The First International Conference on
Machine Learning and Knowledge Discovery (MLKD 2024)
Amirkabir University of Technology, December 18-19, 2024

Enhanced MOGA-DBSCAN: Improving Clustering Quality with a Density-Adjusted Outlier Index

Hossein Eyvazi* Ali Rajaei†

Abstract

Clustering is a crucial aspect of data mining and machine learning, and its performance can significantly depend on parameter selection. The DBSCAN algorithm, known for its efficacy in detecting clusters of arbitrary shapes, relies heavily on its two parameters: *Eps* and *MinPts*. This paper presents an enhanced version of the Multi-Objective Genetic Algorithm (MOGA) for optimizing the parameters of DBSCAN, named Enhanced MOGA-DBSCAN. Our approach incorporates a modified Outlier Index that accounts for the density of clusters, providing a better evaluation of outliers. Additionally, we parallelized the computation of the Outlier Index to significantly reduce the runtime, enabling practical applicability to larger datasets. Experimental results on two benchmark datasets demonstrate that Enhanced MOGA-DBSCAN outperforms the original MOGA-DBSCAN algorithm, achieving higher Silhouette scores and Rand indices while requiring less computational time. This advancement not only improves clustering efficiency but also offers more meaningful insights into the underlying data structure.

1 Introduction

Clustering is a fundamental task in data mining and machine learning [1], which involves grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other than to those in other groups. DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a widely used clustering algorithm known for its ability to find clusters of arbitrary shape and its effectiveness in detecting outliers [2]. However, the performance of DBSCAN is highly dependent on two critical parameters: *Eps* (the radius around a point) and *MinPts* (the minimum number of points required to form a dense region). Selecting appropriate values for these parameters is a challenging problem, particularly in the absence of prior knowledge about the dataset's structure [3].

Multi-Objective Genetic Algorithms (MOGA) have been successfully applied to optimize the parameters of DBSCAN by treating the clustering task as a multi-objective optimization problem. MOGA-DBSCAN leverages genetic algorithms to explore a wide range of parameter settings, optimizing two or more objective functions simultaneously. In this context, the Silhouette index, which measures the compactness and separation of clusters, is used as one of the objective functions. Another objective function considered is the Outlier Index, which focuses on the degree of separation between detected outliers and the clusters [4].

Despite the effectiveness of the original Outlier Index, it treats all clusters equally without considering their density, which can lead to suboptimal results. For instance, an outlier that is equally distant from two clusters, one dense and the other sparse, would be penalized the same way, regardless of the cluster's density. This approach fails to account for the fact that outliers are more significant when they are far from dense clusters compared to sparse ones.

In this paper, we propose an enhanced version of the MOGA-DBSCAN algorithm [4], which incorporates a modified Outlier Index that accounts for cluster density. Our Enhanced Outlier Index provides a more nuanced evaluation by scaling the distance of outliers from clusters based on the density of the clusters. To further improve the algorithm's practicality, we parallelized the computation of the Outlier Index, significantly reducing the runtime and making the algorithm more suitable for larger datasets. This improvement addresses the limitations of the original Outlier Index and leads to more accurate and meaningful clustering results, especially in datasets with varying cluster densities.

The proposed Enhanced MOGA-DBSCAN algorithm is evaluated on two benchmark datasets, demonstrating its superiority not only in terms of clustering quality and outlier detection but also in computational efficiency. In the following sections, we detail the methodology behind the Enhanced MOGA-DBSCAN, present experimental results, and discuss the implications of our findings in the context of density-based clustering.

2 Methodology

The proposed Enhanced MOGA-DBSCAN algorithm aims to optimize the DBSCAN clustering process by framing it as a multi-objective optimization problem. The algorithm utilizes a genetic algorithm to explore

*Department of Computer Science, Tarbiat Modares University, eyvazi_hoseyn@modares.ac.ir
†alirajaei@ac.ir

various combinations of the DBSCAN parameters *Eps* and *MinPts*, with the objective of maximizing two metrics: the Silhouette index and the Enhanced Outlier Index.

2.1 Enhanced Outlier Index

The original Outlier Index computes the average minimum distance of outliers to the nearest clusters, but it does not account for cluster density [5]. This limitation can lead to inaccurate results, especially in datasets with clusters of varying densities. To overcome this, we propose an Enhanced Outlier Index that integrates cluster density into its computation, while addressing two key challenges: (1) improving computational efficiency through parallelization, and (2) normalizing densities to handle very sparse or very dense clusters effectively.

The Enhanced Outlier Index is defined as follows:

$$\frac{1}{n}\sum_{i=1}^{n}\left(\min_{1\le j\le m}(d_{ij})\times \text{Normalized Density}_j\right), \quad (1)$$

where:

- n is the number of outliers.
- d_{ij} is the distance of outlier i from cluster j.
- m is the number of clusters.
- Density_j is the density of cluster j, computed as the inverse of the average distance from the centroid of cluster j.
- $\text{Normalized Density}_j$ is the density of cluster j, normalized relative to the densities of all clusters, and is computed as:

$$\frac{\text{Density}_j - \min(\text{Density}_k)}{\max(\text{Density}_k) - \min(\text{Density}_k)} + \epsilon \quad (2)$$

  where $k \in \{1, 2, \ldots, m\}$ represents all clusters.

2.1.1 Rationale for Normalization

Normalization ensures that all cluster densities lie within the range $[0, 1]$, limiting the influence of extreme density values. This adjustment addresses cases where clusters are either extremely sparse or extremely dense, ensuring that the Enhanced Outlier Index remains robust and meaningful across a wide range of datasets.

2.1.2 Parallelization of Density Computation

The time complexity of density computation is $O(n)$, where n is the number of data points. For large datasets, this computation becomes expensive. To address this issue, we implemented a parallelization strategy for the key steps involved in the Enhanced Outlier Index computation. The process is divided into the following components:

The complete algorithm for parallelizing the Enhanced Outlier Index computation consisted of the following steps:

1. **Preprocessing:**
   - We computed the centroids of all clusters in parallel.
   - Data points and outliers were assigned to workers for further processing.
2. **Parallel Density Calculation:**
   - Clusters were divided among workers.
   - Each worker computed Density_j for its assigned clusters.
   - We gathered all Density_j values to the main process.
3. **Normalize Densities:**
   - We computed $\min(\text{Density}_k)$ and $\max(\text{Density}_k)$ using a parallel reduction.
   - These values were broadcast to all workers.
   - Each worker normalized Density_j in parallel.
4. **Outlier Distance Computation:**
   - Outliers were divided among workers.
   - Each worker computed distances d_{ij} of its assigned outliers to all cluster centroids.
   - Each worker found $\min(d_{ij})$ for each outlier and multiplied it by $\text{Normalized Density}_j$.
   - Partial sums were sent to the main process.
5. **Parallel Aggregation:**
   - We used a parallel reduction to sum the contributions of all outliers across workers.
   - The total was divided by n to compute the final EOI.

2.2 Motivation for the Enhanced Outlier Index

The Enhanced Outlier Index is designed to address the shortcomings of the original Outlier Index, particularly in scenarios with clusters of varying densities:

- **Scenario 1:** An outlier o is located at a distance x from a dense cluster c_1. Figure 1 illustrates this case.
- **Scenario 2:** The same outlier o is located at the same distance x from a sparse cluster c_2. Figure 2 illustrates this case.

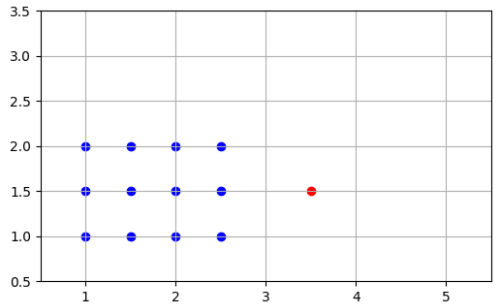

Figure 1: Scenario 1: Outlier o at distance x from a dense cluster c_1.

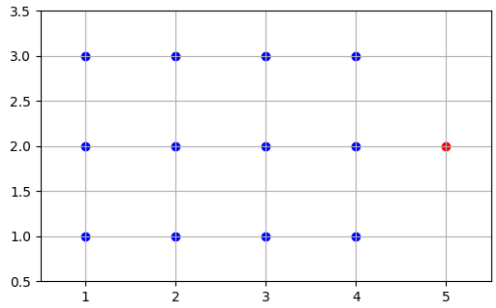

Figure 2: Scenario 2: Outlier o at distance x from a sparse cluster c_2.

In the original Outlier Index, both scenarios would penalize the outlier o equally, as the metric considers only the distance x. However, this approach fails to account for the fact that outliers close to dense clusters should be penalized more heavily than those close to sparse clusters.

The Enhanced Outlier Index resolves this issue by incorporating the normalized density of clusters into the calculation. In Scenario 1, where c_1 is dense, the penalty for the outlier is higher. Conversely, in Scenario 2, where c_2 is sparse, the penalty is lighter. This adjustment provides a more accurate representation of the significance of outliers relative to the clustering structure, resulting in improved clustering quality [6].

2.3 Enhanced MOGA-DBSCAN Process

The Enhanced MOGA-DBSCAN process begins with the initialization of a population of candidate solutions, where each solution represents a pair of DBSCAN parameters *Eps* and *MinPts*. The initial population is generated within bounds determined by Delaunay triangulation, ensuring a diverse and high-quality set of candidate solutions [7].

The algorithm iteratively applies mutation and crossover operators to generate new solutions, which are then evaluated using the two objective functions: the Silhouette index and the Enhanced Outlier Index [8]. A non-dominated sorting approach, combined with a crowding distance mechanism [9], is used to select the next generation of solutions, guiding the population towards Pareto-optimal solutions [10].

To accelerate convergence, a statistical t-test is employed to compare the performance of the current Pareto front with that of previous generations. If the null hypothesis is accepted, indicating no significant improvement, the algorithm terminates early. Otherwise, the process continues until the maximum number of generations is reached or the stopping criterion is satisfied.

At the end of the optimization process, the algorithm outputs a set of Pareto-optimal solutions, allowing users to select the optimal value themselves.[1]

2.4 Time Complexity of MOGA-DBSCAN

The time complexity of MOGA-DBSCAN depends on the time complexities of three primary components: DBSCAN, NSGA-II (Non-dominated Sorting Genetic Algorithm II), and the Delaunay triangulation. The overall time complexity is expressed as follows:

$$O\left(g \times (O(\text{fitness}) + O(n \log n))\right) \tag{3}$$

Where:

- g is the number of generations.
- $O(\text{fitness})$ indicates the time complexity of the cluster validity indices used as objective functions.
- n is the total number of data points.

The time complexity of DBSCAN and the Delaunay triangulation is $O(n \log n)$. NSGA-II has a time complexity of $O(MN^2)$, where M is the number of objectives and N is the population size. Given that M and N are much smaller than the total number of data points (n), the overall complexity of NSGA-II can be considered $O(1)$ relative to the total dataset size.

In the context of MOGA-DBSCAN, $O(\text{fitness})$ consists of the time complexities of the silhouette index and the outlier index. The silhouette index has a time complexity of $O(n^2)$, while the original outlier index has a time complexity of $O(n \times m)$, where m is the number of clusters. Therefore, the overall time complexity of MOGA-DBSCAN is dominated by the silhouette index, yielding:

$$O\left(g \times \left(n^2 + n \log n\right)\right) \tag{4}$$

[1]The implementation details and source code for the Enhanced MOGA-DBSCAN algorithm are available at `https://github.com/HosseinEyvazi/Density-Adjusted-MOGA-DBSCAN/tree/main`.

2.5 Time Complexity of Enhanced MOGA-DBSCAN

In Enhanced MOGA-DBSCAN, the outlier index used in MOGA-DBSCAN is replaced by the Enhanced Outlier Index. The Enhanced Outlier Index is computed as follows:

$$\frac{\text{Density}_j - \min(\text{Density}_k)}{\max(\text{Density}_k) - \min(\text{Density}_k)} + \epsilon, \tag{5}$$

The time complexity of computing the Enhanced Outlier Index is $O(n \times m)$, the same as the original outlier index beacuse the time complexity of computing the density of a cluster is equal to size of that's cluster , also m is O(1). Thus, the time complexity for the fitness function in Enhanced MOGA-DBSCAN is:

$$O(\text{fitness}) = O(n^2) \tag{6}$$

This is still dominated by the silhouette index, so the overall time complexity of Enhanced MOGA-DBSCAN remains:

$$O\left(g \times \left(n^2 + n \log n\right)\right) \tag{7}$$

Given that n^2 dominates $n \log n$, the time complexity can be simplified to:

$$O\left(g \times n^2\right) \tag{8}$$

This shows that the introduction of the Enhanced Outlier Index does not increase the overall complexity of the algorithm compared to the original MOGA-DBSCAN.

2.6 Results on the Custom 3-Cluster and Network Datasets

The evaluation of the Enhanced Outlier Index and runtime improvements was conducted using two datasets: a custom 3-cluster dataset with varying densities and a network dataset. The results, displayed in Figures 3, 4, 5, and 6, highlight the comparative performance of MOGA-DBSCAN and Enhanced MOGA-DBSCAN.

The Enhanced MOGA-DBSCAN demonstrated superior clustering quality and reduced runtime for both datasets. A detailed comparison is provided below:

- **Custom 3-Cluster Dataset:**
  - **MOGA-DBSCAN:**
    Silhouette Score: 0.7095
    Rand Index: 0.9562
    Runtime: 19.8192 seconds
  - **Enhanced MOGA-DBSCAN:**
    Silhouette Score: 0.7369
    Rand Index: 0.9926
    Runtime: 16.4516 seconds

Enhanced MOGA-DBSCAN achieved a higher Silhouette Score and Rand Index, reflecting improved clustering quality. Additionally, it reduced the runtime by approximately 3.37 seconds compared to MOGA-DBSCAN.

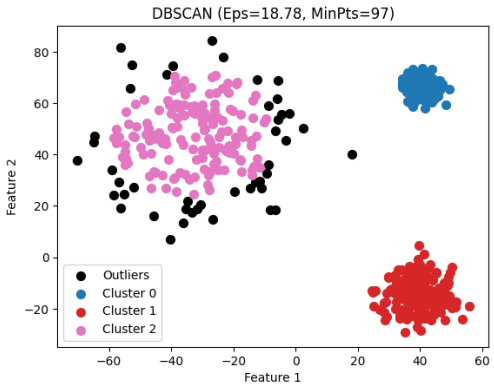

Figure 3: Clustering result of MOGA-DBSCAN on the custom 3-cluster dataset.

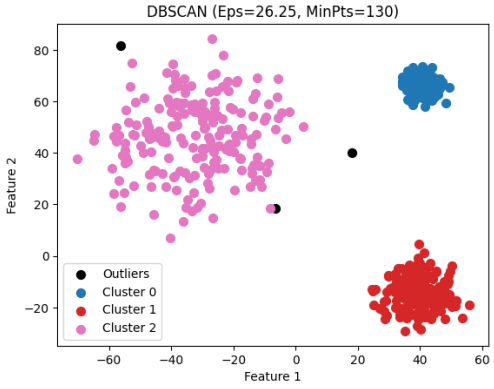

Figure 4: Clustering result of Enhanced MOGA-DBSCAN on the custom 3-cluster dataset.

- **Network Dataset:**
  - **MOGA-DBSCAN:**
    Silhouette Score: 0.4355
    Rand Index: 0.6743
    Runtime: 373.0894 seconds
  - **Enhanced MOGA-DBSCAN:**
    Silhouette Score: 0.7089
    Rand Index: 0.9858
    Runtime: 254.7385 seconds

Enhanced MOGA-DBSCAN significantly improved clustering performance with a much higher Silhouette Score and Rand Index. It also achieved a remarkable runtime reduction of over 118 seconds, showcasing substantial computational efficiency gains.

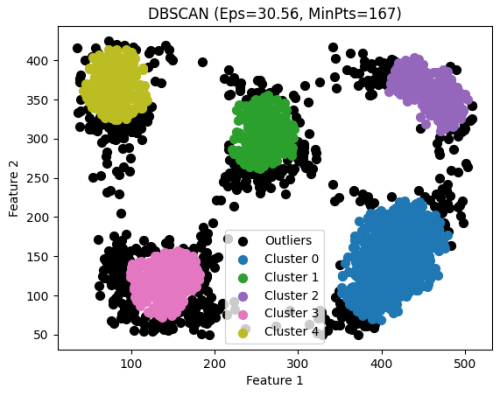

Figure 5: Clustering result of MOGA-DBSCAN on the network dataset.

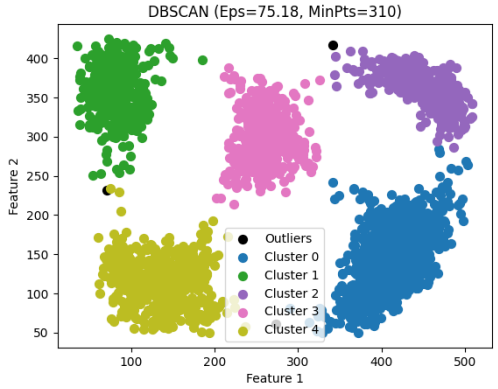

Figure 6: Clustering result of Enhanced MOGA-DBSCAN on the network dataset.

Overall, Enhanced MOGA-DBSCAN consistently delivered better clustering metrics and faster execution times compared to MOGA-DBSCAN, demonstrating the practical benefits of the proposed enhancements.

2.7 Discussion

The experimental results highlight the effectiveness of the Enhanced MOGA-DBSCAN algorithm in addressing the limitations of the original MOGA-DBSCAN. Key findings include:

Clustering Quality: The enhanced algorithm achieved noticeable improvements in both Silhouette Scores and Rand Indices:

- For the custom 3-cluster dataset, the Silhouette Score improved from 0.7095 to 0.7369, and the Rand Index increased from 0.9562 to 0.9926.
- For the network dataset, the Silhouette Score increased from 0.4355 to 0.7089, and the Rand Index rose from 0.6743 to 0.9858.

Computational Efficiency: The Enhanced MOGA-DBSCAN significantly reduced runtime while maintaining or improving clustering performance:

- For the custom 3-cluster dataset, the runtime decreased by approximately 3.37 seconds, offering a balance between accuracy and efficiency.
- For the network dataset, the runtime dropped from 373.0894 to 254.7385 seconds, a reduction of over 30%.

These results confirm that the Enhanced MOGA-DBSCAN algorithm successfully improves clustering quality and computational efficiency, making it better suited for practical, real-world applications.

3 Future Work

While Enhanced MOGA-DBSCAN shows promising results, several avenues for future research remain:

3.1 Time Complexity Optimization

The current time complexity of $O(g \times n^2)$ could be reduced through:

- Implementation of better Silhouette score calculation to achieve lower complexity .
- Optimization of density calculations .

3.2 High-Dimensional Adaptations

To enhance performance in high-dimensional spaces, future work will focus on:

- Development of specialized distance metrics for high-dimensional clustering
- Implementation of feature relevance weighting mechanisms

These improvements would extend the algorithm's applicability to more complex datasets while maintaining its computational efficiency. Additionally, exploring online learning capabilities for streaming data could further enhance the algorithm's practical utility.

References

[1] Y. Ren, J. Pu, Z. Yang, J. Xu, G. Li, X. Pu, S. Y. Philip, and L. He Deep clustering: A comprehensive survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[2] H. V. Singh, A. Girdhar, and S. Dahiya A literature survey based on DBSCAN algorithms. In *2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 751–758, 2022. IEEE.

[3] D. Deng DBSCAN clustering algorithm based on density. In *2020 7th International Forum on Electrical Engineering and Automation (IFEEA)*, pages 949–953, 2020. IEEE.

[4] Z. Falahiazar, A. Bagheri, and M. Reshadi Determining the parameters of DBSCAN automatically using the multi-objective genetic algorithm. *J. Inf. Sci. Eng.*, 37(1):157–183, 2021.

[5] X. Xu, S. Ding, L. Wang, and Y. Wang A robust density peaks clustering algorithm with density-sensitive similarity. *Knowledge-Based Systems*, 200:106028, 2020. Elsevier.

[6] K. Li, X. Gao, X. Jia, B. Xue, S. Fu, Z. Liu, X. Huang, and Z. Huang Detection of local and clustered outliers based on the density–distance decision graph. *Engineering Applications of Artificial Intelligence*, 110:104719, 2022. Elsevier.

[7] Y. Liu and G. Yin The Delaunay triangulation learner and its ensembles. *Computational Statistics & Data Analysis*, 152:107030, 2020. Elsevier.

[8] A. Dudek Silhouette index as clustering evaluation tool. In *Classification and Data Analysis: Theory and Applications 28*, pages 19–33, 2020. Springer.

[9] S. Verma, M. Pant, and V. Snasel A comprehensive review on NSGA-II for multi-objective combinatorial optimization problems. *IEEE Access*, 9:57757–57791, 2021.

[10] Y. Zhou, W. Zhang, J. Kang, X. Zhang, and X. Wang A problem-specific non-dominated sorting genetic algorithm for supervised feature selection. *Information Sciences*, 547:841–859, 2021. Elsevier.