WebLog Journal of Aviation

Research Article
Published: 09 May, 2026

Intelligent Image and Graphics Processing by DBSCAN Method

Mahabub Sultan[1] and Nikunjun Sarker Akash[2]*

[1]*Buffalo State University, Buffalo, USA*

[2]*School of Aeronautics, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, China*

WebLog Open Access Publications
Article ID : wjav.2026.e0902
Author : Mahabub Sultan

Abstract

DBSCAN is a popular method for clustering multi-dimensional objects. Just as notable as the method's vast success is the research community's quest for its efficient computation. The original KDD'96 paper claimed an algorithm with O(n log n) running time, where n is the number of objects. Unfortunately, this is a miss claim; and that algorithm actually requires O(n2) time. There has been a fix in 2D space, where a genuine O(n log n)-time algorithm has been found. Looking for a fix for dimensionality d ≥ 3 is currently an important open problem. In this report, we prove that for d ≥ 3, the DBSCAN problem requires (n4/3) time to solve, unless very significant breakthroughs - ones widely believed to be impossible - could be made in theoretical computer science. This (i) explains why the community's search for fixing the aforementioned mis-claim has been futile for d ≥ 3, and (ii) indicates (sadly) that all DBSCAN algorithms must be intolerably slow even on moderately large n in practice. Surprisingly, we show that the running time can be dramatically brought down to O(n) in expectation regardless of the dimensionality d, as soon as slight inaccuracy in the clustering results is permitted. We formalize our findings into the new notion of ρ-approximate DBSCAN, which we believe should replace DBSCAN on big data due to the latter's computational intractability.

Introduction

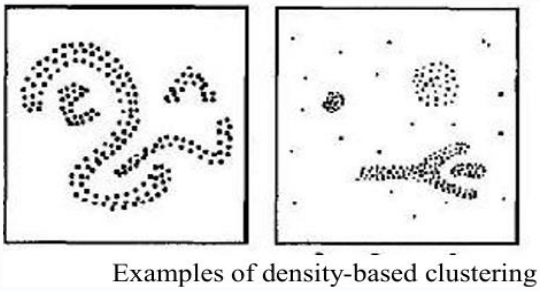
Density-based clustering is one of the most fundamental topics in data mining. Given a set P of n points in d-dimensional space Rd, the objective is to group the points of P into subsets - called clusters - such that any two clusters are separated by "sparse regions" (Figure 1).

The left one contains 4 snake-shaped clusters, while the right one contains 3 clusters together with some noise. The main advantage of density-based clustering (over methods such as k-means) is its capability of discovering clusters with arbitrary shapes (while k-means typically returns ball-like clusters).

Density-based clustering can be achieved using a variety of approaches, which differ mainly in their (i) definitions of "dense/sparse regions", and (ii) criteria of how dense regions should be connected to form clusters. In this report, we concentrate on DBSCAN, which is an elegant approach invented by Ester, Kriegel, Sander, and Xu, and received the test-of-time award in KDD'14. DBSCAN characterizes "density/sparsity" by resorting to two parameters.

Our Contributions

This report makes three contributions. First, we prove that the DBSCAN problem requires (n4/3) time to solve in d ≥3, unless very significant breakthroughs (ones widely believed to be impossible) can



Examples of density-based clustering

Figure 1:

OPEN ACCESS

Correspondence:
Mahabub Sultan, SUNY Buffalo State University, Buffalo, New York, United States,
E-mail: mahabubsultan2@gmail.com

Received Date: *10 Apr 2026*
Accepted Date: *07 May 2026*
Published Date: *09 May 2026*

Citation:
Sultan M, Akash NS. Intelligent Image and Graphics Processing by DBSCAN Method. WebLog J Aviat. wjav.2026.e0902. https://doi.org/10.5281/zenodo.20139361

be made in theoretical computer science. Note that n4/3 is arbitrarily larger than n logc n, regardless of constant c. Second, we introduce a new concept called ρ-approximate DBSCAN as an alternative to DBSCAN on large datasets of d ≥ 3. Approximate DBSCAN comes with very strong assurances in both quality and efficiency. For quality, its clustering result is guaranteed to be "sandwiched" between the results of DBSCAN obtained with parameters (ϱ,MinPts) and (ϱ(1 +ρ),MinPts), respectively. This is much desired in practice, because it is well-known [2] that there is a comfortable range of ϱ that will yield good DBSCAN clusters.

For efficiency, we prove that ρ-approximate DBSCAN can be solved in linear O(n) expected time, for any ϱ, arbitrarily small constant ρ, and in any fixed dimensionality d! It is rather surprising that such a small sacrifice of accuracy can bring this tremendous gain in running time. Third, we perform DBSCAN experiments on datasets significantly larger than those used in all the previous experiments to our awareness. The experiments reveal that, as predicted by theory, none of the exact DBSCAN algorithms has acceptable running time even in 3D (which perhaps explains why the previous evaluation was done only on small datasets). In contrast, our new ρ-approximate DBSCAN algorithm exhibits graceful scalability with respect to all parameters, and outperforms even the fastest exact algorithm by a factor up to three orders of magnitude.

Related Work

As before, let P be a set of n points in d-dimensional space Rd. Given two points p, $q \in$ Rd, we denote by dist (p, q) the Euclidean distance between p and q. Denote by B(p, r) the ball centered at a point $p \in$ Rd with radius r.

Remember that DBSCAN takes two parameters: ϱ and MinPts.

Definition 1 A point p ∈ P is a core point if B(p, ϱ) covers at least MinPts points of P (including p itself). If p is not a core point, it is said to be a non-core point. To illustrate, suppose that P is the set of points in Figure, where MinPts = 4 and the two circles have radius ϱ. Core points are shown in black, and non-core points in white.

DEFINITION 2 A cluster C is a non-empty subset of P such that:

- (Maximality) If a core point p ∈ C, then all the points density-reachable from p also belong to C (Figure 2).

ρ Approximate DBSCAN

The hardness result in Theorem 1 implies that DBSCAN is feasible only in 2D space. Even for d = 3, the computation time becomes strongly polynomial to n such that it will take an intolerably long period of time to calculate the clusters precisely. This opens the door to studying approximate DBSCAN.

DEFINITION 4 A point q ∈ P is ρ-approximate density-reachable from p ∈ P if there is a sequence of points p1, p2, ..., pt ∈ P (for some integer t ≥ 2) such that:

p1 = p and pt = q

p1, p2, ..., Pt−1 are core points

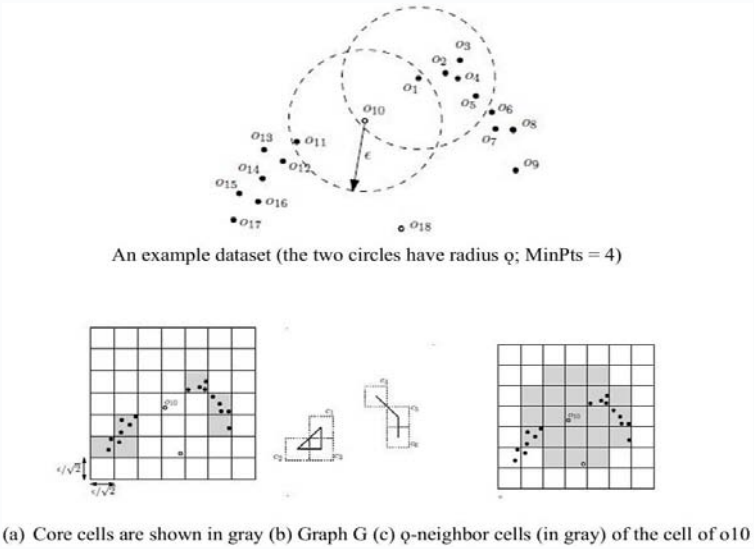
pi+1 ∈ B(pi, **ϱ**(1 + ρ)) for each i ∈ [1, t − 1].

Experiments

We now present an empirical evaluation of the proposed techniques. All the experiments were run on a machine equipped with 3.2GHz CPU and 8 GB memory. The operating system was Linux (Ubuntu 13.04). All the programs were coded in C++, and compiled using g++ with -o3 turned on.

Datasets

Except in a single experiment (for visualization), we focused on dimensionality d ≥ 3 because the 2D case has been well solved in [11]. In all cases, the underlying data space had a normalized domain of [0, 105] for every dimension. We deployed both synthetic and real datasets whose details are explained next. Synthetic: Seed Spreader (SS). A synthetic dataset was generated in a "random walk with restart" fashion. First, fix the dimensionality d, take the target cardinality n, a restart probability ρ restart, and a noise percentage ρ noise. Then, we simulate a seed spreader that moves about in the space, and spits out data points around its current location. The spreader carries a local counter such that whenever the counter reaches 0, the spreader moves a distance of r shift towards a random direction, after which the counter is reset to c reset. The spreader works in steps. In each step, (i) with probability ρ restart, the spreader restarts, by jumping to a random location in the data space, and resetting its counter to c reset;



An example dataset (the two circles have radius ϱ; MinPts = 4)

(a) Core cells are shown in gray (b) Graph G (c) ϱ-neighbor cells (in gray) of the cell of o10

Figure 2:

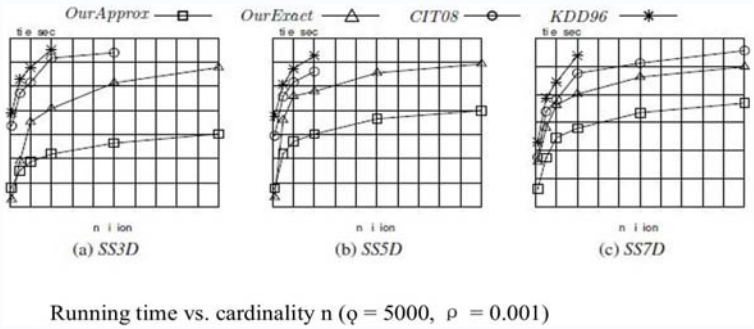


Figure 3:

(ii) no matter if a restart has happened, the spreader produces a point uniformly at random in the ball centered at its current location with radius 100, after which the local counter decreases by 1. Intuitively, every time a restart happens, the spreader begins to generate a new cluster. In the first step, a restart is forced so as to put the spreader at a random location. We repeat in total n(1−ρ noise) steps, which generate the same number of points.

Computational efficiency

We now proceed to inspect the running time of DBSCAN clustering using four algorithms:

• KDD96 [10]: the original DBSCAN algorithm in [10];

• CIT08 [17]: the state of the art of exact DBSCAN, namely, the fastest existing algorithm able to produce the same DBSCAN result as KDD96;

• Our Exact: the exact DBSCAN algorithm we developed in Theorem 2;

• Our Approx.: The ρ-approximate DBSCAN algorithm we proposed in Theorem 4.

Scalability with Cardinality n the first experiment examines how each method scales with the number n objects. For this purpose, we used synthetic SS datasets of 3D, 5D, and 7D by varying n from 100k to 10m. If KDD96 and CIT08 do not have results at a value of n, it means that they did not terminate within 12 hours in the corresponding experiments! Our Exact managed to finish within 104 seconds (less than 3 hours) even on the largest dataset. However, this is dwarfed by the superb efficiency of Our Approx. which took less than 490 seconds in all cases, and were often faster than Our Exact by a factor of two orders of magnitude. As an interesting note, all methods were fast when the dataset was small, e.g., when n = 100k. This is perhaps the reason why most (if not all) of the previous evaluation of DBSCAN algorithms, as far as we are aware, was on datasets of this scale (Figure 3).

Conclusions

DBSCAN is a creative, elegant, and effective technique for density-based clustering, which is very extensively applied in data mining, machine learning, and databases. However, all the existing DBSCAN algorithms have poor scalability with the dataset size in dimensionality d ≥ 3. This is unfortunate because clustering in d ≥ 3 is important, due to the frequent need of using multiple features to accurately model an object. In this report, we explain rigorously why the community's search for a fast DBSCAN algorithm for d≥3 has been unsuccessful. We show that, unless very significant breakthroughs (ones widely believed to be impossible) can be made in theoretical computer science, the DBSCAN algorithm requires (n4/3) time to solve, where n is the size of the underlying dataset. This strongly polynomial complexity essentially states that DBSCAN is computationally intractable in practice, even for moderately large n! This is very disappointing especially given the arrival of the big data era. Motivated by this, we propose the novel concept of ρ-approximate DBSCAN, which is designed to replace DBSCAN on large-scale data. We prove both theoretical and experimentally that ρ-approximate DBSCAN has excellent guarantees both in the quality of cluster approximation and computational efficiency. In fact, almost in all scenarios, it returns exactly the same clusters as DBSCAN but requires computation time only linear to n.

References

1. Agarwal PK, Edelsbrunner H, Schwarzkopf O. Euclidean minimum spanning trees and bichromatic closest pairs. Discrete Comput Geom. 1991; 6: 407-422.
2. Ankerst M, Breunig MM, Kriegel HP, Sander J. OPTICS: Ordering points to identify the clustering structure. SIGMOD. 1999.
3. Arya S, Mount DM. Approximate range searching. Comput Geom. 2000; 17: 135-152.
4. Bache K, Lichman M. UCI machine learning repository, 2013.
5. Böhm C, Kailing K, Kröger P, Zimek A. Computing clusters of correlation connected objects. SIGMOD. 2004.
6. Borah B, Bhattacharyya DK. An improved sampling-based DBSCAN for large spatial databases. Proceedings of Intelligent Sensing and Information Processing. 2004.
7. Chaoji V, Hasan MA, Salem S, Zaki MJ. Sparcl: Efficient and effective shape-based clustering. ICDM. 2008.
8. Erickson J. On the relative complexities of some geometric problems. CCCG. 1995.
9. Erickson J. New lower bounds for Hopcroft's problem. Disc Comp Geom. 1996; 16(4):389-418.
10. Ester M, Kriegel H, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. SIGKDD. 1996.
11. Gunawan A. A faster algorithm for DBSCAN. Technische University Eindhoven. 2013.
12. Han J, Kamber M, Pei J. Data Mining: Concepts and Techniques. Morgan Kaufmann. 2012.
13. Klusch M, Lodi S, Moro G. Distributed clustering based on sampling local density estimates. IJCAI. 2003.
14. Li Z, Ding B, Han J, Kays R. Swarm: Mining relaxed temporal moving

object clusters. PVLDB. 2010; 3(1): 723-734.

15. Liu B. A fast density-based clustering algorithm for large databases. Proceedings of International Conference on Machine Learning and Cybernetics. 2006.
16. Lu ECH, Tseng VS, Yu PS. Mining cluster-based temporal mobile sequential patterns in locationbased service environments. TKDE. 2011; 23(6): 914-927.
17. Mahran S, Mahar K. Using grid for accelerating density-based clustering. CIT. 2008.
18. Matousek J. Range searching with efficient hiearchical cutting. Disc Comp Geom. 1993; 10: 157-182.
19. Milenova BL, Campos MM. O-Cluster: Scalable clustering of large high dimensional data sets. ICDM. 2002.
20. Pal SK, Mitra P. Pattern Recognition Algorithms for Data Mining. Chapman and Hall. 2004.