Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 2, Issue 5, 2024

MULTIOBJECTIVE LEMURS OPTIMIZER ALGORITHM FOR EFFICIENT DBSCAN

Shanzay Amjad*[1], Amjad Mahmood Chaudhry[2]

[1]College of Statistical Sciences, University of Punjab, Lahore, Pakistan.
*[*2] University of Agriculture Faisalabad Punjab, Pakistan*
*[*1]shanzayamjad09@gmail.com*

DOI: https://doi.org/10.5281/zenodo.16757919

Keywords
DBSCAN,
Multiobjective Optimization,
Unsupervised learning,
Cluster validity indices

Article History
Received on 14 October 2024
Accepted on 15 November 2024
Published on 30 December 2024

Corresponding Author: * Shanzay

Abstract
Clustering algorithms are frequently employed in the fields of data processing as unsupervised learning algorithms. Density-based spatial clustering of applications with noise (DBSCAN), a typical method based on density clustering, can build clusters by finding densely populated regions divided by sparsely populated regions based on cluster density. However, the DBSCAN algorithm has an inherent flaw that cannot be avoided. Because the clustering performance is highly sensitive to the DBSCAN parameters settings i.e. ***Eps*** *and* ***Minpts****, there is no theory to guide the setting of these parameters. To optimize the settings of these parameters, this study proposed a hybrid algorithm that combines the DBSCAN method with the Multiobjective Lemurs Optimizer (MOLO). This method approaches the matter of clustering as a multiobjective optimization challenge to minimize particular cluster validity indices, expressed as objective functions characterizing the quality of the clustering solutions. This made it feasible to determine the correct values for the DBSCAN parameters. The outcomes indicate that the suggested MOLO-DBSCAN is still effective in achieving the most accurate settings for these parameters.*

Introduction

Clustering significance to information analysis is expanding every day, as a result of the rapid development of data mining, machine learning (ML), artificial intelligence, and other disciplines [1]. The goal of clustering is to group a collection of data objects based on how similar or close they are, without any prior knowledge [2]. Clustering algorithms are frequently employed in a variety of industries, including healthcare [3], document management [4, 5], marketing [6, 7], and pattern recognition [8, 9].

The fundamental goal of clustering is to group related data objects while keeping distinct objects separate. A distance metric, such as Euclidean distance or cosine similarity, is used to measure the similarity of data objects. There are numerous clustering algorithms available, each with its own set of strengths and drawbacks, and they are generally classified into five categories: partitioning-based, hierarchical-based, density-based, model based, and grid-based algorithms [10].

However, the main focus is on the density-based method, specifically the density-based noise application spatial clustering algorithm (DBSCAN).

They are able to find clusters by examining data point density and do non-convex clustering. According to their density characteristics, data entities are arranged using the DBSCAN method, which was created for clustering by Ester et al. [11]. It defines a cluster as a region that is heavily occupied with data items and is divided into smaller zones of lower density. It is able to recognize groups of various shapes.

DBSCAN is superior to other clustering approaches in several ways, including its skill at handling noise and its resistance to abnormalities. However, using DBSCAN presents the difficulty of adjusting its parameters. The two main parameters of the DBSCAN method are represented by the symbols *Eps* and *Minpts* [12, 13].

According to the characteristics of the data and the user's domain knowledge, the values of *Eps* and *Minpts* are frequently determined individually. This method, nevertheless, can be time-consuming and may not always produce the best results. The parameters for DBSCAN are to be tuned using an adaptive method that is suggested in this research.

1.1. Literature Review

The primary issue concerning the automatic selection of input parameters for the DBSCAN algorithm presents a significant challenge. Nevertheless, achieving an appropriate selection of parameter values is essential, involving the utilization of both Minpts and Eps.To address the issue of parameter configuration, and various frameworks have been created. In order to start, Fan et al. [14] presented a framework for modifying parameters within the clusters of stochastic block models. They used cross-validation to choose the ideal number of clusters, concluding the model selection procedure for a specific clustering situation.

Ditton et al. [15] proposed a similar cross-validation-based methodology to evaluate cluster reliability across several samples that made use of inter-cluster validity indices. Notably, they focused on domain-specific assessments inputted during param-eter refinement rather than external data or ground truth labels when making their judgment.

The DBSCAN algorithm's broad application is accordingly constrained within a narrow scope due to the manual intervention required for the complete parameter selection approach which draws from research experiments or extensive trial and error [12]. Numerous academics have responded by introducing automated algorithms for choosing DBSCAN parameters to get around this problem. A KD tree data format was proposed by Mitra and Nandy [16] to find clusters of varying densities. This structure served as the foundation for the k-dist display and aided in calculating the Eps parameter value. The technique significantly reduced noise and automatically recognized regions of different densities by using pattern distances to their k-nearest neighbors to find points of convergence or knees in the plot.

The grid partition method was employed by the grid-based DBSCAN approach to select appropriate clustering parameters. Its objective was to improve the efficiency and impartiality of choosing the Eps and Minpts parameters for the DBSCAN algorithm. The algorithm could discriminate between noise and clusters with arbitrary shapes due to the autonomous calculation of these parameters, leading to more depend-able clustering results. However, the comprehensive effectiveness of This approach was not thoroughly investigated across a broader spectrum of datasets. [17]. To compute the DBSCAN parameters, the Gaussian mean was employed. The Gaussian mean was used to calculate the DBSCAN parameters. By utilizing the benefits of Gaussian means, the shortcomings of DBSCAN were to be addressed. To estimate the DBSCAN parameters and create compact clusters with pinpointed cluster centers, Gaussian means were used. For datasets containing non-linear or irregularly formed clusters, arithmetic mean and circular clustering algorithms may be less useful [18].

According to [19], the LP-DBSCAN method was used to establish the DBSCAN parameters. Although it requires a user-defined cutoff range, this approach locates data density peaks. While the earlier described heuristic techniques likewise require user-specified parameters and don't fully automate, more current methods have added metaheuristic algorithms to gradually improve clustering results. By incorporating metaheuristic algorithms, researchers have optimized DBSCAN, allowing for better clustering outcomes and automated optimization of DBSCAN settings.

The modified ant clustering algorithm (ACA) and the point density-based method were merged to

create the innovative hybrid algorithm known as PACA-DBSCAN during the data pre-processing stage. Addressing DBSCAN's shortcomings is the goal. The database is split into N segments according to data density using this method, and each segment is then subjected to DBSCAN. Five datasets were used to compare PACA-DBSCAN's outcomes to those of DBSCAN and PDBSCAN. PACA-DBSCAN provides superior clustering accuracy, according to the experimental data. Using met-rics like the F-Measure and a certain criterion (ER), the algorithm's effectiveness was evaluated. According to the analysis, PACA-DBSCAN performs far better than DBSCAN [12]. It was suggested by Hosseini Rad and Abdolrazzagh-Nezhad [20] to combine the earthworm optimization algorithm EWOA and DBSCAN, or EWOA-DBSCAN, and utilize this combination to determine the proper values of DBSCAN's parameters for grouping data cubes. This study highlights the adaptability and autonomy of DBSCAN in detecting no convex clusters. A new similarity metric is created, addresses are assigned to the cube cells, and 3D data is converted into a 2D dataset as part of the preprocessing techniques.

Zhu et al. [46] presented a K-DBSCAN clustering algorithm. By applying the widely used harmony search optimization (HS) algorithm to improve the DBSCAN parameters, this strategy tries to deal with the challenges posed by nonconvex clustering and local optimum. By adding the harmony search method to DBSCAN, K-DBSCAN offers superior clustering results with a predetermined number of K classifications. This study highlights a few K-DBSCAN shortcomings that should be addressed in subsequent research, In order to find the ideal clustering parameters for huge datasets, the innovative HS optimization technique may need to do several rounds, which can take time. The clustering performance of K-DBSCAN on high-dimensional datasets was the second problem.

Xiong et al. [22] suggested a multi-density clustering approach that incorporates the density levels and calculates the Eps for every group based on characteristics of density variation. This approach was developed to address issues with the traditional DBSCAN algorithm.

With updated DBSCAN and Density Peak (DP) algorithms, Li et al. [23] suggested a two-stage clustering method (TSCM), which uses a bat optimization strategy to improve only the Eps parameters of DBSCAN. The Eps and Minpts variables in particular can have an impact on the DBSCAN algorithm's performance. The Silhouette Index [24] is used as a fitness function in this method's bat optimization technique to determine the parameters.

The experiments' findings show that the TSCM outperforms DP and DBSCAN, which effectively eliminates the necessity for manual assistance with DP's cluster center selection and the robustness of DBSCAN parameter selections. Determining a cut-off distance was not discussed, though. We would strictly abide by the recommended cutoff distance. In essence, cut-off distance (dc) is chosen so that an average quantity of neighboring individuals of each data point accounts for between (1%-2%) of the total. The term neighbors here refer to the data points that were just dc distant from one another. This assertion was beneficial and arbitrary. As a result, attention must be paid to the TSCM algorithm's automatic selection of the cutoff distance parameter.

For superior clustering findings, parameter Minpts should also be visualized, as the aforementioned two algorithms merely identified the issue with the Eps parameter and its selection criterion.

The optimal selection of DBSCAN parameters is facilitated by an adaptive clustering approach which integrates the nearest neighbor function with a genetic algorithm [25]. The approach uses DBSCAN directly to cluster datasets, a genetic algorithm (GA) for improving DBSCAN's parameters, and factor analysis (FA) to minimize the number of dimensions. To lessen the uncertainty, DBSCAN's settings need to be adjusted. The precision and entropy of the clustering results were compared to assess the hybrid approach FA+GA-DBSCAN. The results show that the information found in this type of dataset was practically meaningless because the entropy

variance was low.

To determine the optimal number of clusters, a novel hybrid automatic clustering system named GFO-DBSCAN was developed by Balavand [26]. It had a two-step foundation. The grouper-fish-octopus algorithm (GFO) was utilized in the first stage to generate values for the DBSCAN method's parameters. The DBSCAN method was used to do clustering in the second stage using similar parameter values, and the CH validity index was used to assess the effectiveness of the clustering. The results of the trial showed that GFO-DBSCAN consistently locates the actual number of clusters. GFO-DBSCAN demonstrated efficiency owing to DBSCAN's capability to detect non-convex clusters and remove outliers during clustering. However, when examining each dataset, there was a lack of distinct clustering patterns.

Through the use of the bird swarm optimization method, Wang et al. [27] developed an adaptive DBSCAN. The ideal Eps parameter neighboring values were chosen by using the bird swarms approach's global search capability. By enhancing the clustering process through adaptable parameter optimization, human participation was also made unnecessary. Using simulated and actual datasets with a range of clustering evaluation index values, tests were conducted on BSA-DBSCAN's clustering efficiency. The simulation studies indicate that the BSA-DBSCAN strategy finds the optimal Eps parameter value more successfully and with more precision than other approaches. The amount of the Minpts parameter must still be manually modified for the BSA-DBSCAN algorithm, though. As therefore, complete optimization for the same dataset might not be achievable when applying various Minpts settings. The effectiveness of DBSCAN depends on the set parameters Eps and Minpts, which can be difficult to determine correctly for various clustering objects.

Yang et al.[28] designed an improved DBSCAN optimized utilizing an arithmetic optimization algorithm (AOA) with opposition-based learning (OBL) (OBLAOA-DBSCAN), which was the solution to this constraint. By applying standard functions, the OBLAOA optimizer's improved exploration capabilities were demonstrated in comparison to conventional AOA and other meta-heuristic methods.

An adaptive DBSCAN approach was proposed by Karami and Johansson [29] that integrated a binary differential evolution optimization strategy for selecting DBSCAN parameters. The DBSCAN lgorithm uses this optimization method to get the ideal Eps value for each unique Minpts. Notably, the tournament-based selection method used in this approach produced superior outcomes in terms of cluster validity indices like purity, entropy, and the Dunn Index, surpassing previous methods. With purity levels varying from 89.4% to 99%, the approach achieves the highest level of precision. The findings recommend the use of further cluster validity criteria, the fusion of DBSCAN with different metaheuristics tools, and varied data sets for testing the BDE-DBACAN approach.

Multi-Verse Optimizer (MVO) a variable update method suggested by Lai et al.[30] that aims to enhance the optimization of the DBSCAN parameters, is known for its outstanding optimization performance. The improved MVO was designed to swiftly determine the optimal Eps interval and locate DBSCAN's optimum clustering accuracy. According to the experimental results, the updated MVO effectively optimizes the DBSCAN settings, which raises clustering performance. The external validity index of purity is used as the goal function in this hybrid technique. This approach cannot be used with real-life data sets because clusters are not predetermined.In an effort to optimize the DBACAN parameters, Falahiazar et al. [31] used a similar approach with a MOGA application. This deployment, in contrast to the approach suggested by Karami and Johansson [29], sought to choose the best parameters to verify the results without requiring any external ground facts.

The proposed method was put to the test using a multiobjective evolutionary algorithm, which assesses optimization based on numerous fitness factors. Genetic algorithms use the Outlier index, the Dunn [32], and the silhouette indices [24] as fitness functions.

The initial boundaries of the DBSCAN parameters are set using the Delaunay triangulation procedure, which does not need any input parameter.
The experimental findings indicate that the DBSCAN parameters are efficiently and precisely optimized using the MOGA-DBSCAN method. The adaptability of the technique should be investigated, and more developments in parameter estimation and visualization are needed in order to assess how well it handles huge datasets.

1.2. The Gap

The No Free Lunch (NFL) theorem states that there is no approach that can solve all optimization issues. The DBSCAN clustering algorithm's performance was still improving. A combination of more robust ML techniques and metaheuristic algorithms are required to automatically optimize the DBSCAN algorithm settings in order to get correct clustering results.

1.3. The Contribution

This study proposed a novel multiobjective metaheuristic improvement technique called MOLO-DBSCAN to fine-tune DBSCAN's parameters, which combines the benefits of multiobjective Lemurs and DBSCAN. DBSCAN also improved with MOLO and performed well across a variety of datasets, depending on the results of the experiment. As a result, the following are the contributions of this article:

(1) The development of the MOLO-DBSCAN clustering algorithm, which does automatic parameter exploration and clustering.

(2) Multiple external cluster validity indices are used during the clustering process, which enhances the clustering results. An objective function is used, which is the internal cluster validity index.

(3) The suggested MOLO-DBSCAN technique is capable of outperforming traditional metaheuristic optimization algorithms in terms of clustering performance

1.4. The Layout of the Paper

The following sections are arranged as follows: Section 2 provides insights into the operational mechanisms of the DBSCAN and MOLO algorithms. In Section 3, the MOLO-DBSCAN algorithm that is being proposed is presented. Moving on to Section 4, the efficacy of the suggested algorithm is showcased across six distinct datasets. This is achieved by employing various cluster validity indices, thereby facilitating a comparative analysis with several considered metaheuristic optimization algorithms. In Section 5 case study was performed by incorporating real real-life data set for further visualization of the proposed MOLO-DBSCAN algorithm. Finally, Section 6 concisely highlights the main contributions, assesses the implications, and suggests possible directions for further research.

2. Related Concepts

2.1 Density-Based Spatial Clustering of Applications with Noise

A well-known density-based clustering technique called DBSCAN is used to group together data points that are adjacent to one another in high-density regions while classifying data points in low-density regions as noise or outliers. The method locates dense areas in the data space and establishes neighborhoods around data points. The approach is beneficial in situations where the number of clusters is unknown beforehand because it does not require the user to specify it previously.

1. ***Eps*** The greatest distance between the two locations at which one is deemed to be in closest proximity of the other. It specifies a data point's nearest neighbors.
2. ***Minpts*** The minimum number of data points required to form a dense region. Points with at least MinPts points within distance *Eps* are considered core points.

The working of the DBSCAN algorithm involved the following terms:-

Core Point: A point with at least *MinPts* neighbors within *Eps* distance, forming the basis of a cluster.

Density-Reachable: A point reachable from another through a chain of core points within *Eps* distance.

Border Point: A non-core point within a core point's *Eps* radius, helping define cluster edges but not forming clusters independently.

Algorithm 1 The DBSCAN Algorithm

Require: Dataset *D*, *Eps*, *Minpts*

Ensure: Clusters of data points

1. Initialize the visited set as empty

2. Initialize the clusters list as empty

3. For each unvisited point P in D:

a. Mark P as visited

b. N ← REGIONQUERY (P, Eps)

c. If |N| < MinPts:

i. Mark P as noise

d. Else:

i. Create a new cluster C and add P to C

ii. For each point Q in N:

1. If Q is not visited:

a. Mark Q as visited

b. N′ ← REGIONQUERY(Q, Eps)

c. If |N′| ≥ MinPts:

- Add all points in N′ to N

2. If Q is not yet a member of any cluster:

a. Add Q to cluster C

iii. Add cluster C to clusters

4. Return clusters

5. Function REGIONQUERY(P, Eps):

Return the set of points within distance Eps from point P

2.2 Multiobjective Lemurs Optimizer

In order to produce a single optimal solution, the single-objective lemurs optimizer algorithm only optimizes one goal during the search process [33]. However, while solving a large number of real-world issues, it is necessary to simultaneously optimize anumber of conflicting objectives. In order to deal with the problems in real-world set-tings, multi-objective optimization methodologies are used. According to the standard definition, the multi-objective optimization problem is defined by Maulik et al. [34] and Kalyanmoy [35].

The technique of simultaneously optimizing a system for many conflicting objectives is known as "multiobjective optimization." A multiobjective optimization issue can be mathematically stated as follows given a set of choice variables *z*:

Minimize $f(z) = [f_1(z), f_2(z), \ldots, f_m(z)]$

Subject to:

$$g_j(z) \leq 0, j = 1, 2, \ldots, n,$$

$$( ) = \quad , \quad = \quad , \quad , \ldots , \quad ,$$

where $f_i(z)$ stands for the objective function that must be minimized, $g_j(z)$ for the equality condition that must be met, and $h_j(z)$ for the equality condition.

2.3 Pareto Dominance

Finding the set of non-dominated solutions usually referred to as the Pareto front, that indicates a conflict between the competing objectives is the goal of multi-objective optimization. It is said that a solution x dominates the other solution if it dominates another solution y in at least one objective and is superior in none [36].

Formally, one solution x is said to dominate another y if:

$$f_j(x) \leq f_j(y), j = 1, 2, \ldots, n, \text{ and}$$

$$( ) < \quad ( ) \qquad .$$

The group of non-dominated solutions governed by Erfani and Utyuzhnikov [37] is usually referred to as the Pareto front, sometimes known as the Pareto set. When solving a multiobjective optimization problem, there is no one perfect solution. The final selection is made in place of this based on the decision-maker's interests and objectives from a collection of non-dominated alternatives.

2.4. Crowding Distance

Crowding distance as described by [35], is a metric used to assess the diversity of solutions in the objective space after non-dominated sorting in evolutionary algorithms. Solutions with larger crowding distances are preferred as they indicate greater diversity. It is calculated by sorting solutions by each objective and summing the normalized distances between neighboring solutions across all objective dimensions.

This study applies the Pareto dominance approach to the lemurs optimizer algorithm, which uses crowding distance and non-dominated sorting [38]. The objective functions must be properly formulated, and any appropriate modifications to the parameters must be made before the procedure can begin. In order to ensure a more uniform distribution of lemurs across the search space, an initial population of n lemurs is generated. The iteration process begins with the evaluation of the jumping rate, known as the Free Risk Rate, once either the tolerance threshold or a predefined number of iterations has been reached. The formula for this coefficient is as follows:

$$\mathbf{FRR = FRR(High\ Risk\ Rate) - Curr\ Iter \times} \frac{\boldsymbol{High\ Risk\ Rate - Low\ Risk\ Rate}}{\boldsymbol{Max\ Iter}} \quad \textbf{(1)}$$

The search space is divided into two behavioral phases dance-hup and leap-up similar to the single-objective case [33].

1. Dance-Hup Behavior (Exploitation Phase)

This phase aims to intensify the search around high-quality solutions. The algorithm first applies non-dominated sorting to identify the first Pareto front, then selects the solution with the highest crowding distance to preserve diversity. The selected solution is refined using the following update equation:

$$\mathbf{L(j) = l(i,j) + |l(i,j) - near_{solution(j)}| \times (rand0.5) \times 2} \quad \textbf{(2)}$$

If the updated solution L(j)L(j)L(j) yields better fitness than the previous one, it replaces the original.

2. Leap-Up Behavior (Exploration Phase)

To encourage diversity and global exploration, the algorithm re-applies non-dominated sorting and selects a diverse subset of solutions from each front based on crowding distance. The movement of each selected lemur is then guided by the following equation:

$$\mathbf{L(j) = l(i,j) + min(|l(i,j) - lb(j)|, |l(i,j) - ub(j)|) \times (rand - 0.5) \times 2} \quad \textbf{(3)}$$

3. The proposed MOLO-DBSCAN

3.1. Solution Formulation and Search Area

The initializing phase is completed in the MOLO-DBSCAN method, similar to other algorithms, before moving on to the main phases. Given that the metaheuristic algorithm's search space is continuous and our parameters for tuning are mixed integers, a constraint should be applied to the corresponding search space as described by Marler and Arora [38] which is a similar approach to [39, 40]. Every possible outcome has two parameters, *Minpts* and *Eps*, which reflect a predefined range of each solution parameter.

3.2 Validity Indices as objective functions

The objective function defines the clustering quality metrics that will be simultaneously optimized in the optimization of the DBSCAN parameters. The objective function guides the search for an ideal pairing of Minpts and Eps that balances various aspects of clustering quality and aids in the identification of a varied selection of effective alternatives. A detailed description of the two objective functions is stated below:-

3.2.1 Davies Bouldin Cluster Validity Index (DBI)

Davies and Bouldin [41] develop DBI. It is used to gauge cluster separation and compactness. In the framework that is being suggested, the main goal of DBI is to maximize inter-cluster distance while minimizing intra-cluster distance, which measures the distance between data points that are a part of the same cluster. The DBI's range lies between 0 and 1. Clustering results with a lower DBI value suggest greater intra-cluster similarity and lower inter-cluster dissimilarity.DBI would be calculated as:-

$$= = \sum_{=} \max_{k \neq j} \left(\frac{+}{} \right)$$

Where:

- m is the total of all clusters
- sj represents the average distance between the cluster centroid and each sample in the cluster j.
- sk The average distance between the centroid of a cluster and its samples is shown

by item sk.

- Mjk is the distance between the centroids of cluster j and cluster k.

3.2.2 Cluster Validity Density Involved Distance (CVDD)

The density information of the data items is taken into consideration via a new internal validity metric called CVDD [42]. The average distance between sites is calculated while taking into account the local density of points within a neighborhood.

The CVDD index is defined as follows:

$$( ) = \frac{\sum^{K} \quad (\quad)}{\sum_{=} \quad (\quad)}.$$

The proposed internal validity index, CVDD, measures the average performance (π) across all clusters. It can also be calculated by first evaluating each cluster's performance individually, averaging those results, and then computing the final index. A higher CVDD(π) value indicates better clustering quality. As noted in [43], the **duality principle** is useful here it allows the same optimization method to be used for either **maximization or minimization** by simply adjusting the objective function (e.g., multiplying by -1), rather than modifying the entire algorithm. Therefore, the proposed CVDD index is structured to support **minimization problems** through this straightforward transformation.

3.3 Parameter Bounds Settings

In setting the parameters for DBSCAN, past studies suggest that for two-dimensional datasets, **MinPts** is fixed at 4, and **Eps** is calculated as *(2 × dim − 1)* [44].For datasets with more than two dimensions, **MinPts** is adjusted to *2 × dim*. Choosing **Eps** is more complex, as it needs to be small enough to ensure only a portion of data points fall within that distance, allowing the algorithm to detect meaningful clusters. It's important that the number of resulting clusters does not exceed a user-defined maximum [46] and the values of MinPts and Eps should enable most data points to form valid clusters [45].

Traditional or statistical methods for setting these values can be time-consuming and imprecise, which is why metaheuristic approaches are preferred. In MOLO-DBSCAN, a metaheuristic method is used to search for optimal MinPts and Eps values within a normalized, predefined range derived from earlier research.

3.4. Framework of MOLO-DBSCAN

In this part, a multiobjective Lemurs technique is used to optimize the two DBSCAN parameters *Eps* and *Minpts*. Algorithm_2_contains the pseudocode for the suggested MOLO-DBSCAN approach.

Algorithm 2 MOLO-DBSCAN Algorithm

Require: Dataset (d), Population size (N), Number of iterations (t_max), Number of dimensions (dim), lower bound (lb), upper bound (ub), high-risk rate, low-risk rate

Ensure: Clustering findings, visualization, and processing.

1: Generate population of n lemurs y_i, where (i = 1,……. N).
2: Apply the objective functions to the results of the DBSCAN algorithm after running it with a combination of Eps and MinPts: DBSCAN(d, Eps, MinPts).
3: Set itr = 1.
4: while (itr < t_max) do
5: Sort lemurs using non-dominated sorting based on their objective function values.
6: Assign crowding distance values to lemurs based on distances from their neighbors in the same rank. Then, calculate Free Risk Rate (FRR) using **Eq. 1**.
7: Select lemurs with best rank and crowding distance to update global best lemurs list (gbl).
8: for each lemur i do
9: Update best nearest lemurs (bnl) by selecting the one with best crowding distance among closest lemurs.
10: for each decision variable j in lemur i do
11: if rand(0,1) < Free Risk Rate then
12: Modify decision variable j using**Eq.2.**
13: else
14: Modify decision variable j using**Eq.3.**
15: end if
16: Apply boundary limitation to ensure variable j remains within bounds [lb, ub].
17: end for
18: Evaluate the objective function for the updated lemur i.
19: end for
20: Update population if new fitness value < previous fitness value.
21: Rank existing Pareto-optimal global best solutions and locate them.
22: Increment iteration: itr ← itr + 1
23: Determine optimal DBSCAN parameters.
24: end while

3.5 Comparision Algorithms

We have chosen a few well-known multiobjective algorithms and compared them to a suggested approach for DBSCAN parameter optimization in order to assess the performance of the proposed MOLO-DBSCAN algorithm.
The algorithms are listed below:-

3.5.1 Multiobjective Cuckoo Search (MOCUCKOO):

MOCUCKOO is a nature-inspired multiobjective optimization algorithm based on cuckoo breeding behavior. It evolves a population of candidate solutions using Lévy flights and random walks, applying non-dominated sorting and crowding distance to maintain diverse trade-off solutions. Clustering is used post-optimization. Key parameters: Lévy exponent $\beta = 1.5$, 30 nests, discovery probability = 0.25.

3.5.2 Multiobjective Firefly Algorithm (MOFIREFLY):

MOFIREFLY mimics firefly flashing behavior for multiobjective optimization. Each firefly represents a solution, with brightness linked to objective performance. Movement is guided by relative brightness and proximity, iteratively finding Pareto-optimal solutions. Clustering follows for further analysis. Parameters: $\alpha = 1.0$, $\theta = 0.98$ (alpha decay), $\beta = 1.0$ (attractiveness), $\gamma = 0.1$ (absorption).

4 Experiments and Performance Evaluation

4.1 The Datasets

This section employs five synthetic data sets and one real data set to assess the performance of our optimization technique MOLO-DBSCAN. Table 1 displays the specifics of the datasets, including their instances, dimensions, and various clusters,which were taken from the clustering benchmark resource [49, 50]. We employed datasets with real labels to analyze the results of this evaluation with the real labels and the clustered labels to determine how effective the algorithm was. In Table 2 the threshold ranges for the preceding data sets are provided.

Table 1 Properties of the dataset

Datasets	Instance	Dimension	Cluster
Aggregation	788	3	7
Spiral	251	3	7
Flame	240	3	2
Diamond9	3000	3	9

Table 2 Parameter Bounds

Datasets	*Eps*	*Minpts*
Aggregation	[0.1,0.5]	[4,10]
Spiral	[0.2,2.1]	[4,12]
Flame	[0.11,0.8]	[4,10]
Diamond9	[0.1,0.5]	[4,15]

4.2 The External Cluster Validity Index

External Cluster Validity Index (CVI) evaluates clustering effectiveness by comparing cluster assignments to external data like ground truth labels. When such labels are absent, internal validation metrics are used, though they have limitations. This study employs both internal and external CVIs. External CVIs measure alignment between clustering results and actual data structure using metrics like accuracy, Rand Index, Purity, V-Measure, Homogeneity, and Completeness.

Accuracy

It is a proportion of true data points gathered from the complete data set. By analyzing the cluster label K with the actual label C, the data is correctly clustered.

Rand Index

It is a metric that is used to determine how closely two data groupings resemble one another. It is often created by contrasting the clustering with the benchmark labels [20, 53]. It assesses the extent of conformity between two data points regarding the clusters to which they belonged. Given two clustering's:

- A true or benchmark clustering label, often represented as a set of ground truth labels.
- A developed or experimental clustering label, produced by a clustering algorithm.

$$RI = \frac{+}{+ \quad + \quad +}$$

- **W**: Pairs in the same cluster in both clusterings.
- **X**: Pairs in different clusters in both clusterings.
- **Y**: Pairs in the same cluster in the first clustering, but different in the second.
- **Z**: Pairs in the same cluster initially, but different in the second clustering.

Purity Rate

Purity measures how well clustering results match true class labels. It is defined as:

$$Purity = -\sum \max|c_K \cap t_i|,$$

- N: total data points
- c_k: data points in cluster K
- ti: data points in true class i
- |·|: set size

A purity of 1 indicates perfect clustering (each cluster contains only one class); lower values imply more mixed-class clusters.

Homogeneity

The homogeneity metric measures the extent to which every cluster exclusively contains data points from an identical class. If a cluster has only one kind of data point, it is said to be homogenous. The homogeneity score is a numeric value between 0 and 1, where 1 indicates that all clusters only include data points from one class. Its formula is stated as:

$$h = 1 - \frac{(\mid)}{( )}$$

$$H(c) = -\sum_x (P_x \log_2(P_x))$$

where,

c is the collection of actual labels; item x is the collection of predicted labels.

$H(c)$ is the entropy of the actual labels

Completeness

This metric evaluates how well each class's data point is assigned to a particular cluster [55]. When all of the data points in the same class are clustered together, the clustering process is considered complete. The completeness results fall between 0 to 1, where 1 indicates that every data point belonging to the identical class is gathered. Its formula is stated as:

$$C = 1 - \frac{(\mid)}{( )}$$

- c is the group of actual labels.
- x is the group of predicted labels.
- $H(x)$ is the expected labels' entropy.
- $H(x|c)$ evaluates the uncertainty of the class labels provided the clustering placements.

V-Measure

The V-Measure, also referred to as the V-Measure score or the V-Statistic is a metric used to rate the accuracy of clustering results [56]. Some of the challenges associated with V-Measure are solved by combining homogeneity and completeness. It has a value range from 0 to 1, with 1 signifying completeness and perfect homogeneity.

$$= \frac{2 \times (H \times C)}{+}$$

where,

- H denotes the homogeneity score.
- C denotes the completeness score.

Experimental Results

The MOLO-DBSCAN results are evaluated against the best results produced by two metaheuristic algorithms, the MOCUCKOO Search DBSCAN Algorithm and the MOFIREFLY DBSCAN Algorithm, in this section. Implementation of the suggested MOLO-DBSCAN algorithm is done using **Matlab 2021b** software. The maximum number of iterations and population size for each method are set to **1000 and 30**, correspondingly.

The MOCUCKOO-DBSCAN, MOFIREFLY-DBSCAN, and MOLO-DBSCAN meta-heuristic algorithms were used to improve the DBSCAN settings for six different datasets, which are displayed in Table 3. The parameters being improved for each dataset and operation are *Eps* and M i n p t s . The highlighted parameter values have been chosen as the optimized parameter values for the appropriate method and dataset.

Table 3 Optimized DBSCAN parameters for six datasets tuned by different metaheuristic algorithms

Datasets	Parameter Values	MOCUCKOO-DBSCAN	MOFIREFLY-DBSCAN	MOLO-DBSCAN
Aggregation	*Eps*	0.19	0.4	**0.16**
	Minpts	7	9	**8**
Spiral	*Eps*	0.25	0.24	**0.310**
	Minpts	5	6	**4**
Flame	*Eps*	0.27	0.6	**0.28**
	Minpts	5	7	**4**
Diamond9	*Eps*	0.1	0.2	**0.1**
	Minpts	10	9	**11**

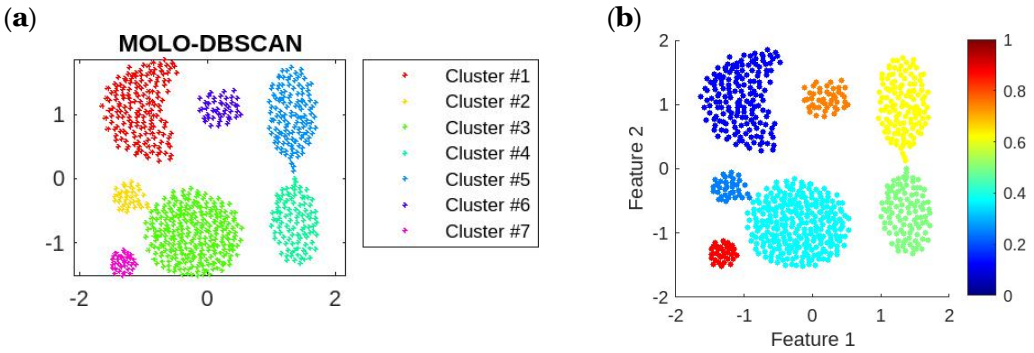
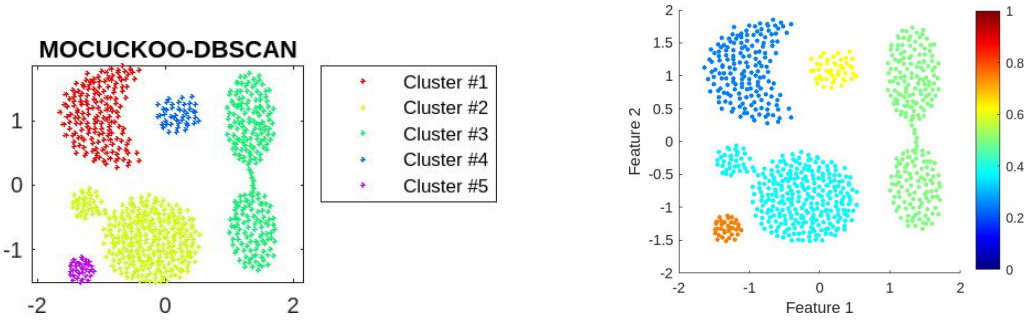


Fig. 1 Plot (a) shows the clustering results for the **aggregation** data set using the MOLO-DBSCAN algorithm, while plot (b) displays the density clustering results.

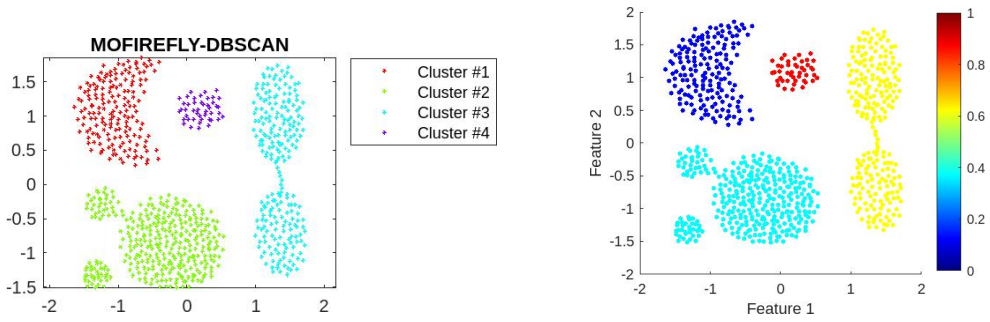
Table 4 The performance evaluation of multiobjective metaheuristic algorithms among six external indices tested on six datasets for optimized DBSCAN

Dataset	External Indices	MOCUCKOO-DBSCAN	MOFIREFLY-DBSCAN	MOLO-DBSCAN
Aggregation	Accuracy	0.84	0.79	**0.99**
	Rand index	0.9273	0.8936	**0.9957**
	Purity rate	0.8274	0.7843	**0.994**
	Completeness	0.96	0.86	**1**
	Homogeneity	0.97	0.88	**1**
	V-Measure	0.9056	0.889	**1**
Spiral	accuracy	0.8365	0.5288	**1**
	Rand index	0.8907	0.6850	**1**
	Purity rate	0.8910	0.6891	**1**
	Completeness	1	0.98	**1**
	Homogeneity	0.84	0.54	**1**
	V-Measure	0.92	0.6918	**1**
Flame	accuracy	0.94	0.65	**0.9583**
	Rand index	0.9510	0.5406	**0.9669**
	Purity rate	0.983	0.645	**0.9917**
	Completeness	1	0.998	**1**
	Homogeneity	0.9417	0.6371	**1**
	V-Measure	0.97	0.7786	**0.9787**
Diamond9	accuracy	0.8823	0.6650	**0.9927**
	Rand index	0.9736	0.8512	**0.9981**
	Purity rate	0.8837	0.6657	**0.9948**
	Completeness	0.8887	0.888	**0.880**
	Homogeneity	0.8820	0.7018	**0.9933**
	V-Measure	0.8853	0.7840	**0.9377**



(**a**) (**b**)

Fig. 2 Plot (a) shows the clustering results for the **aggregation** data set using the MOCUCKOO-DBSCAN algorithm, while plot (b) displays the density clustering results.



(**a**) (**b**)

Fig. 3 Plot (a) shows the clustering results for the **aggregation** data set using the MOFIREFLY-DBSCAN algorithm, while plot (b) displays the density clustering results.

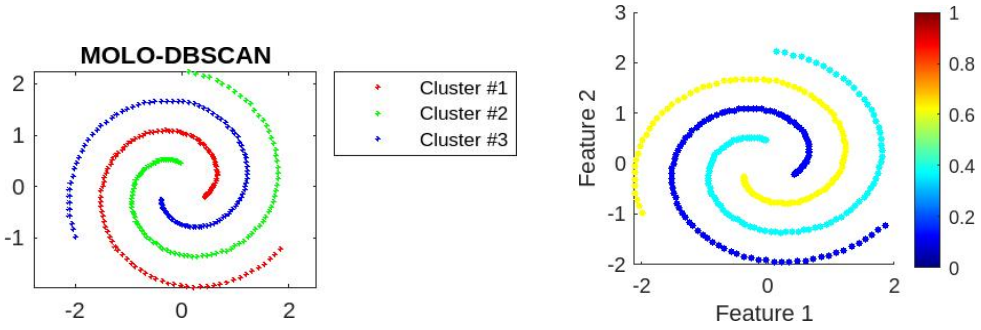


Fig. 4 Plot (a) shows the clustering results for the **spiral** data set using the MOLO-DBSCAN algorithm, while plot (b) displays the density clustering results.

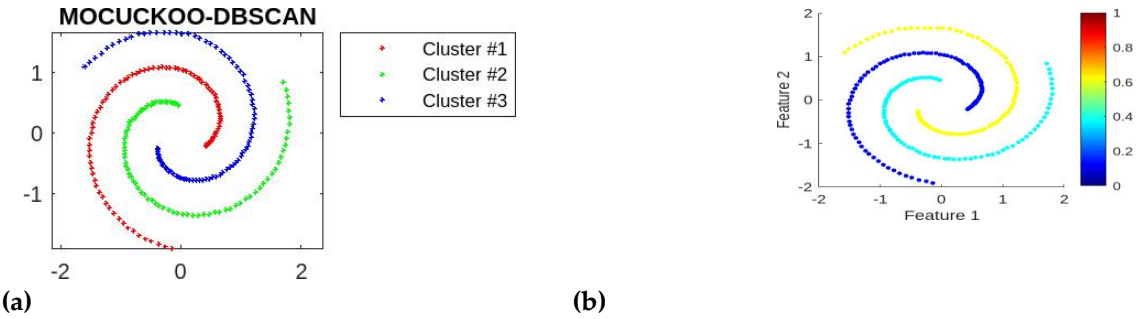


Fig. 5 Plot (a) shows the clustering results for the **spiral** data set using the MOCUCKOO-DBSCAN algorithm, while plot (b) displays the density clustering results.

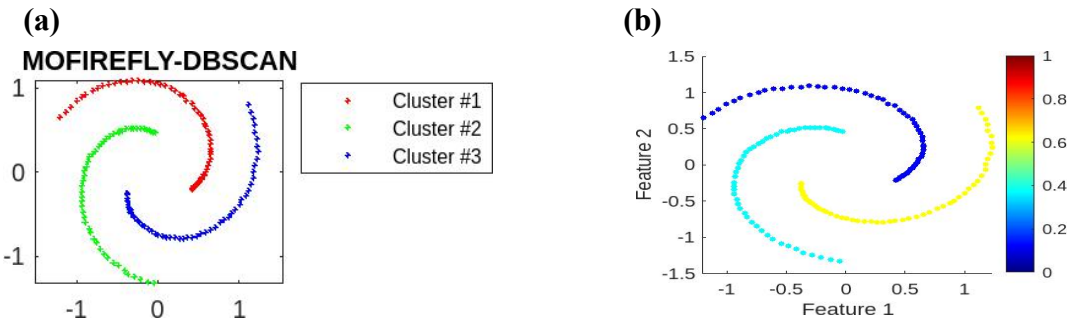
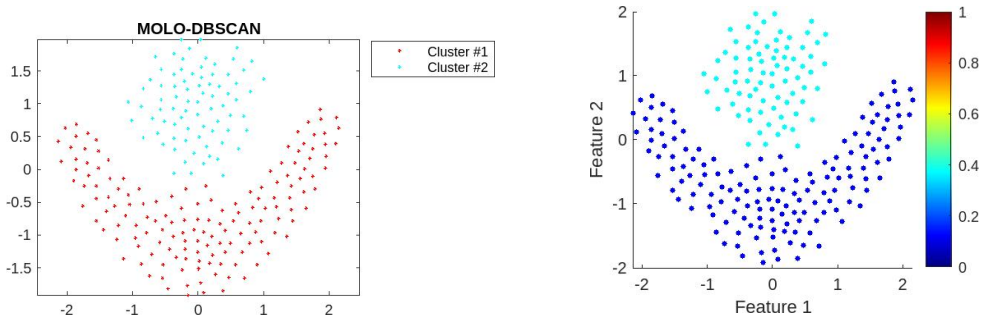
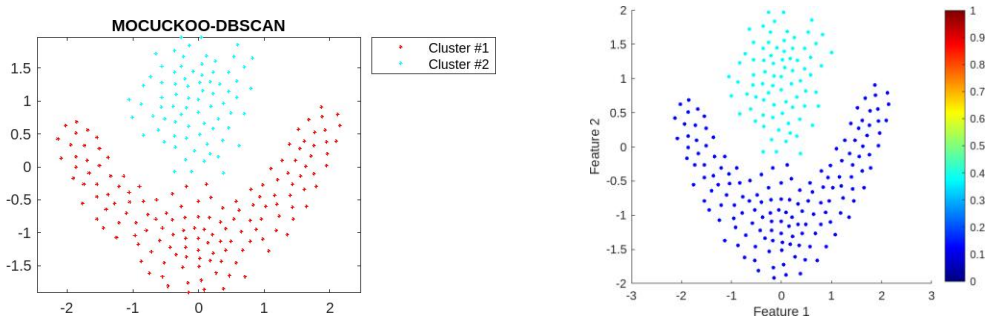


Fig. 6 Plot (a) shows the clustering results for the **spiral** data set using the MOFIREFLY-DBSCAN algorithm, while plot (b) displays the density clustering results.



(a) **(b)**

Fig. 7 Plot (a) shows the clustering results for the **flame** data set using the MOLO-DBSCAN algorithm, while plot (b) displays the density clustering results.



(a) **(b)**

Fig. 8 Plot (a) shows the clustering results for the **flame** data set using the MOCUCKOO-DBSCAN algorithm, while plot (b) displays the density clustering results.

(a) **(b)**

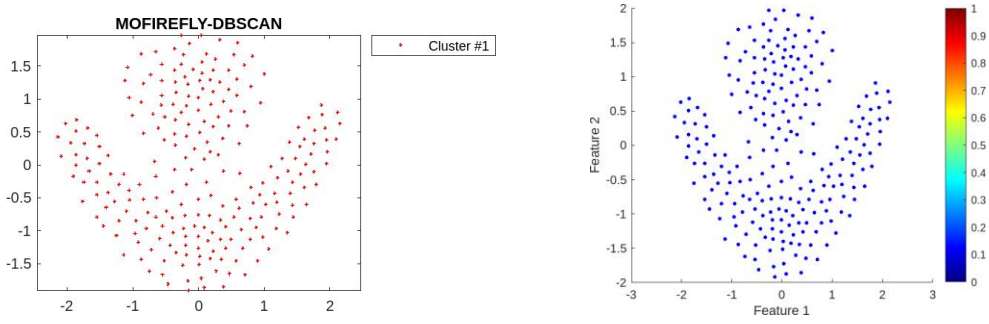
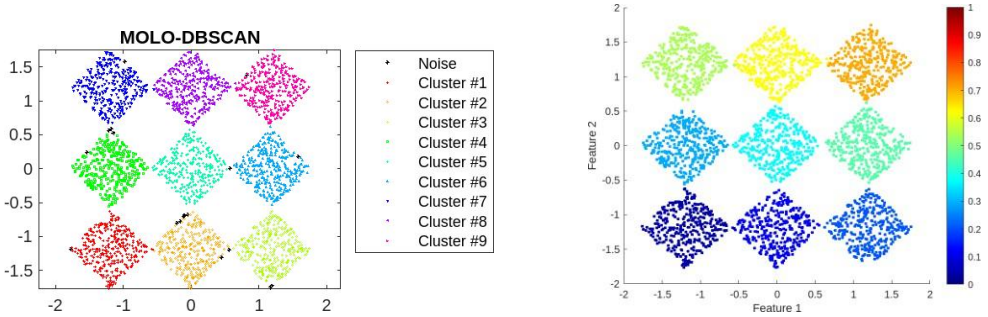
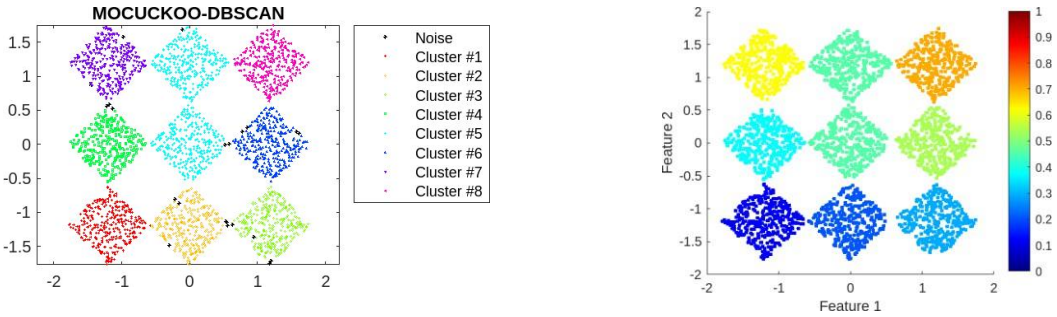


Fig. 9 Plot (a) shows the clustering results for the **flame** data set using the MOFIREFLY-DBSCAN algorithm, while plot (b) displays the density clustering results.



(a) **(b)**

Fig. 10 Plot (a) shows the clustering results for the **diamond9** data set using the MOLO-DBSCAN algorithm, while plot (b) displays the density clustering results.



(a) **(b)**

Fig. 11 Plot (a) shows the clustering results for the **diamond9** data set using the MOCUCKOO-DBSCAN algorithm, while plot (b) displays the density clustering results

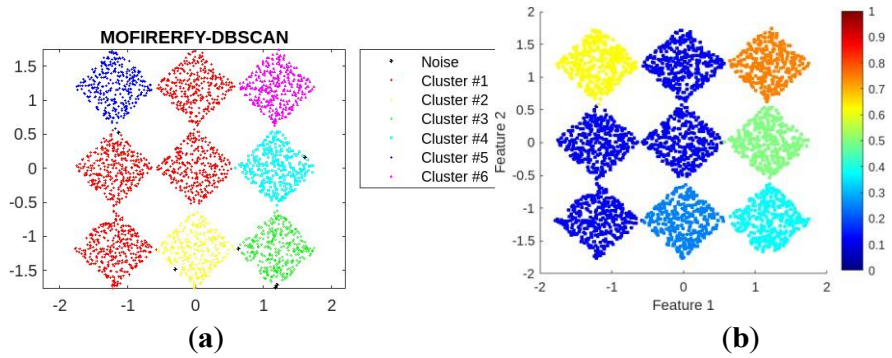


Fig. 12 Plot (a) shows the clustering results for the **diamond9** data set using the MOFIREFLY-DBSCAN algorithm, while plot (b) displays the density clustering results.

The results of three different meta-heuristic algorithms' evaluations, which were utilized to improve the DBSCAN clustering method on the different data sets, are listed in Table 4.
The figures accompanying the tables depict clustering results from the upgraded DBSCAN algorithm using three meta-heuristic variants (MOLO-, MOCUCKOO-,and MOFIREFLY-DBSCAN). Each figure contains two visuals: the left showing clustered data points and the right a heat map indicating density, darker shades represent denser clusters.

Figures (1 - 3) show how the data points in the aggregation data set cluster into 7 separate groups. Only the MOLO-DBSCAN as opposed to others is the only approach that correctly identifies all 7 clusters, with its heat maps showing clear, compact groupings. On the spiral dataset, MOLO-DBSCAN again performs best, yielding accurate, pattern-aligned clusters, while MOCUCKOO is moderately accurate and MOFIREFLY is the least effective as seen in Figures (4 - 6).. Density plots reinforce these findings: high-density clusters appear darker, low-density clusters lighter.

In the Flame dataset, MOLO- and MOCUCKOO-DBSCAN outperform MOFIREFLY in metrics like completeness and V-measure, generating well-shaped clusters Figures (7- 9). On the Diamond09 dataset, MOLO-DBSCAN achieves the highest accuracy (0.9927), followed by MOCUCKOO and MOFIREFLY. Rand Index values support these results, indicating strong clustering similarity, especially for MOLO-DBSCAN.

Final visualizations depicted in Figures (10 - 12), clearly indicates that MOLO-DBSCAN forming tight, accurate clusters, MOCUCKOO showing moderate performance, and MOFIREFLY failing to reflect the true data structure.

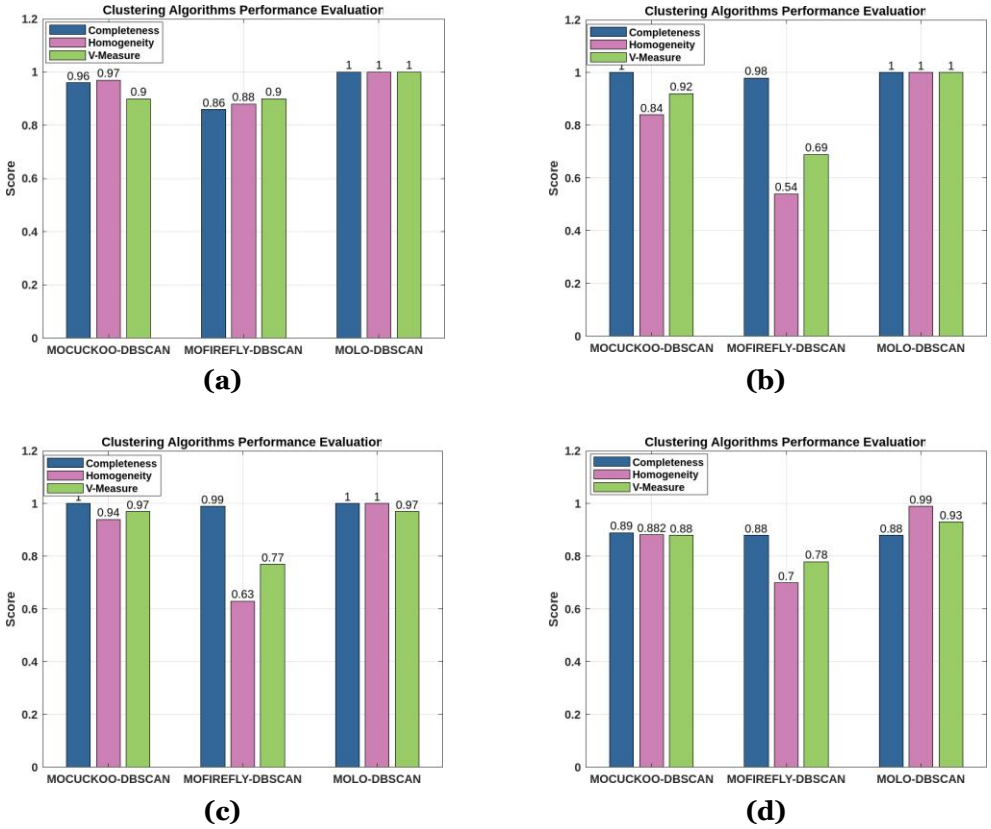


Fig. 13 The assessment of Completeness, Homogeneity, and V-Measure metrics is conducted on six different datasets using three distinct algorithms. These datasets are :(a) aggregation dataset (b) spiral dataset (c) flame dataset (d) diamond9 dataset

a) All three algorithms perform similarly well across all metrics, with **MOLO-DBSCAN** slightly outperforming the others, especially in **Homogeneity** and **V-measure**.

b) **MOLO-DBSCAN** again shows higher performance, with a noticeable drop in **MOFIREFLY-DBSCAN** (Completeness and Homogeneity < 0.6), making it less reliable in this scenario.

c) All algorithms perform closely, with **MOLO-DBSCAN** and **MOCUCKOO-DBSCAN** slightly ahead. **MOFIREFLY-DBSCAN** shows a small dip.

d) **MOLO-DBSCAN** consistently delivers the highest scores across metrics, while **MOFIREFLY-DBSCAN** remains weaker in **Homogeneity** and **V-measure**

5. Case Study

This study employs the DBSCAN algorithm to cluster the locations of weather stations across Canada. The primary objective is to identify groups of weather stations with similar weather conditions. This task involves spatial data, making DBSCAN an ideal choice due to its ability to discover clusters of arbitrary shapes and its resistance to noise. The data set is extracted from the Kaggle repository [57].

The data frame comprised 1341 rows and 25 columns. The features incorporated in this study include Min Temp(T n), Max Temp(T x), and Mean Temp(T m).After dropping NAN values from each row of the above features the data set comprised of 1255 rows and 25 columns.

The longitudes and latitudes in our data frame are further converted into x/y map projection coordinates. These map projection coordinates will serve as attributes for spatially clustering the data points, in conjunction with temperature information. The glimpse of various weather stations in Canada is highlighted in Figure 14.

Afterward, the MOLO-DBSCAN algorithm is applied to the weather data set which generates nine clusters with parameter values of *Eps* = 0.29 and *Minpts* = 9 by passing temperature and map coordinates as input to the clustering algorithm.

Nine clusters (Cluster 0–8) and noise points (-1) were identified, each visualized with distinct colors Figure 15. Average mean temperature was calculated for each cluster, revealing spatial temperature patterns, with details summarized in

Table 5. Cluster 0 showed the highest average (≈6.24°C), and Cluster 8 the lowest (≈-0.55°C), indicating warmer and colder regions, respectively. This clustering highlights regional temperature similarities among weather stations.

Fig. 14 Different weather stations in Canada plotted on a map

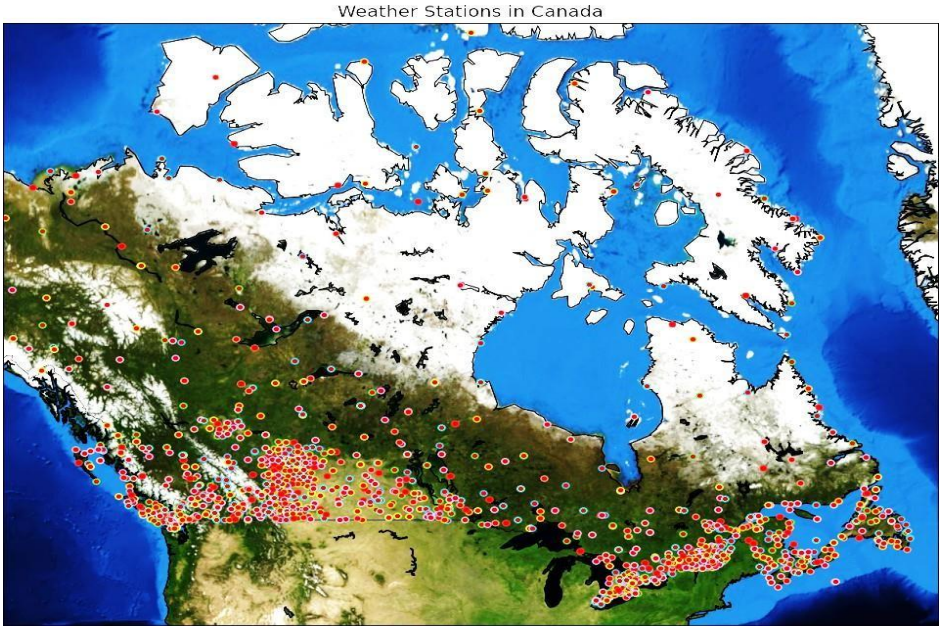


Table 5 Summary of Clustering Results for Mean Temperature

Number of Clusters	Number of Stations	Average Mean Temperature
Cluster # 0	177	6.240
Cluster # 1	22	-0.550
Cluster # 2	8	-2.949
Cluster # 3	250	-13.800
Cluster # 4	53	-4.190
Cluster # 5	319	-15.857
Cluster # 6	6	-22.845
Cluster # 7	9	-7.989
Cluster # 8	16	-4.706

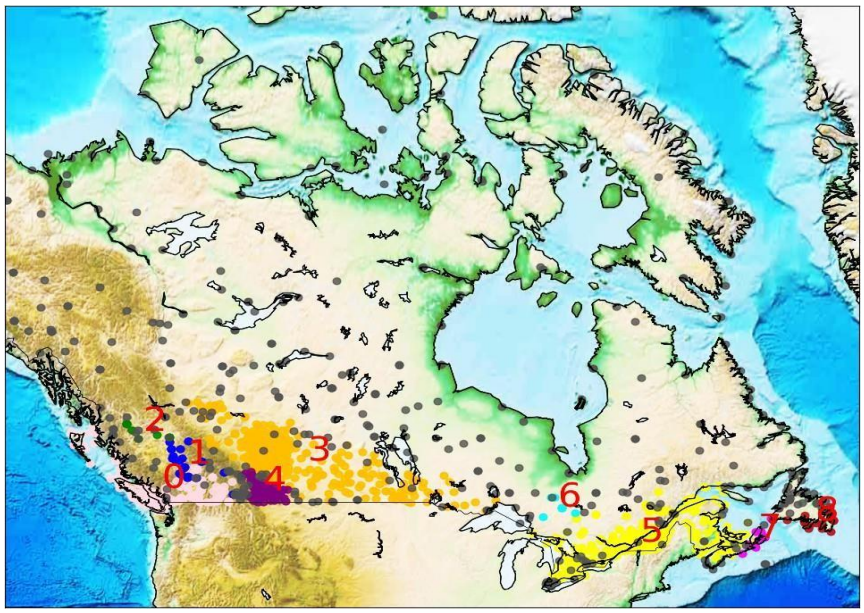


Fig. 15 Temperature trend of Weather Stations in Canada Clustered at *Eps* = 0.29 and *Minpts* = 9

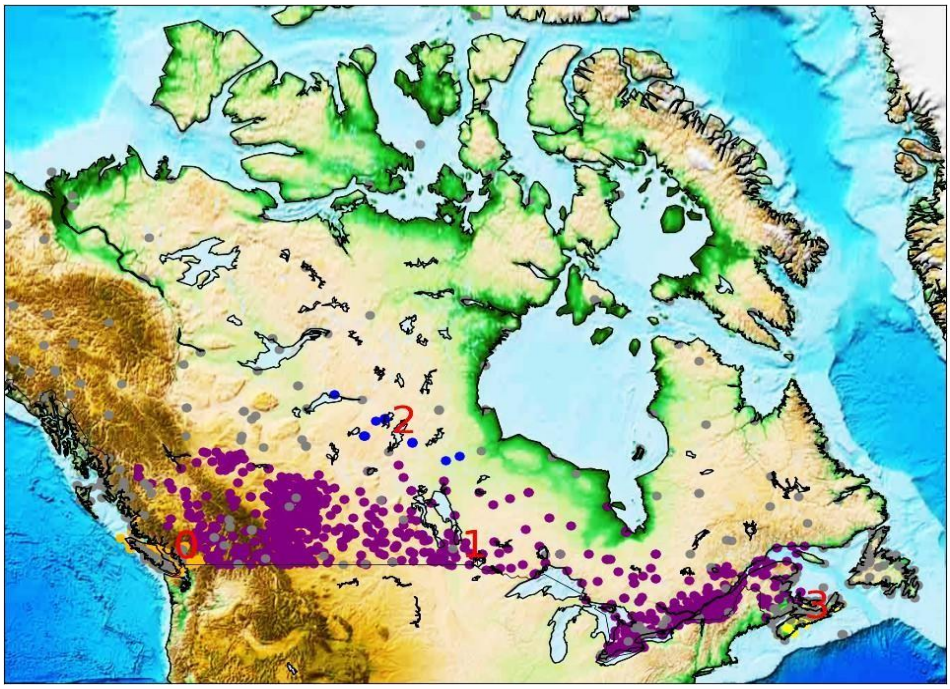
The MOLO-DBSCAN algorithm is applied to the weather dataset using temperature, precipitation, and map coordinates as input features, with parameters set to Eps = 0.48 and MinPts = 10. The clustering process identifies four distinct clusters (Cluster 0–3) and labels outliers as noise (-1). These clusters are visually mapped (Figure 23) and analyzed for precipitation patterns

Table 6, revealing whether data points within a cluster share similar precipitation levels. Precipitation plays a key role in both clustering and interpretation, enabling spatial identification of wetter or drier regions and offering insights into typical weather conditions across clusters. Each cluster is visually distinguishable on the map, as depicted in Figure 16.

Table 6 Summary of Clustering Results for Mean Temperature

Number of Clusters	Number of Stations	Average Mean Temperature	Average Mean Precipitation
Cluster # 0	79	7.663	127.332
Cluster # 1	725	-12.496	27.313
Cluster # 2	11	-25.609	10.008
Cluster # 3	14	-9.336	155.790

Fig. 16 Temperature and Precipitation trend of Weather Stations in Canada Clustered at *Eps* = 0.48 and *Minpts* = 10



6. Conclusion

This study has presented a novel and efficient approach for optimizing the parame-ters of the DBSCAN algorithm using the innovative Multiobjective Lemurs Optimizer Algorithm. The utilization of a multiobjective optimization strategy addresses the challenges associated with tuning DBSCAN parameters, which often involves finding a delicate balance between density thresholds and cluster radii, influencing the quality and efficiency of clustering results. The MOLO algorithm demonstrated its effective-ness in this task, offering a powerful and flexible solution for parameter optimization in the DBSCAN.

The testing findings demonstrated the proposed approach's superiority to both manual parameter adjustment and traditional strategies. The Davies Bouldin and Cluster Validity Density Involved Distance indices are employed as the objective functions that played a significant part in outcomes correctness by exploiting the multiobjective character of the lemurs optimizer algorithm. The proposed approach achieves improved clustering accuracy while simultaneously enhancing the computational efficiency of the DBSCAN algorithm.

This research not only contributes to the field of clustering and data mining but also underscores the significance of utilizing advanced optimization algorithms to handle complex parameter optimization tasks in machine learning and data analysis. Furthermore, the insights gained from this study have broader implications for other clustering algorithms and optimization problems in various domains. The adaptability and efficiency of the MOLO algorithm suggest its potential for addressing complex optimization challenges beyond DBSCAN parameter tuning. The success of this research highlights the importance of exploring innovative techniques that har-ness the power of nature-inspired algorithms for solving real-world problems in data science.

However, it's critical to recognize this study's constraints. The efficiency of the MOLO method on advanced spatial data sets and in more complicated clustering conditions should be explored for additional research despite the algorithm's encour-aging results. Another area for improvement could be to make the MOLO algorithm more scalable by employing more objective functions so that it can successfully handle exceptionally large amounts of data.

References

[1] Armano, G., Farmani, M.R (2016) Multiobjective clustering analysis using particle swarm optimization. Expert Syst Appl 55:184–193

[2] Xie, J., Gao, H., Xie, W., Liu, X., Grant, P.W. (2016) Robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors. Inf Sci 354:19-40 https://api.semanticscholar.org/CorpusID:38141526

[3] Mete, M., Kockara, S., Aydin, K (2011) Fast density-based lesion detection in dermoscopy images. Comput Med Imaging Graph 35:128–36

[4] Santra, A. K., Christy, C. J. (2012) An Efficient Document Clustering by Optimization Technique for ClusterOptimality.Int J Comput Appl 975:8887

[5] Hatamlou, A.S..N.-p.H. A (2012) A combined approach for clustering based on k-means and gravitational search algorithms. Swarm Evol. Comput. 6:47–52

[6] Chen, X. (2015) A new clustering algorithm based on near neighbor influence.Expert Syst Appl 42(21):7746-7758 https://doi.org/10.1016/j.eswa.2015.05.007

[7] Salem, S.B., Naouali, S., Chtourou, Z. (2018) A fast and effective partitional clus-tering algorithm for large categorical datasets using a k-means based approach. Comput Electr Eng 68:463-483

[8] Elshourbagy, M., Hemayed, E., Fayek, M (2016) Enhanced bag of words using multilevel k-means for human activity recognition. Egyptian Info J 17:227–237

[9] Tian, Tian, K., Li, J., Zeng, J., Evans, A., Zhang, L. (2019) Segmentation of tomato leaf images based on adaptive clustering number of K-means algorithm. Comput. Electron Agric 165 https://api.semanticscholar.org/CorpusID:202101023

[10] Han, M.P.J. Jiawei Kamber (1999) Data mining: Concepts and tech-niques.585–631

[11] Ester, M., Kriegel, H., Sander, J., Xu, X. (1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Adv Intel Soft Compu https://api.semanticscholar.org/CorpusID:355163

[12] Jiang, H., Li, J., Yi, S., Wang, X., Hu, X (2011) A new hybrid method based on partitioning-based dbscan and ant clustering. Expert Syst Appl 38:9373–9381 https://doi.org/10.1016/j.eswa.2011.01.13534

[13] Dharni.,C.,B., M (2013) An improvement of dbscan algorithm to analyze cluster for large datasets. International Conf MOOC Innovation and Techno Educ (MITE) 42–46 https://api.semanticscholar.org/CorpusID:19512906

[14] Fan, X., Yue, Y., Sarkar, P., Wang, R. (2020) On hyperparameter tuning in general clustering problems. https://api.semanticscholar.org/CorpusID:221082870

[15] Ditton, E., Swinbourne, A., Myers, T.S., Scovell, M (2021) Applying semi-automated hyperparameter tuning for clustering algorithms. ArXiv:2108.11053 https://api.semanticscholar.org/CorpusID:237289767

[16] Mitra, .N.J. S (2011) Kddclus A simple method for multi-density clustering. pp. 72–76

[17] Darong, H., Peng, W (2012) Grid-based dbscan algorithm with referential parameters. Physics Procedia 24:1166–1170

[18] Smiti, A., Elouedi, Z. (2012). DBSCAN-GM: An improved clustering method based on Gaussian Means and DBSCAN techniques.IEEE 16th Int

Conf.(INES) 573-578

[19] Diao, K., Liang, Y., Fan, J. (2018) An Improved DBSCAN Algorithm Using Local Parameters.

[20] Hosseini Rad, M., Abdolrazzagh-Nezhad, M. (2020) A new hybridization of DBSCAN and fuzzy earthworm optimization algorithm for data cube cluster-ing.Soft Comput 24(20):15529-15549

[21] Zhu, Q., Tang, X., Elahi, A. (2021) Application of the novel harmony search optimization algorithm for DBSCAN clustering. Expert Syst Appl 178(Complete) https://doi.org/10.1016/j.eswa.2021.115054

[22] Xiong, Z., Chen, R., Zhang, Y., Zhang, X. (2012) Multi-density DBSCAN algorithm based on density levels partitioning. J Inf Comput Sci 9(10):2739-2749

[23] Li, M., Bi, X., Wang, L., Han, X. (2021) A method of two-stage clustering learning based on improved DBSCAN and density peak algorithm. Comput Comm 167:75-84

[24] Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math 20:53-65

[25] Perafan-Lopez, J. C., Ferrer-Gregory, V. L., Nieto-Londoño, C., Sierra-P´erez, J. (2022) Performance Analysis and Architecture of a Clustering Hybrid Algorithm Called FA+GA-DBSCAN Using Artificial Datasets. Entropy-Switz 24(7):875 https://doi.org/10.3390/e24070875

[26] Balavand, A. (2022) Combination of a New Metaheuristic Algorithm Based on Cooperative Grouper Fish-Octopus and DBSCAN Algorithm to Automatic Clustering Wang, L., Wang, H., Han, X., Zhou, W. (2021) A novel adaptive density-based spatial clustering of application with noise based on bird swarm optimization algorithm. Comput Comm 174:205-214

[27] Wang, L., Wang, H., Han, X., Zhou, W. (2021) A novel adaptive density-based spatial clustering of application with noise based on bird swarm optimization algorithm. Comput Comm 174:205-214

[28] Yang, Y., Qian, C., Li, H., Gao, Y., Wu, J., Liu, C. J., Zhao, S (2022) An efficient DBSCAN optimized by arithmetic optimization algorithm with opposition-based learning. J Supercomput 78(18):19566-19604

[29] Karami, A., Johansson, R. (2014) Choosing DBSCAN Parameters Automatically using Differential Evolution. Int J Comput Appl 91:1-11

[30] Lai, W., Zhou, M., Hu, F., Bian, K., Song, Q. (2019) A New DBSCAN Parameters Determination Method Based on Improved MVO. IEEE Access 7:104085-104095

[31] Falahiazar, Z., Bagheri, A., Reshadi, M. (2021) Determining the Parameters of DBSCAN Automatically Using the Multi-Objective Genetic Algorithm. J Inf Sci Eng 37:157-183

[32] Dunn, J. C. (1973) A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters.Journal of Cybernetics 3:3,32-57 https://doi.org/10.1080/01969727308546046

[33] Abasi, A. K., Makhadmeh, S. N., Al-Betar, M. A., Alomari, O. A., Awadallah, M. A., Alyasseri, Z. A. A., Hadjouni, M. (2022) Lemurs optimizer: A new metaheuristic algorithm for global optimization. Appl Sci 12(19):10057

[34] Maulik, U., Bandyopadhyay, S., Mukhopadhyay, A. (2011) Multiobjective Genetic Algorithms for Clustering. Appl Data Min Bioinformat

[35] Kalyanmoy, D. (2001) Multi-objective optimization using evolutionary algo-rithms.

[36] Coello Coello, C.A. (1999) A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. Knowl Inf Syst 1:269-308.

[37] Erfani, T. and Utyuzhnikov, S. V. (2011) Directed search domain: a method for even generation of the pareto frontier in multiobjective optimization. Eng Optimiz 43(5):467–484

[38] Marler, R. T., Arora, J. S. (2004) Survey of multi-objective optimization methods for engineering. Structural and multidisciplinary optimization 26:369-395

[39] Ba˘canin, N. (2013) Firefly Algorithm Applied to Integer Programming Problems.

[40] Baghlani, A., Makiabadi, M. H., Sarcheshmehpour, M. (2014). Discrete optimum design of truss structures by an improved firefly algorithm. Advances in Structural Engineering 17(10):1517-1530

[41] Davies, D. L., Bouldin, D. W. (1979). A cluster separation measure. IEEE transactions on pattern analysis and machine intelligence (2):224-227

[42] Hu, L., Zhong, C. (2019) An internal validity index based on density-involved distance. IEEE Access 7:40038-40051

[43] Deb, K. (2012) Optimization for engineering design: Algorithms and examples.

[44] PHI Learning Pvt. Ltd Schubert, E., Sander, J., Ester, M., Kriegel, H. P., Xu, X. (2017) DBSCAN revis-ited, revisited: why and how you should (still) use DBSCAN. ACM Transactions on Database Systems (TODS) 42(3):1-21.

[45] Khan, M. M. R., Siddique, M. A. B., Arif, R. B., Oishe, M. R. (2018) ADBSCAN: Adaptive density-based spatial clustering of applications with noise for identifying clusters with varying densities. IEE Conf Publ (iCEEiCT) pp. 107-111 https:// doi.org/10.1109/CEEICT.2018.8628138

[46] Zhu, Q., Tang, X., and Elahi, A. (2021) Application of the novel harmony search optimization algorithm for dbscan clustering. Expert Syst Appl 178:115054.

[47] Yang, X.-S. and Deb, S. (2013) Multiobjective cuckoo search for design optimiza-tion. Comput Oper Res 40(6):1616–1624

[48] Yang, X.-S. (2013) Multiobjective firefly algorithm for continuous optimization. Eng Comput 29:175–184

[49] Fr¨anti, P., Sieranoja, S. (2018) K-means properties on six clustering benchmark datasets. Appl intel 48:4743-4759

[50] Thrun, M. C., Ultsch, A. (2020). Clustering benchmark datasets exploiting the fundamental clustering problems. Data in brief 30:105501

[51] Liu, Y., Li, Z., Xiong, H., Gao, X., Wu, J. (2010) Understanding of internal clustering validation measures. IEEE Conf Publ (iCEEiCT) pp. 911-916

[52] Craenendonck, T.V., Blockeel, H. (2015) Using internal validity measures to compare clustering algorithms. Int Conf Publ

[53] Gholizadeh, N., Saadatfar, H., Hanafi, N. (2021). K-DBSCAN: An improved DBSCAN algorithm for big data. J Supercomput 77:6214-6235

[54] Zhao, Y., Karypis, G. (2004) Empirical and theoretical comparisons of selected criterion functions for document clustering. ML 55:311-331

[55] Zhang, H., Guo, H., Wang, X., Ji, Y., Wu, Q. J. (2020) Clothescounter: a framework for star-oriented clothes mining from videos. Neurocomputing 77:38-48

[56] Hirschberg, J. B., Rosenberg, A. (2007) V-Measure: a conditional entropy-based external cluster evaluation.

[57] https://www.kaggle.com/