain data streams in the process of online clustering. In addition, the definitions of probability density grid cluster and grid probability density similarity are defined in this paper. In the process of online clustering, PDG-OCUStream labels the dense probability density grid within the current sliding window as a single cluster center, then merges any two adjacent dense or medium probability density grids if the value of grid probability density similarity between the two adjacent
dense or medium probability density grids is less than the threshold of grid probability density similarity and the distance between the centroids of the two adjacent dense or medium probability density grids is less than the length of grid unit, so the probability density grid with large similarity and small distance can be assigned to a cluster. Finally, PDG-OCUStream outputs all grids belonging to the type of dense and medium probability density. Therefore, PDG-OCUStream obtains the clustering results with arbitrary shapes online. The experimental results show that PDG-OCUStream can clustering uncertain data streams online and also can get better clustering purity.

6. Acknowledgment

This work is supported by the Natural Science Foundation of Hebei Province P. R. China under Grant No.F2010001298. The authors also gratefully acknowledge the valuable comments and suggestions of the reviewers, which have improved the presentation.

7. References


