An Ensemble Density-based Clustering Method

Article *in* International Journal of Computational Intelligence Systems · October 2007

DOI: 10.2991/iske.2007.45

2 authors, including:

Luning Xia
Chinese Academy of Sciences
23 PUBLICATIONS **54** CITATIONS

SEE PROFILE

An Ensemble Density-based Clustering Method

Luning Xia Jiwu Jing

Information Security State Key Laboratory, Graduate University of Chinese Academy of Science, Beijing 100049, P. R. China

Abstract

Density based clustering is sound for its great ability of finding arbitrary shapes of clusters and identifying the number of clusters automatically. DBSCAN is a frequently used density based clustering algorithm. In DBSCAN a density threshold, which is hard to be chosen adaptively, should be specified to determine whether an object is dense or sparse. In this paper we introduce the concept of *clustering ensemble* to avoid the difficulty of selecting a single appropriate threshold. Performing DBSCAN multiple times with diverse thresholds picked up from a pre-constructed interval, the final partition can be figured out via a consensus function. Experimental results show that this method can go beyond DBSCAN both in validity and stability, and avoid the inefficiency caused by any inappropriate thresholds.

Keywords: DBSCAN, Clustering ensemble, Consensus function

1. Introduction

Clustering is an important means of data mining. There are four categories of clustering algorithms so far. (1) Partition algorithms. The typical one is k-means [1]. (2) Hierarchical algorithms, such as single-link, complete-link, group average, Ward, BIRCH, CURE, etc. [5] (3) Modal based algorithms, for example EM algorithm [5]. (4) Density based algorithms. According to the different definitions of the *density*, the representative algorithms are DBSCAN [2], OPTICS [4], DENCLULDE [3], etc.

The density based clustering algorithms, grouping objects according to the spatial density of them, are suitable for processing datasets without any priori knowledge. DBSCAN is a classic density based clustering algorithm. It can find arbitrary shapes of clusters, identify outliers (the noise), and determine the number of clusters automatically. In DBSCAN, the density of an object is measured based on the number of the other objects within a hypersphere area around it. Two parameters, *Eps* and *MinPts*, denoting the radii of the hypersphpere and the minimum number of objects around a high-density object, have to be specified as the density threshold. *MinPts* is usually specified to 4 in most DBSCAN practices. The traditional way of selecting the value of *Eps* is to analyze the 4-dist curve, i.e. the sorted 4-nearest distance set, and select a seemingly good value. Theoretically we can find the best value of *Eps* by this way. Unfortunately, the curve is a nonparametric curve so we can not express it with any formula and figure out the theoretically best value by any mathematic means. Some adaptive methods of estimating *Eps* are available in many literatures, but the validity of them is affected by the statistical characteristics of the datasets. There are no methods suitable to estimate the appropriate value of *Eps* for any datasets with diverse statistical characteristics so far.

In this paper we propose an ensemble method for DBSCAN. We do not try to find the best value of *Eps* but present an interval of possible *Eps* values according to the statistical characteristic of the 4-nearest distance set. DBSCAN is iterated for several times with different values of *Eps* selected from the interval, instead of being performed once with a single *Eps* value. Through a consensus function which is used to combine the partitions obtained from these DBSCAN instances, the final partition of the dataset is determined. We call this new method as Ensemble DBSCAN (abbr. EDBSCAN).

The rest of this paper is organized as follows. Section 2 introduces the basic concept of DBSCAN and clustering ensemble. Section 3 describes the Ensemble DBSCAN clustering method. Next, Section 4 summarizes the result of experimental results and analysis. Finally in Section 5, a conclusion is made.

2. Related Works

2.1. DBSCAN Algorithm

The main definitions of DBSCAN are described as follows.

Def. 1 To an object p in a dataset D, the ***Eps-neighborhood*** of p means the hypersphere area with radii *Eps* around p, which is denoted by $N_{Eps}(p) = \{q \in D \mid dist(p,q) \le Eps\}$. Here $D \subseteq R^d$ is a dataset in the d-dimension real space R^d, and $dist(p,q)$ denotes the distance between two objects p and q in dataset D.

Def. 2 Given an integer *MinPts*, if the number of objects within the *Eps*-neighborhood of object *p* satisfies $\left|N_{Eps}(p)\right| \geq MinPts$, the object *p* is defined as a ***core object***. A ***border object*** is one of those objects in the *Eps*-neighborhood of a core object, though it is not a core object itself.

Def. 3 An object *p* is ***directly density-reachable*** from an object *q* if a) $p \in N_{Eps}(q)$ and b) $\left|N_{Eps}(q)\right| \geq MinPts$ (i.e. *q* is a core object)

Def. 4 An object *p* is ***density-reachable*** from an object *q* if there is a chain of objects $p_1, p_2, p_3 \ldots\ldots p_n, p_1 = q, p_n = p$, such that p_{i+1} is directly density-reachable from p_i for $1 \leq i \leq n-1$.

Def. 5 An object *p* is ***density-connected*** to an object *q* if there is an object *o* such that both *p* and *q* are density-reachable from *o*.

Def. 6 A ***density-based cluster*** *C* is a non-empty subset of dataset *D* satisfying the following conditions:

a) $\forall p,q$, if $p \in C$ and *q* is density-reachable from *p*, then $q \in C$;

b) $\forall p,q \in C$, *p* is density-connected to *q*.

Those objects that are not in any clusters are defined as ***noise***.

The validity of the partition obtained from a DBSCAN instance is affected by the value of *Eps*. Selecting a smaller *Eps* may cause many objects to be identified as noise falsely. And a labeled group of objects, i.e. an inherent cluster, may be split to several clusters. Whereas selecting a bigger *Eps* may cause many noise objects to be falsely assigned into some clusters, and several labeled groups to be merged into a single cluster. The original literature of DBSCAN [1] proposed an observation method to select the appropriate value of *Eps*. For an object $p \in D$ the distance between *p* and the 4^{th} nearest object of *p*, which is denoted by $dist_4(p)$, is calculated. *p* goes through the dataset *D* so that the 4-nearest distance set $Dist_4 = \{dist_4(p) \mid p \in D\}$ is formed. Sorting $Dist_4$ from the minimum member to the maximum one and drawing it on the Cartesian coordinates, the 4-dist curve is shown in Fig.1 assumed that the high-density areas and the low-density areas in the dataset are relatively well isolated. The proposal of literature [1] is to select the value of position A as the value of *Eps*. Obviously this work has to be done by the user.

The curve shown in Fig.1 can be considered as a "good" or "ideal" curve for DBSCAN algorithm because the inflexion is obvious. Unfortunately, not all datasets present such a 4-dist graph. The curve in Fig.2 is the 4-dist curve of the OptDigit dataset [16] in which no clear and sharp boundaries between high-density areas and low-density areas. In this situation, selecting the value of position A may cause all high-density objects being assigned into a single cluster. In fact the theoretically best value of *Eps* is close to the value of position B, but an appropriate value is hard to be chosen because the range of valid *Eps* values is rather narrow. A slight drift from the theoretically best value may significantly reduce the validity of the result.

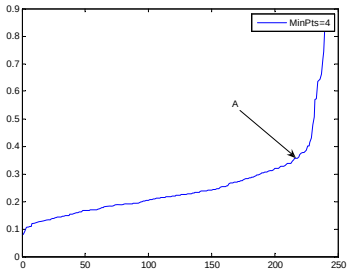

Fig.1: The 4-dist graph.

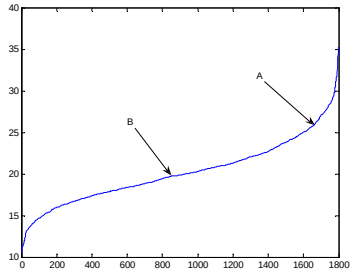

Fig.2: The 4-dist graph of OptDigit dataset.

The methods of estimating *Eps* adaptively proposed in literature [8] that introduced the concept of distance distribution and literature [9] that borrowed the idea of the generic algorithm are all based on the statistical characteristics of the datasets. They are only adapted to these datasets satisfying the specified statistical characteristics and have clear boundary between high-density and low-density areas. Whereas for the datasets like OptDigit described above, selecting the accurate value of *Eps* just by statistical analysis is really hard. In literature [6] several values of *Eps* are selected as candidates. Clustering the dataset with each value, the optical candidate is selected through a clustering validity test. It is equivalent to perform DBSCAN for many times and accept the seemingly best result of them. Still it can't process the datasets like OptDigit well because the range of appropriate values of *Eps* is very narrow, unless spending much time to try a large number of *Eps* values. Literature [4], the OPTICS algorithm, sorted all objects by the so-called density reachable distance. Objects in the same labeled group are aligned closely. By this way we can make out the partition by observation. However, it is not an automatic method.

In fact, there are no adaptive ways to find the appropriate values of *Eps* for datasets with diverse statistical characteristics so far. To avoid the difficulty of selecting one accurate value of *Eps*, we introduce the idea of clustering ensemble and propose a new method in this paper.

2.2. Clustering ensemble

It might be very dissimilar while clustering a dataset with different algorithms. Every clustering algorithm can only be adapted to specific type of datasets. It may cause an invalid or low-quality result if the dataset does not match the assumptive type. So there are no clustering algorithms that can get valid partition for all types of datasets.

Under this condition, the idea of clustering ensemble is proposed by many researchers. Clustering ensemble is a method combining multiple partitions to form a final partition. It is believed that different fractions of the information of the dataset could be derived from each partition. After merging these fractions via a consensus function, the relatively complete information about the dataset will be obtained. Compared with a single clustering algorithm, clustering ensemble can go beyond in both validity and stability [13]. The original idea of clustering ensemble was proposed in literature [12], where it is defined as: using a consensus function to combine the partitions obtained from multiple clustering instances, without resorting to the original object features or algorithms.

Let $X = \{x_1, x_2, ...x_n\}$ denotes a dataset. Multiple partitions to dataset X are denoted by $H = \{h_1, h_2, ...h_r\}$, where r is the count of partitions and h_i (i=1, 2….r) is the i^{th} partition. The consensus function Γ is used to combine these r partitions into a final partition h, which is illustrated in Fig.3.

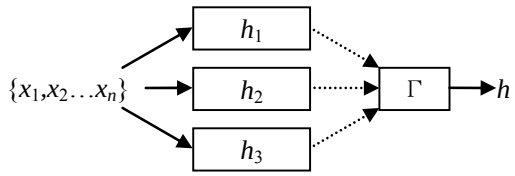

Fig.3: The demo of clustering ensemble.

The r partitions of dataset X can be gotten by performing diverse clustering algorithms such as k-means, DBSCAN or hierarchical methods, as well as iterating one algorithm for r times with different thresholds. For example, iterating k-means with different original core points [14]. As to the consensus function, available methods are voting method [11], hypergraph method [12][15], mixed model method [13], etc. In voting and hypergraph methods the *co-association matrix* is a key component. It is a square matrix of n rows and n columns, where n is the size of dataset X. Let A denotes the co-association matrix, every a_{ij} of matrix A denotes the associability of object x_i and x_j, i.e. the probability that x_i and x_j were in the same cluster in all r partitions.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}$$

$$a_{ij} = asso(x_i, x_j) = \frac{\text{the count that } x_i \text{ and } x_j \text{ are in a same cluster}}{r}$$

The co-association matrix was originally defined in literature [11], in which a voting method based on k-means was introduced as well. The voting method can be briefed as: Partitioning the dataset using k-means with different original core points for r times, then putting object x_i and x_j into the same cluster in the final partition if a_{ij} is equal to or great than 0.5.

3. Ensemble DBSCAN Method (EDBSCAN)

The key idea of EDBSCAN is to select an interval of the probable values of *Eps* according to the statistical characteristic of the 4-nearest distance set ($Dist_4$) described in 2.1. The DBSCAN clustering is performed for r times, each of which uses a different *Eps* selected from the interval. Instead of accepting the seemingly best result like the practice of literature [6], we use a consensus function to combine all the results and figure out the final partition. It should be noticed that DBSCAN is a noise-sensitive clustering algorithm in contrast with k-means. So we must modify the forming method of the co-association matrix described in literature [11] in order to reflect the effect of noise. Finally, the voting method is used to determine the final partition.

The detailed procedures of EDBSCAN are described in following 3 subsections.

3.1. Initialization

There are some work including calculating the distance matrix, obtaining the 4-nearest distance set and identifying the interval of probable *Eps* values should be done before any other procedures.

The density is measured by the number of the objects within a hypersphere area around the target object in DBSCAN. So we must calculate the distance

between every two objects in the dataset. We use a matrix D_X to denote the pair distances of dataset X.

$$D_X = \begin{bmatrix} d_{11} & d_{12} & \dots & d_{1n} \\ d_{21} & d_{22} & \dots & d_{2n} \\ \dots & \dots & \dots & \dots \\ d_{n1} & d_{n2} & \dots & d_{nn} \end{bmatrix}$$

Where $d_{ij}(1 \le i \le n, 1 \le j \le n)$ denotes the distance between object x_i and x_j. Obviously $d_{ij} = d_{ji}$, and the diagonal items $d_{ii}(1 \le i \le n) \equiv 0$ because the distance between an object and itself is zero. So there are $n(n-1)/2$ calculations totally.

Sorting each column of D_X, the 4-nearest distance set $Dist_4$ is the 5th row of D_X (the first row is all zeros because it comprises the distances between all objects and themselves).

Let $d_{\min} = \min(Dist_4)$ and $d_{\max} = \max(Dist_4)$. In order to get the interval described above, first we calculate the mean of $Dist_4$.

$$d_{avg} = E(Dist_4)$$

Then extend d_{avg} to the left for 1/4 (an empirical proportion) times to get the low limit $e_{\min}$ of the interval, as well as to the right for 1/4 times to get the high limit $e_{\max}$.

$$e_{\min} = d_{avg} - \frac{d_{avg} - d_{\min}}{4}, \quad e_{\max} = d_{avg} + \frac{d_{\max} - d_{avg}}{4}$$

The r different values of Eps are denoted by $EPS = \{Eps_i | Eps_i \in [e_{\min}, e_{\max}], 1 \le i \le r\}$. These values can be generated randomly between $e_{\min}$ and $e_{\max}$, or spaced evenly along the interval. Every Eps_i is used in one DBSCAN instance.

The final step of the initialization procedure is to initialize the co-association matrix A to all zero.

3.2. Clustering iteration

In this procedure, the DBSCAN clustering is performed for r times, each of which uses a different Eps in $EPS = \{Eps_i | Eps_i \in [e_{\min}, e_{\max}], 1 \le i \le r\}$. The r partitions denoted by $H = \{h_1, h_2, \dots h_r\}$ are obtained then. The co-association matrix is refreshed according to each partition. a_{ij} is added by 1 if object x_i and x_j are in a same cluster of the partition, otherwise kept unchanged. It should be noticed that the noise objects do not belong to any cluster of the partition, so we keep a_{ij} unchanged if object x_i or x_j are noise. Having combined all r partitions, the co-association matrix is normalized through being divided by r.

For example, for a dataset of 5 objects the 3 partitions are:

$h_1 = \{1,1,2,2,2\}$ $h_2 = \{1,2,2,2,1\}$ $h_3 = \{1,1,2,2,-1\}$

Where 1, 2 denote the cluster labels and -1 denotes the noise. After processing h_1 the co-association matrix is:

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

After processing h_2 the co-association matrix is:

$$A = \begin{bmatrix} 2 & 1 & 0 & 0 & 1 \\ 1 & 2 & 1 & 1 & 0 \\ 0 & 1 & 2 & 2 & 1 \\ 0 & 1 & 2 & 2 & 1 \\ 1 & 0 & 1 & 1 & 2 \end{bmatrix}$$

After processing h_3 the co-association matrix is:

$$A = \begin{bmatrix} 3 & 2 & 0 & 0 & 1 \\ 2 & 3 & 1 & 1 & 0 \\ 0 & 1 & 3 & 3 & 1 \\ 0 & 1 & 3 & 3 & 1 \\ 1 & 0 & 1 & 1 & 3 \end{bmatrix}$$

And after normalization the co-association matrix is formed completely as:

$$A = \begin{bmatrix} 3 & 2 & 0 & 0 & 1 \\ 2 & 3 & 1 & 1 & 0 \\ 0 & 1 & 3 & 3 & 1 \\ 0 & 1 & 3 & 3 & 1 \\ 1 & 0 & 1 & 1 & 3 \end{bmatrix} / 3 = \begin{bmatrix} 1 & 0.67 & 0 & 0 & 0.33 \\ 0.67 & 1 & 0.33 & 0.33 & 0 \\ 0 & 0.33 & 1 & 1 & 0.33 \\ 0 & 0.33 & 1 & 1 & 0.33 \\ 0.33 & 0 & 0.33 & 0.33 & 1 \end{bmatrix}$$

3.3. Clustering ensemble

The voting method used in literature [11] is essentially a single-linkage clustering algorithm with a threshold of 0.5. It is a rather simple but effective consensus function. We use it to get the final partition in following steps.

a) Marking the first object of the dataset as a new cluster C_1;

b) Reviewing the objects from x_2 to x_n. Here we will determine the associability between an object x_i and an existing cluster C_j, which is defined as follows.

$$asso(x_i, C_j) = \max_k asso(x_i, x_j^k), x_j^k \in C_j$$

If $\max_j asso(x_i, C_j)$, i.e. the associability between object x_i and the most associative cluster to x_i is equal to or greater than 0.5, x_i is put into the most associative cluster of it. Otherwise, x_i is marked as a new cluster.

c) While all objects are processed, the final partition $h = \{C_1, C_2 \dots .\}$ is obtained. The isolated clusters that only include one object are marked as noise.

4. Implementation

We use four datasets to analyze and verify the EDBSCAN method.

- Two 2-dimention datasets, DS1 and DS2, shown in Fig.4. DS1 comprises three circular groups of objects, and DS2 comprises 4 irregular groups and some noise objects. They are all "good" dataset, i.e. there are clear boundaries between high-density areas and low-density ones.
- The Pen-Based Recognition of Handwritten Digits dataset from UCI Machine Learning Repository [16] (abbr. PenDigit). It is a digit database of 250 samples from 44 writers. The samples written by 30 writers are used for training, cross-validation and writer dependent testing, and the digits written by the other 14 are used for writer independent testing. We only use the latter in our implementation.
- The Optical Recognition of Handwritten Digits dataset from UCI Machine Learning Repository [16] (abbr. OptDigit). It's composed of the bitmaps of handwritten digits extracted via the preprocessing programs made available by NIST from a total of 43 people, 30 contributed to the training set and different 13 to the test set. Only the test set is used in our implementation.

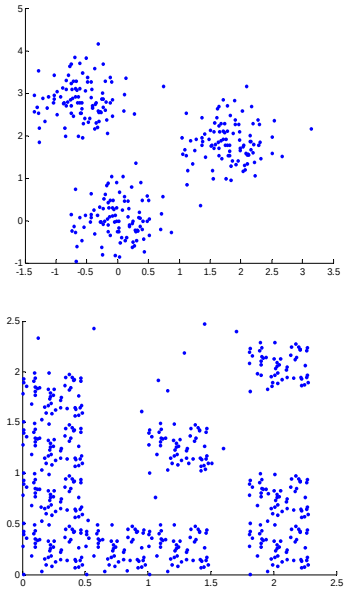

Fig.4: The two 2-dimension datasets DS1 and DS2.

The information of the four datasets is summarized in Table 1.

Dataset	Dimensions	Objects	Groups
DS1	2	300	3
DS2	2	520	4
PenDigit	16	3498	10
OptDigit	64	1797	10

Table 1: summary of datasets

We also perform DBSCAN on the four datasets in order to make a contrast. The value of *Eps* in DBSCAN is selected through the observation method described in 2.1. The validity of clustering results are measured via the F-measure method [5][10]. All experiments are done using Matlab2006b in Pentium IV3.0. The EDBSCAN results of DS1 and DS2 are illustrated in Fig.5. The complete experimental data of EDBSCAN is presented in Table 2, and that of DBSCAN is presented in Table 3.

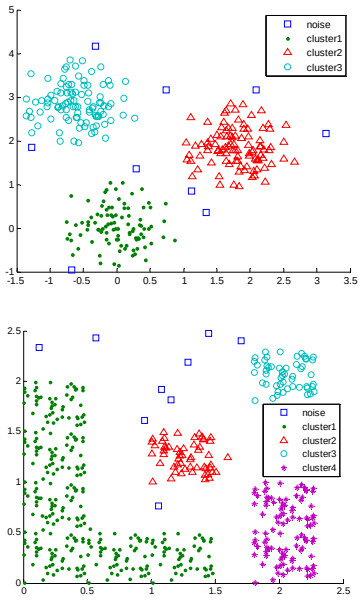

Fig.5: The EDBSCAN clustering results of DS1 and DS2.

Dataset	Interval	*r*	Clusters	Validity	Time (ms)
DS1	[0.18,0.3716]	8	3	96.4%	330
DS2	[0.075,0.191]	8	4	99.1%	919
PenDigit	[22.62,49.01]	8	15	68.1%	53439
OptDigit	[17.74,23.94]	8	13	71.3%	16056

Table 2: summary of EDBSCAN clustering results.

Dataset	Observation value	Clusters	Validity	Time (ms)
DS1	0.241	3	95.3%	184
DS2	0.1448	4	97.7%	501
PenDigit	40	15	66.1%	30764
OptDigit	22	11	66.2%	10655

Table 3: summary of DBSCAN clustering results

4.1. Validity and Stability Analysis

The value of *Eps* is not very sensitive in DS1 and DS2, as well as in other datasets in which the density boundary is obvious. For density based clustering, they are "good" datasets. We select another two values of

Eps by moving the observation value to the left and right and perform DBSCAN on DS1 and DS2 with them, as shown in Table 4.

Dateset	*Eps*	Clusters	Validity
DS1	0.22	3	90.5%
	0.241 (observation value)	3	95.3%
	0.26	3	95.8%
DS2	0.13	4	95.6%
	0.1448 (observation value)	4	97.7%
	0.16	4	97.0%

Table 4: DBSCAN result of DS1 and DS2 using different values of Eps.

From Table 4 we can see that there is no noticeable difference among the results of DS1, as well as those of DS2. It implies that there is a relatively broad range of valid *Eps* value around the theoretically best value. In this situation, the validity of the result of EDBSCAN is better than that of most single DBSCAN instances, as shown in Table 2 and Table 4. It's mainly because every partition of EDBSCAN includes some relatively complete information about the dataset. The information of all partitions is combined and amplified after clustering ensemble.

While dealing with the "bad" datasets in which there is no obvious boundary between high-density areas and low-density areas such as PenDigit and OptDigit, both EDBSCAN and DBSCAN can not achieve high-quality results. But the validity of EDBSCAN is still better than most DBSCAN instances. And more importantly, the stability of EDBSCAN is far better than that of DBSCAN. Some examples are shown in Table 5.

Dateset	*Eps*	Clusters	Validity
PenDigit	34	23	44.2%
	36	19	70.9%
	40 (observation value)	15	66.1%
	44	8	47.2%
OptDigit	18	23	50.0%
	20	17	73.5%
	22 (observation value)	11	66.2%
	24	7	46.5%

Table 5: DBSCAN result of PenDigit and OptDigit using different values of Eps.

Table 5 presents much instable results when performing DBSCAN with different *Eps* values. In this situation, even the observation method can hardly find an appropriate value, not to mention any automatic estimation methods. But the EDBSCAN method can avoid such a difficulty effectively. Though most partitions are low-quality and only have small fractions of the information of the dataset, clustering ensemble can combine and amplify these small fractions and find those highlight clusters correctly at least. So the validity of EDBSCAN is stable and still better than that of most single DBSCAN instances as shown in Table 2 and Table 5.

In addition, the size of the interval cannot affect the stability much. Let's narrow and extend the interval and perform EDBSCAN again, the results are shown in Table 6.

dataset	Groups	Interval	Clusters	Validity
PenDigit	10	[24.03,45.21]	15	68.8%
		[22.62,49.01] (original interval)	15	68.1%
		[17.83,58.61]	9	67.4%
OptDigit	10	[18.93,21.19] (original interval)	11	71.6%
		[17.74,23.94]	13	71.3%
		[15.31,27.69]	14	69.2%

Table 6: EDBSAN results with different size of intervals.

Table 6 reveals the stability of processing PenDigit and OptDigit with different intervals. Though there are slight differences between them, the results are still stable enough to be considered as valid final partitions and are more valid than those of most single DBSCAN instances shown in Table 5.

4.2. Time Consumption Analysis

Undoubtedly the speed of EDBSCAN is slower than that of DBSCAN. However, there is no magnitude difference between them. The time consumption of DBSCAN is mainly used to calculate the distance matrix D_X described in 3.1, whereas the distance matrix is the same to all the r DBSCAN instances of EDBSCAN. So D_X is needed to be calculated only once during the process of EDBSCAN clustering. Let the time consumption of EDBSCAN be $\Delta T_{EDBSCAN}$ and that of DBSCAN be ΔT_{DBSCAN} . The ratio of $\Delta T_{EDBSCAN}$ and ΔT_{DBSCAN} is:

$$R = \frac{\Delta T_{EDBSCAN}}{\Delta T_{DBSCAN}}$$

We calculate the ratio R for the 4 datasets respectively according to the data shown in 0 and 0.

$$R_{DS1} = \frac{330}{184} = 1.79, \quad R_{DS2} = \frac{919}{501} = 1.83$$

$$R_{PenDigit} = \frac{53439}{30764} = 1.73, \quad R_{OptDigit} = \frac{16056}{10655} = 1.51$$

So an EDBSCAN instance with $r = 8$ only consumes twice or less the time of a DBSCAN instance. Obviously, there is no magnitude difference between them. While selecting a very big integer for *r*, the ratio will grow. But the experiments show that an *r* bigger than 10 does few favors for the validity of the final partition.

5. Conclusions

DBSCAN is a classic density based clustering algorithm. The main difficulty of DBSCAN is to choose an appropriate value of *Eps*. To solve this problem we proposed an ensemble DBSCAN method. Instead of trying to find a single accurate value of *Eps*, we introduce the idea of clustering ensemble in which multiple values of *Eps* are picked up from an interval formed according to the statistical characteristic of the 4-nearest distances. With these values multiple DBSCAN clustering are performed and the voting method, a consensus function, is used to combine the results to get the final partition. Experiments show that the EDBSCAN method goes beyond the DBSCAN method both in validity and stability.

The EDBSCAN method greatly improves the ability of adapting to diverse datasets. It also achieves the clustering automation because no parameters need to be specified manually. This method can be used in most unmanned clustering applications such as intrusion detection, network Text Mining, etc. and make significant contributions to the validity and stability of them.

Acknowledgement

This work is partially supported by National High-Tech Research and Development Plan of China (Grant No. 2003AA144050).

References

[1] J. MacQueen, Some methods for classification and analysis of multivariate observations, *Proc. of the Fifth Berkeley Symposium on Math*, pp. 281-297, 1967.

[2] M. Ester, H.P. Kriegel, J. Sander., A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, *Proc. of 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96)*, pp. 226-231, 1996.

[3] A. Hinneburg, D.A. Keim, An Efficient Approach to Clustering in Large Multimedia Databases with Noise, *Proc. Of the 4th Int. Conf. on Knowledge Discovery and Data Mining*, pp. 58-65, 1998.

[4] M. Ankerst, M-M. Breunig, H.P. Kriegel, OPTICS: Ordering Points To Identify the Clustering Structure, *Proc. of ACM SIGMOD'99 Int. Conf. on Management of Data*, pp. 49-60, 1999.

[5] P.N. Tan, M. Steinbach, V. Kumar, *Introduction to Data Mining*, Addison Wesley, US Ed edition, 2005.

[6] P.J. Feng, L.D. Ge, Adaptive DBSCAN-based algorithm for constellation reconstruction and modulation identification, *Proc. of Radio Science Conference*, pp. 177-180, 2004.

[7] M. Halkidi, M. Vazirgiannis. Clustering validity assessment: finding the optimal partitioning of a data set, *Proc. IEEE International Conference*, pp 187-194, 2001.

[8] S.H. Yue, P. Li, J.D.Guo, et al, A statistical information-based clustering approach in distance space, *Journal of Zhejiang University SCIENCE*, 6A(1): 71-78, 2005.

[9] C.Y. Lin, C.C. Chang, C.C. Lin, A New Density-Based Scheme for Clustering Based on Genetic Algorithm, *Fundamenta Informaticae*, Volume 68, Issue 4: 315-331, IOS Press, The Netherlands, 2005.

[10] M. Steinbach, G. Karypis, V. Kumar, A Comparison of Document Clustering Techniques, *Technical Report*, Report Number: 00-034, Minnesota: University of Minnesota-Computer Science and Engineering, 2000.

[11] A.L.N. Fred, Finding Consistent Clusters in Data Partitions, *Proc. Of 2nd Int. Workshop on Multiple Classifier Systems*, LNCS 2364:309-318, 2001.

[12] A. Strehl and J. Ghosh, Cluster ensembles - a knowledge reuse framework for combining multiple partitions, *Journal of Machine Learning Research*, 3: 583-617, 2002.

[13] A .Topehy, A. K Jain, W Punch, A. Mixture Model for Clustering ensembles, *Proc. of the 4th SIAM International Conference on Data Mining*, pp. 379-390, 2004.

[14] Ana Fred, Anil K Jain, Evidence accumulation clustering based on the K-Means algorithm, *Proc. of the International Workshops on Structural and Syntactic Pattern Recognition* (SSPR 2002), pp. 442-451, 2002.

[15] H Ayad, M Kame1, Finding Natural Clusters Using Multi-Clusterer Combiner Based on Shared Nearest Neighbors, *Proc. of the 4th International Workshop on Multiple Classifier Systems(MCS'03)*, Volume 2709 of Lecture Notes in Computer Science, Springer, pp. 166-175, 2003.

[16] http://www.ics.uci.edu/~mlearn/MLSummary.html