Front. Comput. Sci., 2021, 15(1): 151308
https://doi.org/10.1007/s11704-019-9059-3

REVIEW ARTICLE

A survey of density based clustering algorithms

Panthadeep BHATTACHARJEE (✉), Pinaki MITRA

Department of Computer Science and Engineering, Indian Institute of Technology, Guwahati 781039, India

Abstract Density based clustering algorithms (DBCLAs) rely on the notion of density to identify clusters of arbitrary shapes, sizes with varying densities. Existing surveys on DBCLAs cover only a selected set of algorithms. These surveys fail to provide an extensive information about a variety of DBCLAs proposed till date including a taxonomy of the algorithms. In this paper we present a comprehensive survey of various DBCLAs over last two decades along with their classification. We group the DBCLAs in each of the four categories: *density definition, parameter sensitivity, execution mode and nature of data* and further divide them into various classes under each of these categories. In addition, we compare the DBCLAs through their common features and variations in citation and conceptual dependencies. We identify various application areas of DBCLAs in domains such as astronomy, earth sciences, molecular biology, geography, multimedia. Our survey also identifies probable future directions of DBCLAs where involvement of density based methods may lead to favorable results.

Keywords clustering, density based clustering, survey, classification, common properties, applications

1 Introduction

Clustering is an unsupervised learning task that groups data objects or patterns based on similarity measures. Such objects may exist as data points in a $\mathcal{R}^d$ space. Entities belonging to a certain cluster have greater similarity between them than with an entity belonging to a different cluster [1–3]. Cluster analysis is done with the objective of summarization or improved understanding of the data in context, e.g., grouping of related documents for browsing, finding protein structures and genes having analogous functions, or as a technique to compress data [4]. A large number of clustering techniques have been developed for pattern analysis, grouping, decision making, document retrieval, image segmentation, data mining, yet many significant challenges still remain in determining the clusters correctly.

Clustering approaches are broadly classified into partitional, hierarchical and density based methods (Refer to Fig. 1) [1]. Partitional method creates partition of the data instead of a clustering structure. The partitional clustering approach involves squared error method, e.g., K-means algorithm, graph theoretic clustering, mixture resolving, e.g., EM algorithm and mode seeking method [1].

Hierarchical clustering produces a dendrogram that represents the nested grouping of patterns, e.g., Chameleon [5]. Hierarchical method adopts agglomerative or divisive approach to determine the clustering.

Density based clustering depends on the notion of finding density of a region. The objective of DBCLAs is to find clusters at different levels of granularity with appropriate noise filtering. Density notion used by the DBCLAs enables segregation of compact regions in the data space from the noise. In DBCLAs, clusters are identified as areas of higher density than the remainder of the data space [6]. DBCLAs facilitate detecting clusters of arbitrary shapes. Over a period of last two decades, numerous density based clustering techniques have been proposed. These methods aim to extract clusters of relatively uniform densities lying across the data space.

The other prominent clustering paradigms are: nearest neighbor based clustering, fuzzy clustering, clustering based on artificial neural network (ANN) and kernel based techniques [1]. Evolutionary approaches for clustering have also been proposed that makes use of the population of solutions to obtain the globally optimal partitioning of data.

The clustering algorithms come with their own set of challenges. Depending on the properties of data and the mechanism adopted to form clusters, we mention certain drawbacks incurred by various clustering algorithms:

1) **Inability to detect clusters correctly in high dimensional space:** Most of the challenges faced by the clustering algorithms are particularly related to the quality of clusters obtained in voluminous high dimensional space. The phenomenon of "curse of dimensionality" [7] is one of the major bottlenecks due to which correctness of clusters cannot be guaranteed among data points having numerous attributes.
2) **Resource constraint:** With the use of very large datasets, there also exists an issue regarding availability of computational resources. Clustering algorithm such as CLARANS [8] is based on the assumption that its data reside within memory, an approach that fails for very large datasets.
3) **Inability to detect non-globular clusters:** Existing literature have also pointed out the inability of various methods to extract arbitrarily shaped clusters having variable

Received February 17, 2019; accepted September 9, 2019

E-mail: panthadeep@iitg.ac.in; pinaki@iitg.ac.in

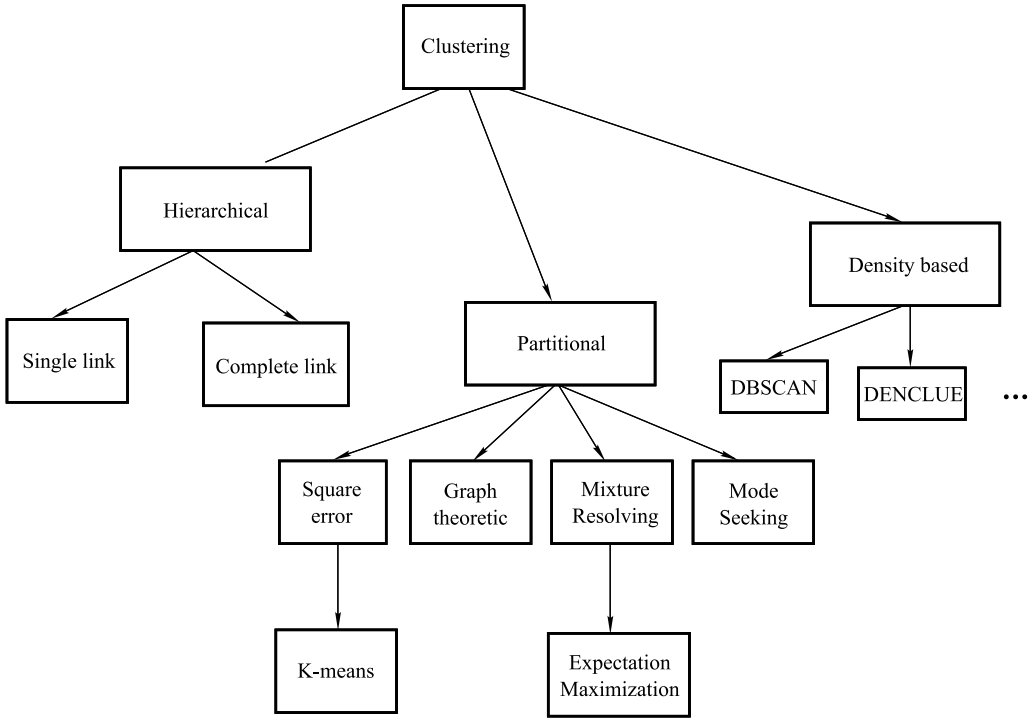

Fig. 1 Representation of various clustering paradigms

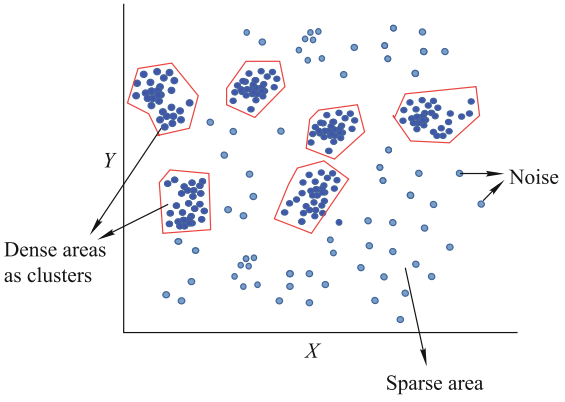

Fig. 2 Dense areas represent the clusters and subsequently the noises are filtered out. DBCLAs aslo enable cluster detection in areas of uniform density irrespective of the size of region

densities. Partitional method such as K-means [9] finds convex shaped clusters. As a result it fails to identify clusters of non-globular shapes. The algorithm also shows its limitations while dealing with noisy data. Agglomerative clustering techniques are not as affected by noise, but they have a tendency towards identifying globular clusters.

4) **Overlooking clusters in compact areas of arbitrary sizes:** For spatial data-bases, we can often find regions of uniform density located at a remote area. Such regions usually go undetected and the probable clusters existing in those regions are potentially overlooked by many clustering algorithms.

In order to address these challenges, the class of DBCLAs holds its own importance. The following points underline the reasons for robustness of the DBCLAs.

1) DBCLAs discover clusters as dense regions in the data space separated from areas of lower density [6] (Fig. 2). The density of any region is given as the number of points within that region or in terms of its kernel density estimate [10].
2) DBCLAs enable appropriate noise filtering [6] (Fig. 2).
3) DBCLAs aim at exploring the data space at varied levels of granularity [11] to detect clusters in regions of uniform density (Fig. 2).
4) Exploring data at higher levels of granularity enables the DBCLAs to reconstruct the entire shape of the data distribution [12] (Fig. 3).
5) DBCLAs facilitate detection of arbitrary shaped clusters with varying sizes and densities [4]. A post processing phase is considered in order to accumulate the dense region into an arbitrary shape (Fig. 3).

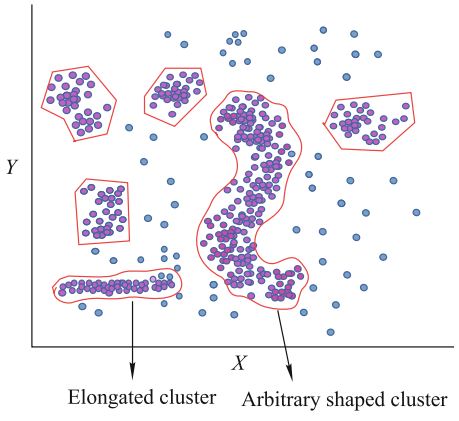

Fig. 3 Detecting clusters of arbitrary shapes at varied levels of granularity

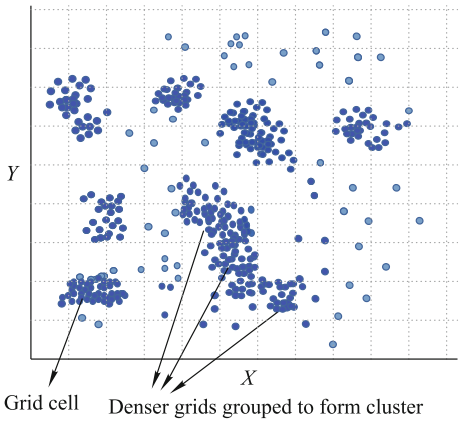

Fig. 4 The construction of grids in the data space. Grid density is determined by the number of points in a grid cell. The denser grids are grouped to form the clusters

6) DBCLAs also use grid based methods to explore individual regions of the data space and form clusters [11] (Fig. 4).
7) Grid like methods can be used in context of higher dimensions since the lower dimensional grids define clusters on subsets of dimensions (Fig. 4).

Contributions of this paper:

1) Provide a brief description of as many as thirty-two DBCLAs from the year 1996 onwards.
2) Provide a classification strategy to group the DBCLAs and based on the strategy adopted, we produce a taxonomy of all the DBCLAs described in this paper.
3) We introduce a set of features that are common to all the DBCLAs, and based on the feature properties exhibited by each algorithm, we relate and differentiate the DBCLAs.
4) The paper also extracts the Citation percentage and Conceptual dependency values of the DBCLAs. These values are used to compare the DBCLAs by showing the variation among different algorithms in their referrals as well as their importance among other DBCLAs.
5) We identify various application domains in which the DBCLAs have been applied and highlight the usage of relevant methods in those applications.

The paper is organized as follows: In Section 2, we present the existing surveys carried out for studying the DBCLAs. This is followed by a description of the classification strategy adopted to group the DBCLAs described in this survey (Section 3). In the next Section 4, we essentially classify the DBCLAs based on the classification strategy adopted in Section 3 as well as provide the description of individual algorithm. We compare various DBCLAs in this paper by providing the relationships and differences among the algorithms in Section 5. Next we present an empirical study of some of the DBCLAs (Section 6). This is followed by highlighting the applications of DBCLAs in Section 7. We mention the future scopes and directions of the DBCLAs in Section 8 followed by the discussion and conclusion in Section 9.

2 Related work

Existing surveys on DBCLAs selectively deal with some pioneer density based algorithms. In effect, the surveys end up covering a limited set of proposed methods. The classification of algorithms done in those surveys are specific and depends upon certain properties of the algorithms of choice. A number of studies were conducted to cover the DBCLAs, however to the best of our observation all such studies fail to extensively provide a taxonomy of the algorithms developed till now. In due course, numerous density based methods have been proposed. The evolution of these methods over the years implies that there exists a variation in the approach based on which the clusters are detected, their performance level and the challenges that these algorithms address upon.

In 2011, a survey on DBCLAs [13] highlighted different variants of the DBSCAN (density-based algorithm for discovering clusters in large spatial databases with noise) [6] algorithm. DBSCAN was the first density based clustering algorithm proposed in 1996. The basis of the study conducted was to identify necessary parameters that were needed to group the DBCLAs. The survey showed its limitations in failing to identify a host of other DBCLAs and is therefore limited only to a range of density based techniques. In addition to the methods that were covered in the 2011 survey, algorithms viz. OPTICS (Ordering Points To Identify the Clustering Structure) [14], DVBSCAN (Density Based Algorithm for Density Varied Clusters in Large Spatial Databases) [15] and ST-DBSCAN (Spatial-Temporal DBSCAN) [16] were also studied. A similar study from the year 2013 [17] investigates the properties of as many as seven DBCLAs along with highlighting their advantages and disadvantages. In 2014, another study [18] reviewed only three DBCLAs [6, 14, 19] thereby providing a limited information about different techniques proposed in this paradigm.

A comparative analysis of density based methods [20] involving DENCLUE (DENsity based CLUstEring) [19], DBCLASD (Distribution Based Clustering of LArge Spatial Databases) [21] and DBSCAN [6] was illustrated in 2011. Two similar studies [22,23] conducted in 2014 and 2016 showed that only three major DBCLAs [6,14,19] were compared. The comparisons in each of the studies were reduced to a small set of algorithms which fails to provide an insight to other DBCLAs cutting across various application domains.

The inception of DBCLAs was mainly motivated due to the volume and variety of clusters that were usually overlooked by other clustering domains. A review conducted in 2013 focused on the density based techniques for large databases. The study mainly dealt with the Soft-DBSCAN (DBSCAN clustering using fuzzy set theory) [24] method, a combination of DBSCAN and fuzzy set theory. OPTICS [20] and a polygon method to detect clusters were other major algorithms presented in the review. A literature on DBCLAs from 2014 by Loh et al. [25] presented a survey of some recent density based techniques viz. PDBSCAN (Parallel DBSCAN) [26], CUDA-DClust (CUDA-Density-based Clustering) [27], GSCAN (Density-Based Clustering Using Graphics Processing Units) [28] for improving the effectiveness of DBSCAN. Both the studies highlight a list of

methods related to DBCLAs but do not provide any classification of the algorithms.

Table 1 highlights the comparison between prior surveys of DBCLAs conducted and our survey. It is evident that the existing surveys are limited in their coverage of a variety of DBCLAs. Our survey conducts a comprehensive study of a wide range of DBCLAs along with providing additional features such as common properties, classification of DBCLAs, conceptual dependency, applications and future directions of the DBCLAs.

In the next section we present the classification strategy adopted to demarcate various DBCLAs in this paper.

3 Strategy for classification of DBCLAs

We identified four categories and subdivided each category with certain number of classes. Based on these classes within each category, we classify the DBCLAs. The adopted categories are:

- Density definition
- Parameter sensitivity
- Execution mode
- Nature of data.

Every studied DBCLA is placed under all the four categories. Distinction between DBCLAs happen only due to the classes which are created within that category itself (Fig. 5). The classification of DBCLAs within a category remains isolated from that of the other categories.

An algorithm may be unique to a class or part of multiple classes within a given category due to its involvement in handling a variety of data sets or adopting a mixture of techniques to address certain challenges. In Section 4, we present a detailed classification chart of the DBCLAs studied in this paper. Next, we describe each of these categories and various classes placed within a category w.r.t. Fig. 5.

1) Density definition DBCLAs classified under the definition of density are segregated based on the way in which the density is computed. We classify the DBCLAs into four classes based on the following definitions of density: *point based, grid based, probabilistic, data dependent.*

   (a) Point based: In this approach, the point density is calculated based on the number of points that lie within its neighborhood.

   (b) Grid based: The grid based technique relies on finding the denser grids and accumulates them to form clusters. The density of a grid cell depends on the quantity of data objects within it. Grid based techniques are helpful in alleviating issues related to high-dimensional datasets.

   (c) Probabilistic: Probabilistic models enable a clustering method to compute the densities based on some density functions. This approach is used to handle voluminous datasets in higher dimensions. DBCLAs dealing with multimedia datasets use probabilistic models to compute the density.

Table 1 Comparison of existing surveys of DBCLAs with our survey

Survey paper (year)	No. of DBCLAs studied	Common properties	Classification of DBCLAs	Conceptual dependency	Applications	Future directions
[13] (2011)	5	×	×	×	√	√
[20] (2011)	3	×	×	×	×	×
[17] (2013)	7	×	×	×	×	×
[18] (2014)	3	×	×	×	×	×
[22] (2014)	3	×	×	×	×	×
[25] (2014)	6	×	×	×	√	√
[23] (2016)	2	×	×	×	×	×
Our survey of DBCLAs	32	√	√	√	√	√

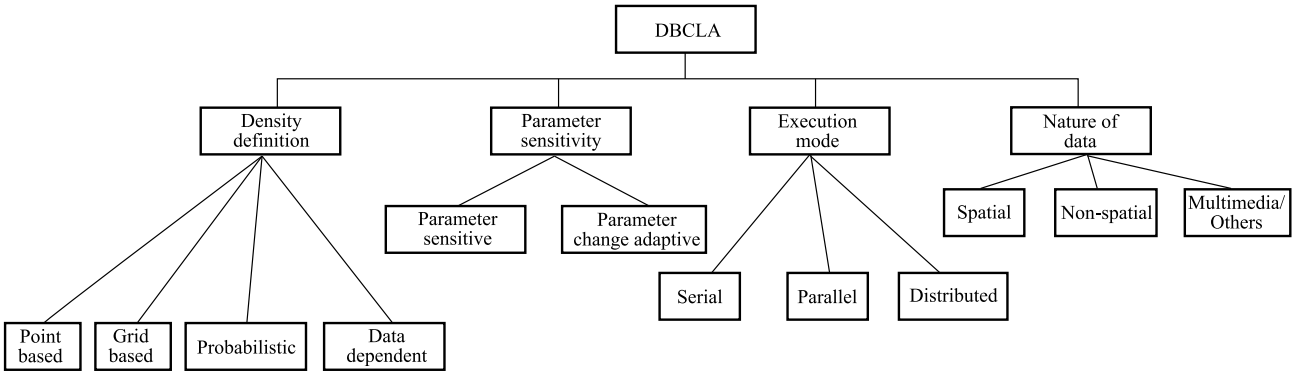

Fig. 5 Classification tree of DBCLAs

(d) Data dependent: Some of the algorithms [4, 29] use data dependent dissimilarity measure to compute the proximity between instances of data objects. The density is therefore computed by identifying the quantity of points in the neighborhood of a data object obtained by using data dependent dissimilarity approach. E.g., In MBSCAN [29], similarity is computed based on the mass of smallest region engulfing two data points. The point density is then found out by detecting the number of points which have a mass less than a certain threshold with the concerned point.

2) Parameter sensitivity: Many of the DBCLAs depend on the use of parameters which potentially influences the clustering output. There are some algorithms which remain adaptive to the changes in parameter values. These algorithms depend on the data distribution to identify the parameter values. Based on parameter dependency, we divide the DBCLAs into two classes: *parameter sensitive and parameter change adaptive*.
   (a) Parameter sensitive: The algorithms whose outcome depend on the chosen values of the parameters.
   (b) Parameter change adaptive: DBCLAs which adapt to the changes in parameter values. The change in parameter values do not influence the final output.
3) Execution mode: Most of the DBCLAs execute in serial mode, however to improve the efficiency, parallel and distributed algorithms have also been proposed by researchers. We classify the density based clustering techniques into three classes: *serial, parallel and distributed* based on their mode of execution.
   (a) Serial: DBCLAs which execute its individual modules one after another in a serial manner lie in this category.
   (b) Parallel: In parallel DBCLAs, data is spread across multiple machines and the execution happens simultaneously in order to speed up the computation.
   (c) Distributed: In order to reduce the cost of execution, distributed DBCLAs have been proposed. Most of the distributed clustering approaches generate global models by aggregating local results obtained on each node. The complexity and quality of solutions depend significantly on the level of aggregation.
4) Nature of data: Some of the DBCLAs are made to work on specific datasets. These datasets may be spatial, non-spatial, image or other multimedia data. Based on the dataset used by the algorithm, we classify the DBCLAs into three classes: *spatial, non-spatial, multimedia/others*.
   (a) Spatial: The spatial datasets directly or indirectly refer to a location on the surface of earth.
   (b) Non-spatial: When a dataset cannot be related to any geographical location, it is referred to as non-spatial data.
   (c) Multimedia/others: Multimedia data sets consist of images, videos, graphics. Some of the DBCLAs highlight their specialty on such data sets as it is often very challenging to deal with such data-sets compared to data with spatial attributes.

Next, we describe the individual DBCLAs based on the properties they exhibit wrt. aforementioned category heads and their classes.

4 Taxonomy of density based clustering algorithms

4.1 Density definition

In this subsection, as per the first category: **Density definition**, we classify the DBCLAs into four classes viz. *Point based, Grid based, Probabilistic and Data dependent*.

4.1.1 Point based density computation:

1) DBSCAN (Density-based algorithm for discovering clusters in large spatial databases with noise) (1996) (Representative algorithm)
   Pros: Extracts clusters of arbitrary shapes with filtration of noises.
   Cons: Fails to detect clusters of variable densities. Unreliable in high-dimensional data space.
   Key issues addressed: Dense regions are detected as clusters. Points in sparser regions are filtered as outliers.
   Description: DBSCAN [6] was proposed in 1996 as the first DBCLA. DBSCAN detects clusters in spatial databases along with filtering noise. DBSCAN takes two parameters *Eps* and *Minpts*. For a given point p, *Eps* signifies the radius of its surrounding region known as the *Eps* neighborhood of p. The literature denotes *Eps* neighborhood of p as $N_{Eps}(p)$. Let D denote the dataset, then for any $p \in D$, its *Eps* neighborhood is given as $N_{Eps}(p) = \{q \in D \mid \text{dist}(p,q) \leqslant Eps\}$. If $|N_{Eps}(p)| \geqslant Minpts$, then p is a dense or core point otherwise p is border point. A border point has at least one core point in its *Eps* neighborhood. For a core point p wrt. *Eps* and *Minpts*, if there exists a point $q \in N_{Eps}(p)$, then q is directly density reachable from p. However q is density reachable from p wrt. *Eps* and *Minpts* only if there exists a chain of points q_1,q_2,q_3,...,q_n, q_1=p, q_n=q such that q_{i+1} is directly density reachable from q_i. Density-reachability is an extension of direct density-reachability. The relation follows transitivity but it is not symmetric.
   DBSCAN classifies each point as either core or border point. Two core points have the same cluster membership if they are directly density reachable. A point in cluster C is density reachable from any core point belonging to C. The cluster expansion takes place by merging the density reachable core points. The algorithm assigns the border points to a cluster of their nearest core point. Points not belonging to any cluster qualify as outliers or noise points.
   DBSCAN detects clusters of arbitrary shapes and sizes with appropriate noise filtering in spatial databases. However, the preset value of the parameter *Minpts* does not allow the algorithm to detect clusters of variable densities. The method also shows its limitations in high dimensional data sets. DBSCAN consumes $O(N \log N)$ time, with N as

the size of the dataset.

2) CLIQUE (CLustering In QUEst) (1998)
Pros: Extracts clusters in high dimensional data space. The clusters produced do not depend on the order of the data. The algorithm is scalable.
Cons: The quality of clusters cannot be guaranteed.
Key issues addressed: Scales linearly with the size of the input. Facilitates automatic high dimensional subspace clustering.
Description: CLIQUE [12] enables discovery of dense clusters embedded in high dimensional subspaces. The algorithm provides cluster description through DNF expressions. The algorithm strives to detect projections of input data automatically. These projections are mapped into a subset of attributes including regions of high density. The algorithm provides no prior estimation of any distribution of the data in its mathematical form. The results produced by the algorithm are order independent wrt. presentation of input records.

Table 2 Common notations used in this article

Notation	Description
N	dataset size
n	#labeled points in the dataset
C	#clusters
s	number of parallel OPTICS run [14]
k	number of dimensions of a subspace [12]
K	number of cells at the bottom layer
D	dataset
c	any constant
S_i	dataset size in i^{th} machine [26]
p	total no. of core points
q	total no. of non-core points and outliers

The estimated density of data points is calculated by partitioning the data space and finding the numerical strength of points within each cell of the partitioning . Each dimension is partitioned into identical equi-spaced intervals. The volume of each unit is equal and the quantity of points in each volume approximates the density.
Clusters are identified in the projections from the appropriate subspaces. The valleys of the density function separate the data points. Clusters are formed by taking the unions of the high density components. A description of the clusters is obtained by covering a cluster with overlapping rectangles and describing it as a union of these rectangles. CLIQUE runs to completion in $O(c^k + Nk)$ time with c being a constant, k is the maximum number of dimensions and N is the size of the dataset.

3) GDBSCAN (Generalized DBSCAN) (1998)
Pros: Ability to cluster both point objects as well as spatially arranged objects.
Cons: No clear heuristics specified to determine the parameters of the algorithm.
Key issues addressed: Clustering of spatially extended objects having both spatial and non-spatial attributes.
Description: GDBSCAN [30] proposed by Ester et al. is the generalized version of the DBSCAN [6] algorithm. GDBSCAN has the ability to group points along with spatial objects based on their spatial and non-spatial features. The method is implemented on important application domains such as astronomy, molecular biology, earth sciences, geography. The algorithm mainly prescribes two notions: *a*) the neighborhood definition is based on symmetric and reflexive binary predicates, *b*) use of simple non-spatial measures to compute the density e.g., mean height of students to define the strength of a class.
GDBSCAN introduces three concepts: *NPred- neighborhood*, *MinCard* and *wCard* function. *NPred* is a reflexive and symmetric binary predicate on D. The *NPred*-neighborhood of an object $o \in D$ is defined as $N_{NPred}(o) = o' \in D \mid NPred(o, o')$ [30]. *wCard* represents a function which measures the cardinality for sets of objects. It is a weighted function and can be defined as $wCard : 2^D \rightarrow \mathbb{R}^{\geqslant 0}$ [30]. If $MinCard \in \mathbf{R}^+$, then for a set of objects S, the predicate *MinWeight* is true only if $wCard(S) \geqslant MinCard$. For a given object p, GDBSCAN extracts all the other objects that are density reachable [30] from p wrt. *NPred* and *MinWeight*. A cluster membership is assigned to p if it qualifies as a core object. If no other object is density reachable from p and p is a non-core object, then it is classified as an outlier or noise. For unclassified objects this procedure is iteratively performed to find the clusters. The run time of GDBSCAN is $O(N^2)$ with no index support. On using a spatial index, the running time reduces to $O(N \log N)$. For a grid based object organization, the time complexity becomes linearly proportional to the quantity of data objects.

4) Inc-DBSCAN (Incremental-DBSCAN) (1998)
Pros: Clusters data objects dynamically after periodic updates.
Cons:Unable to handle bulk insertion or deletion of data objects. The algorithm is sensitive to change in parameter values.
Key issues addressed: Introduced first incremental clustering method based on DBSCAN. Addresses the bottleneck of redundant computation involved in non-incremental methods.
Description: Inc-DBSCAN (Incremental DBSCAN) [31] is the incremental version of DBSCAN. The algorithm deals with dynamic datasets. Patterns in database e.g.,log database alters temporally with new logs being added to and previous records are deleted from the database. The algorithm identifies affected parts of the existing clusters by an update in the database. Based on this underlying idea of selective handling of the updated dataset, the algorithm proves to be more efficient. After insertion of new points, some non-core objects may turn into core forming novel density connections. Points which were not density reachable [6] earlier might become density reachable. Similarly upon performing deletion some core objects turn into non-core resulting in removal of existing connections. If an object p is inserted or deleted, then $N_{Eps}(p)$ [6] becomes the affected region. In addition, all the other objects in $D \cup \{p\}$ also become a part of the affected region. The unaffected points retain the same cluster membership. The number of region queries performed

by Inc-DBSCAN is determined experimentally. Let r_i and r_d denote the mean number of region queries during incremental insertion and deletion respectively. Let f_i and f_d be the percentage of insertions and deletions. Then the cost incurred by Inc-DBSCAN for making r updates to the dataset incrementally is: $r \times (f_i \times r_i + f_d \times r_d)$.

5) PDBSCAN (Parallel DBSCAN) (1998)
Pros: Uses partitioning scheme to execute DBSCAN concurrently in multiple machines.
Cons: Load balancing across machines may become a bottleneck.
Key issues addressed: Faster detection of clusters of arbitrary shapes in dense regions.
Description: PDBSCAN [26] is a parallel version of DBSCAN. The algorithm uses a distributed data structure as a part of the shared nothing architecture. Many machines are connected with replication of data indexes. PDBSCAN introduced a distributed index structure known as dR*-tree. According to this structure, data is spanned across many workstations with replication of data indexes in each one of them. As per this method, the main advantage of the "shared nothing" architecture is that it can be scaled up to thousands of computers.
The algorithm executes parallel DBSCAN by using partitioning strategy. PDBSCAN involves three major steps. Firstly it creates multiple partitions by dividing the input. The created partitions are distributed across all the available computers. In the second step, the partitions which were created in the first step are subjected to clustering. For this purpose the DBSCAN algorithm is run concurrently. The third step involves merging of clusters obtained in individual computers and create a clustering of the whole database. The least efficient machine determines the run time of PDBSCAN. The algorithm has to ensure a proper load balancing among different machines in function. Let S_i be the size of the individual partitions, then by using the dR*-tree index structure, the running time is reduced to $O(S_i \log S_i)$.

6) OPTICS (Ordering points to identify the clustering structure) (1999)
Pros: Produces an augmented ordering of the database instead of explicit clustering structure. Efficiently extracts intrinsic clustering structure. The algorithm is parameter change adaptive.
Cons: For high dimensional data spaces, no index structures go on to exist for efficient support of hyper-space range queries.
Key issues addressed: The issue of parameter sensitivity has been resolved. Interactive cluster analysis in an $\mathcal{R}^d$ space is possible.
Description: OPTICS [14]was introduced to create an augmented ordering of the dataset. This ordering represents density-based clustering organization of the dataset. The algorithm is not meant for clustering the data explicitly. The information contained in the clustering structure equivalently represents the density based clustering matching a large range of parameters. To produce a consistent result, the algorithm adheres to certain ordering where the data items are handled during cluster expansion.
The algorithm selects an object and writes it to an ordered file. The directly reachable points from the core point maintain an order in a seed list. The points are ordered in increasing order of their reachability from the closest core object from where they are directly density reachable. Closest core point is selected, and its reachability distance is calculated. The point is then written to an ordered file. If the current object is a core object, then further candidates for expansion may be inserted into the seed list. The running time is $O(N^2)$ however, if tree based method is used then its complexity becomes $O(N \log N)$, where N is the size of the dataset.

7) SDBSCAN (Sampling based DBSCAN) (2000)
Pros: Combines sampling technique with DBSCAN for clustering large spatial databases .
Cons: Cannot detect clusters of variable densities.
Key issues addressed: I/O cost and memory usage are reduced drastically.
Description: SDBSCAN [32] is a combination of sampling technique with DBSCAN algorithm to cluster voluminous spatial databases. SDBSCAN introduces two sampling based techniques: SDBSCAN-1 or sampling inside DBSCAN and SDBSCAN-2 or sampling outside DBSCAN. SDBSCAN -1 cuts down the time consumed on region query operation and memory requirement for storing a core object. In order to reduce the memory and I/O costs to speed up the DBSCAN, algorithm samples some representatives rather than taking all the objects in the neighborhood of a core object as new seeds.
Apart from sampling, an efficient labeling mechanism is adopted in order to label the un-sampled data based on R* tree. SDBSCAN-2 samples the database *DB* to produce a sampled dataset *sdb*. It then creates R* trees *DB* and *sdb*. The sampled data *sdb* is clustered through DBSCAN. The algorithm runs to completion in $O(N \log N)$ time.

8) SNN-DBSCAN (Shared nearest neighbor-DBSCAN) (2003)
Pros: Employs a data dependent dissimilarity technique to evaluate the proximity measure between data points for clustering. Removes the use of distance based measure to find the similarity scores.
Cons: Unable to function for non-spatial data e.g., transaction data, genomic data.
Key issues addressed: Finding clusters of uniform densities across sub-spaces. Challenges involved in high-dimensional clustering.
Description: SNN-DBSCAN [4] is a robust graph based clustering algorithm which detects arbitrary shaped clusters having variable densities. The algorithm uses the concept of shared nearest neighbor [33] to decide the proximity between two data points. If two nodes have sufficient number of points in their shared neighborhood, a link is constructed between them. The graph obtained in this manner is known as the shared nearest neighbor(SNN) graph. A data object is categorized as core if it has sufficient number of adjacent links to it. Two core

points sharing a common link are placed in the same cluster. The non-core points are assigned to a cluster of their nearest core point. Points which fail to obtain any cluster membership qualify as noise points. The definition of core points allow SNN-DBSCAN to identify arbitrarily shaped clusters with varying sizes. The use of shared nearest neighbor instead of any distance measure enables the algorithm to address the issues related to high dimensionality. The algorithm runs to completion in $O(N^2)$ time. The construction of SNN graph in the form of a similarity matrix results in quadratic time complexity.

9) DBDC (Density based distributed clustering) (2004)
Pros: Clusters data from different sources locally and extracts suitable representatives from these clusters.
Cons: Self induced quality criterion used to judge the clusters.
Key issues addressed: Issue of transmission cost, inefficient global clustering.
Description: DBDC [34] clusters the data locally and extracts suitable cluster representatives. Globally located servers store the cluster representatives in order to restore the clustering information on the basis of local representatives. The efficiency of the method lies in the fact that the local clustering is carried out very quickly and are independent of each other. On the basis of limited quantity of representatives, clustering at the global level can also be performed effectively. DBDC is a combination of both local and global clustering.

10) IDBSCAN (Improved density based spatial clustering with application of noise) (2004)
Pros: Uses *Marked Boundary Objects* [35] to expand computing directly without actual dataset selection.
Cons: Unable to find clusters of variable densities.
Key issues addressed: Addresses the issue of greater I/O cost and memory requirements involved in clustering.
Description: IDBSCAN [35] applies the idea of *Marked Boundary Object* or MBO in order to identify points from an expansion seed. It searches for a neighboring region for adding to the expansion seed. If an object P is core satisfying the condition of set density, the algorithm aims to find in the neighborhood the nearest point to the MBOs and sets the points as the seeds for expansion. The algorithm selects seeds, using numerous MBOs as a result of which it demands only a single input instance. (3^d-1) *MBO*s are added with d representing the number of dimensions of the dataset. Number of quadrants is 2^d and the number of seeds selected is at most (3^d-1). The algorithm outperforms DBSCAN and runs to completion in $O(N \log N)$ time.

11) ST-DBSCAN (Spatial-temporal DBSCAN) (2006)
Pros: Discovers clusters in accordance with spatial, non-spatial and temporal values of the objects.
Cons: Doesn't support parallel execution.
Key issues addressed: Resolves the issue of finding clusters among spatio-temporal data.
Description: ST-DBSCAN [16]proposes three extensions to the DBSCAN algorithm related to detection of core objects and outliers along with clusters that are adjacent to each other. The algorithm mainly fosters the discovery of clusters among non-spatial, spatial and temporal data objects. One of the major flaws of DBSCAN is its inability to detect variable density clusters. ST-DBSCAN addresses the issue by allocating each cluster a density factor. ST-DBSCAN has the ability to cluster spatio-temporal data based on its temporal, spatial and non-spatial attributes. The value of core points within a cluster can be significantly dissimilar from the values of border or non-core points on the other side. If the non-spatial values of neighboring objects have small differences, the clusters remain adjacent to one other. ST-DBSCAN addresses this issue by comparing the average value of a cluster with an incoming value. Spatio-temporal values are indexed according to the space and time dimensions. The algorithm runs to completion in $O(N \log N)$ time.

12) KIDBSCAN (K-means and IDBSCAN)(2006)
Pros: Outperforms DBSCAN by combining K-Means with IDBSCAN. Not limited by memory while dealing with large datasets.
Cons: Doesn't support parallel execution.
Key issues addressed: Cluster accuracy along with I/O cost.
Description: KIDBSCAN [36] combines the features of K-means clustering [9] along with IDBSCAN [35] to find the high-density center points. IDBSCAN is used to expand clusters from these high-density center points. The algorithm consists of three phases: firstly, the variables required by K-means and IDBSCAN are initialized and are included in the dataset. Secondly, K-means algorithm is applied to the input dataset to yield K-high density center points. The points that are closest to the center points are found. In the third phase, IDBSCAN is executed on an adjusted dataset determined by K-means method, and the clustering result is generated. The algorithm outperforms DBSCAN.

13) CUDA-DClust (CUDA-density-based clustering) (2009)
Pros: Exploits the advantages of GPUs to achieve a higher speedup as compared to DBSCAN.
Cons: Since data structures are stored in shared memory, this affects the number of threads to be executed in parallel.
Key issues addressed: Cost due to extreme parallelization.
Description: CUDA-DClust [27] uses the graphics processing unit (GPU) due to its ability to provide high parallelism combined with a greater bandwidth. The transfer of memory is done at a reduced cost. The result obtained is assured to be similar to DBSCAN but with a high speedup. GPUs facilitate massive parallelism and is used for various computationally intensive tasks. CUDA-DClust introduces a concept called *chain* to allow high degree of parallelism in DBCLAs. Chain represents the data objects belonging to a common density based cluster. Each chain is assigned to a single cluster, however a cluster may consist of several chains. CUDA-DClust considers a chain a tentative cluster. Instead of performing

a single cluster expansion like DBSCAN, different cluster expansions are started at different points through different chains. The algorithm also handles collision when two chains become a part of a single cluster. The approach of chains also uses parallelism while determining the core points and also for making its neighbors as new seed points.

14) DVBSCAN (Density based algorithm for density varied clusters in large spatial databases) (2010)
Pros: Handles local density variation within clusters.
Cons: Parameters cannot be determined automatically.
Key issues addressed: Finding clusters in uniform density regions.
Description: DVBSCAN [15]enables handling of local density variation within a cluster. This property does not exist in the DBSCAN algorithm. DVBSCAN computes the increasing average density of the cluster along with cluster density variance for the core point aiming for further expansion. The algorithm considers the density of its ϵ neighborhood. If cluster density variance for a core point lies within the boundary of a threshold value satisfying the cluster similarity index, the algorithm allows the core point for expansion. The running time of DVBSCAN is $O(N \log N)$.

15) P-DBSCAN (Photo-DBSCAN) (2010)
Pros: Analyzes places and events using geo-tagged photos.
Cons: Gathering of data becomes a hectic process.
Key issues addressed: Optimization of dense area search, convergence speed of the algorithm.
Description: P-DBSCAN [37] is used to analyze locations and events utilizing an assembly of geo-tagged photos. The algorithm is based on DBSCAN [6]. P-DBSCAN introduces two novel concepts: *density threshold* and *adaptive density*. The *density threshold* is defined according to the number of people within neighboring region while *adaptive density* nurtures fast convergence towards high density regions. The algorithm aims to discover interesting places or significant events that are characterized by high photo activity in a specific area. The algorithm starts with a photo that is neither core nor a noise object. A core photo is ascribed to a cluster while the neighboring photos are queued up for further processing. The processing of photos continues till the queue is empty. For a core photo the algorithm invokes its adaptive density version. Adaptive density threshold is the ratio of present and previous density of a photo object p. Points lying in the neighborhood of p are placed into the current cluster as long as the density ratio exceeds 1 or it is more than the threshold density. P-DBSCAN runs to completion in $O(N \log N)$ time, with N representing the quantity of photos.

16) DADBC (Distributed approach for density-based clustering) (2011)
Pros: Adopts a distributed structure while reducing the communication overhead involved in aggregation of global models.
Cons: Complexity and quality of clusters depend on the nature of aggregation.
Key issues addressed: Higher cost involved in merging results from local nodes involving heterogeneous data.
Description: DADBC [38] reduces the communication overhead and improves the quality of the global models by considering the shapes of local clusters. Global models are obtained by aggregating the localized nodes. The model proposed by the algorithm is a three step process. Firstly, it clusters datasets located on each local node and selects good local representatives. Secondly, all the local models are sent to the server. Local models are not directly merged to build the global model. The local models are extracted from the local data such that their sizes are small enough to send through the network. The algorithm regenerates the data objects on the server basing on the local model representatives. The objective of this step is to improve the quality of the global model.

17) MR-DBSCAN (MapReduce-DBSCAN) (2011)
Pros: Adopts a distributed structure while reducing the communication overhead involved in aggregation of global models.
Cons: Complexity and quality of clusters depend on the nature of aggregation.
Key issues addressed: Higher cost involved in merging results from local nodes involving heterogeneous data.
Description: MR-DBSCAN [39] is a parallel DBCLA that implements Map-Reduce in four stages. A quick partitioning strategy is adopted for large scale non-indexed data. First stage involves data preprocessing through load balancing while removing skewness from the data. In the second stage, a local DBSCAN algorithm is run. All the related data is prepared for every single reducer, where the local DBSCAN function executes. The local DBSCAN is a modified version of the PDBSCAN [26]. The third stage involves merging of clusters from subspaces. In the fourth stage, a global view of cluster mapping is built. This stage also involves streaming of all the local clustered records over the map-reduce process and replacing their local cluster identity with a new global cluster identity.

18) HDBSCAN (Hierarchical clustering method using DBSCAN)(2013)
Pros: Finds clusters of variable densities.
Cons: Runs is quadratic time complexity.
Key issues addressed: Sensitivity to change in parameter values.
Description: HDBSCAN [40] is a density-based, hierarchical clustering method producing a clustering hierarchy. It is an improvement over the OPTICS [14]algorithm. From this clustering hierarchy, a simplified tree of significant clusters is reconstructed. The algorithm proposes a cluster stability measure and formulates a method with the objective of finding an optimal solution to the problem of identifying stable clusters. HDBSCAN is not sensitive to the changes in parameter values. HDBSCAN has the potential to detect clusters of varying densities. In addition, the algorithm automatically simplifies the hierarchy into a representation involving the most significant clus-

ters. HDBSCAN produces a clustering tree that contains all partitions obtainable by DBSCAN [6] in a hierarchical nested manner.

HDBSCAN uses Prim's algorithm based on an ordinary list search to construct the minimum spanning tree(MST). The method is implemented in $O(dN^2)$ time, with the data consisting of d features. At a given point of time, the algorithm takes into account the current hierarchical level, as a result of which the HDBSCAN algorithm keeps in memory the extended MST.

19) Cludoop (Distributed density-based clustering on Hadoop) (2015)
Pros: Efficiently clusters large scale data in parallel using Hadoop.
Cons: Workload balancing problem of clustering on Hadoop platform.
Key issues addressed: Density based clustering for big data applications.
Description: Cludoop [41] uses Hadoop for efficiently dealing with big data. It is a distributed clustering technique. Initially, a serial algorithm *CluC* is proposed to find fast clusters. *CluC* classifies points by utilizing the relationship of cells that are connected instead of any expensive neighbor query method. *CluC* is put into the parallel mapper using existing data partition on Map/Reduce platform. Cell based principles guide a three-step properties: Merge-Refine-Merge in order to combine clusters over the cover of pre-clustering results upon the reducer.

20) DSets-DBSCAN (Dominant Sets-DBSCAN) (2016)
Pros: Removes dependency of clustering on change of parameters.
Cons: Designed to work only for image segmentation .
Key issues addressed: Parameter dependency.
Description: DSets-DBSCAN [42] enables clustering of image pixels as a part of the image segmentation technique. In order to remove the dependence of clustering results on user specified parameters, the algorithm provides a parameter-free approach based on the dominant sets (DSets [43]) and DBSCAN [6]. At first, a histogram equalization is applied to the pairwise similarity matrix of input data to make the clustering results independent of user-specific parameters. By merging the merits of DSets [43] and DBSCAN, the algorithm generates arbitrarily shaped clusters with no input parameter. DSets-DBSCAN extracts clusters in a sequential process. The clusters obtained from the dominant sets are extended by means of DBSCAN.

The algorithm achieves parameter free technique in two steps. First, histogram equalization is applied to the similarity matrices corresponding to different parameters (σs) before they are used in clustering. In the second step, DSets is run to obtain the dominant sets clusters followed by cluster extension process based on DBSCAN and extract the clusters sequentially. DSets clustering uses the pairwise similarity matrix of the data to be clustered as input and does not require the data to be represented in vector spaces. By means of histogram equalization transformation, DSets-DBSCAN makes different σs generate identical similarity matrices which result in identical clustering results. With the involvement of DBSCAN, the algorithm runs into completion in $O(N \log N)$ time.

21) Dynamic density based clustering (2016)
Pros: Provides computational hardness in various updates schemes under dynamic clustering.
Cons: Convergence of multiple concepts .
Key issues addressed: Inefficient clustering in case of dynamic updates.
Description: This work [44] investigates the algorithmic principles for dynamic clustering by DBSCAN and proposes *ρ-approximate* [44] to bring down the computational hardness on static data. The work also proves that the *ρ-approximate* version suffers from the same hardness when the dataset is dynamic in nature. However, it also shows that this *hardness* disappears when a tiny further relaxation is made. The quality of result obtained is same as that while handling the static data. This phenomenon is known as the "sandwich guarantee" of *ρ-approximate* DBSCAN. The algorithms guarantee near-constant update processing. The approximate version takes $O(N)$ time while the unit spherical emptiness checking(*USEC*) method consumes $o(N^{4/3})$ time in worst case.

22) MBSCAN (Mass-based clustering of spatial data with application of noise) (2016)
Pros: Introduces generic data dependent dissimilarity measure to find proximity between data objects. Removes the disadvantages associated with distance based clustering.
Cons: Parameter sensitive. Random parameter selection can impact the final results.
Key issues addressed: Hard density problem, issues related to distance based clustering.
Description: MBSCAN [29] is the first DBCLA to introduce a general interpretation of dissimilarity depending on data distribution. The algorithm proves that the data dependent dissimilarity is better than any geometrical model used for clustering. MBSCAN replaces the distance function with data dependent dissimilarity measure removing the shortcomings of methods based on distance functions. In MBSCAN, the dissimilarity measure depends on the probability mass of the smallest region covering two data instances. Let D be the dataset from probability density function F and $H \in \mathbb{H}(D)$ is a hierarchical partitioning model of the space into non-overlapping regions. For two data instances a,b, the smallest local region covering a and b wrt H and D is given as:

$$R(a, b|H; D) = \underset{r \subset H | \{a,b\} \in r}{\operatorname{argmin}} \sum_{z \in D}^{n} 1(z \in r), \tag{1}$$

where 1(.) is an indicator function. The mass based dissimilarity estimated from a finite number of models $H_i \in \mathbb{H}(D)$, $i = 1,2,3,4,\ldots,t$ is:

$$m_e(a, b|D) = \frac{1}{t} \sum_{i=1}^{t} P(R(a, b|H_i; D)), \tag{2}$$

where $P(R) = \frac{1}{|D|} \sum_{z \in D} 1(z \in R)$. Instead of any con-

ventional distance metric to measure the proximity, the similarity between two points $m_e(a,b)$ is measured by using Eq. (2). Unlike DBSCAN, instead of using ϵ-neighborhood, μ-neighborhood is used. The definition of μ-neighborhood is given as:

$$M_\mu(a) = \#\{b \in D | m_e(a,b) \leqslant \mu\}. \quad (3)$$

From Eq. (3), it is evident that only those points b are included as seed points for an object a if the mass based dissimilarity between a and b is less than a certain threshold μ. Unlike DBSCAN which involves a radius to decide the neighborhood while finding point density, MBSCAN relies on mass based measure. The algorithm runs to completion in $O(N \log N)$ time.

4.1.2 Grid based density computation:

1) STING (STatistical INformation Grid) (1997) (Representative algorithm)
Pros: Statistical grid based approach to reduce the cost of clustering spatial data objects.
Cons: Grid size may impact the final outcome.
Key issues addressed: Parameter dependency.
Description: STING [45] is a statistical technique for clustering. It adopts a grid oriented approach to limit the computational cost involved in clustering. STING retrieves statistical information from spatially arranged cells to answer the queries involved in clustering. This procedure is independent of each data item. The algorithm uses top-down approach while answering queries from spatial data. In this method, a higher level cell is fragmented into smaller lower level cells. Statistical information [45] such as mean, standard deviation, maximum, minimum, type of distribution in each grid cell are stored after computation. The parameters of cells at the lower level enables the algorithm to calculate the statistical information pertaining to each cell.
The algorithm generates the hierarchy of cells with their associated parameters when the data is loaded. The statistical parameters are computed directly from the data. A *confidence interval* is measured for individual cells in current level. A cell is labeled as relevant or irrelevant depending on the value of *confidence interval*. After processing the current layer of cells, the next layer of cells is processed. Instead of going through all the cells, the algorithm looks at only those cells which have been children of the relevant cells. STING finds all the regions formed by relevant cells and returns them. The algorithm scans the dataset only once to determine the parameters associated with the grid cells at the lower level. The time taken is linearly proportional to the number of cells K(say) at the bottom layer and is given as $O(K)$.
2) DENCLUE (DENsity based CLUstEring) (1998)
Pros: Models points density analytically as the sum of influence function. Extracts the peak of curve as density attractors.
Cons: Lacks extensive comparison with other clustering methods.
Key issues addressed: Clustering large multimedia datasets.
Description: DENCLUE [19]fosters cluster discovery in multimedia databases filled with noise. As per the algorithm, the total density at a point is modeled analytically as the summation of influence functions. Clusters of arbitrary shapes belonging to any $\mathcal{R}^d$ space are described through mathematical equations. These equations represent the generic density function in context of the algorithm. DENCLUE puts data points into following classes: *density attractor* and *density attracted*. For a given point p, if the sum of influence function of any other point q upon p crosses a certain threshold, then p is designated as a *density attractor*. From the point p, if q becomes density reachable, then point q is classified as a *density attracted* point. For a continuous and differentiable function, e.g., Gaussian influence function, a hill climbing algorithm [46] can detect the density attractor. Peaks of the influence function which consists of local maxima are density attractors. The points in the neighborhood of the peaks are usually the density attracted points.
DENCLUE is a two step algorithm. The first step constructs a map of the necessary region of the data space. A minimal bounding hyper-rectangle is fragmented to a multi-dimensional hyper cube. Only hyper-cubes which contain the data points are determined. The numbering of hypercubes is done based on a relative origin. The map speeds up the computation of density function which requires an efficient access to the nearby regions. The second step is the real clustering step. In this step DENCLUE identifies the *density attractors* and *density attracted* points. The densely filled cubes are connected to other cubes to determine the clusters. DENCLUE runs in $O(N \log N)$ time.
3) OPTIGRID (Optimal Grid-Clustering)(1999)
Pros: Creates optimal grid partitioning to perform efficient high-dimensional clustering.
Cons: Nature of grid partitioning may have an impact on the final clustering results.
Key issues addressed: Addresses the issue related to "curse of dimensionality" for clustering in higher dimensions.
Description: OPTIGRID [47] uses grids to perform clustering. OPTIGRID was introduced by Hinneburg et al. in order to address various issues related to high dimensional data. The algorithm strives to remove the curse of dimensionality in high dimensional space. OPTIGRID shows that different condensation based approaches like BIRCH [48], STING [45] has effectiveness issues in high dimensional space. OPTIGRID constructs the grid based partitioning of data in an optimal way. This is done by calculating the hyperplanes that are best partitioning for each dimensions using certain projections of the data. The algorithm uses kernel density function to make the density estimate. To efficiently determine cutting planes, the algorithm uses the concept of contracting projections. A contracting projection is a linear transformation defined on all points. For a given point x, its density in contracting projection acts as an upper bound of its planar density.

Initially, the algorithm determines the set of contracting projections after which it calculates all projections of the dataset D. A set of best cutting planes is determined as *BEST_CUT*. If *BEST_CUT* is empty, then the dataset D is the cluster obtained. Otherwise, the cutting planes which obtain the maximum score from the set *BEST_CUT* are selected. This is followed by the construction of a multi-dimensional grid G. All the data points $x \in D$ are inserted into G. The highly populated grid cells are chosen as the clusters and are added to a cluster set C. The procedure takes $O(N)$ time to completion.

4) WaveCluster (Wavelet-based clustering)(1999)
Pros: Finds clusters of arbitrary shapes, insensitive to noise, independent of the order of input data.
Cons: Not designed to work for non-spatial data.
Key issues addressed: Extraction of clusters at higher levels of granularity with appropriate noise filtering.
Description: WaveCluster [49] uses the multi-resolution property of wavelet transforms to determine clusters of arbitrary shapes in high dimensional space. Wavecluster is insensitive to noise and the order of the input data. The algorithm partitions the feature space into non-overlapping hyper rectangles called as *c*ells. Each cell is an intersection of one interval from each dimension. Each cell has a list of statistical parameters such as mean, variance, aggregation and the probability distribution of points associated with it. Each cell has information about the data density of the cell. The quantity of points within a cell is the single chosen statistic used in the algorithm. The algorithm then applies wavelet transformation over the feature space. Connected components are found in the sub bands of the transmuted feature space. The cells are assigned labels and the objects are mapped to form the clusters. WaveCluster chooses all the points within a cluster as representative of the cluster. The algorithm runs to completion in $O(N)$ time.

5) IPCLUS (Interactive Projected CLUStering algorithm)(2001)
Pros: Leverages the use of human intervention and computer to determine clusters in high dimensional space.
Cons: Human intuition may be less accurate at times.
Key issues addressed: Issue of extracting meaningful clusters in high dimensional data space.
Description: IPCLUS [50] uses human intuition to define the cluster and characterize its meaningfulness. IPCLUS utilizes the credibility of both human and computers. The algorithm proposes a system which performs high dimensional clustering through efficient coordination between man and machine. Initially, the algorithm determines the projection of data points. Such projections are referred to as well polarized projections. A subspace of the data is determined using the well polarized clusters. The algorithm uses kernel density estimates to determine the data density at each point of the data projection. The data is sampled on a iterative basis such that the most dominant subspace clusters are discovered.

6) D-Stream (Density-based clustering framework for data streams)(2007)
Pros: Finds arbitrarily shaped clusters from data streams using a grid based approach. No degradation of cluster quality happens.
Cons: Size of grids may impact the clustering result.
Key issues addressed: Dynamic changes in data stream are handled by the usage of density decaying technique [51].
Description: D-Stream [51] enables clustering of streaming data using a density based technique. The algorithm uses an on-line constituent to map individual data object to a grid and an off-line constituent to evaluate density of the grid. Clustering of grids is done based on their density. D-Stream introduces a density decaying technique in order to extract the dynamic changes in the data stream. The algorithm exploits the relationship between the decay factor, data density and the structure of the clusters efficiently generating and handling clusters in real time. D-Stream removes grids which are mapped to by the outliers for enhancing space and time efficacy. This technique enables clustering speedy data streams without destroying the cluster quality. D-Stream takes $O(C)$ time for completion with C representing the cluster count.

7) DBCLASD (Distribution Based Clustering of LArge Spatial Databases)(2010)
Pros: Adaptive to change in parameters unlike DBSCAN.
Cons: Not suitable for non-spatial data objects.
Key issues addressed: Addresses the issue of dynamic dataset through incremental updates.
Description: DBCLASD [21]assumes that the objects within a cluster are distributed uniformly. The algorithm dynamically determines the quantity and conformation of clusters in a database without involving any input parameter. The algorithm is efficient for large databases. DBCLASD incrementally augments an initial cluster with the points within its neighborhood. This procedure continues till the set of nearest neighbor distances of the resultant cluster fits the estimated distance distribution. A point which is not yet a part of the current cluster but needs to be examined for possible cluster membership is a candidate point. Candidates failing the cluster membership test in their first attempt are called unsuccessful candidates. Unsuccessful candidates are not overlooked. They are considered at a later time. Objects belonging to any cluster might shift to another cluster later. The running time of DBCLASD is approximately twice that of DBSCAN or $O(N \log N)$.

8) GSCAN (Density-based clustering using Graphics Processing Units)(2014)
Pros: Eliminates redundant distance computation for clustering using a grid structure.
Cons: Size of grids may impact the final clustering results.
Key issues addressed: Parallelization of density based clustering.
Description: GSCAN [28] aims to reduce unnecessary distance computation using a grid structure. GSCAN is an extension of CUDA-DClust [27] algorithm. Both DB-

SCAN and CUDA-DClust perform distance computations for each unclassified object p in dataset D to compute the ϵ neighborhood $N_\epsilon(p)$. Most of these objects reside beyond the ϵ neighborhood of p and thus most of the distance computations become an overhead. GSCAN reduces the unnecessary distance computations by forming a grid covering the entire data space D. The grid dimension d' is less than or equal to data dimension d. For each cell in the grid, a list of objects contained in the cell is generated. The algorithm performs distance computations from an unclassified object p to only the objects contained in the cells overlapping with the ϵ-range from p. A GPU kernel function is used to efficiently find the data objects contained in each grid cell. The entire dataset is evenly divided into *grid size* data subsets and each subset is assigned to a kernel block. The data subset assigned to a block consists of data objects residing adjacently in device memory to make the best utilization of the device memory cache.

4.1.3 Probabilistic based density computation

1) STING [45] (Representative algorithm): Once a layer is determined, for every cell elonging to this layer, the confidence interval of probability is computed. This is done in order to determine the relevance of a cell to a given query. Moreover the algorithm extracts statistical information like mean, standard deviation from spatial cells to respond against clustering queries. DENCLUE [19]: A probability measure is used to ascertain whether the number of density attractors in the whole dataset D and a dataset containing clusters D_{clus}(say) is identical or not. When the number of outliers in D tends to $\infty(infinity)$, this probability equates to one. OPTIGRID [47]: With a decreasing grid size, the neighboring grids to be connected reduces. At this juncture, the use of probability lies in identifying the likelihood of cutting planes hitting the cluster centers.
   IPCLUS [50]: The algorithm makes use of kernel density estimate(KDE) [52] to find the point density. KDE resembles a probability density function which estimates the likelihood of a point being drawn from a data sample. This probability can be interpreted as the density of a data point.
2) FOPTICS (Clustering of Fuzzy data objects using OPTICS) (2005)
   Pros: Expresses the similarity between two fuzzy objects with distance probability functions.
   Cons: The variable probability value to a distance may impact the final clustering result.
   Key issues addressed: Issue of performing hierarchical density based clustering on uncertain and fuzzy data.
   Description: FOPTICS [53] uses distance probability functions to express the closeness between two fuzzy objects. The algorithm analyzes uncertain data which are naturally occurring in diverse fields. FOPTICS' working principle is based on the OPTICS [14]algorithm. The result set adds to itself the first element from the seed list [14]and the range query is carried out. FOPTICS uses *monte-carlo* sampling to compute the reachability values. The algorithm integrates fuzzy distance functions into data mining algorithms. The run time of FOPTICS is $O(s^2N^2)$ with s denoting the frequency of parallel OPTICS run.
3) FDBSCAN (Clustering of Fuzzy data objects using DBSCAN)(2014)
   Pros: Expresses the similarity between two fuzzy objects with distance probability functions.
   Cons: The variable probability value to a distance may impact the final clustering result.
   Key issues addressed: Issue of performing density based clustering on uncertain and fuzzy data.
   Description: FDBSCAN [54] uses distance probability function to evaluate the proximity between two fuzzy objects. A distance value is assigned a probability value by the fuzzy distance functions. The distance functions provide information that are fully exploited by performing integration of these functions. The integration is directly used into the mining algorithm. In real applications there often exists no sharp boundary between clusters. Fuzzy clustering is better suited for such applications. FDBSCAN adopts a monte-carlo based sampling technique for sampling purposes and has an approach similar to DBSCAN for clustering the fuzzy objects. The running time of FDBSCAN is $O(N^2)$.

4.1.4 Data dependent density computation

SNN-DBSCAN [4] (Representative algorithm): The use of shared nearest neighbors in determining the proximity score between data points is a measure of data dependent computation. The algorithm does not rely on any geometrical model e.g., distance to determine the similarity measure between data points. MBSCAN [29]: MBSCAN was the pioneer algorithm in adopting a generic definition of data dependent dissimilarity measure. The algorithm relies on the strength of combined neighborhood between a pair points evaluated through construction of $iForest$ [29].

4.2 Parameter sensitivity

In this subsection, as per the second category: Parameter sensitivity, we classify the DBCLAs into two classes viz. *Parameter sensitive, Parameter change adaptive*. Since we have already described all the 32 DBCLAs studied in this survey (Refer Subsection 1). As a result, we only cite the relevant DBCLAs belonging to rest of the categories and their classes.

4.2.1 Parameter sensitive

DBSCAN [6] (Representative algorithm): Depends on parameters Eps (radius of a point neighborhood), $MinPts$ (minimum number of points required to be within the neighborhood of a point for it to be called as a dense point). STING [45]: For each cell in a given layer there are parameters which depend on attributes and otherwise. Some important parameters involved are: n (dataset size), m (mean of all the cell values), s (standard deviation of all the cell values), min (minimum cell values), max (maximum cell value).

DENCLUE [19]: The influential parameters are σ (determines the influence of a point in its neighborhood) and ξ (points out if a density-attractor is significant or not). CLIQUE [12]:

The parameters involved here are ξ (number of partitions per dimension), τ which represents the density threshold. These parameters influence the clustering results of the algorithm. GDBSCAN [30]: The algorithm depends on following parameters viz. a neighborhood predicate $NPred$, a weight function $wCard$ and a minimum weight $MinCard$. Variation in these parameters affects the overall functioning of the algorithm. Inc-DBSCAN [31]: Similar to DBSCAN, parameters Eps and $MinPts$ determine the performance of the algorithm. PDBSCAN [26]: Similar to DBSCAN, parameters Eps and $MinPts$ affect the outcome of the algorithm.

OPTIGRID [47]: Uses the kernel density estimate (KDE) [52] to find the point density. The related parameters in the KDE function can potentially influence the density attractors and attracted points involved in clustering. WaveCluster [49]: Multiple parameters are used such as number of dimensions d, number of paritions per dimension m which makes the algorithm parameter sensitive.

SDBSCAN [32]: Extends the DBSCAN algorithm and is therefore sensitive to parameters. IPCLUS [50]: The KDE approach is used to evaluate the point density. KDE comprises of a smoothing factor or kernel bandwidth h which influences the outcome of the algorithm. SNN-DBSCAN [4]: SNN-DBSCAN mainly involves three parameters: Eps, $MinPts$ and $similarity(p, q)$ where $p, q \in D$ (input dataset). Each of these parameters influence the final set of clusters and outliers obtained. DBDC [34]: DBDC relies on the aggregation of information from local sites to a global site for clustering. In this process, the algorithm uses certain parameters that affects the final set of clusters.

IDBSCAN [35]: A total of five parameters influence the IDBSCAN algorithm. These are: the neighbor list size k, the distance for spatially and non-spatiallly arranged objects $Eps1, Eps2$, $MinPts$ similar to DBSCAN and a threshold value ϵ. FOPTICS [53]: Two fuzzy distance functions are used as algorithm parameters making FOPTICS parameter sensitive. ST-DBSCAN [16]: The involved parameters are $Eps1, Eps2$, $MinPts$ and $\Delta\epsilon$ potentially affecting the outcome of the algorithm. KIDBSCAN [36]: KIDBSCAN consists of three parameters viz. ϵ (radius) $MinPts$ and cluster impacting the result of the algorithm. D-Stream [51]: An integer parameter is involved which adjusts the clusters after that amount of time. Variation in the parameter may impact the cluster adjustment time due to the offline component of the algorithm.

CUDA-DClust [27]: The algorithm is mainly influenced by two parameters $\epsilon, MinPts$ similar to DBSCAN [6]. The various components of the algorithm use certain other parameters making the method sensitive to change in the parameter values. DVBSCAN [15]: The parameters involved are ϵ (radius), μ (equivalent to $MinPts$ of DBSCAN), α and λ which control the local density variation. P-DBSCAN [37]: The density parameter used is $MinOwners$ having similar functionality as $MinPts$ of DBSCAN. The other parameter known to have influence on the algorithm is ϵ (the radius component). DADBC [38]: Variants of the parameters in DBSCAN are used by DADBC. These are $MinPts_{global}$ and ϵ_{global} where parameter $MinPts_{global} = f(MinPts, \epsilon_{global})$.

MR-DBSCAN [39]: $Eps, MinPts$ and m(maximum size of data per node) are the prominent parameters having an impact on the outcome of the algorithm. FDBSCAN [54]: Since the method extends DBSCAN while dealing with uncertain data, the involved parameters go onto have an influence on the clustering of fuzzy data objects. GSCAN [28]: In case of GSCAN, the parameters used are the number of data items N, number of dimensions of data d and the minimum number of points $MinPts$ within an object's neighborhood making the algorithm sensitive to parameter changes. Cludoop [41]: The algorithm uses Eps and $MinPts$ as parameters similar to DBSCAN leading towards parameter sensitivity. MBSCAN [29]: MBSCAN relies on parameters such as number of $iTrees(t)$, μ-neighborhood mass(μ), the core point formation threshold(δ_{core}) to produce the clusters. Any change in parameter values influences the final clustering result.

4.2.2 Parameter change adaptive

OPTICS [14](Representative algorithm): OPTICS produces cluster ordering that contains information corresponding to density based clusters. A wide range of parameters are covered by the mechanism adopted by OPTICS making it a parameter change adaptive algorithm. DBCLASD [21]: The algorithm functions without the involvement of any parameters. HDBSCAN [40]: Executes DBSCAN for a range of ϵ values and aggregates the result to cluster points having stability over ϵ. As a result the algorithm is more resistant to parameter changes. DSets-DBSCAN [42]: Applies the concept of histogram equalization to pairwise similarity matrix of input dataset. This enables the algorithm to create an approach independent of user defined parameter setting. Dynamic density based clustering [44]: The algorithm leverages the idea of *Sandwich Guarantee* for the newly introduced ρ-approximate DBSCAN. On varying ϵ to a maximum value of $\rho\epsilon$, the clusters obatined do not change much for a smaller value of ϵ.

4.3 Execution mode

4.3.1 Serial

DBSCAN [6] (Representative algorithm): The algorithm first detects the core and non-core points from the dataset and aggregates the core points to form clusters. Subsequently, the points which are not a part of any cluster get filtered as noise points. STING [45]: The designation of different layers of cells and processing the relevant cells to extract clusters happen in a serial manner. DENCLUE [19]: DENCLUE functions in two serial steps. These are mapping of appropriate regions of dataspace followed by the clustering step. CLIQUE [12]: The algorithm enables discovery of dense clusters in high dimensional dataspace. The partitioning of data into cells as well as cluster formation take place serially. GDBSCAN [30]: GDBSCAN extends DBSCAN for both point objects and spatially arranged objects. The algorithm functions in a serial manner.

Inc-DBSCAN [31]: It is the incremental version of DBSCAN which handles data updates dynamically. The updates are processed sequentially. OPTIGRID [47]: The partitioning of dataspace into grids, creation of hyper-planes and the task of clustering are serial in nature. OPTICS [14]: The augmented ordering of the database and building of the clustering structure take place in a serial manner. WaveCluster [49]: The algorithm partitions the feature space into cell based hyper rectangles. In

the subsequent stage, wavelet transform is applied over the feature space resulting to produce clusters. SDBSCAN [32]: The sampling of data representatives is followed by an efficient labeling of the un-labeled data. This is done in a sequential manner.

IPCLUS [50]: Leverages the intervention of human and machines to extract clusters serially in a high dimensional data space. SNN-DBSCAN [4]: The steps of SNN-DBSCAN starting from the construction of K-sparsified SNN graph to detection of core and non-core points happen sequentially. Each step is dependent on the previous step. The final step is clustering making the algorithm a serial procedure. IDBSCAN [35]: The search for a region in the neighborhood for aggregating the expansion seed happens serially. FOPTICS [53]: The probability value used to determine the proximity score between points followed by clustering is a serial process. ST-DBSCAN [16]: The algorithm addresses the issue of variable density in a cluster. The clustering of spatio-temporal data or non-spatial data takes place in serial manner. KIDBSCAN [36]: KIDBSCAN functions in three steps supporting the integration of K-means clustering [9] with the IDBSCAN [35] algorithm. Starting from variable initialization to applying K-means leading into the execution of IDBSCAN take place sequentially.

D-Stream [51]: The algorithm facilitates clustering of data streams using grids. The clustering of grids is done depending on their density. The steps of the algorithm are serial in nature. DBCLASD [21]: DBCLASD is an incremental approach towards clustering. The algorithm dynamically determines the clusters in a sequential way without involving any input parameters. DVBSCAN [15]: The algorithm handles the intra-cluster density variation in a serial manner P-DBSCAN [37]: P-DBSCAN analyzes locations and events through the use of geo-tagged photos. The algorithm is based on DBSCAN thereby adopting a serial method of execution. DADBC [38]: The model proposed by DADBC is a three step process aggregating the information received from the localized nodes. The steps are carried out serially. HDBSCAN [40]: Adopts a hierarchical scheme of clustering based on density computations. The algorithm focuses on finding the most significant clusters in a serial manner. FDBSCAN [54]: Extends DBSCAN while finding the similarity between fuzzy data points in a sequential manner by using probability based distance functions.

GSCAN [28]: Adopts a mixture of serial and parallel schemes to perform clustering. The algorithm extends CUDA-DClust [27] by building a grid like structure while extracting clusters. DSets-DBSCAN [42]: The algorithm steps involve histogram equalization thereby combining the benefits of DSets [43] and DBSCAN [6] for clustering to happen in a sequential manner. Dynamic density based clustering [44]: The algorithm facilitates dynamic clustering of data on frequent updates. Various theoretical schemes have been proposed to reduce the computational hardness. The algorithm functions in a serial manner. MBSCAN [29]: MBSCAN computes the similarity measure between data objects, detects the core and non-core points and performs clustering in sequential way.

4.3.2 Parallel

PDBSCAN [26] (Representative algorithm): PDBSCAN demonstrates parallelization of DBSCAN using a distributed data-structure as a part of shared nothing architecture. CUDA-DClust [27]: Uses the high parallelism facility of the GPUs in combination with a greater bandwidth for efficient clustering. MR-DBSCAN: Implements the Map-Reduce in four stages to ensure parallelism. The algorithm locally executes DBSCAN for extracting the clusters. [39], GSCAN [28]: GSCAN extends CUDA-DClust by eliminating redundant distance computation through the use of grid structure.

4.3.3 Distributed

DBDC [34] (Representative algorithm): Clusters data locally in different sources and restores the cluster information in a globally located server using a distributed structure. Cludoop [41]: Uses a distributed structure for efficient clustering of large scale data.

4.4 Nature of data

4.4.1 Spatial

DBSCAN [6] (Representative algorithm): Uses the SEQUOIA 2000 [55] datasets representing Earth Science tasks. Each of the contained datasets can be spatially arranged. STING [45]: Similar to DBSCAN, the SEQUOIA 2000 [55] datasets were used for evaluation of the STING algorithm. CLIQUE [12]: Uses synthetic as well as real world datasets with varying number of attributes that can be spatially placed. GDBSCAN [30]: Synthetic databases viz. “bell shaped clusters, non-convex shaped clusters and arbitrary shaped clusters” were used for comparisons along with the SEQUOIA 2000 datasets [55]. Inc-DBSCAN [31]: 2D spatial databases were used along with WWW access log database. PDBSCAN [26]: The algorithm uses Birch [56] and SEQUOIA datasets [55] which are spatial in nature.

OPTIGRID [47]: High dimensional spatial datasets were used removing the phenomenon of “curse of dimensionality”. WaveCluster [49]: Large synthetic datasets were generated for evaluating the algorithm. Each spatial dataset were composed of 1,000,000 data objects. SDBSCAN [32]: Similar to DBSCAN large spatial databases were used along with the SEQUOIA 2000 benchmark datasets [55]. IPCLUS [50]: Spatial datasets (See UCI Machine Learning Repository) were used. SNN-DBSCAN [4]: The spatial datasets used in SNN-DBSCAN used were 2D dataset, datasets related to earth sciences. DBDC [34]: The algorithm used three 2D point sets for its evaluation. IDBSCAN [35]: Large spatial databases from different resources e.g., satellite images, geographical images, medical images etc. were used for evaluating the algorithm.

ST-DBSCAN [16]: Spatial data such as sea surface temperature, sea winds, wave height, sea surface height have been used for evaluation. KIDBSCAN [36]: Four spatial datasets with polygonal clusters have been used for assessing the algorithm. CUDA-DClust [27] Large spatial mining databases were adopted for comparing the efficiency of CUDA-DClust over its sequential counterpart. DBCLASD [21]: Large spatial databases comprising polygonal clusters, earthquake databases were used for comparisons. DVBSCAN [15]: The algorithm used 2D synthetic dataset containing 4000 points for the evaluation purposes. DADBC [38]: For evaluation purpose, a spatial 2D dataset viz. DS9 was used. MR-DBSCAN [39]: The GPS

location records from vehicular traffic were used to asses the effectiveness of the algorithm. HDBSCAN: 2D spatial dataset was used while judging the performance of the algorithm [40]. GSCAN [28]: Compares with CUDA-DClust [27] and DBSCAN [6] over spatial dataset by varying the parameter values. Dynamic density based clustering [44]: Both static and dynamic datasets were used proving the effectiveness of the algorithm. MBSCAN [29]: Well known spatial datasets such as Libras and Segments (See UCI Machine Learning Repository) were used for peformance evaluation.

4.4.2 Non-Spatial

GDBSCAN [30] (Representative algorithm): Enables clustering of objects having both spatial and non-spatial attributes. Data related to geography, biological data, astronomical data were looked upon by the algorithm. FOPTICS [53]: The algorithm adopts fuzzy (non-spatial) objects and expresses similarity between them using probability functions. ST-DBSCAN [16]: The algorithm facilitates cluster discovery for both non-spatial and temporal data, e.g., *record of temperatures during day time*. D-Stream [51]: Synthetic data and KDDCup'99 intrusion detection dataset (See UCI Machine Learning Repository) were primarily used to judge the quality of the said stream clustering algorithm.

FDBSCAN [54]: An artificial dataset (ART) consisting of 1000 2D points and an engineering dataset (PLANE) with 5000 42D points were taken into consideration. Cludoop [41]: Cludoop uses the Hadoop platform for carrying out distributed clustering. The datasets included a trajectory dataset from the project *Geolife* [57] and a *Taxi* data [58] containing around 15 million points. In addition, synthetic datasets viz. Ssyn (4413 points) and Lsyn (2 billion) points were also used to evaluate the algorithm.

4.4.3 Multimedia/others

DENCLUE [19] (Representative algorithm): Datasets related to CAD and molecular biology were used for assessing the performance level of DENCLUE. OPTICS [14]: Visualizing large high-dimensional datasets were taken into account while evaluating OPTICS. Industrial parts dataset, synthetic 2D datasets, 64D color histograms gathered from TV snapshots were also employed for evaluation. P-DBSCAN [37]: The algorithm uses a collection of geo-tagged photos from Flickr for analyzing places and events. HDBSCAN [40]: Gene-expressions datasets [59] viz. "Cell-Cycle237", "Cell-Cycle384" and "YeastGalactose" were used as three of the major datasets. Among the other major datasets considered were Wine, Glass, Iris (See UCI Machine Learning Repository). DSets-DBSCAN [42]: The algorithm mainly dealt with the task of image segmentation carried on Berkley Segmentation dataset (See the Berkeley Segmentation Dataset and Benchmark).

In order to summarize the classification of DBCLAs, we extract the following details:

- A classification chart of the DBCLAs (Refer Fig. 6).
- Extract the percentage of DBCLAs belonging to all classes under any given category (Refer Table 3).
- Identifying multi-class DBCLAs (Refer Table 4).

Table 3 Percentage of DBCLAs in each class

Category	Class	#DBCLAs	#Total DBCLAs	Percentage of DBCLAs
Density definition	Point based	23	32	71.875 %
	Grid based	8		25 %
	Probabilistic	7		21.875 %
	Data dependent	2		6.25 %
Parameter sensitivity	Parameter sensitive	28		87.5 %
	Parameter change adaptive	6		18.75 %
Execution mode	Serial	27		84.375 %
	Parallel	4		12.5 %
	Distributed	3		9.375 %
Nature of data	Spatial	25		78.125 %
	Non-spatial	6		18.75 %
	Multimedia/others	5		15.625 %

From the classification chart (Fig. 6) we observe that each of the four categories: *Density definition, Parameter sensitivity, Execution mode* and *Nature of data* contain different classes. Each category classifies thirty-two DBCLAs into these designated classes per category.

The classification chart also enables us to extract the percentage of DBCLAs belonging to a particular class. Table 3, represents the percentage share of individual classes within each category. From Table 3, it is evident that nearly 71% of the DBCLAs adopt point based technique to compute the density. A higher percentage of the algorithms are also sensitive to change in parameter values.

One of the primary goals of DBCLAs is to handle large voluminous datasets in $\mathcal{R}^d$ space. These datasets pose challenges while grouping objects in higher dimensions. As a result, an increasing number of DBCLAs are involved in dealing with large non-spatial or multimedia datasets. Around 18% of the DBCLAs deal with non-spatial data while a significant 15% handle multimedia datasets e.g., DENCLUE [19], GDBSCAN [30]. It has also been observed that the point based technique of finding densities from these datasets in $\mathcal{R}^d$ space has many limitations [4, 19, 47]. As a result, nearly 24.24% and 21.875% of DBCLAs use probabilistic and grid based approach to compute the neighborhood density in the dataspace.

A single DBCLA can exhibit a combination of properties to address certain challenges and therefore it may belong to multiple classes under a certain category. Table 4 provides an account of algorithms that belong to more than a single class. Eg: Under the category of *Density definition*, around 21% of DBCLAs use more than one technique to find the density of a neighborhood while an estimated 9.375% of DBCLAs handle multiple types of datasets. The presence of multi-class algorithms signifies the evolution of DBCLAs aiming to address a variety of issues related to dealing with high dimensional datasets.

5 Relationship and differences among DBCLAs

In this section, first we present a set of common features shared by various DBCLAs in order to relate and differentiate between the algorithms. Next, we provide the citation percentage value of each algorithm showing the difference in referrals for

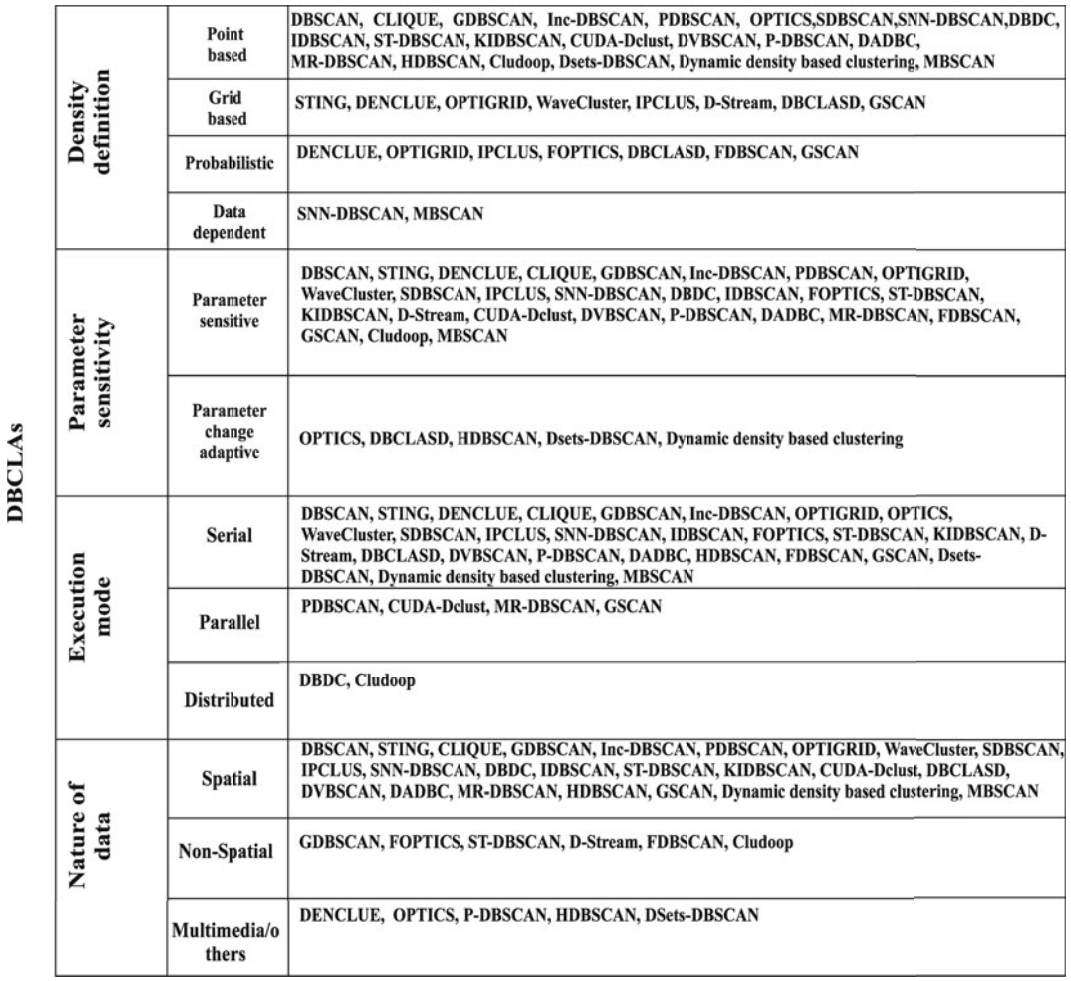

Fig. 6 Classification chart of the DBCLAs

Table 4 Percentage of DBCLAs belonging to multiple classes

Category	DBCLA	#Classes	Class name	Percentage of multi-class DBCLAs
Density definition	DENCLUE	2	Grid based, Probabilistic	21.875 %
	OPTIGRID	2	Grid based, Probabilistic	
	IPCLUS	2	Grid based, Probabilistic	
	SNN-DBSCAN	2	Point based, Data dependent	
	DBCLASD	2	Grid based, Probabilistic	
	GSCAN	2	Grid based, Probabilistic	
	MBSCAN	2	Point based, Data dependent	
Parameter sensitivity				
Execution mode	GSCAN	2	Serial, Parallel	3.125 %
Nature of data	GDBSCAN	2	Spatial, Non-spatial	9.375 %
	ST-DBSCAN	2	Spatial, Non-spatial	
	HDBSCAN	2	Spatial, Multimedia/others	

individual DBCLA. This is complemented by the extraction of conceptual dependencies of DBCLAs on one another highlighting the inter-algorithm relationship.

5.1 Relationship and differences through common features of DBCLAs:

Based on our study of DBCLAs in this paper, we extract a set of generic features: *data preprocessing, density evaluation, dense subspace extraction* and *cluster build* for comparing the DBCLAs (Refer to Tables 5 and 6). The description of these features are as follows:

5.1.1 Data preprocessing

Different data pre-processing techniques adopted by DBCLAs:

- Transforming raw data to meaningful information.

- Reducing the dimensions of data.
- Using data generators.
- Processing voluminous data.
- Data reduction.

Data preprocessing step requires translating raw data into a meaningful and understandable form. Real world data at most times is complex, partial or inconsistent. Therefore it remains a challenge to identify patterns or predict behavior from such data. Depending on the context in which the algorithm is used, it is often desirable to present the input data in a processed format. A processed input on a given algorithm often enables to smoothly interpret the outcome of the method. In context of the DBCLAs, there are methods which adopt a strategy of preparing data as per the requirements of algorithm. For instance in OPTIGRID [47], the projections of points are considered instead of the raw data. The algorithm mainly strives towards addressing the curse of dimensionality in high dimensional space.

Data mining tasks often demand data to be presented in reduced dimensions. This phenomenon of dimensionality reduction is often carried out by techniques such as the Principal Component Analysis (PCA) [60], kernel PCA [61], Linear discriminant analysis (LDA) [62], generalized discriminant analysis [63]. Algorithms such as IPCLUS [50] employ the PCA technique to find the optimum subspace in which the momentum of a single set of points about their centroid is minimal.

Data generators are used to synthetically produce the data. The structure of datasets and their size are controlled by the data generators. Data generators influence important parameters such as the records, features and the range of values for

Table 5 Relating and differentiating DBCLAs based on their common features

Algorithm	Preprocessing	Density evaluation	Dense subspace extraction	Cluster build
DBSCAN [6]	No	Points within a region of specified radius	Regions beyond threshold density	Arbitrary shapes, uniform density
STING [45]	No	Relies on statistical information	Populated grids	Arbitrary shapes
DENCLUE [19]	Yes	Analytically, sum of influence functions	Density attractors, kernel functions	Arbitrary shapes in $\mathbb{R}^d$ space
CLIQUE [12]	Yes	No. of points in a subspace	Dense subspace	DNF expressions, $C \in \mathbb{R}^d$
GDBSCAN [30]	Yes	Points within a region of specified radius	Regions beyond threshold density	Arbitrary shapes, uniform density
Inc-DBSCAN [31]	No	Points within a region of specified radius	Regions beyond threshold density	Arbitrary shapes, uniform density
PDBSCAN [26]	Yes	Points within a region of specified radius	Regions beyond threshold density	Arbitrary shapes, uniform density
OPTIGRID [47]	Yes	Points within a region of specified radius	Regions beyond threshold density	Highly populated grid cells
OPTICS [14]	Yes	Probability density function	Density threshold dependent	Arbitrary shapes
WaveCluster [49]	Yes	Grid based	Wavelet transforms on feature space	Arbitrary shapes, wavelet transformation
SDBSCAN [32]	Yes	Points within a region of specified radius	Regions beyond threshold density	Sampling technique, core point cluster formation
IPCLUS [50]	Yes	kernel density estimation	Curve peaks	Arbitrary shape eye estimation
SNN-DBSCAN [4]	No	Strong links associated to a point	Uniformly compact regions	Arbitrary shapes
DBDC [34]	Yes	Points within a region of specified radius	Regions beyond threshold density	From aggregated information
IDBSCAN [35]	Yes	Points within a region of specified radius	Regions beyond threshold density	Arbitrary shapes, core point cluster expansion
FOPTICS [53]	Yes	Probability density function	Density threshold dependent	Fuzzy objects
ST-DBSCAN [16]	Yes	Points within a region of specified radius, assigns density factor	Regions beyond threshold density	Arbitrary shape on spatio-temporal data
KIDBSCAN [36]	Yes	Points within a region of specified radius	Regions beyond threshold density	Arbitrary shapes
D-Stream [51]	No	Grid based	Grids with high density	Combining grids, arbitrary shapes

Table 6 Relating and differentiating DBCLAs based on their common features

Algorithm	Preprocessing	Density evaluation	Dense subspace extraction	Cluster build
CUDA-DClust [27]	No	High no. of chains, multiple seed list	Regions beyond threshold density	Cluster expan -sion kernel, chains
DBCLASD [21]	Yes	Parameterless, probability distribution	Arbitrary shapes, grid based approach polygons	Arbitrary shapes, large databases
DVBSCAN [15]	No	Compute cluster density mean	Depends on cluster density variance	Core point cluster formation
P-DBSCAN [37]	Yes	Number of people, owners etc. adaptive density	Adaptive density threshold based	Small packed clusters with high density
DADBC [38]	Yes	Points within a region of specified radius	Regions beyond threshold density	From aggregated information
MR-DBSCAN [39]	Yes	Points within a region of specified radius	Regions beyond threshold density	From aggregated information
HDBSCAN [40]	No	Points within a region of specified radius	Regions beyond threshold density	From aggregated information
FDBSCAN [54]	Yes	Core object probability	Peak regions from the curve	Fuzzy clustering
GSCAN [28]	Yes	GPU kernel function	Grid based	Arbitrary shapes, grid based cluster formation
Cludoop [41]	No	Points within a region of specified radius	Regions beyond threshold density	From aggregated information
DSets- [42] DBSCAN	Yes	Histogram equalization	Regions beyond threshold density	Arbitrary shapes
Dynamic density based clustering [44]	No	Points within a region of specified radius	Regions beyond threshold density	Merging or splitting of clusters
MBSCAN [29]	Yes	Mass based calculation	Regions beyond threshold mass	Arbitrary shapes

each feature. In methods such as CLIQUE [19], the clusters are represented as hyper rectangles in reduced number of dimensions. The user provides cluster description which includes the hyper-rectangle subspace along with the range for each attribute in the subspace.

Clustering algorithms dealing with multimedia data e.g., images, videos, geographical maps etc. requires extensive preprocessing of data. The data is presented in its furnished form to the method for smooth interpretation of the result. e.g., one of the major DBCLAs viz. GDBSCAN [30] combines five pictures consisting of 1,024,000 intensity values with 8 bits per pixel. The method represents individual points over the surface by a 5-D vector. This corresponds to an earth surface area equaling 10^6 m^2. Extracting clusters in these kind of feature spaces is a common task applicable in remote sensing digital image analysis for constructing thematic maps in GIS (geographic information systems).

Some of the methods consider a reduced yet meaningful information from the dataset to perform their tasks. The reduction technique involves transformation of numerical or alphabetical digital information derived experimentally or through empirical studies into a corrected, ordered, and simplified form. The goal is to reduce voluminous data down to meaningful parts. The data reduction technique may be done due to a number of reasons. This may include the constraints involved in availing requisite computational resources. A number of complex datasets e.g., molecular biology usually contain a huge amount of information to be processed. Handling of such datasets poses a significant challenge and may have an impact on the outcome of the experiment. Data reduction techniques are employed to alleviate such issues and draw an inference from a given application domain.

5.1.2 Density evaluation

DBCLAs perform density computation by adopting the following approaches:

- Determining number of points in a neighborhood.
- Use of mathematical functions.
- Using Kernel Density Estimate (KDE).
- Through data dependent similarity measure.

Density of a region is calculated by determining the number of points lying within that region. Most of the DBCLAs employ this basic methodology to compute the density of various subspaces. DBSCAN [6] is the pioneer algorithm in the field of DBCLAs. In this algorithm, for a single point, its region of neighborhood wrt. a radius (*Eps*) must comprise of a minimum number of points (*MinPts*) for it to be called as a core or dense point otherwise it qualifies as a non-core or border point. Figure 7 depicts a generic method adopted by most DBCLAs to compute the density.

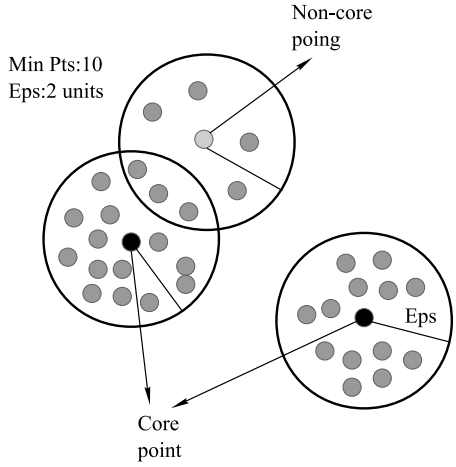

Fig. 7 Density computation based on number of points in a region

Apart from defining density in terms of number of points within a region, some algorithms rely on mathematical functions to compute the point density. DENCLUE [19] identifies such functions as influence functions. Influence functions are a mathematical description of the influence of a data object $x \in \mathcal{R}^d$. The aggregation of influence functions for all data objects at a point x provides the density at that point. For a set of feature vectors $D = \{x_1, x_2, x_3, \ldots, x_r\} \subset \mathcal{R}^d$, the density function is defined as in Eq. (4):

$$f_B^D(x) = \sum_{j=1}^{r} f_B^{x_i}(x). \tag{4}$$

Here $f_B^{x_i}(x)$ represents the influence of x_i on x. Influence function are arbitrary in nature. A specific influence function can be a distance function $d : \mathcal{R}^d \times \mathcal{R}^d \to \mathcal{R}_0^+$. Illustrations of influence functions can be Square Wave influence function, Gaussian influence function (Refer to Eqs. (5) and (6)). Let σ be a certain threshold in context of the influence functions then,

Square Wave function:

$$\begin{aligned} f_{Square}^D(x, y) &= 0, \text{if } \mathrm{d}(x,\ y\ ) > \sigma, \\ &= 1, \text{if } \mathrm{d}(x,\ y\ ) \leqslant \sigma, \end{aligned} \tag{5}$$

Gaussian Influence function:

$$f_{Gaussian}^D(x, y) = e^{-\frac{d(x,y)^2}{2\sigma^2}}. \tag{6}$$

Another technique known as kernel density estimation(KDE) has been effectively used to compute densities of different DBCLAs. For any random variable, KDE estimates the probability density function by using non-parametric technique. The kernel density function is defined as follows: Let D be a collection of r points belonging to a $\mathcal{R}^d$ space and h represents level of smoothness, then on the basis of kernel density estimator K, the density function f^D is given by Eq. (7) as:

$$f^D(x) = \frac{1}{rh} \sum_{j=1}^{r} K\Big(\frac{x - x_j}{h}\Big). \tag{7}$$

The non-negative function $K(.)$ which is the kernel integrates to 1. It has a mean 0 and $h > 0$, where h is a smoothing parameter known as bandwidth. In OPTIGRID [47] method, KDE proves to be a robust technique for detecting clusters in voluminous datasets.

Studies related to statistics have pointed out various kernels K, e.g., the square wave function [64]. The kernel bandwidth h strongly influences the final estimation. A smoother kernel is a statistical technique for estimating a real valued function by using its noisy observation. The estimated function is smooth and the level of smoothness is set by a single parameter. In OPTIGRID, the density function consists of maxima that are above a certain threshold noise ξ. These maxima define the clusters of the algorithm. Methods such as FOPTICS [53], IPCLUS [50], DBCLASD [21] make use of such density functions.

Data dependent dissimilarity measure also enables to compute point densities. The similarity between points is calculated based on the distribution of data instead of any geometrical model. MBSCAN [29] was the first algorithm to introduce a generic approach of finding data dependent dissimilarity in order to overcome the weaknesses of distance based methods. MBSCAN employs the concept of *iForest* [29] to find the mass of smallest region including the data points. The mass value between a pair of points gives the measure of their proximity. For any data item, the density is defined as the total number of points with whom the mass falls below a certain threshold.

Graph based clustering technique viz. SNN-DBSCAN [4] also employed the idea of data dependent dissimilarity, e.g., shared nearest neighbor [33] to determine the proximity between data points. In SNN-DBSCAN, a link between two points is constructed based on the number of shared nearest neighbors instead of any distance measure. The number of shared links that are adjacent to a given point gives the measure of its density. A given point is designated as core if the number of adjacent links crosses a certain predefined threshold value. The remaining points are classified as either non-core or noise points.

5.1.3 Dense subspace extraction

DBCLAs identify dense regions through a varied set of techniques as follows:

- Identify areas with higher frequency of points.
- Select grid cells containing a minimum number of data objects.
- Determining the peaks of the curve obtained from any density function as the center of a dense area.

The dense region of a data space is identified based on the aggregation of points in a certain locality. The regions which exceed a minimum number of points determined by a predefined threshold are known as dense regions. This technique of detecting dense region applies for a number of DBCLAs.

The grid based techniques identify the dense regions by determining the grids which contain a specified minimum number of data objects within a cell.

Methods which adopt the kernel based techniques to find the density, detect the peaks of the curve that are obtained from the probability density function. Such peaks are usually identified as core points and act as the attractors of other points. The neighborhood of these peaks are designated as dense regions.

5.1.4 Cluster build

DBCLAs form clusters through following different ways:

- Grouping core points.
- Accumulating the dense grids.
- Merging denser geometrical shapes.

The cluster formation scheme mainly involves grouping of the core points. The cluster membership of a border point is identical to that of the core point nearest to it. If two core points are connected directly then they are placed in the same cluster. In a similar manner the core points which are interconnected with one another through other core points share the same cluster membership. The non-core points without any cluster membership qualify as noise or outliers. The use of core points allows the DBCLAs to detect clusters of arbitrary shapes. However, the role of core points is not restricted to find globular shaped clusters only. The various ways in which the grouping of core points take place also enables a particular method to find non-globular or elongated shaped clusters in different subspaces.

DBCLAs which use the probability density functions to compute the density of data objects aggregate the core points to form the clusters. The use of density functions to identify the core points and the denser regions enables efficient cluster detection in a database with significant noise quantity.

Methods which adopt the grid based technique accumulate the dense grids to form the clusters. The grid based technique adopted by various algorithms prove useful while detecting high dimensional clusters.

While detecting clusters by polygonal method, certain geometrical shapes are constructed and are treated as clusters. The denser geometrical shapes are merged based on their reachability from each other. The regions which are merged together assume the shape of a cluster.

5.2 Relationship and differences through variation in citation percentage and conceptual dependency of DBCLAs

The variation in citation percentage and conceptual dependency (Refer Fig. 8) represents the importance of a DBCLA in influencing other density based algorithms. The pioneer algorithm DBSCAN (1996) [6] is a highly cited method till date having more than 11,000 citations (See Google Scholar). The DBCLAs proposed in the subsequent years consistently refer to DBSCAN while inheriting concepts from the algorithm. The point density in DBSCAN relies on the numerical strength of points in its neighborhood. Although DBSCAN fails to detect clusters of varying densities, the algorithms which address such issues cite the method extensively. From Fig. 8 it is clear that DBSCAN is the most highly cited algorithm with more than 90% referrals.

OPTICS [14] is the second most highly cited algorithm having around 50% citations. OPTICS is a parameter change adaptive algorithm which orders the data point based on the reachability distance from their nearest core point. Many algorithms which address the issue of parameter dependency refer to OPTICS as a benchmark algorithm. In addition, algorithms such as FOPTICS, GDBSCAN, FDBSCAN depend directly on the

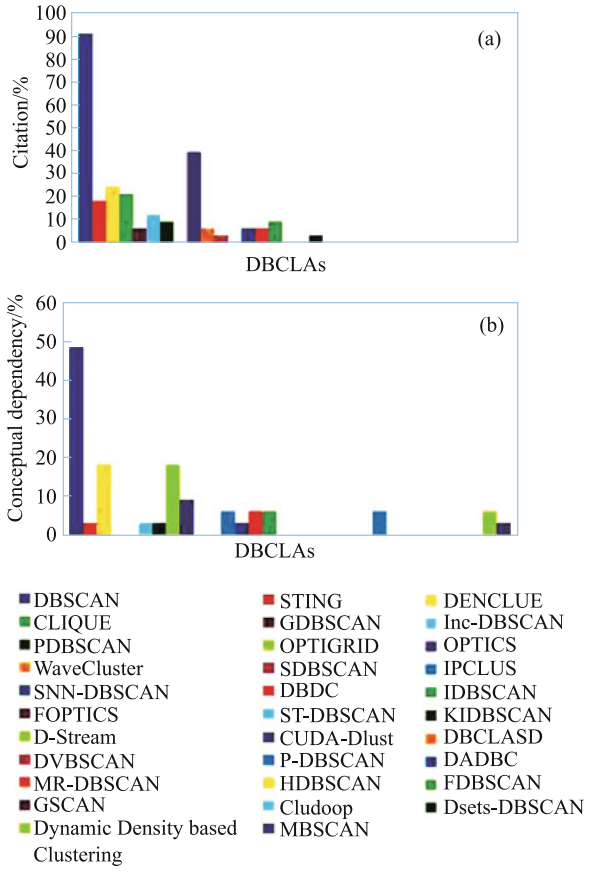

Fig. 8 Citation percentage and Conceptual dependency percentage variation of DBCLAs. (a) Citation percentage variation of DBCLAs; (b) conceptual dependency percentage variation of DBCLAs

pioneer DBCLAs like DBSCAN, OPTICS, DENCLUE with minor variations. As a result the referral percentage to the algorithms proposed in the initial years from 1996 are considerably high. Algorithms such as STING, DENCLUE, CLIQUE have more than 10% citations while some of the methods which have been proposed in recent years have not received considerable amount of citations as yet.

We also provide the conceptual dependency upon a certain algorithm by all the other DBCLAs. Through the conceptual dependency chart in Fig. 8, we identify the percentage of algorithms that inherit the concept of its parent algorithm in its own working methodology. DBSCAN has the highest conceptual dependency with close to 50% dependents. Other major DBCLAs like DENCLUE and OPTIGRID have about 20% dependency. DENCLUE handles clustering of large multi-media databases while OPTIGRID is a grid based clustering technique designed to address issues related to high dimensional data. Over the years many grid based techniques like D-Stream, IPCLUS, DBCLASD have been proposed which depend on ideas conveyed in OPTIGRID.

Parallel and distributed methods have gained importance in recent years due to their higher efficiency as compared to the serial methods. DBCLAs such as MR-DBSCAN makes use of map-reduce techniques to parallelize the detection of clusters while Cludoop is a distributed method relying on hadoop properties to detect the clusters. Subsequently, most of the methods which are based on parallel execution inherit the methodolo-

gies proposed in Cludoop or MR-DBSCAN. The depency on DBCLAs such as IPCLUS, SNN-DBSCAN, IDBSCAN, FOPTICS lie below 10%. A number of variants of DBSCAN proposed over a period of time, e.g., ST-DBSCAN, DVBSCAN have no conceptual dependency till now.

6 Empirical study of DBCLAs

We adopt a qualitative approach to conduct an empirical study of some representative DBCLAs under various categories covered. Based on the direct observation of the DBCLAs studied in this survey, we aim to present a comparison between the concerned algorithms.

1) DBSCAN: DBSCAN [6] is compared with CLARANS [8] and based on visual inspection both the algorithms were evaluated. Datasets of SEQUOIA 2000 benchmark data [55] were selected for comparisons. The included datasets were raster data, point data, polygon data and directed graph data. For DBSCAN, the amount of noise was set to 0% for the first dataset, while it was 10% for the remaining two datasets. DBSCAN was able to detect all the clusters along with segregation of noise from the involved datasets. On the contrary CLARANS had no clear notion of noise filtration. As a result all the points end up obtaining a definite cluster membership. Experimental results showed that the execution time of DBSCAN is marginally higher than $O(n)$, where n is the number of points in the dataset. Moreover, DBSCAN outperformed CLARANS by a factor ranging between 250 and 1900 which increases with the size of the database.
2) STING: STING [45] is compared with CLARANS [8], BIRCH [48] and DBSCAN [6] algorithm. In this case too, the SEQUOIA 2000 datasets were used for comparing STING with the other clustering approaches. The experiments were conducted with a data of size ranging between 1252 and 12512 points. DBSCAN proved to be about 15 times faster than CLARANS. Furthermore, BIRCH outperformed CLARANS by about 20 to 30 times. In effect STING also outperformed BIRCH by a significant margin. In Fig. 9 taken from [45], we provide the necessary comparison plot between STING and DBSCAN which shows the effectiveness of the former.
3) SNN-DBSCAN: The SNN-DBSCAN [4] algorithm was tested on following datasets: 2D (Chameleon [5] and CURE [65]) datasets, NASA Earth Science data and KDDCup'99 [66] dataset. The value of the similarity threshold involved in SNN-DBSCAN potentially influences the quality of clusters detected. Figure 10 taken from [4] shows that the combination of the Jarvis-Patrick [33] scheme and DBSCAN [6] enables detection of better clusters as compared to only DBSCAN running on the same datasets [4]. In case of NASA Earth Science data, a monthly measurement of sea level pressure was take from 1982 to 1993. The data was produced in form of a 2.5° (144 horizontal by 72 vertical divisions). The SNN clustering yielded clusters that were geographically contiguous due to the spatial correlation of the underlying data. A total of 30 clusters were taken into conderation. In case of K-means [9], out of 100 clusters 30 most cohesive clusters were retained while the rest were discarded. The clusters obtained by SNN-DBSCAN were better than that of the K-means algorithm.
4) OPTICS: OPTICS [14] was executed on 64 dimensional color histogram derived from television snapshots. For different parameter values, the algorithm shows a scale up of between 10000 to 100000 data items thereby proving its efficiency. The algorithm runs in $O(n)$ time where n is the number of data objects.
5) DENCLUE: The polygonal CAD data is used for proving the effectiveness of DENCLUE [19]. The data is converted into a eleven dimensional feature vector using a Fourier transformation. The size of the dataset was varied from 10000 to 20000. The parameters of DENCLUE and

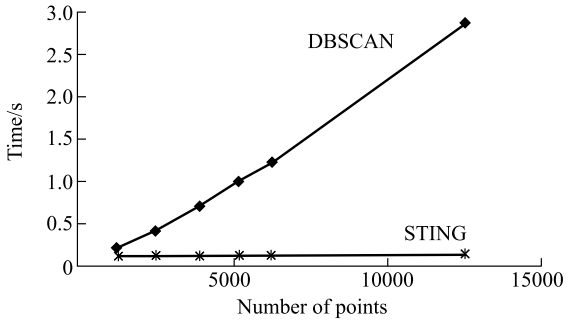

Fig. 9 Performance comparison between STING and DBSCAN [45]

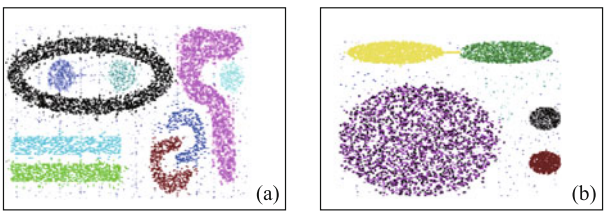

Fig. 10 Clusters detected by the SNN-DBSCAN algorithm on Chameleon and CURE datasets [4]. (a) Chameleon data set; (b) CURE data set

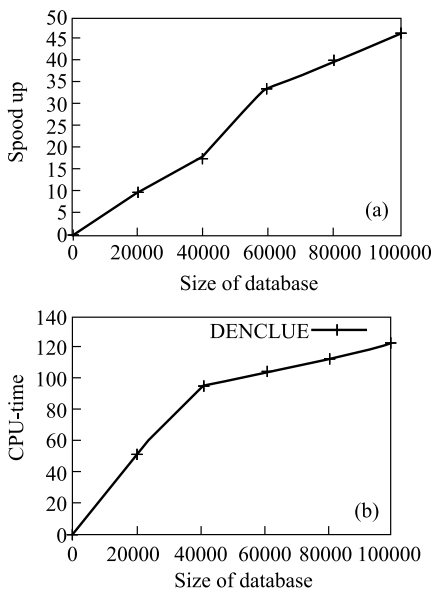

Fig. 11 Performance of DENCLUE [19]. (a) Speed up; (b) DENCLUE

DBSCAN were chosen such that both the algorithms produced identical clusters. Experimental results showed that DENCLUE is about 45 times speedier than the DBSCAN algorithm.

From Fig. 11 taken from [19], it is shown that DENCLUE takes $O(\log n)$ time to execute with n being the number of data objects. The logarithimic time complexity of the algorithm is due to the fact that only data points lying in the densely populated regions are considered for cluster formation scheme.

7 Applications of the DBCLAs

In this section, we present the applications relevant to DBCLAs (Refer to Table 7 for tabular summary). For individual application, we mention the following aspects:

- Describe the application.
- Nature of data used.
- DBCLAs involved in the application.

7.1 Earth sciences

In this application spectral space obtained from satellite images is clustered. This is a task commonly used in remote sensing image analysis. One of the prominent DBCLAs which finds its importance in this application is GDBSCAN [30] which uses a 5-D space. This 5-D space is gathered from the satellite imagery of a region on earth's surface. After preprocessing, the images of various spectral channels of a region are merged. Individual point corresponds to an area of 1000×1000 m^2 on the surface of the earth. Clusters obtained through this procedure finds importance in analysis of digital images in remote sensing.

Algorithms such as SNN-DBSCAN [4] also finds its application in earth sciences data. In particular researchers are interested in finding the areas of ocean whose behavior correlates well to climate events on the earth's land surface. In earth science domain, clusters which are relatively uniform and not as dense are also equally important.

7.2 Molecular biology

Bio-molecules, proteins, DNA and other cell components consist of millions of atoms within them. These cell components interact with each other e.g., protein protein interaction (PPI). PPI contains dense regions that can be identified based on the data connectivity [67]. According to molecular biologists, the spatial arrangement of the molecular structures at the site of interaction holds importance in addition to physio-chemical molecular behavior. In a PPI, several geometric and physio-chemical measures are computed for proteins on the surface. The phenomenon of such interactions is called *docking* [30]. Crystallographic experiments are used to determine the atomic positions of proteins and their structures. For example, solid angle (SA) [68], a geometrical property that describes the degree of convexity or concavity of the area in the neighboring region of the concerned part.

A database for docking of protein processes queries for complementary protein surfaces. The segments on the protein surface should possess a healthy correlation with the known protein docking spot. To find segmentation over the surface of the protein, application pertaining to clustering algorithms is necessary. GDBSCAN is one such DBCLA which finds its applicability in this task of protein segmentation.

7.3 Astronomy

The data acquired from celestial objects may lead to the discovery of various patterns. Grouping of such data is necessary for detecting the presence of any pattern or any other relevant information for mining. Surveys detecting celestial objects of interest enable their statistical studies. Such observations extract sporadic or irregular objects. Modern surveys can potentially produce images in terms of thousands. Each image can consume a space of 10 GB to 1 TB [30]. In order to maximize an yield from a survey it needs a precise and effective approach to detect the source. Conventional approach to separate single sources from outliers requires the sources to cross a predefined threshold. Alternative methods such as [69, 70] make use of the classifier systems [71] or statistics.

On applying density based method [30] for such purpose, the final clustering yield a better result than the known statistical [69, 70] techniques. One of the main reasons to adopt DBCLAs for such purpose is its ability to segregate compact regions from the sparse areas in the data space. This enables detection of clusters in the galactic survey along with appropriate noise filtering.

7.4 Geography

DBCLAs can be used to retrieve two-dimensional polygons based on the similarity measure of non-spatial attributes. In order to characterize the proximity between attributes, the domain is segregated to some distinct classes. The values lying within same class are said to be similar. Sets that have the least or highest feature values become the influence regions. GDBSCAN [30] algorithm extracts such regions of influence.

In the area of geospatial clustering, DBSCAN [6] plays an important role in GIS spatial analysis techniques such as

Table 7 Tabular summary of the applications relevant to DBCLAs

Sl No.	Application domain	Utility	Key function	Nature of data	DBCLAs involved
1.	Earth Sciences	Remote sensing image analysis	Clustering satellite images	Image, Spatial	GDBSCAN, SNN-DBSCAN
2.	Molecular Biology	Identify physio-chemical molecular behavior	Finding segments over protein surface	Image	GDBSCAN
3.	Astronomy	Satellite imagery	Grouping patterns of celestial objects	Image, Spatial	GDBSCAN
4.	Geography	GIS	Merge polygons	Image, Spatial	DBSCAN
5.	Multimedia/others	Urban planning	Cluster photos, events	Multimedia	P-DBSCAN, DENCLUE

polygon overlay [72]. In order to improve the usefulness of clustering, it is essential to study constraint-based geospatial cluster analysis. DBSCAN Ontology (DBSCANO) [73] has been developed in accordance with the DBSCAN process and geographic information data characteristics. C-DBSCANO [73] was developed by combining DBSCAN and DBSCANO. These methods enable clustering under constraints based on the geographical background which leads to reasonable clustering results.

7.5 Multimedia/others

When dealing with multimedia data, many algorithms fail to cluster multi attribute vectors. As a result the efficiency of such methods degenerate at rapid pace. DBCLA such as DENCLUE [19] works efficiently for high dimensional data sets allowing random noises while finding clusters. DBCLA such as P-DBSCAN [37] analyses locations along with utilizing a collection of geographically tagged photos. P-DBSCAN finds interesting places or significant events that are characterized by high photo activity in a specific area. Determining such places and events prove to be beneficial for local authorities, service providers and urban planners. Density based clustering is helpful in such cases because of the following reasons: a) no prior knowledge about the quantity of clusters is available, b) high photo activity can be measured by density and compared to different parts of the region under investigation, c) clusters are of arbitrary shapes and d)sparse regions are treated as noise.

In the next section, we present the future directions and scopes where application of DBCLAs can lead to favorable results.

8 Future directions

In this section, we identify the probable future directions where the application of DBCLAs may be lead to gain favorable results. After studying as many DBCLAs in this article, the following directions have been identified:

1) Heterogeneous datasets: In our study, a large number of proposed techniques focus only on homogeneous datasets. In effect, there lies a scope for evaluating the performance of various DBCLAs while dealing with heterogeneous datasets. With increase in the volume of online data, and a need to integrate structured and unstructured data [74], the application of density based clustering on these datasets can be an important direction. Data related to studying genome sequence [75], gene functional classification [76], fault slip data sets for tectonic movements [77], cosmological data sets [78] are few of the examples where the application of DBCLAs can be advantageous.
2) Intra cluster density variation: The role of clusters having significant intra cluster density variation remains an open issue. Although a seminal work on local density variation is presented [15], the method is parameter sensitive. Also for high dimensional datasets, to the best of our observation, no known method is familiar with local density variation. To make the algorithms adaptive to parameter changes and detect clusters with intra cluster density variation, application of DBCLAs can be useful.
3) Graph based applications: Role of DBCLAs in graph based applications can be of significant importance. Some real world applications can be mapped to graph domain. Involvement of DBCLAs in graph based application can gather importance as DBCLAs can be used to separate out dense regions in the graph from sparse ones e.g., SNN-DBSCAN [4].
4) Social networking: The role of DBCLAs in social networking domain also remains an interesting challenge. A recent study [79] relying on density based technique identifies clusters based on the textual heterogeneity on Twitter using spatio-textual information. Applications of DBCLAs in this field may lead to meaningful observations in the future.

9 Discussion and conclusion

In this article, we presented a comprehensive review of various DBCLAs from the year 1996. Contrary to the previous surveys of DBCLAs, this paper studied as many as thirty-two DBCLAs. Individual algorithms have been described highlighting their key features, time complexity, cluster extraction techniques.

We introduced the notion of common properties for the DBCLAs. The common properties extracts the features that are involved in designing of a DBCLA. The features of the common properties are uniform to all the DBCLAs and acts as an appropriate platform to compare and understand the properties of all the DBCLAs.

The paper also provided a structured representation of the classification of DBCLAs. Four distinct categories were identified and each of these categories independently classified all the DBCLAs in various classes under that category. We chose to group DBCLAs under different categories in an isolated manner because of the presence of a variety of properties in each of the DBCLAs. In addition, we also evaluated the share of DBCLAs belonging to each class within a category. The presence of multi-class algorithms shows the evolution of DBCLAs adopting a mixture of techniques to address challenges related to high dimensional clustering. We limited ourselves to provide an empirical study of some of the representative DBCLAs out of all the algorithms covered in this survey. We did not use any statistical techniques to prove any hypotheses. Rather, based on the study conducted on various algorithms, we presented the behavior of some of the relevant DBCLAs supported by experimental evidences. In future studies, a comprehensive empirical study of the all the major DBCLAs may be conducted.

While providing the differences between the DBCLAs, the goal behind finding the conceptual dependency and citation percentage chart of DBCLAs was to identify the set of algorithms that have had a major influence in the field of density based clustering. The survey also highlights the relevant application areas and the DBCLAs involved in it along with the nature of dataset used. We conclude by identifying the probable future directions where the application of DBCLAs may lead to favorable results.

References

1. Jain A K, Murty M N, Flynn P J. Data clustering: a review. ACM Computing Surveys, 1999, 31(3): 264–323

2. Yan Z, Luo W, Bu C, Ni L. Clustering spatial data by the neighbors intersection and the density difference. In: Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing Applications and Technologies (BDCAT). 2016, 217–226
3. Xu R, Wunsch D. Survey of clustering algorithms. IEEE Transactions on Neural Networks, 2005, 16(3): 645–678
4. Ertöz L, Steinbach M, Kumar V. Finding clusters of different sizes, shapes and densities in noisy, high dimensional data. In: Proceedings of the 2003 SIAM International Conference on Data Mining. 2003, 47–58
5. Karypis G, Han E H, Kumar V. Chameleon: hierarchical clustering using dynamic modeling. Computer, 1999, 32(8): 68–75
6. Ester M, Kriegel H P, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceeedings of International Conference on KDD. 1996, 226–231
7. Keogh E, Mueen A. Encyclopedia of Machine Learning and Data Mining. Curse of Dimensionality. 2nd ed. Springer, Boston, MA, 2017, 314–315
8. Raymond T N, Han J. Clarans: a method for clustering objects for spatial data mining. IEEE Transactions on Knowledge and Data Engineering, 2002, 14(5): 1003–1016
9. Hartigan J A, Wong M A. Algorithm as 136: a k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics), 1979, 28(1): 100–108
10. Silverman B W. Density Estimation for Statistics and Data Analysis. Chapman and Hall/CRC press, 1998
11. Aggarwal C, Reddy C K. Data Clustering: Algorithms and Applications. CRC press, 2013
12. Agrawal R, Gehrke J, Gunopulos D, Raghavan P. Automatic subspace clustering of high dimensional data for data mining applications. In: Proceedings of the ACM SIGMOD International Conference on Management of data. 1998, 94–105
13. Parimala M, Lopez D, Senthilkumar N C. A survey on density based clustering algorithms for mining large spatial databases. International Journal of Advanced Science and Technology, 2011, 31(1): 59–66
14. Ankerst M, Breunig M M, Kriegel H P, Sander J. Optics: ordering points to identify the clustering structure. ACM Sigmod Record, 1999, 28: 49–60
15. Ram A, Jalal S, Jalal A S, Kumar M. A density based algorithm for discovering density varied clusters in large spatial databases. International Journal of Computer Applications, 2010, 3(6): 1–4
16. Birant D, Kut A. ST-DBSCAN: an algorithm for clustering spatial-temporal data. Data & Knowledge Engineering, 2007, 60(1): 208–221
17. Bhuyan R, Borah S. A survey of some density based clustering techniques. In: Proceedings of the 2013 Conference on Advancements in Information, Computer and Communication. 2013
18. Chaudhary S, Chauhan R, Batra P. A survey of density based clustering algorithms. International Journal of Computer Science and Technology, 2014, 5(2): 169–171
19. Hinneburg A, Keim D A. An efficient approach to clustering in large multimedia databases with noise. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining. 1998, 58–65
20. Nagpal P B, Mann P A. Comparative study of density based clustering algorithms. International Journal of Computer Applications, 2011, 27(11): 421–435
21. Xu X, Ester M, Kriegel H P, Sander J. A distribution-based clustering algorithm for mining in large spatial databases. In: Proceedings of the 14th IEEE International Conference on Data Engineering. 1998, 324–331
22. Thiagarasu V, Prabahari R. A comparative analysis of density based clustering techniques for outlier mining. International Journal of Engineering Sciences and Research Technology, 2014, 3(11): 132–136
23. Rajavat A, Dubey P. Comparative study between density based clustering - DBSCAN and OPTICS. International Journal of Advance Computational Engineering and Networking, 2016, 4(12): 34–37
24. Smiti A, Eloudi Z. Soft DBSCAN: improving DBSCAN clustering method using fuzzy set theory. In: Proceedings of the 6th International Conference on Human System Interactions. 2013, 380–385
25. Loh W K, Park Y H. A survey on density-based clustering algorithms. In: Jeong Y S, Park Y H, Hsu C H, Park J, eds. Ubiquitous Information Technologies and Applications. Springer, Berlin, Heidelberg, 2014, 775–780
26. Xu X, Jäger J, Kriegel H P. A fast parallel clustering algorithm for large spatial databases. In: Guo Y, Grossman R, eds. High Performance Data Mining. Springer, Boston, MA, 1999, 263–290
27. Böhm C, Noll R, Plant C, Wackersreuther B. Density-based clustering using graphics processors. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management. 2009, 661–670
28. Loh W K, Moon Y S, Park Y H. Erratum: fast density-based clustering using graphics processing units. IEICE TRANSACTIONS on Information and Systems, 2014, 97(7): 1947–1951
29. Ting K M, Zhu Y, Carman M J, Zhu Y, Zhou Z H. Overcoming key weaknesses of distance-based neighbourhood methods using a data dependent dissimilarity measure. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016, 1205–1214.
30. Sander J, Ester M, Kriegel H P, Xu X. Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications. Data Mining and Knowledge Discovery, 1998, 2(2): 169–194
31. Ester M, Kriegel H P, Sander J, Wimmer M, Xu X. Incremental clustering for mining in a data warehousing environment. In: Proceedings of the 24th International Conference on Very Large Data Bases. 1998, 323–333
32. Zhou S, Zhou A, Cao J, Wen J, Fan Y, Hu Y. Combining sampling technique with DBSCAN algorithm for clustering large spatial databases. In: Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining. 2000, 169–172
33. Jarvis R A, Patrick E A. Clustering using a similarity measure based on shared near neighbors. IEEE Transactions on Computers, 1973, 100(11): 1025–1034
34. Januzaj E, Kriegel H P, Pfeifle M. DBDC: density based distributed clustering. Advances in Database Technology-EDBT 2004, 2004, 529–530
35. Borah B, Bhattacharyya D K. An improved sampling-based DBSCAN for large spatial databases. In: Proceedings of IEEE International Conference on Intelligent Sensing and Information Processing. 2004, 92–96
36. Tsai C F, Liu C W. KIDBSCAN: a new efficient data clustering algorithm. In: Proceedings of International Conference on Artifical Intelligence and Soft Computing. 2006, 702–711
37. Kisilevich S, Mansmann F, Keim D. P-DBSCAN: a density based clustering algorithm for exploration and analysis of attractive areas using collections of geo-tagged photos. In: Proceedings of the 1st International Conference and Exhibition on Computing for Geospatial Research and Application. 2010
38. An N, Khac L, Kechadi M T. On a distributed approach for density-based clustering. In: Proceedings of the 10th International Conference on Machine Learning and Applications and Workshops. 2011, 283–286
39. He Y, Tan H, Luo W, Mao H, Ma D, Feng S, Fan J. MR-DBSCAN: an efficient parallel density-based clustering algorithm using MapReduce. In: Proceedings of the 17th IEEE International Conference on Parallel and Distributed Systems. 2011, 473–480
40. Campello R J, Moulavi D, Sander J. Density-based clustering based on hierarchical density estimates. In: Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining. 2013, 160–172
41. Yu Y, Zhao J, Wang X, Wang Q, Zhang Y. Cludoop: an efficient distributed density-based clustering for big data using hadoop. International Journal of Distributed Sensor Networks, 2015, 11(6): 379–391
42. Hou J, Gao H, Li X. DSets-DBSCAN: a parameter-free clustering algo-

rithm. IEEE Transactions on Image Processing, 2016, 25(7): 3182–3193
43. Pavan M, Pelillo M. Dominant sets and pairwise clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006, 29(1): 167–172
44. Gan J, Tao Y. Dynamic density based clustering. In: Proceedings of the 2017 ACM International Conference on Management of Data. 2017, 1493–1507
45. Wang W, Yang J, Muntz R. Sting: a statistical information grid approach to spatial data mining. In: Proceedings of the 23rd International Conference on Very Large Data Bases. 1997, 186–195
46. Davis L. Bit-climbing, representational bias, and test suite design. In: Proceedings of the 4th International Conference on Genetic Algorithms. 1991, 18–23
47. Hinneburg A, Keim D A. Optimal grid-clustering: towards breaking the curse of dimensionality in high-dimensional clustering. In: Proceedings of the 25th International Conference on Very Large Data Bases. 1999
48. Zhang T, Ramakrishnan R, Livny M. BIRCH: a new data clustering algorithm and its applications. Data Mining and Knowledge Discovery, 1997, 1(2): 141–182
49. Sheikholeslami G, Chatterjee S, Zhang A. WaveCluster: a multi-resolution clustering approach for very large spatial databases. In: Proceedings of the 24th International Conference on Very Large Data Bases. 1998, 428–439
50. Aggarwal C C. A human-computer interactive method for projected clustering. IEEE Transactions on Knowledge and Data Engineering, 2004, 16(4): 448–460
51. Chen Y, Tu L. Density-based clustering for real-time stream data. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2007, 133–142
52. Aggarwal C C. Outlier Analysis. In: Data Mining. Springer, 2015, 237–263
53. Kriegel H P, Pfeifle M. Hierarchical density-based clustering of uncertain data. In: Proceedings of the 5th IEEE International Conference on Data Mining. 2005
54. Kriegel H P, Pfeifle M. Density-based clustering of uncertain data. In: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining. 2005, 672–677
55. Stonebraker M, Frew J, Dozier J. The SEQUOIA 2000 Project. In: Proceedings of International Symposium on Spatial Databases. 1993, 397–412
56. Zhang T, Ramakrishnan R, Livny M. BIRCH: an efficient data clustering method for very large databases. ACM Sigmod Record, 1996, 25(2): 103–114
57. Zheng Y, Xie X, Ma W Y. Geolife: a collaborative social networking service among user, location and trajectory. IEEE Data Engineering Bulletin, 2010, 33(2): 32–39
58. Yuan J, Zheng Y, Xie X, Sun G. T-drive: enhancing driving directions with taxi drivers' intelligence. IEEE Transactions on Knowledge and Data Engineering, 2011, 25(1): 220–232, 2011
59. Yeung K Y, Fraley C, Murua A, Raftery A E, Ruzzo W L. Model-based clustering and data transformations for gene expression data. Bioinformatics, 2001, 17(10): 977–987
60. Bishop C M. Pattern Recognition and Machine Learning. Springer, 2006
61. Lu C D, Zhang T Y, Du X Z, Li C P. A robust kernel PCA algorithm. In: Proceedings of 2004 International Conference on Machine Learning and Cybernetics. 2004, 3084–3087
62. Balakrishnama S, Ganapathiraju A. Linear discriminant analysis-a brief tutorial. Institute for Signal and Information Processing, 1998, 18: 1–8
63. Baudat G, Anouar F. Generalized discriminant analysis using a kernel approach. Neural Computation, 2000, 12(10): 2385–2404
64. Epanechnikov V A. Non-parametric estimation of a multivariate probability density. Theory of Probability & Its Applications, 1969, 14(1): 153–158
65. Guha S, Rastogi R, Shim K. CURE: an efficient clustering algorithm for large databases. ACM Sigmod Record, 1998, 27(2): 73–84
66. Tavallaee M, Bagheri E, Lu W, Ghorbani A A. A detailed analysis of the KDD cup 99 data set. In: Proceedings of IEEE Symposium on Computational Intelligence for Security and Defense Applications. 2009, 1–6
67. Bader G D, Hogue C W V. An automated method for finding molecular complexes in large protein interaction networks. BMC Bioinformatics, 2003, 4(1): 2
68. Connolly M L. Measurement of protein surface shape by solid angles. Journal of Molecular Graphics, 1986, 4(1): 3–6
69. Becker R H, White R L, Helfand D J. The first survey: faint images of the radio sky at twenty centimeters. The Astrophysical Journal, 1995, 450: 559
70. Zepka A F, Cordes J M, Wasserman I. Signal detection amid noise with known statistics. The Astrophysical Journal, 1994, 427: 438–445
71. Weir N, Fayyad U M, Djorgovski S. Automated star/galaxy classification for digitized POSS-II. The Astronomical Journal, 1995, 109: 2401
72. Chrisman N R, Dougenik J A, White D. Lessons for the design of polygon overlay processing from the odyssey whirlpool algorithm. In: Proceedings of the 5th International Symposium on Spatial Data Handling. 1992, 401–410
73. Du Q, Dong Z, Huang C, Ren F. Density-based clustering with geographical background constraints using a semantic expression model. ISPRS International Journal of Geo-Information, 2016, 5(5): 72
74. Hendler J. Data integration for heterogenous datasets. Big Data, 2014, 2(4): 205–215
75. Allen T E, Herrgård M J, Liu M, Qiu Y, Glasner J, Blattner F R, Palsson B. Genome-scale analysis of the uses of the escherichia coli genome: model-driven analysis of heterogeneous data sets. Journal of Bacteriology, 2003, 185(21): 6392–6399
76. Pavlidis P, Weston J, Cai J, Grundy W N. Gene functional classification from heterogeneous data. In: Proceedings of the 5th Annual International Conference on Computational Biology, 2001, 249–255
77. Angelier J. Tectonic analysis of fault slip data sets. Journal of Geophysical Research: Solid Earth, 1984, 89(B7): 5835–5848
78. Kowalski M, Rubin D, Aldering G, Agostinho R J, Amadon A, Amanullah R, Balland C, Barbary K, Blanc G, Challis P J, et al. Improved cosmological constraints from new, old, and combined supernova data sets. The Astrophysical Journal, 2008, 686(2): 749
79. Nguyen M D, Shin W Y. DBSTexC: density-based spatio–textual clustering on twitter. In: Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. 2017, 23–26

Panthadeep Bhattacharjee is currently a PhD research scholar in the Department of Computer Science & Engineering, Indian Institute of Technology (IIT) Guwahati, India. He obtained his BTech in Information Technology from Assam University Silchar, India in 2010 and MTech in Computer Science & Engineering from National Institute of Technology Durgapur (NIT), India in 2012. He worked as assistant professor in the School of Computer Engineering, Kalinga Institute of Industrial Technology (KIIT), Bhubaneswar, India from June 2012 to December 2013. He joined IIT Guwahati in January 2014. His research interest includes pattern recognition, machine learning, data science, online algorithms. He has been a winner in the Artificial Intelligence-Design Challenge Championship held at International Institute of Information Technology (IIIT) -Bangalore in 2018.

Pinaki Mitra is currently an associate professor at the Department of Computer Science & Engineering, Indian Institute of Technology (IIT) Guwahati, India. He obtained his BTech in Computer Science & Engineering from Jadavpur University, Kolkata, India in 1987 and MTech in Computer Science & Engineering from Indian Institute of Science (IISc) Bangalore, India in 1989. He received his PhD from Simon Fraser University, Canada in 1994. He worked as a Research Scientist at Center for Microprocessor Training Education and Research, Jadavpur University. Subsequently he joined National Institute of Management, Kolkata and served as an assistant professor. He joined IIT Guwahati in December 2004. His research interest includes cryptography, network security, computer graphics, multimedia, pattern recognition and machine learning.