Multi-Density Datasets Clustering Using K-Nearest Neighbors and Chebyshev's Inequality

Amira Bouchemal, Mohamed Tahar Kimour
Laboratory of Embedded Systems, Badji Mokhtar-Annaba University, Algeria EM-Alger Business School, Algiers, Algeria
E-mail: bouchemal.ami@gmail.com , rahatkimm@gmail.com

Keywords: clustering, density-based clustering, DPC, DBSCAN

Received:

Density-based clustering techniques are widely used in data mining on various fields. DBSCAN is one of the most popular density-based clustering algorithms, characterized by its ability to discover clusters with different shapes and sizes, and to separate noise and outliers. However, two fundamental limitations are still encountered that is the required input parameter of Eps distance threshold and its inefficiency to cluster datasets with various densities. For overcoming such drawbacks, a statistical based technique is proposed in this work. Specifically, the proposed technique utilizes an appropriate k-nearest neighbor density, based on which it sorts the dataset in ascending order and, using the statistical Chebyshev's inequality as a suitable means for handling arbitrary distributions, it automatically determines different Eps values for clusters of various densities. Experiments conducted on synthetic and real datasets have demonstrated its efficiency and accuracy. The results indicate its superiority compared with DBSCAN, DPC, and their recently proposed improvements.

Povzetek: Predlagana metoda izboljšuje metodo gručenja DBSCAN z uporabo najbližjih sosedov in neenakosti Chebysheva.

1 Introduction

Clustering is a powerful machine learning tool for detecting structures in datasets. Detecting, analyzing, and describing natural clusters within a dataset, is of fundamental importance to a number of various fields [1-4]. Such fields concern bioinformatics to identify similar genes, marketing to segment customers to establish market research, social sciences, psychology, biology, security, computer vision and image processing in various areas such as the medical or industrial fields, and so on.

Clustering techniques are generally divided into four classes: 1) partitioning [5-7], 2) hierarchical [8], 3) model-based [9], and 4) density-based [10-18]. Due to their ability to support clusters of arbitrary shapes, density-based clustering algorithms are naturally widely used. They consider clusters as being dense regions separated by less dense or scattered areas. They determine the clusters as well as their number, while identifying the noise.

As among the most popular density based clustering algorithm, DBSCAN [10] has several key advantages. First, it groups data into clusters of arbitrary forms. Second, it does not require that the number of clusters be given as input. The number of clusters is determined by the nature of the data and the values of *Eps* and *minPts*. Third, it is insensitive to the order of entry of points in the dataset. All these strengths are very important for any clustering algorithm.

However, DBSCAN requires the user to provide a global separation threshold *Eps* and the minimum number of points for any cluster.

DPC (density-based clustering) also requires the user to enter the distance threshold *dc*.

For these two types of algorithms and their variants, the clustering results depend on these input parameters which are difficult to determine by the
user, in the absence of precise rules for their calculation. Though they provided important advances in the clustering techniques and more and more widely used, they are still exhibiting limits to efficiently deal with complex datasets having clusters of varying shapes, sizes and densities [19-21].

In this article, we propose a new density-based clustering approach that overcomes these limitations. It is based on the main idea of characterizing the data points in the datasets with their local densities and using Chebyshev's inequality [22-23] to estimate local *Eps* for every cluster. In doing so, we not only automatically determine *Eps*, but also we efficiently handle clusters with various densities and shapes.

We define the concept of local neighbors of a given data point *x*, from which it determines the local threshold *Eps* for the current cluster. Indeed, for each data point *x* in the dataset, we calculate the local density so that it is the smallest in a dense region and the largest in a sparse region. Next, we sort the dataset on the density key, in descending order.

In doing so, the first data point with the highest density will initiate the construction of the first cluster by calculating its local separation threshold *Eps* using the Chebyshev inequality [22] and its distances, applied to the k-nearest neighbors. Following which, our algorithm extends the cluster initially built from these k-neighbors in the same way as DBSCAN.

When no new Eps- neighbor is found, the algorithm closes the current cluster and repeats the same procedure for unclassified data points. It stops when the dataset is empty and thus, all the clusters are automatically determined, without requiring any distance threshold nor number of clusters. In doing so, the local *Eps* can be adapted to a specific cluster density, which is an advantage of the proposed method compared to DBSCAN and its improvements.

The rest of the article is organized as follows. Section 2 presents the related work. Section 3 Presents DBSCAN and DPC algorithms. Section 4 describes our approach to cluster datasets with various densities, shapes and sizes. Section 5 presents the experimental evaluation. Section 6 presents a discussion. Finally, section 7 concludes the paper and outlines the future work.

2 Related work

The literature presents several data clustering techniques, such as hierarchical, model-based, and density-based algorithms. Partitioning-based clustering algorithms such as k-means and the likes [5] specify an initial number of clusters at the start and represent each cluster by a centroid, and objects close to the same centroid are considered to be similar. These algorithms determine all the clusters at the same time by aiming for a strong intra-group similarity. They require a number of user-defined clusters that are often unknown in practice. In addition, they build spherical clusters, which are not suitable for arbitrarily shaped clusters.

Hierarchical clustering algorithms [8] are bottom-up approaches such that each data point begins in a separate cluster, and the pairs of clusters at the bottom are merged as we move up the hierarchy. In hierarchical clustering algorithms, clustering process handles clusters of various shapes well but they are not suitable for clusters of varying density.

Model-based clustering methods [9] estimate a model for data by incorporating a measure of probability or uncertainty into cluster assignments. Model-based clustering attempts to address this concern and provide flexible allocation when observations are likely to belong to each cluster. In addition, model-based clustering offers the added benefit of automatically identifying the optimal number of clusters.

In density-based clustering methods, Rodriguez and Laio [16] proposed a new clustering algorithm by finding density peaks which are the potential centers of the cluster. In this algorithm, each data point can obtain its own local density and its distance to the nearest neighbor which has a higher density. Depending on the density and distance of the points, the result of the grouping can be obtained in a single step without further iteration. Since clustering methods based on the search for density peaks produce clusters of arbitrary shapes, the notion of "center" is somewhat misleading [17-18]. These clustering algorithms based on density peaks have a deficiency in the allocation process, which is likely to trigger a domino effect. They require a user-defined parameter (cut-off distance) and visual judgment to estimate cluster centers based on the decision diagram. In addition, they cannot process certain non-spherical data sets such as Spiral.

DBSCAN [10] is probably the most popular density-based clustering algorithm for building clusters of arbitrary shape. A cluster is created if there is a *minPts* of objects in a dense region with an *Eps* (distance threshold); *minPts* (minimum cluster size), which are user defined. DBSCAN starts with an arbitrary starting point. The Eps-neighborhood of this point is retrieved and a cluster is created, if it contains a sufficient number of points. Otherwise, the point is considered as noise. The selected point and its accessible points are considered as a cluster. The cluster will increase several times by adding other points which are in the distance *Eps* of the accessible points as new accessible points. The algorithm continues until all the points have either a cluster label or there are less *minPts* points around them and they are therefore considered as noises.
Table 1 summarizes the comparison between ones of the most notable works on density-based clustering.

Method	**Key features**	**limitation**	**reference**
DBSCAN	Users need to define two parameters: What distance *Eps* to determine for each observation the Eps-neighborhood? What is the minimum number of neighbor's *minPts* necessary to consider that an observation is a core observation?	DBSCAN fails to discover clusters of varying densities and requires two user-defined input parameters *Eps* and *minPts*.	[10]
DPC	User needs to select cluster centers from a decision graph to identify clusters	Sensitive to user input and results are dependent on the center selected by user	[16]
GMDBSCAN	uses a space division technique and considers each grid as a separate part, then it estimates the independent *minPts* for each grid according to its density	applies several DBSCANs on every grid and finally, it uses a distance-based method to improve the boundaries	[11]
DPCSA	User needs to select cluster centers from a decision graph to identify clusters in two stages	Sensitive to user input	[27]

Table 1: Comparison using key features and limitations

DBSCAN and its aforementioned variants [9-10][12-15] have significant advantages: i) discovering clusters with various shapes, sizes and noises, ii) robust with outliers, iii) no need to specify the number of clusters in the data, iv) insensitive to the order of the points. However, these algorithms fail to discover clusters of varying densities and require two user-defined input parameters *Eps* and *minPts*.

The GMDBSCAN method [11] uses a space division technique and considers each grid as a separate part, then it estimates the independent *minPts* for each grid according to its density, after which it applies several DBSCANs on every grid and finally, it uses a distance-based method to improve the boundaries.

Hou et al. proposed a clustering algorithm without parameters based on the combination of DSets (dominant sets) and DBSCAN algorithms [10]. In the AGED algorithm [15], the value of the *Eps* as defined in DBSCAN is determined according to local densities. Wang et al. proposed a modified DBSCAN algorithm to automatically determine the *Eps* values for different data distributions [13].

In [24], the author introduced a method which estimates the local density - for each point of the dataset - as the sum of the distances to the k-nearest neighbors. By arranging the data points in ascending order according to the local density, the proposed algorithm determines clusters of various densities. However, the density threshold associated with the cluster initiator sample lacks precise definition, which leads to clustering failures in certain situations.

In [27], the authors proposed the DPCSA algorithm as an improvement of DPC [16], however, users are still need to select cluster centers from a decision graph to identify clusters in two stages.

3 Density–based clustering

A clustering process partitions a dataset into groups of data points according to their proximity to each other. It takes as input a set of data points and a measure of similarity between them, and produces as output a set of partitions describing the general structure of the dataset. Density-based clustering techniques are one of the most popular unsupervised learning techniques used in machine learning algorithms. They proceed by detecting areas where data points are concentrated and where they are separated by areas that are sparse or empty. Points that do not belong to any cluster are labeled as noise.

Density-based clustering analysis is well known and widely appreciated for two desirable features. First, it does not require a predefined number of clusters as an input parameter, and is robust to noise. Second, it can discover clusters with arbitrary shapes, and not just ball-shaped clusters. Density based clustering algorithm has played a vital role in finding non linear shapes structure based on the density.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is the most widely used density-based algorithm. It uses the concept of density reachability and density connectivity.

DBSCAN is a simple algorithm that defines clusters using local density estimation. It can be divided into 4 steps:

Step 1: For each observation, we look at the number of points at most a distance *Eps* from it. This area is called the Eps -neighborhood of the observation.

Step 2: If an observation has at least a number of neighbors including itself, it is considered a core observation. A high-density observation was then detected.

Step 3: All observations in the vicinity of a core observation belong to the same cluster. There may be core observations close to each other. Consequently, step by step, we obtain a long sequence of core observations which constitutes a single cluster.

Step 4: Any observation that is not a core observation and that does not include a core observation in its vicinity is considered an anomaly.

Therefore, to apply DBSCAN, users need to define two parameters: What distance *Eps* to determine for each observation the Eps-neighborhood? What is the minimum number of neighbor's *minPts* necessary to consider that an observation is a core observation? These two parameters are provided freely by the user.

Unlike the k-means algorithm or hierarchical ascending classification, there is no need to define the number of clusters, which makes the algorithm less rigid. Another advantage of DBSCAN is that it also allows to manage outliers or anomalies.

On the other hand, the concept of distance and choice of *Ep*s and *minPts* are critical for the clustering quality. In this algorithm two points are fundamental:

What is the metric used to evaluate the distance between an observation and its neighbors? What is the ideal *Eps*?

In the DBSCAN we generally use the Euclidean distance, let $p = (p_1, \dots, p_n)$ and $q = (q_1, \dots, q_n)$:

$$d(\mathbf{p},\mathbf{q}) = \sqrt{(p_1-q_1)^2 + (p_2-q_2)^2 + \cdots + (p_i-q_i)^2 + \cdots + (p_n-q_n)^2} = \sqrt{\sum_{i=1}^{n}(p_i-q_i)^2}.$$

At each observation, to count the number of neighbors at a distance *Eps* at most, we calculate the Euclidean distance between the neighbor and the observation and check if it is less than *Eps*.

4 Proposed method

In this section, we present our method to address the abovementioned problems in clustering datasets with various densities. The main idea is to use local density concept and the statistical Chebyshev's inequality [22- 23] to automatically determine a separation threshold for every density class in a dataset, and we build every cluster with its associated distance threshold, leading to clusters with various densities and shapes. Our algorithm we call MDC (Multi-

Density Clustering) uses the K-nearest-neighbors' information to define a local density of every data point as follows:

$$\rho_i = \sum_{j \in KNN(i)} \exp\left(-d_{ij}\right) \qquad (1)$$

Where ρ_i is the local density of a data point i, and d_{ij} is the distance between data point i and a neighboring data point j in the neighborhood $KNN(i)$. In doing so, the density value is reduced from the global density to the local density of the k neighbors closest to the data point, and the greater the distance between a data point and its neighborhood k, the smaller is the local density. This finding will better reflect the local information in the dataset.

4.1. Chebyshev's inequality

Chebyshev's inequality is adequate for completely arbitrary distributions [22]. It constitutes an efficient estimation technique facilitating the calculation of the probability solely on the basis of the mathematical expectation and the variance, without requiring information on the distribution. Chebyshev's inequality is highly useful in that it can be applied to completely arbitrary distributions [2].

Chebyshev's inequality holds in the case when x (random variable) with:

a mean $(M(x) = \mu < \infty)$ and a variance $(V(x) = \sigma 2 < \infty)$.

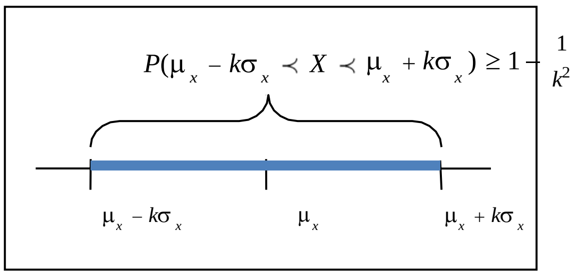

$P(\mu_x - k\sigma_x \prec X \prec \mu_x + k\sigma_x) \geq 1 - \frac{1}{k^2}$
$\mu_x - k\sigma_x$
μ_x
$\mu_x + k\sigma_x$

Figure 1: Interval of μx

This inequality is expressed as follows:

$$\mathrm{P}\{|\mathrm{x} - \mu| \geq \mathrm{k}\sigma\} \leq \frac{1}{\mathrm{k}^2} \qquad (2)$$

and it is valid for any $k > 1$.

We can express a variant of Chebyshev's inequality as the following expression:

$$\mathrm{P}\{|\mathrm{x} - \mu| < k\sigma\} > 1 - \frac{1}{\mathrm{k}^2} \qquad (3)$$

and we reformulate in the following inequality (Figure 1) :

$$P(\mu_x - k\sigma_x < X < \mu_x + k\sigma_x) \geq 1 - \frac{1}{k^2} \qquad (4)$$

It means that "greater than $(1 - \frac{1}{k^2})$ of population falls within k $(k > 1)$ standard deviations (i.e. $k\sigma$) from the population mean μ". In other words, it states that for a distribution, the percentage of observations that lie within k standard deviations is at least: $1 - \frac{1}{k^2}$ (Figure 1).

4.2. The proposed algorithm

In the clustering step, the proposed clustering algorithm is used to extract spatial clusters from the dataset. Finally, the outputs are evaluated and visualized. Algorithm 1 takes as input a dataset and produces clusters with different shapes and various densities.

It takes the point of the highest density as a cluster initiator CI that absorbs its k nearest neighbors $\{v1, v2, \ldots, vk\}$ to build a new cluster. Thus the current initiated cluster is composed of data points $C = \{CI, v1, v2, \ldots, vk\}$. At this step, we use Chebyshev's inequality to estimate the local Eps of the cluster. We calculate pairwise distances of C as distances between every two data points in C to have $D = \{d1, d2, \ldots, dm\}$. Eps is estimated as the bound of D. To estimate the maximum extending of the current cluster, we consider the collection of distances D and we apply the following Chebyshev's formula:

$$Eps = mean + 3.5 \times std \qquad (5)$$

Where mean is the mean of D and std is the standard deviation of D. Afterward, we remove the classified data points from the dataset. We initiate a new cluster by taking out a new initiator and proceed in the same manner, while removing the classified data points from the dataset, while using $minPts$ to control the minimum density allowed within a cluster, and so on till obtaining an empty dataset. In doing so, an Eps value is estimated for every new cluster, leading to a set of clusters with various densities.

```
Algorithm 1: multi-density datasets clustering (MDC)
Input: dataset $X$, $k$ (for the k-nearest neighbors) and $minPts$ (may be 4 or 5)
Output: a set of multi-density clusters

Step 1: Calculate the density of every data point in the dataset $X$ using equation (1), store the result in the vector $RO$ and create an empty list Clusters.
Step 2: Sort $RO$ in decreasing order of densities.
Step 3: Take out the first data point $p1$ in $RO$ as a cluster initiator.
Step 4: Construct a new cluster $C = \{p1,v1,...,vk\}$, where $v1,...,vk$ are the k-neighbors of $p1$, and remove these data points from $RO$.
Step 5: From $C$, construct $D = \{d1,d2,...,\}$, pairwise distances between every two data points in $D$.
Step 6: From $D$, calculate the local $Eps$, using formula (5).
Step 7: Using $RO$, expand $C$ using the local $Eps$ and $k$, while controlling the size of the cluster using $minPts$.
Step 8: Append $C$ to Clusters, and remove from $RO$ all the data points in $C$.
Step 9: If $RO$ is not empty, repeat from step3 to step 9.
Step 10: Output Clusters.
```

In step 1, for every data point in the dataset, it calculates its distances from the k neighboring data points, calculates the local density according to equation (1) for every data point, stores the result in RO and creates an empty list Clusters. In step 2, it sorts RO in decreasing order of their local densities. In step 3, it takes out the first data point $p1$ (it has the highest density) as a new cluster initiator. In step 4, algorithm 1 builds a new cluster $C = \{p1,v1,.., vk\}$, where $v1,v2,...,vk$ are the k-neighbors of $p1$, and removes these points from RO.

To be able to search the remaining members of the cluster from RO, we need to estimate a local Eps for this cluster. To this end, at step 5, algorithm 1 constructs an initial set of pairwise distances, (between every two data points in the current cluster) and stores the result in D. In step 6, it applies Chebyshev's inequality on D to determine the local Eps of the current cluster using equation (5).

In step 7, it expands the current cluster C by searching reachable data points using the calculated Eps in the same manner as in DBSCAN, while removing from the dataset all the data points that compose the current cluster. Moreover, $minPts$ is utilized to control the minimum size of the cluster. $minPts$ value may be 4 or 5.

In step 8, it appends C to Clusters and removes all new classified data points from RO. In step 9, it checks the datasets, if it is empty it stops clustering and execute step 10 to output clusters, otherwise, it initiates a new cluster in the same manner and so on.

5 Experimental results

In order to validate the efficiency of the proposed MDC algorithm, we conducted dataset clustering experiments not only on synthetic datasets, but also on real data, with clusters of various densities, shapes and sizes [25-26], while comparing the results with those obtained by DBSCAN and DPC. The algorithms are implemented in the Integrated Development Environment of Python (Python IDE version 3.7.5). For DBSCAN, we used the scikit-learn library in Python.

In DPC the required dc is calculated using percentage value 2% as advised in [16]. We looked for the optimal parameter value of 0.7 to 3.0 with a step of 0.3. In order to simplify the implementation of DPC, we select M points with the first values of $\rho \times \delta$, directly as cluster centers. The *Eps* value varies in the range of 0.5 to 3.0 with the step is 0.3. The *minPts* values vary from 4 to 20.

Dataset	**DBSCAN**		**DPC**		**MDC (proposed)**	
	ARI	F1	ARI	F1	ARI	F1
Flame	0.968	0.987	0.99	1.00	1.00	1.00
Jain	1.00	1.00	0.722	0.923	1.00	1.00
Compound	0.932	0.939	0.630	0.777	1.00	1.00
Aggregation	0.908	0.938	1.00	1.00	0.991	0.996

Table 2: Comparison using ARI and F-measure metrics

Table 2 shows the quality measures of the algorithms, when applied to the datasets in Figure 2. By using two cluster validity indexes, F-measure with $\alpha = 1$ (F1) and adjusted rand index (ARI), we evaluate and discuss the performances of these three methods. F-measure is the weighted average (or harmonic mean) of Precision and Recall. Using F1 score as a metric, we are sure that if the F1 score is high, both precision and recall of the classifier indicate good results. ARI has the maximum value 1, and its expected value is 0 in the case of random clusters. A larger adjusted rand index means a higher agreement between two partitions. ARI is recommended for measuring agreement even when the partitions compared have different numbers of clusters.

Below we show through Figure 2, the results of clustering produced by different algorithms. Flame dataset has two components with similar densities but different shapes. And, they are very close to each other. MDC and DPC clustered it correctly, but DBSCAN identified seven outliers as noise (Figure 2). Jain is divided in two components with different densities. DBSCAN, DPC, and MDC correctly identified the two components.

It is difficult to correctly distinguish the components in Compound, composed of six components with varying densities and shapes, with complex spatial relationships. We see in the upper left corner, two contiguous components. A small disc-like component is surrounded by a ring-like component. They are located at the bottom left. On the right, an irregularly shaped component is surrounded by a set of scattered, spatially overlapping points.

As shown in Figure 2, MDC correctly identified the six clusters of the compound dataset. Unlike DBSCAN and DPC where the first detected only five clusters, and a number of points considered as noise, while the second divided the ring component into two parts and merged the scattered component with the dense one. In the aggregation dataset, there are seven components with various sizes and shapes. Except the two slightly connected component pairs, the other components are independent of each other. DBSCAN groups each of these two pairs in one cluster. MDC and DPC produced correct results.

6 Discussion

Density-based clustering proceeds by calculating the density of each sample point. DBSCAN and DPC algorithms [16] are ones of the most emerging density based clustering algorithms. However, these clustering algorithms have certain defects.

In DBSCAN, required *Eps* and *minPts* are difficult to be determined and only global values of such parameters are used, leading to inaccurate clustering result of multi-density datasets.

In DPC algorithm, subjective selection of cluster centers using a decision graph, and the *dc* parameter with only a unique value does not handle the multi-density data. Aiming at the problem of the selection of the parameter *dc* of the DPC algorithm, in [27] the authors proposed the use of the parameters optimized according to the local distance of data points. However, with multi-density data, this cannot obtain stable clustering effect. To tackle these limitations, our method addresses the abovementioned problems in clustering datasets with various densities. Thanks to the Chebyshev's inequality, our approach allows to automatically determine a separation threshold for every density class in a dataset, and we build every cluster with its associated distance threshold.

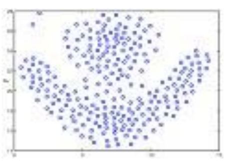
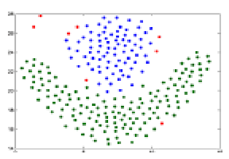
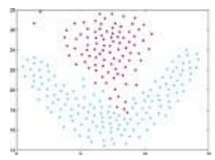
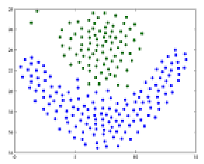
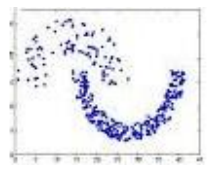
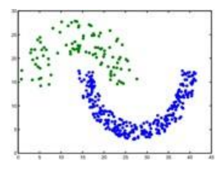
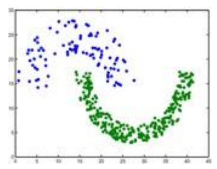
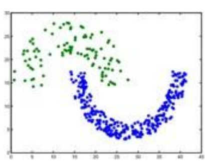
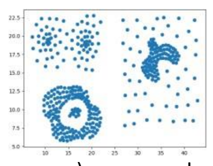
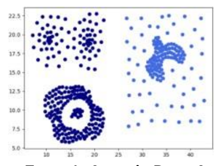
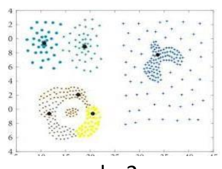
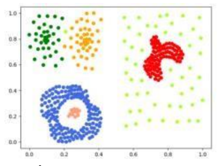
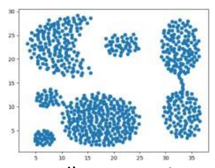
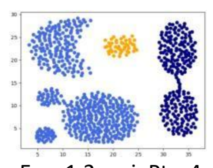
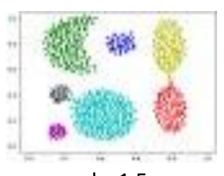
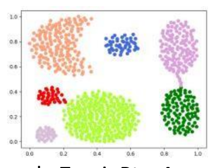
Original	DBSCAN	DPC	MDC
 *a) flame*	 Eps=2.5 , minPts=4	 dc=2	 k=5, minPts=4
 b) jain	 Eps=1.5 , minPts=4	 dc=1.4	 k=5, minPts=4
 c) compound	 Eps=1.4 , minPts=4	 dc=2	 k=5, minPts=4
 d) aggregation	 Eps=1.2 , minPts=4	 dc=1.5	 k=7, minPts=4

Figure 2: Clustering results of DBSCAN, DPC and MDC through a) flame, b) jain, c) compound, and d) aggregation

7 Conclusion and future work

This paper shows how to automatically cluster data into groups with multiple densities and shapes through determining multiple set of parameters *Eps* and *minPts*. To execute DBSCAN, users need to introduce *Eps* and *minPts* that are difficult to be determined. Moreover, DBSCAN simply uses the global *Eps* and *minPts* so that the clustering result of multi-density data is inaccurate.

To tackle such shortcomings, a new dataset clustering technique to efficiently handle clusters with various densities and shapes is presented in this paper. It introduced simple modification on DBSCAN. The main idea lies on the definition of local density of data points in the dataset and the use of Chebyshev's inequality to define a local threshold separation at the initiating step of each cluster, while using *minPts* to control the minimum density allowed within a cluster. The experiments conducted on synthetic and real-world datasets showed that the proposed algorithm provides important enhancements of clustering techniques on datasets with various density levels and shapes. As a future work, we plan to focus on applying MDC to solving practical problems in digital image processing such as segmentation and edge detection.

References

[1] Han, J.; Kamber, M.; Pei, J. Data mining: Concepts and techniques. Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2011; p. 696.

[2] Hennig, C.: What are the true clusters? Pattern Recognition Letters 64 (2015) 53-62

[3] T. Vo Van and T. Pham-Gia, Clustering probability distributions, Journal of Applied Statistics, vol. 37, no. 11, pp. 1891–1910, 2010.

[4] Jiawei Han, Micheline Kamber, Data Mining Concepts and Techniques, Morgan Kaufmann Publishers, Chapter 8, Pp. 363- 364, 2001.

[5] Ashish Mohan Yadav , A Survey on Unsupervised Clustering Algorithm based on K-Means Clustering, International Journal of Computer Applications 156(8):975-8887 dec. 2016

[6] Ahmed Fahim. An extended DBSCAN Clustering Algorithm. (IJACSA) International Journal of advanced computer science and applications, vol. 13, No. 3, 2022.

[7] Law Siew Xue, NazatulAini Abd Majid, Elankovan

A. Sundararajan, A Principal Component Analysis and Clustering based Load Balancing Strategy for Cloud Computing, TEM Journal. Vol. 9, Issue 1, Pages 93-100, Feb. 2020.

[8] Zheng Zheng, Fukai Cao, Song Gao, Amit Sharma. Intelligent Analysis and Processing Technology of Big Data Based on Clustering Algorithm. (2022).

[9] Rasmussen CE (1999),the Infinite Gaussian Mixture Model. In: Advances in Neural Information Processing Systems 12, MIT Press, Cambridge, MA, pp 554–560

[10] Yang Yang, Chen Qian, Haomiao Li, Yuchao Gao, Jinran Wu, Chan-Juan Liu, Shangrui Zhao. An efficient DBSCAN optimized by arithmetic optimization algorithm with opposition-based learning. The journal of supercomputing (2022) 78:19566-19604.

[11] Xiaoyun, C, Yufang, M, Yan, Z and Ping, W., GMDBSCAN: Multi-Density DBSCAN Cluster Based on Grid, IEEE International Conference on E- Business Engineering, ICEBE'08, pp. 780–783.

[12] Bryant, A.; Cios, K. RNN-DBSCAN: A Density-Based Clustering Algorithm using Reverse Nearest Neighbor Density Estimates. IEEE Trans. Knowl. Data Eng. 2018, 30, 1109–1121.

[13] W.-T. Wang, Y.-L. Wu, C.-Y. Tang, and M.-K. Hor, Adaptive Density-Based Spatial Clustering of Applications with Noise (DBSCAN) according to data, in Proc. ICMLC, Guangzhou, 2015, pp. 445-451.

[14] M. Ankerst, M. M. Breunig, H-P Kriegel, OPTICS: Ordering Points to Identify the Clustering Structure, Proceedings of ACM SIGMOD, pp. 49-60, (1999).

[15] N. Soni and A. Ganatra, AGED: Automatic Generation of Eps for DBSCAN, Int. J. of Computer Science and Information Security (IJSIS),, vol. 14, no. 5, pp. 536-559, 2016.

[16] Rodriguez, A.; Laio, A. Clustering by Fast Search and Find of Density Peaks. Science 2014, 344, 1492.

[17] Mehmood, R.; Zhang, G.; Bie, R.; Dawood, H.; Ahmad, H. Clustering by Fast Search and Find of Density Peaks via Heat Diffusion. Neurocomput. 2016, 208, 210–217.

[18] Aida Chefrour, Labiba Souici Meslati. AMF-IDBSCAN : Incremental Density Based Clustering Algorithm using Adaptive Median Filtering Technique. (2019).

[19] Yizhang Wang, Di Wang, Wei Pang, Chunyan Miao, Ah-Hwee Tan, You Zho, A Systematic Density-Based Clustering Method using Anchor Points, Neuro-computing , March 2020

[20] Stiphen Chowdhury and Renato Cordeiro de Amorim, An Efficient Density-Based Clustering Algorithm Using Reverse Nearest

Neighbor,
Computing Conference 2019, London, Nov. 2018

[21] T. Vo-Van A. Nguyen-Hai M. V. Tat-Hong and T. Nguyen-Trang, A New Clustering Algorithm and Its Application in Assessing the Quality of Underground Water, Scientific Programming, March 2020

[22] Jin Chu Wu, Charles L. Wilson, Using Chebyshev's Inequality to Determine Sample Size in Biometric Evaluation of Fingerprint Data, Interagency/Internal Report (NISTIR) – 7273, November 1, 2005

[23] Zhang Kun, Wang Cui Rong, WSN Cong., Adaptive Threshold Background Modeling Algorithm Based on Chebyshev’s Inequality, Computer Science, 40(4): 287–291. 2013.

[24] Ahmed Fahim , Clustering Algorithm For Multi-Density Datasets, Romanian Journal of Information Science and Technology, Vol. 22, Number 3–4, 2019, 244–258

[25] Clustering datasets: http://cs.uef.fi/sipu/datasets/. Accessed on: January 23, 2020.

[26] M.L. K. Bache, UCI machine learning repository, 2013, http://archive.ics.uci.edu/ml., Accessed on: January 2

[27] Donghua Yu, Guojun Liu, Maozu Guo;Xiaoyan Liu; Shuang Yao, Density Peaks Clustering Based on Weighted Local Density Sequence and Nearest Neighbor Assignment, IEEE Access, vol. 7, 2019.