Interdisciplinary Journal of AI, Machine Learning & Data Science (IJAIMLDS)

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

Article - e013

A TAXONOMY OF DBSCAN VARIANTS ADDRESSING PARAMETER DEPENDENCE AND HIGH-DIMENSIONALITY

Anunaya Manoj[1], Masood Husain Siddiqui[2], Deepak Kumar Singh[3]

[1,2]Department of Statistics, University of Lucknow, Lucknow
[3]Department of Information & Technology, Indian Institute of Information Technology, Lucknow

Received: 15/04/2026 **Revision Received**: 08/05/2026 **Accepted**: 29/05/2026

ABSTRACT

Density-based clustering algorithms have emerged as a significant area of research within unsupervised machine learning due to their capability to identify clusters based on data density. Among these methods, the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm has gained considerable attention owing to its robustness against noise and outliers. However, DBSCAN's sensitivity to input parameters, its inability to detect clusters of varying densities, and its poor performance in high-dimensional spaces limit its applicability. Considering these challenges, this review categorizes variants of DBSCAN into four major groups: (1) parameter-independent, (2) parameter-reduced, (3) high-dimensional, and (4) hybrid algorithms. Within each category, we provide a detailed description of parameter estimation and clustering strategies. Our analysis highlights the use of graph-based methodologies, tree-based and grid structures, and other techniques including RNN, LSH, DLT, and triangular inequality to help automate parameter estimation and accelerate neighborhood searches. Moreover, hybrid strategies incorporating feature selection, feature extraction, graph-based modeling, game theory, and deep learning have shown promising potential in addressing both challenges simultaneously. By consolidating insights across these research directions, this review offers a comprehensive perspective on the evolution of DBSCAN, equipping researchers and practitioners with a deeper understanding of its limitations, existing solutions, and avenues for future exploration.

KEYWORDS: DBSCAN, Parameter Estimation, High-Dimensional, Clustering, Density-Based Algorithms

Interdisciplinary Journal of AI, Machine Learning & Data Science (IJAIMLDS)
ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

1. INTRODUCTION

Unsupervised machine learning algorithms analyse unlabelled data points to identify underlying patterns within the dataset. Amongst several unsupervised machine learning techniques, clustering is the most employed unsupervised learning algorithm. Clustering is a technique which captures meaningful clusters from a set of datapoints by forming internally homogeneous groups. Clustering finds its application in several fields including gene expression [1], market segmentation [2], image processing [3], medicine [4], geospatial analysis [5], and astronomy [6]. Among various clustering methods existing in the literature [7], density-based clustering algorithms have emerged as a popular approach by adopting the notion of density while clustering the points. These techniques can identify clusters in different density regions of data. Most widely used density-based approaches are DBSCAN [8], OPTICS [9], and DENCLUE [10].

In recent years, the notion of high-dimensionality has gained prominence due to the substantial amounts of data generated, encompassing hundreds or even millions of features. In high-dimensional datasets, data points are scattered over the data space, and clusters may be hidden by irrelevant features or noise. Consequently, these datasets become a challenge owing to their complexity and the substantial volume of information inherent in each data point. Thus, in 1961, Richard Bellman termed this phenomenon as "curse of dimensionality." High-dimensional datasets provide challenges such as overfitting, extensive computations, and inappropriate distance metrics. Subsequently, in [11], Silverman elucidated the influence of increased dimensions on density distributions, therefore expanding the applicability of DBSCAN in this domain.

The Density Based Spatial Clustering of Applications with Noise (DBSCAN) technique, developed by [8] possess the ability to find varying shaped clusters along with its robustness against outliers or noise. Nonetheless, DBSCAN possesses numerous limitations. Some of the prominent limitations are: (1) it is sensitive to input parameters, (2) it cannot recognize clusters with varying densities, and (3) it is inefficient in extracting clusters in higher dimensions. Previous research has investigated the difficulties associated with DBSCAN, leading to the development of several versions of the algorithm. However, a consolidated review of these variants under the taxonomy of parameter estimation and high dimensionality is still unexplored. Hence, we move a step ahead in combining all such adaptations of DBSCAN in one single review specifically evaluating them based on their efficacy, encompassing runtime and space analysis. The key research contributions of this review are highlighted as follows:

1. A comprehensive review of DBSCAN variants that aim to alleviate the issues related to parameter dependency and poor scalability in high-dimensional datasets.
2. We grouped the DBSCAN adaptations into four distinct categories: (1) parameter-independent algorithms (2) parameter-reduced algorithms, (3) high-dimensional algorithms, and (4) hybrid algorithms, as depicted in figure 1.
3. An in-depth analysis of different methodologies employed along with the algorithmic design, computational complexity, and parameter requirements.
4. The paper outlines emerging research opportunities for researchers to enhance their work in the field of unsupervised clustering.

Interdisciplinary Journal of AI, Machine Learning & Data Science (IJAIMLDS)
ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

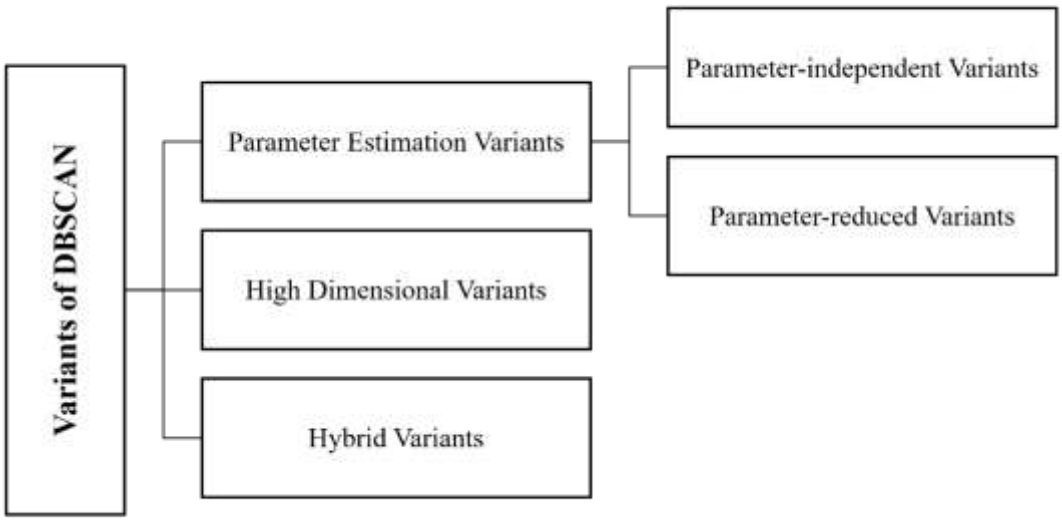

Figure 1. Taxonomy of Variants of DBSCAN

2. OVERVIEW OF DBSCAN ALGORITHM

The DBSCAN [8] algorithm is a density-based clustering algorithm that determines the density of a region around a specific point through two user-defined parameters ε and *minPts*. ε represents the radius of a circle around a point, whereas *minPts* denote the least number of points inside the ε radius. The formation of clusters depends on classifying a point into three categories: core, border, and noise. If the number of points within a point's ε-neighbourhood is more than the specified *minPts*, the point is a core point. Likewise, if the number of points within a core point's ε-neighbourhood is less than the specified *minPts*, the point is a border point. A point which is neither a core point nor a border point is termed as noise. Additionally, the DBSCAN algorithm defines the reachability between pairs of data points through the concepts of directly density reachability, density reachability, and density connectivity.

The DBSCAN method begins by selecting a random point x and proceeds to explore its neighbouring points for specified ε and *minPts* by employing a similarity metric. Next, the point x is categorised as a core point, border point, or noise. A cluster C is created when x is a core point, which is further extended by including the points density-reachable from point x. This procedure continues until no other clusters are detected. If x and y are two core points that are density-reachable from each other, then either of the point will be allocated to a different cluster. Clustering points in this manner amplifies DBSCAN's capability to identify clusters of varying shapes in higher density regions. Nonetheless, the DBSCAN method presents certain restrictions. These encompass parameter sensitivity, varying density clusters, and high dimensionality.

The clustering process by DBSCAN significantly depends on its parameters ε and *minPts*, and to obtain an optimal value for each parameter is a challenging task that necessitates an extensive knowledge of the dataset. Schubert et al. [12] emphasized the significance of setting of these parameters. For many two-dimensional datasets, *minPts* can be kept at a default value [8], whereas ε should be chosen as small as possible [12]. The parameter ε exhibits an inverse relationship with the number of clusters [13]. This indicates that as ε value increases, larger neighbourhoods are formed which result in a decrease in the number of clusters. Likewise, as

Interdisciplinary Journal of AI, Machine Learning & Data Science (IJAIMLDS)
ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

suggested by [14], larger *minPts* generate a smaller number of clusters, whereas smaller *minPts* reduces influence of noise. Moreover, the significance of selecting an appropriate distance measure prevents DBSCAN from detecting clusters with different densities [13]. This metric is crucial for generating homogeneous clusters as it establishes homogeneity, hence influencing the actual clustering process. Also, DBSCAN is susceptible to the "curse of dimensionality" when the dataset has more than two dimensions, resulting in a time complexity of $\Omega(n^{4/3})$ [15].

These concerns have been examined in prior scholarly works, and various approaches have been suggested in diverse clustering algorithms to mitigate them. In the following sections, we will discuss different adaptations of DBSCAN algorithm that aim to address the challenges posed by the sensitivity of the parameters ε and *minPts*, as well as the issue of high dimensionality, as illustrated in figure 2.

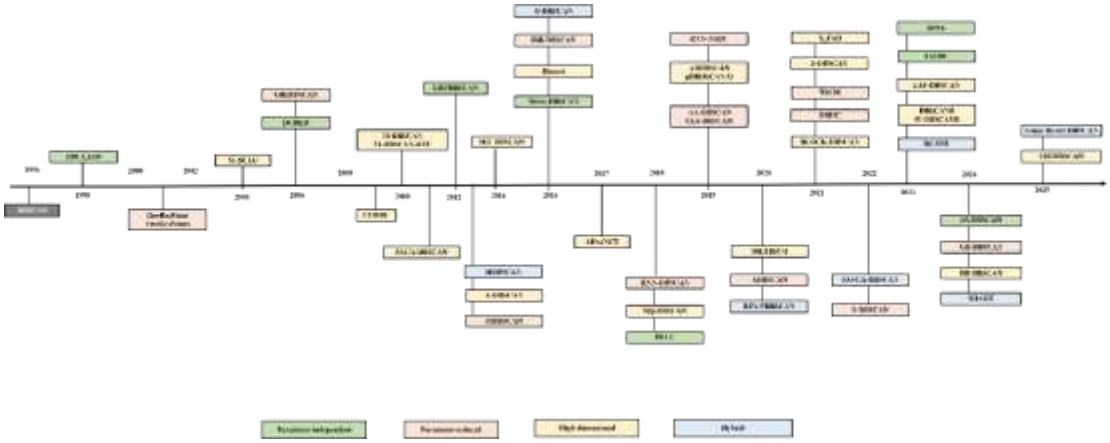

Figure 2. Evolution of Variants of DBSCAN

3. VARIATIONS IN DBSCAN

The existing body of research [16], [17], [18], [19], [14], [20], [21], [22], [23], [24], [25], [26] has extensively explored a wide range of clustering methodologies aimed at overcoming the inherent limitations of the conventional DBSCAN algorithm. Several studies have focused on eliminating DBSCAN's sensitivity to parameter selection through graphical techniques such as histogram equalization, *k*-distance plots, and elbow KNN method. Other works have proposed approaches, including reverse nearest neighbor (RNN), influence space, density layer tree (DLT), and nearest neighbor graph (NNG) to minimize the parameter selection. In addition, methods designed for high-dimensional data, such as locality-sensitive hashing (LSH), tree-based models, triangular inequality, feature weighting, and WAND, have been introduced to enhance clustering performance. Accordingly, this section categorizes these algorithms based on the specific limitations they address, providing a structured perspective on the evolutionary development of DBSCAN variants over time.

3.1 Parameter-independent variants of DBSCAN

One of the primary limitations of the DBSCAN algorithm is its dependence on global parameters. The user input parameters ε and *minPts* provide a density estimation of the data points and setting them necessitates a comprehensive understanding of the dataset to get optimal clustering results. Therefore, prior studies [14], [18], [27], [28] have put forth various adaptive techniques that autonomously calculate the optimal values of ε and *minPts* to address

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

this drawback of the DBSCAN algorithm. Table 1 summarises the articles on the DBSCAN variants based on parameter estimation.

Table 1: Parameter Estimation Variants of DBSCAN

S. No	Year	Author	Algorithm	Parameters	Time complexity
Parameter-independent variants					
1.	1998	[27]	DBCLASD	None	Not Specified
2.	2006	[18]	DCBRD	None	$O(nk + m^2k + nm)$; $m = \frac{n}{k}$
3.	2012	[31]	GRPDBSCAN	None	Not Specified
4.	2016	[14]	DSets-DBSCAN	None	Not Specified
5.	2018	[29]	PELC	None	Not Specified
6.	2020	[32]	Not Specified	None	Not Specified
7.	2023	[23]	FADBC	None	$O(in)$
8.	2023	[28]	DPPA	None	$O(n)$
9.	2024	[30]	SS-DBSCAN	None	$O(n^2)$ (worst-case complexity)
Parameter-reduced variants					
10.	2001	[16]	OneResPoint, TwoResPoints	ε	Not Specified
11.	2006	[40]	GRIDBSCAN	$\lambda, perc$	Not Specified
12.	2013	[33]	ISDBSCAN	k	$O(n.\log(n))$
13.	2016	[37]	ISB-DBSCAN	k	Not Specified
14.	2018	[38]	RNN-DBSCAN	k	$O(n^2)$
15.	2019	[39]	RNN-NSDC	α	$O(n.\log(n))$
16.	2019	[34]	AA-DBSCAN, *k*AA-DBSCAN	*minPts*	Not Specified
17.	2020	[42]	ADBSCAN	k	$O(n.\log(n))$
18.	2021	[43]	TSCM	d_c	Not Specified
19.	2021	[41]	DBHC	k	$O(n^2 + \sqrt{n}.nlogn + n^2)$
20.	2022	[35]	S-DBSCAN	g_r	$O(g_r \times nlog\ (n))$
21.	2024	[36]	GB-DBSCAN	*Ratio*	$O(nd\ log\ (n))$

** n is the dataset size; k is the number of circles; m is the average number of points in each circle; i is the number of iterations; g_r is the granularity; d is the dimension number*

In [27], Xu et al. introduced a novel clustering algorithm, Distribution Based Clustering of Large Spatial Databases (DBCLASD) assuming points to be uniformly distributed inside a cluster. This algorithm can cluster data points without requiring any input parameters. As an iterative method, the DBCLASD algorithm enhances an initial cluster by including its neighboring points, disregarding the entire cluster or the database. The neighbourhood points are retrieved efficiently through R*-trees. DBCLASD's non-parametric nature allows it to efficiently cluster massive geographical databases and produce high-quality clusters of any shape.

The Data Point Positioning Analysis (DPPA) algorithm [28] is a parameter-free clustering approach which automatically determines its clustering criteria by analyzing geometric relationships and distance distributions among data points. DPPA is based on two key concepts: the First Nearest Neighbor (1-NN), which represents the closest neighboring point within an automatically estimated radius range, and the Maximum Nearest Neighbor (Max-NN), defined as a sequence of data points connected through successive 1-NN links that terminate at a boundary point. The clustering procedure consists of three main stages. First, DPPA automatically estimates a radius range (λ) by computing the minimum distance from

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

each data point to all others and identifying the global minimum and maximum of these distances. Next, the algorithm constructs neighbor-link structures by identifying 1-NN relationships and forming Max-NN chains, rather than using fixed-radius neighborhood queries. Finally, clusters are generated by merging data points connected through Max-NN chains, with clusters progressively expanded along dense connectivity paths.

Some researchers [14], [29], [23], [30] devised variants of DBSCAN which estimated the parameters ε and *minPts* through the clustering process. For instance, Fahim et al. [18] introduced a novel algorithm called Density Clustering Based on the Radius of Data (DCBRD) based on the fundamental concept of DBSCAN. DCBRD first partitions the data space into overlapping circular regions, ensuring that every data point belongs to at least one region, with some points potentially appearing in multiple regions. To determine region size, DCBRD computes the overall data space radius (R) from the dataset center and derives the region radius (Rad), which adaptively increases with dimensionality to handle data sparsity. The ε parameter is automatically estimated by averaging the far nearest pair distances within solid regions, while MinPts is fixed at 3 based on empirical findings. Finally, a DBSCAN-based clustering process is applied using the estimated parameters to obtain the final clusters.

Grid-based clustering algorithms have been known for their efficiency in identifying clusters by focusing on the grids rather than individual datapoints. Keeping this view, Darong and Peng [31] proposed an algorithm which combines grid-based and density-based methods, Grid-based DBSCAN Algorithm with Referential Parameters (GRPDBSCAN). To estimate the parameters required by the algorithm, non-overlapping grids of uniform size are constructed. Each of these grids contain at least a single data point and the grids that include the greatest number of data points are designated as ε-neighbour and the points inside them are referred to as *minPts*.

Hou et al. [14] proposed a novel clustering method called DSets-DBSCAN which combines the dominant sets algorithm with DBSCAN. This algorithm overcomes the limitations of both DSets and DBSCAN which rely on parameters – regulation parameter, ε and, *minPts*. To eliminate dependency on regulation parameter, DSets employs histogram equalization technique to standardize DSets which produces over-segmented clusters of spherical shapes. The over-segmentation problem is addressed using DBSCAN which is executed on dominant sets to generate clusters with arbitrary shapes. Bataineh & Alzahrani [23] developed a two-stage novel clustering algorithm called the Fully Automated Density-based Clustering (FADBC). In the parameter estimation stage, a histogram technique is employed to estimate the optimal parameters ε and *minPts* for dataset of any size and shape. These parameters are subsequently used to extract clusters by examining every data point of the dataset, and labelling them based on their surrounding density. The high-density regions are considered as clusters, whereas low-density clusters do not constitute a cluster.

Another parameter-free clustering algorithm was proposed by Yang et al. [29] called Laplacian Centrality Peaks Clustering based on Potential Entropy (PELC) that integrates Laplacian centrality, potential entropy, and a DBSCAN framework to cluster the data points. The potential entropy method identifies an optimal threshold at which entropy reaches its minimum. This threshold serves as the neighborhood radius (ε) in DBSCAN. Additionally, a sequence of steps is undertaken to ascertain the *minPts* parameter. Initially, the dataset is segmented into two subsets, which are further partitioned into smaller subsets until the cluster centers of both subsets are density-reachable. In the last stage, the *minPts* parameter is set to the boundary density of each subset. Another variant of DBSCAN that employed a graphical method to determine parameters of DBSCAN automatically was proposed by [32]. In this paper, Starczewski et al. proposed that the estimation of parameters ε and *minPts* begin by sorting the *k*-nearest neighbour distances and subsequently plotting them on a graph. The graph typically shows a point range known as the *knee,* where a sharp increase in distances

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

occurs. This *knee* is crucial for selecting a suitable ε value, as points beyond this *knee* might be considered noise. In case of varying cluster densities, the ε value is adjusted using a distance factor (d_p) and a bias to account for different sizes of the *knee*. Likewise, *minPts* is estimated using d_p and the dimensionality of the dataset. On similar lines, a semi-supervised clustering technique (SS-DBSCAN) was developed by Abdulhameed et al. [30] that incorporated a graphical approach to determine the ε parameter. Although ε was determined using two approaches: Elbow-KNN and Trial & Error, the authors considered Elbow-KNN method as an appropriate approach for determining the ε value. Further, the *minPts* parameter was evaluated when the dataset is either noisy or noiseless. This algorithm is termed as semi-supervised due to its behaviour of selecting core points on a specific criterion. This specification is user-defined which means it must depend on the clustering objective.

This section addressed several parameter-free variations of DBSCAN, including the formation of overlapping circles [18] and non-overlapping grids of uniform size [31]. Graph-based methodologies such as the *k*-dist function [32], histogram [23], histogram equalization [14], elbow-KNN [30], were used to ascertain the parameters of DBSCAN. Additionally, [27] presumed a uniform distribution of points, whereas [28] utilized nearest neighbours for clustering purposes. Nonetheless, prior literature developed several variations of DBSCAN that need a minimal number of parameters to be set. Consequently, in the next part, we provide an overview of these methods.

3.2 Parameter- reduced variants of DBSCAN

The traditional DBSCAN algorithm necessitates adequate domain understanding for deciding the input parameters. An inappropriate choice of ε and *minPts* in DBSCAN can cause two issues – (a) cluster splitting and (b) cluster merging. Cluster splitting occurs with a small ε or large *minPts*, whereas cluster merging happens with a large ε or small *minPts* [31]. Keeping this in mind, past researchers [16], [33], [34], [35], [36] devised several adaptations of DBSCAN requiring minimal input parameters. This section presents some of these variations of DBSCAN for a comprehensive knowledge of the literature.

Gradual Clustering Algorithms (GCAs) cluster with a limited number of points and, if results are relatively adequate, advances to a greater number of points. Using this aspect, Wu and Gardarin [16] proposed two variants of GCA: OneResPoint GCA, which employs a single vantage point to partition the data space, and TwoResPoints GCA, which uses two representative points – selected either randomly or as extreme points – to enhance filtering and improve pruning efficiency. GCA incorporates a heuristic-based attribute selection process that normalizes attribute values, computes a radius reflecting each attribute's contribution to distance variation, and ranks attributes to identify the most influential dimensions for initial processing. The algorithm then precomputes distances between all data objects and one or two vantage points in the *n*-dimensional space. These distances are sorted and stored using an index structure such as an M-tree. GCA is then integrated into DBSCAN, modifying the neighbor retrieval step. During clustering in the higher-dimensional space, the algorithm applies a "close enough" test based on the triangle inequality. Two points are considered potential neighbors only if the absolute difference between their precomputed distances to the vantage point does not exceed ε. Only points that satisfy this condition undergo full distance computation in the higher-dimensional space, thereby significantly reducing unnecessary calculations.

The notion of space stratification works on the principle of ranking the data objects based on their densities in the space along with detecting outliers in the dataset. In [33], Cassisi et al. defined stratification as a linear combination of Influenced Outlierness (INFLO) function and *k*-nearest neighbour (*k*-NN) distances. The INFLO function establishes a novel neighborhood relationship termed influence space (IS), which incorporates both nearest neighbors (NNs) and reverse nearest neighbors (RNNs). Regarding this, the IS_{DBSCAN} algorithm [33] was

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

proposed which utilizes STRATIFY algorithm to remove global as well as local outliers and focuses on the expansion of clusters within the IS_k-neighbourhood rather than the ε-neighbourhood. The symmetric nature of IS_k prevents objects belonging to clusters of different densities from being considered as neighbours. IS_{DBSCAN} necessitates the prior setting of only one parameter, k, number of k-nearest neighbours which estimates the IS_k of each data point. The formation of a cluster begins by selecting a random point until a boundary point or an outlier is reached. On similar lines, [37] developed the ISB-DBSCAN technique, which operates on p-stable Local Sensitive Hashing (LSH) and influence space.

The LSH technique reduces time of neighbourhood query search. However, for further minimizing the time required for the neighbour query in DBSCAN, the authors presented an enhanced version of the LSH algorithm, RLSH (Randomness-based Local Sensitive Hashing) which makes ISB-DBSCAN scalable. This approach depends on an approximate nearest neighbour scheme. Furthermore, the problem of parameter sensitivity in DBSCAN was resolved by utilizing the influence space (IS), by requiring only one parameter, the number of k-nearest neighbours. ISB-DBSCAN performs better than DBSCAN and IS_{DBSCAN} in the sense that it clearly distinguishes between border and noise points where border points are classified based on the nearest core point within their influence space. Moreover, ISB-DBSCAN introduces the core density reachability concept which ensures that cluster expansion involves only core points.

Subsequently in [38], Bryant and Cios devised an algorithm, Density-based Clustering Algorithm using Reverse Nearest Neighbour (RNN-DBSCAN) which integrated the concept of reverse nearest neighbour with DBSCAN. The clustering process begins by selecting a core point and forming clusters by modifying the core definitions of reachability and connectivity. RNN-DBSCAN outperforms IS_{DBSCAN} since it necessitates the specification of just one parameter, k, whereas IS_{DBSCAN} requires the parameter k and a threshold that is to be determined a priori. Furthermore, the outlier elimination process in IS_{DBSCAN} introduces errors by misclassifying certain data points as noise [38]. On similar lines, Dai et al. [39] proposed a novel clustering algorithm, RNN-NSDC based on natural reverse nearest neighbour framework. A point q is said to be a natural neighbour of a point p, if both points p and q consider each other as the neighbour of the other. The clustering process begins by finding the core objects, which are derived through determining the reverse nearest neighbor of each object. The core objects are subsequently grouped based on the neighborhood structure information obtained via the natural neighbor technique. The unallocated points are assigned to the nearest cluster. RNN-NSDC is advantageous over the traditional DBSCAN in that it can effectively handle patterns with varied densities.

The GRId Density-Based Spatial Clustering of Applications with Noise (GRIDBSCAN) algorithm, proposed by Uncu et al. [40], integrates the DBSCAN and grid-based clustering algorithms. A three-level clustering approach was employed, involving grid selection, grid merging, and the implementation of DBSCAN. The selection phase involves selecting appropriate grids so that there is a uniform density inside each grid. During the merging phase, cells with similar densities are merged and for each grid the optimal values for ε and *minPts* are obtained. The DBSCAN method is implemented at the third level utilizing the identified parameters. Another version of DBSCAN, a DBSCAN-based hierarchical clustering algorithm (DBHC) was developed by Latifi-Pakdehi & Daneshpour [41]. This algorithm clusters data points in a hierarchical manner by determining the parameter ε using a k-dist plot. Next, for each ε value, the DBHC implements the DBSCAN algorithm and subsequently combines the clustering outcomes in a bottom-up approach.

Another algorithm devised by Kim et al. [34] devised an approximate version of adaptive DBSCAN (AA-DBSCAN) to improve the clustering speed when determining the parameters for DBSCAN. The algorithm constructs density layers (DLs) using a quadtree-based Density

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

Layer Tree (DLT). The DLT improves computational efficiency by restricting neighborhood checks to points within adjacent nodes, rather than scanning the entire dataset. For each density layer, AA-DBSCAN computes an approximate adaptive ε-distance. To address the limitation of using a static ε-distance, an enhanced variant called kAA-DBSCAN was introduced. This extension derives the adaptive ε-distance from a dataset-specific k-distance. However, incorporating k-nearest neighbor distance computations introduces additional overhead, resulting in increased runtime.

Later, in [42], Li et al. devised an algorithm known as Adaptive DBSCAN (ADBSCAN), utilizes k-NN and Nearest Neighbour Graph (NNG) to cluster the data points in the three basic steps. At the initial step, ADBSCAN constructs a NNG to identify local high-density samples, without requiring any user-defined parameters. The second step involves filtering out these samples located in the noisy regions by employing k-NN distances under the assumption that the logarithm of these k-NN distances follows a Gaussian-like distribution. This filtering step uses the parameters k and $noise_{percent}$, $0 \leq noise_{percent} \leq 1$. Lastly, the closest subgraphs are assigned to the same cluster. Similar in series, Cheng et al. [36] proposed the Granular-Ball DBSCAN (GB-DBSCAN) algorithm to enhance the efficiency of DBSCAN. Granular-balls are compact, coarse-grained representations of data that group together points and their neighbors that share the same class. The algorithm follows four steps: (1) generating granular-balls via a novel bottom-up unsupervised partitioning method that uses the FLANN algorithm to find each point's k-nearest neighbors; (2) classifying granular-balls as core or non-core based on density, which is inversely proportional to their radius; (3) merging core GBs using DBSCAN's reachability concept; and (4) allocating non-core GBs to the nearest core GBs if none of the points are classified.

Li et al. [43] developed a Two-stage Clustering Method (TSCM) that integrates Density Peaks (DP) with DBSCAN to overcome the limitations of both approaches. Specifically, DBSCAN suffers from parameter sensitivity, while DP relies on a decision graph that requires manual selection of cluster centers. To mitigate parameter sensitivity, TSCM employs a bat optimization technique to determine the ε value, and the enhanced DBSCAN algorithm generates the initial clusters. The DP algorithm utilizes these initial clusters to determine the number of cluster centers. Low-density cluster centers are automatically identified from the two-dimensional decision graph, resulting in the final clustering output. Nevertheless, determining an appropriate cut-off distance remains a challenging issue.

Likewise, a Self-tuning version of DBSCAN (S-DBSCAN) was developed by Ros et al. [35] which works on the concept of density peaks and detects clusters in a sequential manner. The algorithm is structured as a hierarchical process that integrates the concepts of distance, k-nearest neighbors, and density peaks. S-DBSCAN implements its core method, S-DBSCANCORE, using suitable input parameters to systematically scan the database through selected seeds, according to the notion of identifying density peaks. The data patterns densely packed together are clustered by S-DBSCANCORE. S-DBSCAN outperforms other methods in the sense that it does not require cluster number as input. Moreover, in case where clusters exist naturally, S-DBSCAN detects them without requiring modifications.

Most of the adaptions of DBSCAN require a clustering approach to create efficient clusters with minimum dependence on parameters. Clustering strategies involved the use of influence space [33], [37], RNN [38], granular-balls [36], quadtree-based DLT [34], and nearest neighbour graph [42]. Additionally, certain variants of DBSCAN have integrated grid-based [40], density peaks [35], [43], and hierarchical-based [41] clustering algorithms with DBSCAN. GRIDBSCAN evaluated parameters using homogenous density grids, whereas TSCM employed a bat optimization technique, and DBHC utilized a k-dist plot. These algorithms demonstrated exceptional accuracy relative to the conventional DBSCAN method. Nevertheless, these methods failed to emphasize the high-dimensional characteristics of the

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

majority of real-world datasets. Consequently, in the next section, we provide alternate versions of DBSCAN that primarily focus on high dimensionality.

3.3 High Dimensional variants of DBSCAN

Conventional clustering algorithms such as K-Means, DBSCAN, OPTICS, etc., operate in the full-dimensional feature space, considering all attributes of each data object. However, as the number of dimensions increases, applying these algorithms becomes computationally expensive, a phenomenon commonly referred to as the "curse of dimensionality." In high-dimensional spaces, data points become increasingly sparse, which diminishes the effectiveness of distance-based similarity measures [22]. Consequently, meaningful cluster structures are often confined to a small subset of relevant dimensions, while irrelevant features and noise conceal the true clusters. To address these challenges, prior studies [15], [17], [44], [45], [46], [47], [48] have proposed various strategies to mitigate the effects of high dimensionality in DBSCAN (summarized in Table 2). Accordingly, the following section reviews key advancements of DBSCAN that effectively tackle dimensionality-related issues.

Table 2: High dimensional and Hybrid Variants of DBSCAN

S. No	Year	Author	Algorithm	Parameters	Time complexity
High dimensional variants					
1.	2004	[17]	SUBCLU	ε, *minPts*	Not Specified
2.	2009	[49]	CODBU	*t, MinDen*	$O(ns)$
3.	2010	[44]	TI-DBSCAN TI-DBSCAN-REF	ε, *minPts*	Not specified
4.	2011	[55]	PACA-DBSCAN	k_{pick}, k_{drop}, s, γ, α	Not specified
5.	2013	[45]	A-DBSCAN	ε, *minPts*	Not specified
6.	2014	[46]	HD_DBSCAN	μ, k	Not specified
7.	2016	[52]	Dboost	ε, *minPts*	Not specified
8.	2017	[20]	ADvaNCE	ε, *minPts*	$O(n)$
9.	2018	[21]	NQ-DBSCAN	ε, *minPts*	$O(n.\log(n))$
10.	2019	[51]	µDBDSCAN, µDBDSCAN-D	ε, *minPts*	$O(n.\log(m) + n.\log(r))$; $n = mr$
11.	2020	[47]	METRIC-1	ε, *minPts, r*	Not specified
12.	2021	[15]	BLOCK-DBSCAN	ε, *minPts*	$O(n.log(n))$
13.	2021	[54]	*h*-DBSCAN	ε, *minPts*	$O(log(n)) + O(n.\log(n)) + O(ik) + O(l)$
14.	2021	[22]	S_FAD	*Iterations, D, N, betamin, betamax, trial, limit, p*	$O(2^d).(n\log(n))$
15.	2023	[13]	DBSCANR W-DBSCANR	k, β	$O(n^2)$
16.	2023	[24]	LAF-DBSCAN	α	Not specified
17.	2024	[48]	BD-DBSCAN	λ, δ	Not specified
18.	2025	[26]	SRRDBSCAN	ε, *minPts, L*, δ, *c*	$O\begin{pmatrix} d.\max(nKL, \\ n^{1+\rho(c)}minPts^{1-\rho(c)}, \\ \boldsymbol{n^{\rho(c)}minPts^{-\rho(c)}N}) \end{pmatrix}$
Hybrid variants					
19.	2013	[19]	HDBSCAN	*minPts*	Not specified
20.	2016	[56]	G-DBSCAN	ε	$O(n + nd)$
21.	2020	[58]	BFA-PDBSCAN	None	Not specified

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

22.	2022	[59]	FA+GA-DBSCAN	None	Not Specified
23.	2023	[57]	DCSNE	None	$O(n^{1.5} + n.log(n))$
24.	2024	[60]	SHADE	μ	Not specified
25.	2025	[25]	Game theory-based DBSCAN	None	Not specified

** n is the dataset size; s is the number of subspaces; m is the number of micro-clusters; i is the initial cluster number; d is the dimension number; k is the number of neighbors; l is the number of border points*

Subspace clustering goes one step beyond the traditional clustering by assuming that the clusters can be represented by a subset of features rather than all features. This notion facilitates clustering in high dimensional spaces by constructing grids; however, the idea of using density-connectivity with subspace clustering was generated by Kailing et al. [17]. The algorithm Density-connected Subspace Clustering (SUBCLU) employed a bottom-up approach which systematically explores various subspaces from the original feature space and applies a density threshold which is defined by parameters *minPts* and *ε*. This process allows the algorithm to efficiently prune irrelevant subspaces and focus on those that contain meaningful clusters. As a result, SUBCLU can effectively uncover hidden patterns in complex datasets. Another study by Liu et al. [49] attempted to use subspace clustering by constructing a novel approach Clustering by Ordering Density-Based Units (CODBU) which is based on ordered subspace to find clusters. CODBU arranges the units in a density-wise manner, where the highest density unit occupies the first level of a cluster followed by other units arranged accordingly. If the density of a unit is greater than that of its any other neighboring units in density, it forms a new cluster. If a unit has a deficit in density relative to its adjacent neighbors, it cannot be regarded as a density peak and will join3 the lowest density-cluster.

Later, in [46], Jahirabadkar & Kulkarni proposed High Dimensional DBSCAN (HD_DBSCAN) which applied two approaches: (a) assessing cluster quality of each dimension and (b) determining the appropriate value of *ε*. For the first approach, the *k*- nearest neighbour is used along with statistical methods like mean and divergence to evaluate the density distribution around each data point. The quality value of 1 indicates a clearer clustering structure in that dimension. Additionally, a threshold value of 0.4 is used to prune the dimensions, thereby making it suitable for high dimensional data by focusing on relevant dimensions. Next, the algorithm determines *ε*-distance for each dimension as well as each subspace in high dimensional dataset. Another clustering algorithm S_FAD (Self-tuned FAD) was devised by Agarwal et al. [22] employing subspace clustering to address the challenges of high-dimensional data. This algorithm integrates self-tuned DBSCAN with FAD algorithm [50] in a bottom-up strategy using Apriori principle. S_FAD employs a self-tuned version of the DBSCAN algorithm that automatically determines *minPts* and *ε* based on the specific dataset. To optimize these parameters, S_FAD uses a hybrid algorithm called FAD, which is a combination of Flower Pollination (FP), Artificial Bee Colony (ABC), and Differential Evolution (DE). This amalgamation ensures a balance between exploring the entire search space and refining local solutions. Moreover, to ensure it only finds maximal and non-redundant clusters, S_FAD assigns unique 15-digit signatures to data points. Matching signatures in a hash table allows the algorithm to merge attributes into subspaces efficiently without comparing every data point individually.

Distance computations play a fundamental role in forming efficient clusters. In the context of DBSCAN, past scholarly works [44], [20], [21], [51], [26] have proposed several variations that can efficiently cluster data points in a high dimensional scenario by reducing unnecessary distance computations. This list includes a novel method TI-DBSCAN and its variant TI-DBSCAN-REF, introduced by Kryszkiewicz & Lasek [44]. This algorithm integrates triangle

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

inequality with DBSCAN to effectively find clusters. The triangular inequality property can effectively reduce the neighborhood search space, therefore speeding the clustering process. The clustering begins in a series of steps. The first step requires a dataset D, in which data points are arranged in descending order based on their distance from a reference point. Subsequently, it produces a label for the first cluster and then analyses each point in D. At last, each point is assigned to a certain cluster, whilst the other points are categorized as noise. The choice of reference point distinguishes TI-DBSAN from its variant TI-DBSCAN-REF, wherein a single reference point is utilised by TI-DBSCAN and multiple reference points are used by TI-DBSCAN-REF.

Another adaption of DBSCAN, Anytime-DBSCAN (A-DBSCAN) proposed by Mai et al. [45], incorporates lower-bounding (LB) functions to enhance the efficiency of the clustering process. LBs facilitate in quickly eliminating points that are not suitable to be a part of a cluster, thereby minimising the distance calculations. A-DBSCAN operates on several levels in relation to the sequence of LBs generated by the Discrete Wavelet Transform (DWT) and the outcome produced at each level is determined by the preceding level outcome. A-DBSCAN primarily relies on two algorithmic frameworks – (a) Effective distance upgrading, and (b) Localized re-clustering. For each level of LBs, the effective distance upgrading approach restricts distance computations to core objects. Localized re-clustering approach limits update operations to relevant objects only. Later, in, Zhang et al. [53] proposed an efficient approach, named Dboost, that minimises the density computations to accelerate the clustering speed of DBSCAN. Dboost employed an adaptation, WAND$^{\#}$, of the ranked retrieval method WAND [54], that could efficiently retrieve a broader neighborhood for a point p. This neighborhood is utilized to assess the density of p's ε-neighbors, hence reducing the global search procedure conducted for each member inside p's ε-neighborhood.

On similar lines, Li et al. [20] introduced a novel approach called Approximate DeNsity-Based ClustEring (ADvaNCE) which employed two approximations – (a) Locality Sensitive Hashing (LSH) and (b) Representative Points to enhance speed the clustering process. LSH approximates distance between points which facilitates a quick identification of the exact points lying in the ε-neighborhood. Representative Points limits the number of points within each cell so that time required for merging of clusters and identification of core points can be minimised. Likewise, a study by Okkels et al. [26] used LSH to handle high-dimensional data efficiently by constructing an approximate version of DBSCAN, SRRDBSCAN (Spherical Range Reporting DBSCAN). At the core of SRRDBSCAN is a multi-level LSH indexing structure that adapts to variations in data density. The algorithm operates in three main phases. First, it builds the LSH index, hashing data points multiple times at different levels and selecting the most suitable level for each point based on its local density, thereby reducing unnecessary distance computations. Next, the algorithm identifies core points by counting neighbors within a distance threshold (ε), restricting comparisons to points that fall into the same LSH buckets. Finally, density-reachable core points are merged using a union–find structure to form the final clusters.

Ding & Yang [47] developed an efficient algorithm that performs a coarse partitioning of the dataset using a randomized k-center clustering approach. This technique only selects the key points, neglecting the unnecessary ones, to represent initial clusters, that further enables the algorithm to perform range queries efficiently. The algorithm does so by assuming that inliers (core and border objects) have a low doubling dimension. Another adaptation of DBSCAN, called h-DBSCAN was proposed by Weng et al. [54] to minimize the neighbourhood search queries thereby improving the clustering quality. The algorithm does so by employing an approximate multi-layer graphic structure called Hierarchical Navigable Small World (HNSW) to identify the k-nearest neighbours. The algorithm employs a three-step procedure to cluster the points. First, the clustering begins by applying HNSW to create an initial cluster. This step results in core point and its ε-neighbor along with border points and noise.

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

In the second stage, these initial clusters are merged as well as the density-connected clusters. In the final step, the border points are filtered out that remained unassigned to any cluster in addition to the noise.

Jiang et al. [55] introduced a Partitioning-based DBSCAN with Ant Clustering (PACA-DBSCAN) that partitions data using two concepts: (a) Point Density (PD) and (b) modified Ant Clustering Algorithm (ACA) during the data preprocessing phase. Two-dimensional data was analyzed utilizing point density, while multi-dimensional data was handled with a modified ant clustering algorithm. Subsequently, for each partition, the PACA-DBSCAN algorithm works in a series of steps. The first step involves construction of a R*-tree to quickly fetch neighborhood queries, followed by plotting a *k*-dist graph to determine the ε parameter whereas *minPts* is set to 4. Subsequently, DBSCAN is executed using the estimated parameters to obtain partial clusters. Finally, PACA-DBSCAN combines the sufficiently proximate clusters from various partitions, thereby accurately reflecting the intrinsic structure of the data. This algorithm can handle clusters with varying densities. Likewise, Chen et al. [21] proposed a novel clustering algorithm NQ-DBSCAN, to prune unnecessary density computations. To do so, NQ-DBSCAN employed a quadtree-like hierarchical tree grid as an indexing technique, and a local neighbour query method as the neighbourhood searching technique. The points p and q are said to have similar neighbours, if, for a given ε, p and q are close to each other. The proximity between two points determines the similarity level of their neighbours. NQ-DBSCAN has superior performance in datasets characterised by noise and high dimensionality.

Later, in, Sarma et al. [51] developed a micro-cluster based DBSCAN algorithm, called µDBSCAN, which employed two strategies: µR-tree and reachability within micro-clusters to identify the core points. These techniques facilitate in minimizing the processing time of neighborhood queries. A parallelized variant of µDBSCAN, termed µDBSCAN-D, was developed to use distributed memory architecture for the efficient clustering of extensive datasets, accommodating up to one billion data points. Another study by Chen et al. [15] proposed an approximate variant of DBSCAN, called BLOCK-DBSCAN which can reduce the redundant distance computations in high dimensional datasets. For the purpose, BLOCK-DBSCAN employs two techniques: (a) Cover Tree and (b) $\frac{\varepsilon}{2} - norm\ ball$. The cover tree algorithm accelerates the process of computing the nearest neighbours within ε-neighbourhood of a point. The $\frac{\varepsilon}{2} - norm\ ball$ method was used to identify Inner Core Blocks with higher efficiency as it finds more core points at one time.

Later in [24], a novel approach, LAF-DBSCAN, was developed by Wang & Wang to accelerate the DBSCAN algorithm using Learned Accelerator Framework (LAF) by employing a cardinality estimator (α). This cardinality estimator is plugged in before each range query to select only the core point associated range queries for clustering. Further, a post-processing module rectifies the core points regarded as non-core or noise points and merges the wrongly separated clusters. This step compensates for the loss caused by prediction error in producing quality clusters. Additionally, the LAF is specifically optimized for cosine distance due to its bounded range (0 to 2) and its dominance in high-dimensional neural embeddings. The authors have utilized LAF in DBSCAN as well as its sampling variant DBSCAN++. Another study by Chowdhury et al. [13] integrated DBSCAN with reverse nearest neighbor and feature weighting to develop W-DBSCANR, which necessitates the estimation of two parameters, k and β. The algorithm initially categorizes each point in the dataset into one of three classifications: core, reachable, or outlier. Secondly, DBSCANR uses the current data partition to repeatedly adjust feature weights. The feature weights indicate the significance of each feature, facilitating a more generic approach to feature selection.

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

Xing & Zhao [48] developed an improvised version of DBSCAN, called Block-Diagonal guided DBSCAN (BD-DBSCAN) employing the block-diagonal property of similarity graphs to efficiently cluster high-dimensional data. The algorithm comprises three phases of the similarity graph: construction, permutation, and segmentation. The graph construction stage creates a similarity graph with high-dimensional points and solves the constrained optimization problem using a gradient descent method. This ensures that the resulting similarity graph inherently possesses a block-diagonal structure after an unknown permutation of the data points. Next, the graph permutation stage rearranges the similarity graph into a block-diagonal form by using a density-based cluster traversal algorithm. This algorithm uses a reachable similarity concept to handle varying cluster densities. The traversal algorithm systematically processes clusters and records the order, which is then used to permute the graph. Finally, the graph segmentation stage identifies the diagonal blocks in the permuted graph to determine the clusters.

The above section listed some previous scholarly works that attempted to alleviate the curse of dimensionality faced by the DBSCAN algorithm. These techniques applied several methods like LSH [20], [26], R*-tree [55], Cover Tree [15], Lower-bounding function [45], WAND$^{\#}$ [52], etc. to reduce unnecessary distance computations. Further, some algorithms utilised the concept of subspace clustering to deal with high dimensional data. Kailing et al. [17] integrated subspace clustering with density connectivity, [49] introduced the concept of ordered subspaces, Jahirabadkar & Kulkarni [46] applied *k*-NN distances to subspace clustering, and Agarwal et al. [22] combined FAD algorithm with self-tuned DBSCAN. These approaches had successfully addressed the issue of high dimensionality in DBSCAN, however, none of them worked on the hyperparameter tuning [17], [22], [24], [46]. Hence, the literature surrounding DBSCAN requires some works related to both parameter estimation and high dimensionality. The subsequent section clearly explains the methodologies employed by variants of DBSCAN to address these drawbacks.

3.4 Hybrid variants of DBSCAN

Addressing parameter sensitivity and high dimensionality independently has led to significant progress in enhancing the DBSCAN algorithm. However, a growing body of research seeks to tackle both issues simultaneously by combining parameter estimation techniques with high-dimensional data handling. These integrated approaches leverage graph-based models, feature selection, evolutionary algorithms, and deep learning to improve clustering accuracy and scalability. The subsequent section gives a detailed explanation of these algorithms.

Campello et al. [19] developed a novel density-based hierarchical clustering (HDBSCAN) algorithm employing *minPts* as a smoothing parameter for efficient clustering. The algorithm constructs a complete Mutual Reachability Graph that is utilized in constructing an extended Minimum Spanning Tree (MST). The extended MST incorporates self-edges for each point, with weights equal to their core distances and extracts the hierarchy by iteratively removing edges in decreasing order of weight. Each removal defines a hierarchical level and connected components become clusters or noise based on whether they retain edges. This potentially complex hierarchy can optionally be simplified by filtering out connected components deemed spurious based on a minimum cluster size, often set equal to *minPts*. Finally, to obtain a single, non-hierarchical result, an optimal flat partition is extracted from the (simplified) hierarchy by selecting clusters that maximize overall cluster stability.

Kumar & Reddy [56] proposed a Groups DBSCAN (G-DBSCAN) algorithm that employs a Groups technique to improve the effectiveness of nearest neighbor search queries by structuring the datapoints inside a graph-based framework. This graph contains vertices, where each vertex represents a group, and if two groups are mutually reachable, they are connected through an edge. Each group denotes a hypersphere, with a central point identified as the master pattern and the surrounding points categorized as slave patterns. The master

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

pattern has a maximum radius of ε. A slave pattern positioned at a distance of no more than ε from master pattern results in merging of the neighbouring patterns into clusters. The Groups approach ensures that, for a particular pattern to be included in a group, the neighborhood search confines point movement to a distance of $5*\varepsilon$. This provides an advantage over the original DBSCAN, which examines all points in the dataset.

Likewise, the Density-based Clustering using Graph Shared Neighbors and Entropy (DCSNE) algorithm [57] employed two concepts: (a) spare similarity graph and (b) information entropy to cluster high-dimensional datasets efficiently without user-specified parameters. DCSNE groups the data points into four distinct stages. The initial phase captures the local neighbourhood information of a point by constructing a spare similarity graph. This was done using Minimum Spanning Tree edges. The second step involves use of information entropy to obtain the density of each point. This effectively captures the random and disordered nature of the data points. In the third phase, smaller, disjoint density regions are formed by linking data points to their nearest neighbors of high density. Lastly, clusters are created using these smaller, denser regions by analysing their shared proximity and density distribution.

Garg et al. [58] developed a multi-stage model, BFA-DBSCAN, which incorporates feature selection, feature partitioning, and anomaly detection. The initial phase utilizes the Boruta algorithm for selecting relevant features which are further partitioned using an extended *k*-medoid (EKM) algorithm. The EKM algorithm requires *k*, the number of centroids, that was estimated using the Davies Bouldin Index (DBI)-based Firefly Inspired Partitioning (FIP) method. In the final phase, the partitioned dataset employed Extended partitioning-based DBSCAN (E-PDBSCAN) for anomaly detection, utilizing the Local Sensitive Hashing (LSH) technique and *k*-distance graph. The LSH method addresses the nearest neighbor search problem, with the parameters ε and *minPts* being automatically determined through the *k*-distance graph. Another novel algorithm FA+GA-DBSCAN [59] automatically cluster datasets without setting the input parameters of DBSCAN. The technique FA+GA-DBSCAN is a synthesis of Factor Analysis, Genetic Algorithm, and DBSCAN. This algorithm follows a step-wise procedure to cluster data points in high-dimensional spaces. The first step involves reducing the dimensions of the dataset by factor analysis followed by parameter estimation through genetic algorithm and nearest neighbour. Lastly, the DBSCAN algorithm was executed to group the data points.

Beer et al. [60] proposed a novel clustering algorithm called Structure-preserving High-dimensional Analysis with Density-based Exploration (SHADE) which autonomously clusters data points using a two-step methodology. The preliminary stage trains a deep autoencoder and introduces a novel loss function to capture density-connectivity from the original high-dimensional space. In the final phase, SHADE performs clustering directly in the low-dimensional embedded space by autonomously identifying the most stable clustering within the hierarchy without necessitating any user input. The most stable clustering is determined using a two-step process including bottom-up flagging and top-down cluster determination of a simplified tree structure. Next, the unallocated points are automatically identified as noise.

The game theory–based DBSCAN approach [25] was proposed as an extension of the classical DBSCAN algorithm to enhance parameter selection, improve clustering accuracy, and accelerate processing for high-dimensional big data. The method integrates game-theoretic concepts, particularly Nash equilibrium and dense games, to automatically determine optimal clustering parameters and to effectively distinguish noise from meaningful clusters. The clustering procedure is carried out in four stages: (1) partitioning the data space into grids, (2) identifying game players to estimate the ε parameter, (3) forming the initial clusters based on the game outcomes, and (4) merging these clusters to obtain the final clustering result.

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

These hybrid approaches illustrate the growing convergence of parameter estimation, dimensionality reduction, and density-based clustering. By blending traditional DBSCAN mechanisms with graph theory [56], entropy [57], feature selection and extraction algorithms [58], [59], game theory [25], and deep learning [60], these methods offer scalable, automated solutions for complex datasets. While no single algorithm fully solves all challenges, they mark significant progress toward robust and adaptable clustering techniques for modern data environments.

4. CONCLUSION AND FUTURE DIRECTIONS

The exponential growth of data over the past few years have posed significant challenges on its evaluation. The machine learning (ML) techniques working well in low-dimensional contexts are unable to perform simpler tasks on high-dimensional datasets. Thus, the efficiency of the existing algorithms is minimised and computational costs have increased. Clustering, an unsupervised ML technique, is incapable of handling high-dimensional datasets. In high-dimensional spaces, the clusters are hidden by irrelevant features or noise, making incorrect interpretations about true clusters. DBSCAN, as a density-based clustering algorithm, has seen notable developments in recent years owing to its multiple challenges. Although DBSCAN excels at identifying arbitrarily shaped clusters and handling noisy data, its reliance on sensitive input parameters and inefficiency in high-dimensional spaces limit its broader applicability. This review has systematically explored and categorized numerous modifications of DBSCAN related to parameter estimation and high dimensionality along with a comprehensive overview of several clustering methodologies, including Reverse Nearest Neighbor (RNN), Local Sensitive Hashing (LSH), Density Layer Tree (DLT), Cover Tree, and Triangular Inequality, utilized for estimating DBSCAN parameters or tackling high-dimensionality issues.

Conventional machine learning methods have long prevailed in the domain of clustering; nevertheless, deep learning techniques are emerging as powerful tools for accurately detecting clusters within complex datasets, necessitating little to no parameters. Unlike ML techniques, deep learning algorithms exhibit superior scalability in high-dimensional contexts, effectively manage complex data structures, and provide enhanced speed, efficiency, adaptability, and flexibility across a broader range of applications. Recent studies have begun integrating deep learning with density-based clustering, offering promising new avenues for scalability and automatic parameter tuning.

Despite these advancements, none of these algorithms have offered a comprehensive solution to the shortcomings inherent in DBSCAN. Most solutions are tailored to specific data characteristics or application domains, and many still struggle with trade-offs between computational efficiency, accuracy, and interpretability. As real-world datasets continue to grow in scale and complexity, future research should prioritize the development of generalized, automated clustering frameworks that combine the strengths of multiple paradigms. Emerging directions such as deep learning, parallel and distributed clustering, manifold-based representations, and multi-label feature selection offer promising avenues. These techniques have the potential to bridge existing gaps and enable clustering methods to scale efficiently, and minimize user intervention—paving the way for more intelligent and robust unsupervised learning solutions.

5. FUTURE SCOPE

The present survey concentrates on DBSCAN variants that address parameter sensitivity and high dimensionality. The future scope may encompass the following aspects:

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

- Subsequent surveys may broaden their coverage by examining other inherent limitations of DBSCAN beyond those addressed in this work, offering a more comprehensive understanding of the algorithm's shortcomings.
- Adopting a systematic literature review methodology in future studies would ensure a more structured, protocol-driven, and reproducible synthesis of the existing literature compared to the conventional survey approach employed here.
- As a natural extension of this survey, future research may explore the integration of deep learning architectures with clustering methods and benchmark their performance against conventional machine learning techniques to assess comparative effectiveness.
- The evaluation of DBSCAN variants and their respective modifications may be extended to a wider spectrum of real-world application domains, including but not limited to healthcare, cybersecurity, remote sensing, and social network analysis, to better assess their generalizability and practical utility.
- Future work may also encompass the establishment of standardized evaluation protocols and benchmark datasets specifically tailored for assessing the performance of clustering algorithms across datasets of varying dimensionalities and noise levels, facilitating more objective and consistent comparisons.

ACKNOWLEDGMENTS

The authors declare that no financial or institutional support was received for this research.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

REFERENCES:

[1] Agapito G, Milano M, Cannataro M (2022) A Python clustering analysis protocol of genes expression data sets. *Genes* 13(10):1839. https://doi.org/10.3390/genes13101839

[2] Wang X, et al. (2020) Electricity market customer segmentation based on DBSCAN and k-means: A case on Yunnan electricity market. *Proc Asia Energy and Electrical Engineering Symposium (AEEES)*:869–874. https://doi.org/10.1109/AEEES48850.2020.9121413

[3] Hou J, Liu W, E X, Cui H (2016) Towards parameter-independent data clustering and image segmentation. *Pattern Recognition* 60:25–36. https://doi.org/10.1016/j.patcog.2016.04.015

[4] Ananthi VP, Balasubramaniam P, Kalaiselvi T (2016) A new fuzzy clustering algorithm for the segmentation of brain tumor. *Soft Computing* 20(12):4859–4879. https://doi.org/10.1007/s00500-015-1775-5

[5] Du Q, Dong Z, Huang C, Ren F (2016) Density-based clustering with geographical background constraints using a semantic expression model. *ISPRS International Journal of Geo-Information* 5(5):72. https://doi.org/10.3390/ijgi5050072

[6] Kowalski M, et al. (2008) Improved cosmological constraints from new, old, and combined supernova data sets. *Astrophysical Journal* 686(2):749–778. https://doi.org/10.1086/589937

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

[7] Hancer E, Xue B, Zhang M (2020) A survey on feature selection approaches for clustering. *Artificial Intelligence Review* 53(6):4519–4545. https://doi.org/10.1007/s10462-019-09800-w

[8] Ester M, Kriegel HP, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. *Proc KDD*:226–231.

[9] Ankerst M, Breunig MM, Kriegel HP (1999) OPTICS: Ordering points to identify the clustering structure. *ACM SIGMOD Record* 28(2):49–60.

[10] Hinneburg A, Keim DA (1998) An efficient approach to clustering in large multimedia databases with noise. *Proc KDD*:58–65.

[11] Silverman BW (1998) *Density estimation for statistics and data analysis*. New York: Chapman & Hall.

[12] Schubert E, Sander J, Ester M, Kriegel HP, Xu X (2017) DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Transactions on Database Systems* 42(3):1–21. https://doi.org/10.1145/3068335

[13] Chowdhury S, Helian N, Cordeiro de Amorim R (2023) Feature weighting in DBSCAN using reverse nearest neighbours. *Pattern Recognition* 137:109314. https://doi.org/10.1016/j.patcog.2023.109314

[14] Hou J, Gao H, Li X (2016) DSets-DBSCAN: A parameter-free clustering algorithm. *IEEE Transactions on Image Processing* 25(7):3182–3193. https://doi.org/10.1109/TIP.2016.2559803

[15] Chen Y, Zhou L, Bouguila N, Wang C, Chen Y, Du J (2021) BLOCK-DBSCAN: Fast clustering for large scale data. *Pattern Recognition* 109:107624. https://doi.org/10.1016/j.patcog.2020.107624

[16] Wu F, Gardarin G (2001) Gradual clustering algorithms. *Proc DASFAA*:48–55. https://doi.org/10.1109/DASFAA.2001.916364

[17] Kailing K, Kriegel HP, Kröger P (2004) Density-connected subspace clustering for high-dimensional data. *SIAM Proceedings Series*:246–256. https://doi.org/10.1137/1.9781611972740.23

[18] Fahim AM, Salem AM, Torkey FA, Ramadan MA (2006) Density clustering based on radius of data. *International Journal of Applied Mathematics and Computer Science* 3(2).

[19] Campello RJGB, Moulavi D, Zimek A, Sander J (2015) Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Transactions on Knowledge Discovery from Data* 10(1):1–51. https://doi.org/10.1145/2733381

[20] Li T, Heinis T, Luk W (2017) ADvaNCE-efficient and scalable approximate density-based clustering based on hashing. *Informatica* 28(1):105–130. https://doi.org/10.15388/Informatica.2017.122

[21] Chen Y, Tang S, Bouguila N, Wang C, Du J, Li H (2018) A fast clustering algorithm based on pruning unnecessary distance computations in DBSCAN for high-dimensional data. *Pattern Recognition* 83:375–387. https://doi.org/10.1016/j.patcog.2018.05.030

[22] Agarwal P, Mehta S, Abraham A (2021) A meta-heuristic density-based subspace clustering algorithm for high-dimensional data. *Soft Computing* 25(15):10237–10256. https://doi.org/10.1007/s00500-021-05973-1

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

[23] Bataineh B, Alzahrani AA (2023) Fully automated density-based clustering method. *Computers, Materials & Continua* 76(2):1833–1851. https://doi.org/10.32604/cmc.2023.039923

[24] Wang Y, Wang DZ (2023) Learned accelerator framework for angular-distance-based high-dimensional DBSCAN. *Proc EDBT*. https://doi.org/10.48786/EDBT.2023.42

[25] Kazemi U, Soleimani S (2025) A new approach data processing: DBSCAN clustering using game-theory. *Soft Computing* 29(3):1331–1346. https://doi.org/10.1007/s00500-025-10405-5

[26] Okkels CB, Aumüller M, Thomsen VB, Zimek A (2025) High-dimensional density-based clustering using locality-sensitive hashing. *Proc EDBT*. https://doi.org/10.48786/EDBT.2025.56

[27] Xu X, Ester M, Kriegel HP, Sander J (1998) A distribution-based clustering algorithm for mining in large spatial databases. *Proc ICDE*:324–331. https://doi.org/10.1109/ICDE.1998.655795

[28] Mustapha SMFDS (2023) An alternative parameter free clustering algorithm using data point positioning analysis. *International Journal of Innovative Computing, Information and Control* 19(6):1805–1825. https://doi.org/10.24507/ijicic.19.06.1805

[29] Yang XH, Jin LB, Ye W, Xiao J, Zhang D, Xu XL (2018) Laplacian centrality peaks clustering based on potential entropy. *IEEE Access* 6:55462–55472. https://doi.org/10.1109/ACCESS.2018.2871500

[30] Abdulhameed TZ, Yousif SA, Samawi VW, Al-Shaikhli HI (2024) SS-DBSCAN: Semi-supervised density-based spatial clustering of applications with noise. *IEEE Access* 12:131507–131520. https://doi.org/10.1109/ACCESS.2024.3457587

[31] Darong H, Peng W (2012) Grid-based DBSCAN algorithm with referential parameters. *Physics Procedia* 24:1166–1170. https://doi.org/10.1016/j.phpro.2012.02.17

[32] Starczewski A, Goetzen P, Er MJ (2020) A new method for automatic determining of the DBSCAN parameters. *Journal of Artificial Intelligence and Soft Computing Research* 10(3):209–221. https://doi.org/10.2478/jaiscr-2020-0014

[33] Cassisi C, Ferro A, Giugno R, Pigola G, Pulvirenti A (2013) Enhancing density-based clustering: Parameter reduction and outlier detection. *Information Systems* 38(3):317–330. https://doi.org/10.1016/j.is.2012.09.001

[34] Kim JH, Choi JH, Yoo KH, Nasridinov A (2019) AA-DBSCAN: An approximate adaptive DBSCAN for finding clusters with varying densities. *Journal of Supercomputing* 75(1):142–169. https://doi.org/10.1007/s11227-018-2380-z

[35] Ros F, Guillaume S, Riad R, El Hajji M (2022) Detection of natural clusters via S-DBSCAN. *Knowledge-Based Systems* 241:108288. https://doi.org/10.1016/j.knosys.2022.108288

[36] Cheng D, et al. (2024) GB-DBSCAN: A fast granular-ball based DBSCAN clustering algorithm. *Information Sciences* 674:120731. https://doi.org/10.1016/j.ins.2024.120731

[37] Lv Y, et al. (2016) An efficient and scalable density-based clustering algorithm for datasets with complex structures. *Neurocomputing* 171:9–22. https://doi.org/10.1016/j.neucom.2015.05.109

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

[38] Bryant A, Cios K (2018) RNN-DBSCAN: A density-based clustering algorithm using reverse nearest neighbor density estimates. *IEEE Transactions on Knowledge and Data Engineering* 30(6):1109–1121. https://doi.org/10.1109/TKDE.2017.2787640

[39] Dai QZ, Xiong ZY, Xie J, Wang XX, Zhang YF, Shang JX (2019) A novel clustering algorithm based on the natural reverse nearest neighbor structure. *Information Systems* 84:1–16. https://doi.org/10.1016/j.is.2019.04.001

[40] Uncu O, Gruver WA, Kotak DB, Sabaz D, Alibhai Z, Ng C (2006) GRIDBSCAN: Grid density-based spatial clustering of applications with noise. *Proc IEEE SMC*:2976–2981. https://doi.org/10.1109/ICSMC.2006.384571

[41] Latifi-Pakdehi A, Daneshpour N (2021) DBHC: A DBSCAN-based hierarchical clustering algorithm. *Data & Knowledge Engineering* 135:101922. https://doi.org/10.1016/j.datak.2021.101922

[42] Li H, Liu X, Li T, Gan R (2020) A novel density-based clustering algorithm using nearest neighbor graph. *Pattern Recognition* 102:107206. https://doi.org/10.1016/j.patcog.2020.107206

[43] Li M, Bi X, Wang L, Han X (2021) A method of two-stage clustering learning based on improved DBSCAN and density peak algorithm. *Computer Communications* 167:75–84. https://doi.org/10.1016/j.comcom.2020.12.019

[44] Kryszkiewicz M, Lasek P (2010) TI-DBSCAN: Clustering with DBSCAN by means of the triangle inequality. *Rough Sets and Current Trends in Computing* 6086:60–69. https://doi.org/10.1007/978-3-642-13529-3_8

[45] Mai ST, He X, Feng J, Böhm C (2013) Efficient anytime density-based clustering. *Proc SIAM SDM*:112–120. https://doi.org/10.1137/1.9781611972832.13

[46] Jahirabadkar S, Kulkarni P (2014) Algorithm to determine ε-distance parameter in density based clustering. *Expert Systems with Applications* 41(6):2939–2946. https://doi.org/10.1016/j.eswa.2013.10.025

[47] Ding H, Yang F (2020) On metric DBSCAN with low doubling dimension. *arXiv preprint arXiv:2002.11933*.

[48] Xing Z, Zhao W (2024) Block-diagonal guided DBSCAN clustering. *IEEE Transactions on Knowledge and Data Engineering* 36(11):5709–5722. https://doi.org/10.1109/TKDE.2024.3401075

[49] Liu K, Zhou D, Zhou X (2009) Clustering by ordering density-based subspaces. *Proc Visual Data Mining Workshop*:197–201.

[50] Agarwal P, Mehta S (2019) ABC_DE_FP: A novel hybrid algorithm for complex continuous optimisation problems. *International Journal of Bio-Inspired Computation* 14(1):46–61.

[51] Sarma A, et al. (2019) μDBSCAN: An exact scalable DBSCAN algorithm for big data exploiting spatial locality. *Proc IEEE CLUSTER*:1–11. https://doi.org/10.1109/CLUSTER.2019.8891020

[52] Zhang Y, Wang X, Li B, Chen W, Wang T, Lei K (2016) Dboost: A fast algorithm for DBSCAN-based clustering on high dimensional data. *Advances in Knowledge Discovery and Data Mining* 9652:245–256. https://doi.org/10.1007/978-3-319-31750-2_20

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/s5y5tk64

[53] Broder AZ, Carmel D, Herscovici M, Soffer A, Zien J (2003) Efficient query evaluation using a two-level retrieval process. *Proc CIKM*:426–434. https://doi.org/10.1145/956863.956944

[54] Weng S, Gou J, Fan Z (2021) h-DBSCAN: A simple fast DBSCAN algorithm for big data. *Proceedings of Machine Learning Research*:81–96.

[55] Jiang H, Li J, Yi S, Wang X, Hu X (2011) A new hybrid method based on partitioning-based DBSCAN and ant clustering. *Expert Systems with Applications* 38(8):9373–9381. https://doi.org/10.1016/j.eswa.2011.01.135

[56] Kumar KM, Reddy ARM (2016) A fast DBSCAN clustering algorithm by accelerating neighbor searching using groups method. *Pattern Recognition* 58:39–48. https://doi.org/10.1016/j.patcog.2016.03.008

[57] Maheshwari R, Mohanty SK, Mishra AC (2023) DCSNE: Density-based clustering using graph shared neighbors and entropy. *Pattern Recognition* 137:109341. https://doi.org/10.1016/j.patcog.2023.109341

[58] Garg S, Kaur K, Batra S, Kaddoum G, Kumar N, Boukerche A (2020) A multi-stage anomaly detection scheme for augmenting security in IoT-enabled applications. *Future Generation Computer Systems* 104:105–118. https://doi.org/10.1016/j.future.2019.09.038

[59] Perafan-Lopez JC, Ferrer-Gregory VL, Nieto-Londoño C, Sierra-Pérez J (2022) Performance analysis and architecture of a clustering hybrid algorithm called FA+GA-DBSCAN. *Entropy* 24(7):875. https://doi.org/10.3390/e24070875

[60] Beer A, et al. (2024) SHADE: Deep density-based clustering. *Proc IEEE ICDM*:675–680. https://doi.org/10.1109/ICDM59182.2024.00075