Journal of Electrical and Computer Engineering Innovations
(JECEI)
Journal homepage: http://www.jecei.sru.ac.ir

Research paper

A Novel Clustering Algorithm based on Natural Neighborhood and Radial Distribution Function

Mohammad Asadpour [1,*], ***Shahin Pourbahrami*** [2]

[1] *Faculty of Electrical and Computer Engineering, university of Tabriz, Tabriz, Iran.*

[2] *Department of Computer Engineering, National University of Skills (NUS), Tehran, Iran.*

Article Info

Article History:

Received 06 Octobber 2025
Reviewed 03 December 2025
Revised 21 December 2025
Accepted 30 December 2025

Keywords:

Clustering
Density peaks
Natural neighbor
Radial distribution function
Cut-off parameter
Density points

[*]Corresponding Author's Email Address:
m_asadpour@tabrizu.ac.ir

Abstract

Background and Objectives: Density Peak Clustering (DPC) is a modern density-based algorithm designed to efficiently identify cluster centers by constructing a decision graph. In this graph, points with high local density and large distances from other high-density points are selected as cluster centers. After the centers are determined, the remaining non-central points are assigned to clusters based on their proximity to the nearest center. Despite its advantages, DPC performs poorly on manifold datasets with varying densities and is highly sensitive to the choice of the cut-off distance parameter.

Methods: To overcome these limitations and enhance clustering performance, this study proposes an approach that utilizes the radial distribution function in combination with a neighborhood graph to quantify the relationship between data points and high-density regions. This function allows estimating the probability of finding neighboring points around a central or dense point, and the resulting histogram provides a representation of these relationships.

Results: The approach was implemented using the natural neighbor algorithm combined with the radial distribution function in both MATLAB and Python environments.

Conclusion: Experimental results demonstrate that the proposed algorithm significantly improves clustering accuracy while reducing execution time compared to existing approaches.

How to cite this paper:
M. Asadpour, S. Pourbahrami, "A novel clustering algorithm based on natural neighborhood and radial distribution function," J. Electr. Comput. Eng. Innovations, 14(2): 351-364, 2025.

DOI: 10.22061/jecei.2025.12365.870

URL: https://jecei.sru.ac.ir/article_2493.html

Introduction

Clustering algorithms represent a fundamental component of data mining for analyzing unlabeled data, with diverse applications across many domains. The objective of clustering is to automatically group data objects such that intra-cluster distances are minimized while inter-cluster distances are maximized [1]-[3]. These algorithms have found widespread use in various fields, including power transformer health assessment [4], big data analytics [5], wireless networks [6], economics [7], [8], and social data analysis [9], [10]. One of the most prominent clustering paradigms is density-based clustering [11]. This method is based on the concept that clusters are regions of higher data density, separated by areas of lower density. In such algorithms, points that lie within a certain radius (i.e., in the same neighborhood) are grouped, with a minimum density threshold specified to ensure that only regions meeting this threshold are considered true clusters; sparse regions are ignored or treated as noise.

Density Peak Clustering (DPC) is a novel density-based algorithm that identifies cluster centers by constructing a decision graph and then assigning non-central points to those centers to form clusters [12], [13]. Although DPC is capable of detecting clusters of arbitrary shapes, it suffers from notable limitations: (1) performance on datasets with varying densities: when clusters in a dataset have differing density levels, DPC often struggles, leading to suboptimal clustering accuracy; (2) sensitivity to the cutoff parameter: the algorithm's results depend heavily on the chosen cutoff distance, making it difficult to find a universally optimal parameter across datasets.

To address these challenges and improve clustering robustness, this paper integrates the concepts of natural neighborhood and radial distribution function to more reliably detect dense data points. In our approach, points with higher neighbor counts are treated as candidate dense centers. We then use the radial distribution function to quantify relationships between data points and these dense points, such that the strength of the relation decays with distance. By visualizing these relationships as strong peaks, we can estimate the likelihood of surrounding points belonging to those dense centers and build a histogram of these associations to guide cluster formation.

Natural Neighborhood and Radial Distribution Function: This method addresses some of the key challenges in DPC, such as the ability to handle datasets with varying densities or noisy data, making it more suitable for complex real-world applications.
Overview: It combines natural neighborhood concepts with the radial distribution function to improve local density estimation.
Improvements: It calculates the density relationships between data points and dense regions, considering how these relationships diminish with distance.
Benefit: It enables more accurate detection of dense clusters and outliers in datasets with varying density.

The structure of this paper is as follows: Section 2 provides a review of related works, particularly focusing on various density-based clustering methods. Section 3 introduces the core concepts underlying the proposed algorithm. Section 4 offers an analysis of the method's complexity. The experimental results and performance evaluations are detailed in Section 5, while Section 6 presents the conclusions and final remarks.

Research Background

Clustering is a machine learning technique that focuses on grouping data points based on their similarities. Given a set of data points, a clustering algorithm categorizes each point into a specific group, or cluster. Ideally, the points within a single cluster should share similar characteristics or features, while points in different clusters should exhibit distinct differences. As an unsupervised learning method, clustering is widely used in statistical data analysis and has applications across numerous fields. In this section, three groups of commonly used clustering methods have been examined as related works, which include kernel-based clustering, density-based clustering used to improve the DPC algorithm, and geometry-based clustering algorithms [14].

Algorithms Based on Kernel

In 2019, Liu et al. investigated an enhanced multi-kernel clustering algorithm [15]. They observed that conventional multiple kernel clustering (MKC) methods, which classify data by combining information from a set of predefined kernels, often neglect the local density surrounding individual data points. This oversight limits the representational capacity of the "optimal" kernel and reduces clustering effectiveness. To remedy these drawbacks, they proposed an algorithm called Optimal Neighborhood MKC with Adaptive Local Kernels (ON-ALK). In ON-ALK, adaptive local kernels are built for each data point by considering its local density; thus, each sample can choose a distinct number of neighbors. The algorithm refines the neighborhood region by applying a weighted combination of the predefined kernels, thereby strengthening the kernel's representational power and improving clustering performance.

In a 2024 paper, Zhu et al. proposed a clustering algorithm that leverages neighborhood relationships in conjunction with a Gaussian kernel [16]. Spectral clustering is widely employed for clustering, particularly when dealing with non-convex datasets and complex structures. Its success, however, hinges crucially on the

construction of the similarity graph. To enhance clustering quality, Zhu et al. introduced a novel similarity measure called SC-NR, which computes similarity using the Gaussian kernel while incorporating neighborhood order information. Recognizing that raw Euclidean distance may inadequately capture deeper relationships between data points, SC-NR refines distance measurement by applying a weight based on each point's rank among its nearest neighbors. This results in a weighted Euclidean distance that more accurately reflects local affinities and yields a better similarity matrix for spectral clustering.

A new and efficient Apollonius kernel introduces an innovative method for managing Gaussian kernel bandwidth [17]. The core idea involves replacing the standard inner multiplication in the objective function with a nonlinear function that maintains many of the original properties while offering greater flexibility. The Apollonius kernel is characterized by a normal distribution and automatically adjusts the neighborhood bandwidth (variance) based on the dataset, removing the need for manual sigma parameter tuning. The RCNAK clustering algorithm, which utilizes this Apollonius kernel, follows four main steps [18]: 1. Natural Neighbor Estimation: Identifying natural neighbors of each data point based on local density and neighborhood relationships. 2. Identifying the Main Points: Detecting core or central points that represent each cluster. 3. Merging Cluster Cores: Grouping core points belonging to the same cluster and separating them from cores of other clusters. 4. Apollonius Kernel Assignment: Using the Apollonius kernel to allocate the remaining data points to clusters based on the distance between core points and farthest points, and detecting noise or outliers.

This algorithm is inspired by the concept of neighborhood relationships among data points, formulated through the geometric properties of the Apollonius circle [19], [20].

Algorithms Based On Density for Improved DPC

The Improved Density Peaks Clustering algorithm based on Natural Neighbor with Merging Strategy (IDPC-NNMS) enhances density-based clustering by leveraging natural neighbors and a pooling approach [21]. The algorithm first adaptively identifies a natural neighbor set for each data point to calculate its local density, mitigating the sensitivity to the cutoff parameter, a common limitation in traditional density peak clustering methods. It then selects as many central (or core) points as possible based on their local density. Afterward, labels are assigned to non-central points to form initial sub-clusters. Finally, using a novel merging strategy, the algorithm iteratively combines sub-clusters until the final clustering criterion is satisfied.

In 2023, Ding et al. introduced a study exploring K-means clustering with natural density peaks for cluster detection [22]. While K-means is widely used due to its simplicity, its performance is highly sensitive to the selection of initial cluster centers, which can make accurately identifying multiple clusters challenging. To address this limitation, the authors proposed an improved *K*-means algorithm that enhances both the speed and quality of initial cluster center selection, resulting in better clustering outcomes.

Zhi et. al, (2024) explored a clustering algorithm known as SFKNN-DPC [23], which introduces a method based on density peaks using a weighted distance approach. Traditionally, Euclidean distance is applied to density peak clustering algorithms and their variations. To address the limitations of using Euclidean distance alone, they proposed a standard deviation weighted distance to enhance the effectiveness of the distance measure. This weighted distance accounts for the specific contribution of each feature in determining the similarity between data points. Additionally, the study introduced a divide-and-conquer allocation strategy, based on the proposed weighted distance, incorporating semi-supervised learning and the reciprocal *K*-nearest neighbor (*K*NN) assumption.

Geometry-Based Clustering Algorithms

In their 2019 study, Pourbahrami et al. examined the Neighborhood Construction Algorithm using the Apollonius Area (NCAR) [24]. The primary objective of this algorithm is to identify and reveal geometric patterns within data through data mining, with a particular focus on accurately determining neighborhood locations using the Apollonius circle. This approach enhances the identification of data point neighborhoods. Rather than evaluating all points, NCAR concentrates on those located within the Apollonius circles, thereby improving efficiency and reducing computational time and resources. The application of Apollonius circles allows for more precise neighborhood identification, ultimately contributing to improved clustering results.

In a subsequent study, Pourbahrami et al. (2020) introduced the Neighborhood Construction with Apollonius Region algorithm based on Density (NCARD) [25]. This algorithm determines the neighbors of data points using density information. Its primary objective is to identify sub-clusters within the Apollonius region that exhibit varying densities, thereby enhancing the accuracy and efficiency of density-based clustering methods. By focusing on density variations within these regions, NCARD aims to improve clustering performance, particularly in complex datasets.

In 2022, Pourbahrami et al., studied a Geometric Clustering method based on Natural Neighborhoods (GCNN) [26]. They proposed a clustering approach that

utilizes the concept of natural neighborhoods to determine the local density of data points. The algorithm begins by identifying dense points with a significant number of natural neighbors, which are used to form initial clusters. Cluster heads are then determined at these dense points using a local natural density criterion. A separate method is employed to detect points with fewer natural neighbors, categorizing them as weak or noisy points. The final clusters are produced by excluding these weak or noisy points, leading to more accurate clustering results.

Non-Parametric Clustering Algorithms

Hierarchical clustering techniques can expose the nested relationships within data by presenting patterns in a tree-like structure. However, when applied to complex datasets, they often produce densely packed trees that are hard to interpret. To overcome this limitation, a novel method named Discovery of Multi-Density Hierarchical Structures (DMDHC) has been introduced. This technique defines a fresh form of cluster tree that more effectively encodes hierarchical information [27]. It automatically generates multiple local slices at various levels of the tree, and by leveraging density-based insights, DMDHC represents complex structures in a more compact and intelligible format.

Rather than outputting a fixed clustering, the OPTICS algorithm (Ordering Points to Identify the Clustering Structure) arranges data into an ordering that reflects its density-based structure across a broad spectrum of parameter values. This ordering—known as the cluster-ordering—allows for the discovery of clusters of arbitrary shape and varying density, thereby removing the necessity of choosing a single parameter (such as ε in DBSCAN) [28]. As a result, OPTICS provides a flexible foundation for both automated and interactive cluster analysis, along with intuitive visualization tools for unveiling the underlying structure in the data.

HDBSCAN, proposed by Campello et al. (2015), offers an integrated framework for density-based clustering, outlier detection, and data visualization. At its core, the method computes hierarchical estimates of density level sets, inspired by Hartigan's model of density-contour clusters and cluster trees [29]. This approach generalizes and strengthens prior density-based clustering methods by constructing a full hierarchy of every possible cluster across an infinite continuum of density thresholds under a nonparametric model. The resultant hierarchy can be used directly for various visualization and exploratory tasks. In post-processing, two key outputs are derived: (*i*) a normalized "outlines" score for each object, which unifies local and global perspectives on outliers into a single definition; and (*ii*) a flat clustering solution obtained via local cuts on the cluster tree, thereby supporting both unsupervised and semi-supervised analysis.

Hierarchical clustering constructs a layered arrangement of clusters by iteratively merging or dividing data point groups at different levels, which demands efficient algorithms to handle the process. Within this framework, the choice of linkage method plays a critical role in determining how clusters are merged [30]. Single linkage defines inter-cluster distance by the closest pair of points, often capturing elongated or chain-like clusters but being sensitive to noise. Complete linkage, on the other hand, uses the farthest pair of points, producing more compact and spherical clusters though sometimes disrupting natural structures. Ward's method seeks to minimize the total within-cluster variance, typically yielding clusters that are balanced and interpretable.

Technical Work Preparation

A. Overview

In this section, we present a robust clustering algorithm that leverages neighborhood estimation and the radial distribution function. The proposed algorithm aims to address common challenges faced by density-based algorithms, such as the density peak algorithm, and consists of four key steps: 1) estimation and identification of natural neighbors [31], 2) checking the radial distribution of data using its physical concept, 3) extracting the nuclei of a cluster as dense atoms, 4) assigning the remaining points to the clusters and finding outlier points based on their position in their radial distribution. In this research, the concept of radial distribution function has been used, so that the relationship between data points and high-density points can be calculated. As the distance between data points increases, the relationship between them decreases, and these are shown by using strong peaks. It is also possible to find the probability of particles around the central atom or dense data points and form a histogram of these connections. By obtaining the natural neighborhood of all the points and then the density of the data points by calculating the distance, the temperature and the density of the points are determined as candidate points for clustering using the radial distribution function, which is the advantage of this method over other methods. Based on the density peaks, there is no need for the distance cut-off parameter, therefore the introduced method is considered an optimal method in terms of accuracy and time.

B. Explaining the Research Method and Implementation Steps of the Proposed Algorithm

In the natural neighbor method, compared to the two-way nearest neighbor approach, it provides a better reflection of data characteristics. The core concept of

this method is based on density division, where data points in high-density areas have more neighbors, points in medium-density areas have fewer neighbors, and points in low-density areas have no neighbors. Using the natural neighborhood algorithm [31], natural neighbors are identified for all data points in the dataset. After determining the natural neighbors, the number of neighbors for each point is calculated. The points are then sorted in ascending order, placing those with the highest number of natural neighbors at the top of the list.

The Radial Distribution Function (RDF), also known as the pair correlation function, describes how particle density varies as a function of distance from a reference particle in a system. It is commonly used to analyze the spatial distribution of particles such as atoms, molecules, or colloids, indicating the probability of finding a particle at a certain distance relative to another particle. RDF provides insights into the structural arrangement and interactions within a material or system, helping to understand its physical properties.

The radial distribution function is a key concept in statistical mechanics for pure fluids and fluid mixtures. It is particularly important because, in addition to providing information about the structure of a liquid, it allows for the calculation of all thermodynamic properties of systems containing molecules with spherical symmetry, assuming a pairwise summable approximation for the potential energy. RDF reflects the ordering of atoms around a reference atom, and more specifically, it is proportional to the probability of finding two atoms separated by a distance $r+\Delta r$. Most simulation programs calculate this function, making it a fundamental tool in statistical mechanics for deriving structural and thermodynamic properties of molecular systems.

C. The Implementation Steps of the Proposed Algorithm

In the proposed algorithm, the radial distribution function is employed to assess the connections between data points and dense points. This is done because the connection between these points weakens as the distance between them increases. These connections are visualized through strong peaks in the RDF. By applying this function, the probability of finding particles around dense data points is calculated, and a histogram is created to depict the relationships between them.

During this phase, data points are sorted based on their distance from each other and their local density. The final dense points are identified through this sorting process. The use of the natural neighborhood concept helps to extract these dense data points, effectively. Specifically, the RDF measures how the density changes as a function of distance from a reference point, capturing the relationships between neighboring points. Importantly, while the dependencies between particles are considered, these dependencies decrease as the distance between them increases. Therefore, cluster heads are selected based on two criteria: having the smallest neighborhood radius and not sharing any natural neighbors with other points. This ensures that the cluster heads are well-separated and representative of distinct dense regions within the dataset.

The procedure can be summarized in the following step-by-step breakdown:

Data Entry

The data is entered into the system to be analyzed.

1. **Natural Neighbor Search:**

A natural neighbor search algorithm checks all the points around a reference point (KDTree) [15].

$$\left(\forall x_i\right)\left(\exists x_j\right)\left(r\in n\right)\vee\left(x_i\neq x_j\right)\rightarrow\left(x_i\in NN_r\left(x_j\right)\right)\vee\left(x_j\in NN_r\left(x_i\right)\right) \quad (1)$$

In (1), the point X_i is the natural neighbor (*NN*) of point X_j, if X_j considers the point X_i as its neighbor and vice versa. The *r* is the search speed (rth-neighborhood of *x*), which increases from 1 to *λ (λ ≤ n)*. *λ* is a parameter for finding natural neighbors.

All points in the natural neighborhood are identified and added to a list called *NN*-List.

This list contains the natural neighboring points for each data point.

The complexity of this algorithm is of order $\mathrm{O}\left(n\log n\right)$.

2. **Neighborhood Analysis:**

For each point in the *NN*-List, its neighbors are examined.

The points with the highest proximity (closest neighbors) are selected and added to the list (GraphSAGE embedding).

3. **Parameter Calculation (Density, Distance, Temperature):**

For each point, the density, distance, and temperature parameters are determined using the function *g (r, t, ρ).*

The probability of finding a point at distance r is calculated using specific equations.

In (2), the local density *ρ(r)* for each point is computed.

In statistical mechanics, the radial distribution function, denoted as g(r), describes how particle density varies as a function of distance from a reference particle. It is defined as:

$$g(r)=\frac{1}{4\pi r^2\rho N}\sum_{i=1}^{N}\sum_{j\neq i}\delta(r-\left\|x_i-x_j\right\|) \quad (2)$$

In the above relationship, g(r) is a factor that is multiplied by the density ρ and produces the local density ρ(r) around an atom.
where:
- N is the total number of particles,
- ρ = N/V is the number density in volume V,
- x_i denotes the position vector of particle i,
- δ(·) is the Dirac delta function (implemented numerically by binning).

4. **Selecting Dense Points**

Among all the points analyzed, the ones with the highest density are selected based on density peak clustering.

In (3), and (4), each point is evaluated as a possible neighbor of others, and the ones with the highest local density are chosen.

$$\begin{cases} \rho_i = \sum_{j \in L.\lambda NN} \frac{1}{d_i + d_{ij}}.\omega. 1 \le L \le 4 \\ \omega = \exp(-1) \end{cases} \quad (3)$$

$$d_i = \sum_{j \in 4\lambda NN} d_{ij} \quad (4)$$

At these equations, the parameter L is the number of layers, d_{ij} is the distance between points X_i and X_j, and λNN represents the nearest neighbor. In equation (3), the maximum number of layers is 4. In a multi-layered system, the points in the first layer represent those closest to the center, while the points in the fourth layer are the farthest from it. Since each layer contributes differently to the calculation of the center point density, the weighting of these layers must also vary accordingly. Layers closer to the center often have a greater impact on density calculations and are typically assigned higher weights, whereas layers further out contribute less and receive lower weights. This differential weighting ensures an accurate representation of the center point density, reflecting the true distribution and influence of surrounding points.

D. Threshold for Distance

O A threshold limit for the distance between points is defined.

O This threshold prevents issues when points are too close together (e.g., zero or very small distances).

O The threshold distance is set at 0.46 based on reference texts.

In the final stage, clustering is performed, and the number of clusters is determined by the number of dense samples. Each dense sample, relative to other points in the initial list, is identified as a cluster head, thereby defining the total number of clusters in the dataset. Any data point that shares a natural neighbor with one of the dense points is assigned to the corresponding cluster. For each point, the sharing of its natural neighborhood with all previously processed points in the sorted list is checked. If the natural neighborhood of the current point overlaps with that of any previous point, the current point is removed. The pseudocode of the proposed algorithm is presented in Fig. 1.

```
Algorithm RDF-Neighborhood (GraphSAGE)-Clustering (X, y, k, pca_n, max_r, dr, iters)
Input:
  X: dataset (n × d)
Output:
  labels, metrics (ARI, NMI, ACC)
1. X ← Standardize(X)
2. X_pca ← PCA (X, n_components=pca_n)
3. Tree ← Build_KDTree(X_pca)
4. g_func ← Compute_RDF (Tree, max_r, dr)
     For each sample subset:
       Compute pairwise distances
       Accumulate histogram of distances
       Compute normalized RDF g(r)
5. A ← Build_ Neighborhood Graph (X_pca, k)
     For each point i:
       Find k nearest neighbors N_i
       For each neighbor j in N_i:
         If i ∈ N_j then:
           w_ij ← exp (-d_ij² / (σ_i * σ_j)) * g_func(d_ij)
           Add edge (i, j, w_ij)
6. H ← X_pca
   For t in 1 to iters:
     For each node i:
       h_i ← α * h_i + (1 - α) * mean_{j ∈ N(i)} h_j
7. Compute local densities:
     ρ_i ← 1 / mean (distance_to_k_neighbors(i))
   Compute δ_i:
     distance to nearest point with higher density
8. γ_i ← ρ_i × δ_i
   Centers ← points with γ_i > mean(γ) + std(γ)
9. Assign each point to nearest center in H-space
10. Evaluate clustering:
     ARI ← AdjustedRandIndex (y, labels)
     NMI ← NormalizedMutualInfo (y, labels)
     ACC ← ClusteringAccuracy (y, labels)
11. Return labels, (ARI, NMI, ACC)
```

Fig. 1: Pseudocode of the proposed algorithm.

Complexity Analysis

The computational complexity of this algorithm can be analyzed in stages. The preprocessing step, including data standardization and PCA, requires $o(nd^2)$ time in

the worst case, where *n* is the number of samples and *d* is the number of features. The construction of the KDTree for nearest neighbor queries has a complexity of $o(n\log n)$, and each *k*-NN search in build_mutual _graph takes $o(n\log n+nk)$, where *k* is the number of neighbors per point. Since the algorithm only keeps mutual connections, the graph construction phase runs in roughly $o(nk\log n)$. The RDF estimation step, which samples pairwise distances and computes histograms, adds approximately $o(nk)$ complexity but operates on a limited subset of points, keeping it near linear time.

For the embedding phase, GraphSAGE performs two iterations of sparse matrix–vector multiplications, which scale as $o(|E|d)$ where $|E|\approx nk/2$ is the number of edges in the mutual *k*-NN graph. The Density Peak Clustering stage involves building another KDTree in the embedded space $o(n\log n)$and computing local densities and distances to higher-density neighbors, resulting in $o(nk)$operations. Overall, the dominant cost across stages is $o(nk\log n+|E|d)$, which scales almost linearly with *n* for moderate *k* and low-dimensional embedding. Memory complexity is $o(nk+nd)$, dominated by storing sparse graph connections and embedding vectors. Thus, the algorithm is computationally efficient and scalable to large datasets when $k\ll n$.

Experimental Results

This section evaluates the effectiveness of the proposed algorithm. It begins by introducing the pre-processing steps and the datasets utilized in the experiments. Following this, the results of the conducted experiments are presented, along with comparisons to other algorithms.

Evaluation Criteria

In (5)-(9), the evaluation criteria—Accuracy (ACC) [32]-[34], Adjusted Rand Index (ARI), the Normalized Mutual Information-score (NMI), and Adjusted Mutual Information (AMI) are shown. These are considered effective metrics in assessing clustering correctness and the quality of cluster separation. These criteria are key in determining the effectiveness of the proposed algorithm.

Accuracy measures how well the clustering results align with the ground truth labels. Since cluster labels are arbitrary, accuracy is computed by first finding the optimal one-to-one mapping between predicted clusters and true classes (typically using the Hungarian algorithm). The metric then reports the proportion of correctly assigned data points relative to the total number of samples. A higher accuracy value indicates better alignment between the clustering solution and the underlying class structure.

The Adjusted Rand Index evaluates the similarity between two partitions (predicted clusters vs. ground truth labels) by considering all pairs of samples and checking whether they are placed in the same or different clusters in both partitions. Unlike the plain Rand Index, ARI corrects for chance grouping, producing values between -1 and 1. A score of 0 indicates random clustering, values close to 1 reflect strong agreement, and negative values suggest worse-than-random performance.

Adjusted Mutual Information measures the agreement between two clustering based on information theory. It quantifies how much information is shared between the predicted clusters and the ground truth classes, while adjusting for chance overlap. AMI values range from 0 (no mutual information, i.e., random clustering) to 1 (perfect matching). Unlike raw Mutual Information, the adjustment makes AMI robust when comparing clustering with different numbers of clusters, making it especially useful for fair performance evaluation across algorithms.

In classification, y_i represents the actual labels and z_i the predicted labels. For accuracy calculation, if a predicted label *x* matches the true label y ($x=y$), the value of $\delta(x,y)$ is set to 1; otherwise, is set to 0.

$$ACC=\sum_{i=1}^{n}\delta(y_i.\frac{map(z_i)}{n}) \tag{5}$$

The Adjusted Rand Index is a measure of the similarity between two data clustering, considering the agreement between the predicted and true labels while adjusting for random chance. It is based on the Rand Index (*RI*), which calculates the proportion of correctly classified point pairs (both in the same cluster or different clusters). *ARI* improves upon RI by incorporating the expected value of the Rand Index under random classification, denoted as $\mathrm{E}[\mathrm{RI}]$, which adjusts for the chance grouping of points. The maximum value of RI, represented as $\max(\mathrm{RI})$, occurs when the two clustering are perfectly aligned. ARI ranges from -1 (indicating no similarity) to 1 (indicating perfect clustering).

$$ARI=\frac{RI-E[RI]}{\max(RI)-E[RI]} \tag{6}$$

$$AMI=\frac{I(U,V)-E\{I(U,V)\}}{\sqrt{H(U)H(V)}-E\{I(U,V)\}} \tag{7}$$

In classification and clustering tasks, *U* represents the true labels, while *V* denotes the predicted labels. The entropy of *U*, denoted as $\mathrm{H}(\mathrm{U})$, measures the degree of uncertainty or randomness within the distribution of the true labels. In contrast, the mutual information $I(U,V)$, on the other hand, quantifies the amount of information shared between *U* and *V*, reflecting how effectively the predicted labels correspond to the actual labels.

In this study, the accuracy of the clustering results produced by the proposed algorithm is assessed using the Normalized Mutual Information-score (NMI) metric. The score is computed according to Equations (8) and (9). This evaluation provides a robust and interpretable measure of how well the algorithm captures the underlying structure of the data.

$$NMI(Y,\ C) = \frac{MI(Y,\ C)}{\sqrt{H(Y)\,H(C)}} \tag{8}$$

$$MI(Y,\ C) = \sum_{y_i \in Y,\ c_i \in C} P(y_i, c_i) \log\left(\frac{P(y_i, c_i)}{P(y_i)\,P(c_i)}\right) \tag{9}$$

Y: the clustering outcomes. C: the actual clustering labels. n: the total data points. n_{ij}: the number of points same in both clusters c_i and y_i cluster. $n_{i.}$ and $n_{.j}$: respectively the number of the same points of y_i and c_i clusters. The probability of a point belonging to both clusters c_i and y_i simultaneously is denoted as $P(y_i, c_i)$ ($\mathrm{P}(\mathrm{y_i}, \mathrm{c_i}) = \frac{\mathrm{n_{ij}}}{\mathrm{n}}$. $\mathrm{P}(\mathrm{y_i}) = \frac{\mathrm{n_i}}{\mathrm{n}}$ and $\mathrm{P}(\mathrm{c_i}) = \frac{\mathrm{n_i}}{\mathrm{n}}$: the probabilities that an instance to clusters y_i and c_i, respectively.

Datasets

In this paper, both artificial and real UCI datasets have been used to evaluate the proposed solution. The synthetic datasets include Aggregation, Flame, Jain, Spiral, Two-circles, Three-circles, Lineblob, Sticks, R15, Blobs, and D31. The Aggregation dataset contains 788 two-dimensional samples, divided into 7 categories. The Flame dataset consists of 240 two-dimensional samples, divided into 2 categories. The Jain dataset includes 373 two-dimensional samples, also divided into 2 categories, while the Spiral dataset contains 312 two-dimensional samples divided into 3 categories. The two-circles dataset contains 400 two-dimensional samples categorized into 2 groups. Other datasets include Lineblob with 266 2D data samples divided into 3 categories, sticks with 512 2D data samples divided into 4 categories, R15 with 600 2D data samples in 5 categories, and D31 with 3100 2D data samples categorized into 4 groups Table 1. provides detailed specifications for each dataset used.

Table 1: Real and synthetic dataset

Synthetic dataset	**#clusters**	**#features**	**#samples**
Jain	2	2	373
Spiral	3	2	312
two-circles	2	2	400
Lineblob	3	2	266
Flame	2	2	240
Aggregation	7	2	788
Compound	6	2	399
Toy	2	2	1480
S2	15	2	5000
R15	5	2	600
D31	4	2	3100
Blobs	4	4	600
Blobs-large1	6	5	1200
Blobs-large2	6	5	15000
Real dataset	**#clusters**	**#features**	**#samples**
Iris	3	4	150
Seeds	3	13	210
Ecoli	8	7	336
Ionosphere	2	34	351
Parkinsons	2	23	195
Dermatology	6	34	366
WDBC	2	30	569
Segmentation	7	19	210
Wine	3	13	178
Optical digits	10	65	5620
Pendigits	10	16	10,992

Results and Discussion

In Fig. 2, which pertains to the Jain and Aggregation datasets, the results obtained after applying the natural neighborhood algorithm and the radial distribution function reveal the identification of dense points. These dense points are recognized as cluster heads in both datasets.

Fig. 3 illustrates the results obtained for the two-circles and Lineblob datasets after applying the proposed algorithm. The outcomes indicate that the method successfully identified dense points, which were designated as cluster heads, on both datasets.

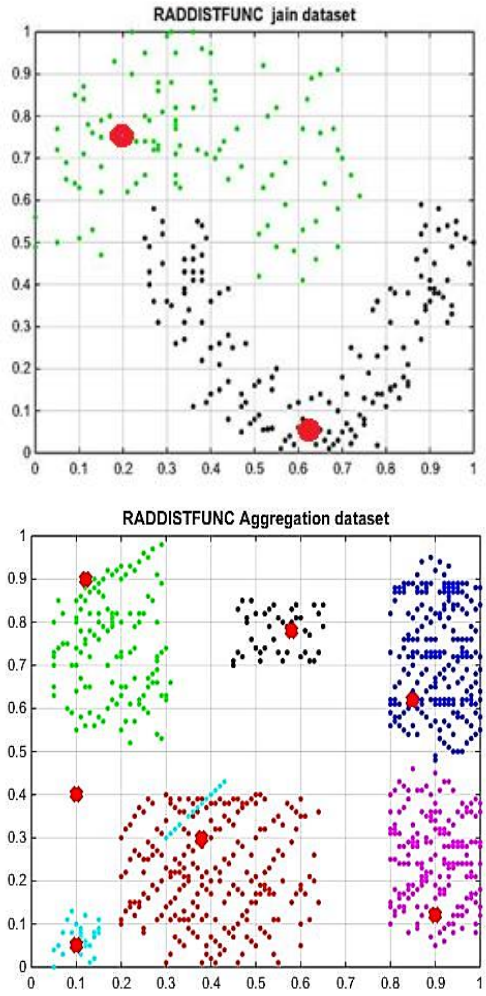


Fig. 2: The result of Jain and Aggregation data set clustering using the proposed algorithm (Large red dots are cluster heads Small dots are clusters separated by the proposed algorithm).

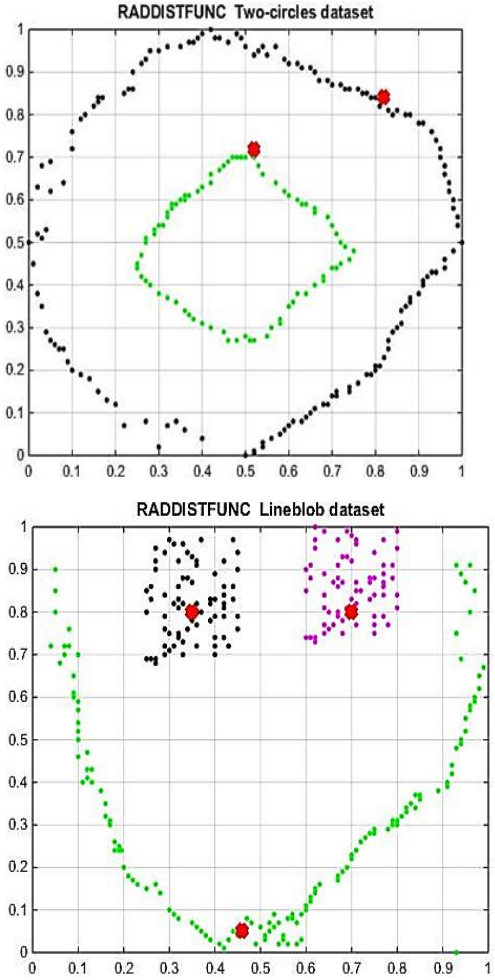


Fig. 3: The result of the two-circles e and Lineblob dataset clustering using the proposed algorithm (Large red dots are cluster heads small dots are clusters separated by the proposed algorithm).

The values obtained for the artificial dataset using the proposed algorithm is compared with those obtained using the IDPC-NNMS clustering method in Table 2. The comparison is based on the criteria of ARI, AMI, and ACC. As shown, the execution time of the proposed algorithm is shorter than that of the IDPC-NNMS algorithm across all synthetic datasets. In terms of ACC, the proposed algorithm consistently performs very well, reaching 0.99 on several datasets, including Jain, Spiral, and R15. However, the AMI and ARI values are generally lower for the proposed algorithm compared to IDPC, suggesting that while the proposed algorithm achieves high accuracy, IDPC may better preserve the underlying structure of some datasets, as reflected by higher AMI and ARI scores, especially on datasets like Jain and Spiral. RDF's execution time is comparable to IDPC, with both methods showing similar times across all datasets. Overall, the proposed algorithm demonstrates robust performance in terms of accuracy but slightly trails IDPC in maintaining internal consistency as measured by AMI and ARI on some datasets.

In Table 2, the results of comparing accuracies using different criteria and the execution times between the proposed algorithm and the method from the reference article on real datasets are presented. The table shows a detailed comparison, highlighting the accuracy of both methods across various evaluation metrics, such as ARI, AMI, and ACC. Additionally, the number of clusters obtained by both methods is also listed, providing insight into the performance and efficiency of each approach on real-world data. The table demonstrates how the proposed algorithm performs in terms of accuracy and time efficiency compared to the reference method. For example, on the Iris dataset, the proposed algorithm achieves an ACC of 0.96 and an ARI of 0.90, comparable to IDPC's results of 0.88 ARI and 0.85 AMI. RDF's execution time is consistently close to that of IDPC. While the proposed algorithm tends to deliver slightly lower AMI and ARI values on some datasets, it shows robust clustering performance across datasets like Wine, WDBC, and Seeds. This suggests that the proposed algorithm is effective for both time efficiency and clustering quality across different datasets.

Table 2: Comparison of the results obtained from the implementation of the proposed algorithm compared to the IDPC method on real data sets and synthetic data sets

Dataset Type	Synthetic/Real Data Set	#Clusters	ACC (Proposed)	ACC (IDPC)	AMI (Proposed)	AMI (IDPC)	ARI (Proposed)	ARI (IDPC)	Time (s) (Proposed)	Time (s) (IDPC)
Synthetic	Jain	2	0.99	0.74	0.78	0.55	0.68	2	310	260
Synthetic	Spiral	3	0.99	0.73	0.77	0.51	0.66	3	300	275
Synthetic	Flame	2	0.97	0.72	0.76	0.48	0.64	2	300	265
Synthetic	Aggregation	7	0.88	0.71	0.75	0.45	0.62	7	310	260
Synthetic	Two-circles	2	0.82	0.70	0.74	0.42	0.60	2	310	260
Synthetic	Three-circles	3	0.99	0.69	0.73	0.40	0.58	3	305	265
Synthetic	Iineblob	3	0.98	0.68	0.72	0.38	0.56	3	300	260
Synthetic	Sticks	4	0.86	0.67	0.71	0.37	0.54	4	300	265
Synthetic	R15	5	0.99	0.66	0.70	0.35	0.52	-	305	262
Synthetic	D31	4	0.98	0.65	0.69	0.34	0.50	-	300	259
Real	Iris	2	0.96	0.85	0.90	0.88	0.90	2	260	255
Real	Seeds	2	0.97	0.84	0.87	0.85	0.87	2	300	260
Real	Glass	2	0.95	0.82	0.86	0.82	0.85	2	300	275
Real	Heart	2	0.98	0.81	0.85	0.79	0.83	2	300	276
Real	Ionosphere	2	0.88	0.80	0.84	0.74	0.80	2	310	265
Real	Libras Movement	3	0.87	0.79	0.83	0.72	0.78	3	305	265
Real	Dermatology	2	0.98	0.78	0.82	0.68	0.76	2	300	260
Real	WDBC	2	0.99	0.77	0.81	0.64	0.74	2	300	265
Real	Pima	4	0.96	0.76	0.80	0.61	0.72	4	306	265
Real	Titanic	5	0.86	0.75	0.79	0.58	0.70	5	305	265
Real	Wine	2	0.69	0.67	0.68	0.32	0.48	-	310	260

Table 3 compares the proposed clustering algorithm against several contemporary and widely used methods—such as DPC, NCAR, GCNN, and IDPC-NNMS—across real-world datasets (WDBC, Seeds, Iris, Wine, Ionosphere, and Dermatology).

These tables assess key performance metrics— ARI, AMI, and ACC —to evaluate both clustering effectiveness and efficiency. The proposed algorithm consistently achieves superior accuracy across most datasets. For instance, it attains 0.990 on WDBC and Wine, and 0.970

on Seeds, with correspondingly high AMI and ARI values, such as AMI 0.810 and ARI 0.740 on WDBC.

At the same time, other algorithms demonstrate unique strengths.

DPC runs quickly but lacks a defined accuracy outcome on WDBC, while IDPC-NNMS shows moderate performance (AMI 0.770, ARI 0.640) and longer runtime. NCAR offers faster processing on Seeds but at a slightly lower accuracy (0.962).

On the Iris dataset, GCNN and the proposed algorithm are tied, each scoring 0.962 ACC and 0.900 for both AMI and ARI.

For Ionosphere, the proposed algorithm achieves 0.880 ACC (AMI 0.840, ARI 0.800), and in Dermatology, IDPC-NNMS records AMI 0.780 and ARI 0.680. Overall, the proposed algorithm demonstrates a clear advantage in clustering accuracy and competitive efficiency compared to other modern algorithms.

The paired t-tests show statistically significant differences between the proposed algorithm and IDPC (In Table 2).

- For ACC, t=6.1011 and $p = 1.1552 \times 10^{-4}$. Since $p < 0.05$, the higher ACC scores of the proposed algorithm over IDPC are statistically significant.
- For AMI, t=−12.2511 and $p = 2.4039 \times 10^{-7}$. The negative t indicates that IDPC's AMI is significantly higher than that of the proposed algorithm, and the p value confirms this difference is highly significant.
- For ARI, t=−5.2865 and $p = 3.5430 \times 10^{-4}$. Again, $p < 0.05$ shows the difference is significant, and the negative t suggests IDPC's ARI is significantly higher than the proposed algorithm's.

In sum, the proposed algorithm significantly outperforms IDPC in ACC, but IDPC shows significantly better AMI and ARI under these tests.

Table 3: Provides a comparison of the results obtained from implementing; the proposed algorithm against other well-known clustering algorithms on real datasets

ALGORITHMS	WDBC_ACC	WDBC_AMI	WDBC_ARI	SEEDS_ACC	SEEDS_AMI	SEEDS_ARI	IRIS_ACC	IRIS_AMI	IRIS_ARI	WINE_ACC	WINE_AMI	WINE_ARI	IONOSPHERE_ACC	IONOSPHERE_AMI	IONOSPHERE_ARI	DERMATALOGY_ACC	DERMATALOGY_AMI	DERMATALOGY_ARI
DPC	-	0.461	0.471	-	0.721	0.734	-	0.781	0.720	-	0.707	0.672	-	0.137	0.218	-	0.757	0.575
NCAR	0.978	0.440	0.480	0.962	0.891	0.625	0.921	0.809	0.736	0.976	0.833	0.847	0.864	0.251	0.243	0.947	0.770	0.581
NCARD	0.976	0.450	0.440	0.963	0.907	0.635	0.932	0.908	0.762	0.986	0.925	0.982	0.871	0.268	0.260	0.943	0.793	0.594
GCNN	0.981	0.550	0.540	0.961	0.913	0.785	0.954	0.881	0.879	0.986	0.930	0.980	0.860	0.270	0.264	0.965	0.812	0.610
The proposed algorithm	0.990	0.810	0.740	0.970	0.861	0.870	0.962	0.900	0.900	0.990	0.680	0.840	0.880	0.840	0.800	0.980	0.820	0.760
IDPC-NNMS	-	0.770	0.640	-	0.840	0.840	-	0.850	0.880	-	0.640	0.320	-	0.800	0.740	-	0.780	0.680

Table 4 presents a comparative evaluation of the proposed algorithm against several non-parametric clustering methods—OPTICS, HDBSCAN, single-linkage, complete-linkage, average-linkage, and Ward's method—across four real-world datasets using the NMI metric.

For Pendigits and Optical Digits, OPTICS and HDBSCAN both achieve the highest NMI (0.710 and 0.677 respectively), outperforming the proposed algorithm (0.525 and 0.585).

Table 4: Comparison of the Proposed Algorithm with Non-Parametric Clustering Algorithms on Real-World Datasets Using the NMI Metric

Dataset	OPTICS	HDBSCAN	Single-linkage	Complete-linkage	Average-linkage	WARD	The proposed algorithm
Pendigits	0.710	0.710	0.101	0.225	0.284	0.284	0.525
Optical digits	0.677	0.677	0.101	0.284	0.284	0.284	0.585
Dermatology	0.344	0.344	0.237	0.461	0.457	0.457	0.810
Ecoli	0.559	0.559	0.112	0.112	0.112	0.112	0.569

In the Dermatology dataset, however, the proposed algorithm markedly surpasses all alternatives, achieving an NMI of 0.810 compared to 0.344 for OPTICS/HDBSCAN and up to 0.461 for linkage methods. For Ecoli, the proposed algorithm attains a slight edge with NMI = 0.569 over OPTICS/HDBSCAN (0.559), whereas linkage methods perform poorly (≈ 0.112). These results indicate that while the proposed algorithm may not always match density-based methods on all datasets, it exhibits strong robustness and competitive performance—especially in datasets where linkage methods fail.

Limitation of the proposed algorithm is its inability to capture varying levels of connectivity among neighbors in highly mixed datasets, which may affect its performance in detecting subtle structural nuances.

Conclusion

In this paper, we presented a clustering algorithm that leverages point density using the natural neighbor method and the radial distribution function.

The method was implemented in both MATLAB and Python and tested on real and synthetic datasets.

The results demonstrate that the proposed algorithm improves clustering accuracy as well as the ARI and AMI metrics across all datasets.

Compared to previous methods, significant improvements were observed, and the execution time was reduced to an acceptable level.

Based on the results for the evaluated parameters, it can be concluded that the proposed algorithm outperforms other approaches and exhibits strong overall performance.

In future work, neuro-fuzzy algorithms could be integrated with the proposed algorithm to achieve optimal clustering and to more precisely capture relationships between clusters.

Author Contributions

M. Asadpour and Sh. Pourbahrami designed the experiments. Sh. Pourbahrami carried out the data analysis and wrote the code. M. Asadpour, and Sh. Pourbahrami interpreted the results and wrote the manuscript.

Acknowledgment

The authors gratefully acknowledge the reviewers for their valuable and insightful comments on the earlier version of this paper.

Funding

This research received no external funding.

Conflict of Interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data fabrication and, or falsification, double publication and, or submission, and redundancy have been completely witnessed by the authors.

Abbreviations

RDF	Radial Distribution Function

References

[1] A. O'Hagan, A. White, "Improved model-based clustering performance using Bayesian initialization averaging," Comput. Stat., 34(1): 201-231, 2019.

[2] G. J. Oyewole, G. A. Thopil, "Data clustering: application and trends," Artificial Intell. Rev., 56(7): 6439-6475, 2023.

[3] M. Sharma, S. K. Dinkar, "A survey on evolutionary clustering algorithms and applications," in Applications of Advanced

Optimization Techniques in Industrial Engineering, pp. 23–34: CRC Press, 2022.

[4] K. Gorgani Firouzjah, J. Ghasemi, “Ensemble learning algorithm for power transformer health assessment using dissolved gas analysis,” J. Electr. Comput. Eng. Innovations, 13(2): 387-402, 2025.

[5] T. BaniRostam, “Presenting a model of data anonymization in big data in the context of in-memory processing framework,” Available at SSRN 5161949, 2024.

[6] M. Jubair, R. Hassan, A. H. M. Aman, H. Sallehudin, Z. G. Al-Mekhlafi, B. A. Mohammed, M. S. Alsaffar, “Optimization of clustering in wireless sensor networks: techniques and protocols,” Appl. Sci., 11(23): 11448, 2021.

[7] E. Feser, H. Renski, H. Goldstein, “Clusters and economic development outcomes: An analysis of the link between clustering and industry growth,” Econ. Dev. Q., 22(4): 324–344, 2008.

[8] G. Anderson, “Industry clustering for economic development,” Econ. Dev. Rev., 12(2):26, 1994.

[9] H. Zhang, Y. Peng, “Image clustering: An unsupervised approach to categorize visual data in social science research,” Sociolog. Methods Res., 53(3): 1534–1587, 2024.

[10] N. Iam-On, T. Boongoen, N. Naik, L. Yang, “Leveraging ensemble clustering for privacy-preserving data fusion: Analysis of big social-media data in tourism,” Inf. Sci., 686: 121336, 2025.

[11] M. A. Sorkhi, E. Akbari, M. Rabbani, H. Motameni, “A dynamic density-based clustering method based on K-nearest neighbor,” Knowl. Inf. Syst., 66(5): 3005–3031, 2024.

[12] A. Rodriguez, A. Laio, "Clustering by fast search and find of density peaks,” Science, 344: 1492–1496, 2014.

[13] Y. Wang, J. Qian, M. Hassan, X. Zhang, T. Zhang, C. Yang, X. Zhou, F. Jia, “Density peak clustering algorithms: A review on the decade 2014–2023,” Expert Syst. Appl., 238: 121860, 2024.

[14] S. Pourbahrami, M. A. Balafar, L. M. Khanli, Z. A. Kakarash, “A survey of neighborhood construction algorithms for clustering and classifying data points,” Comput. Sci. Rev., 38: 100315, 2020.

[15] J. Liu, X. Liu, J. Xiong, Q. Liao, S. Zhou, S. Wang, Y. Yang, “Optimal neighborhood multiple kernel clustering with adaptive local kernels,” IEEE Trans. Knowl. Data Eng., 34(6): 2872–2885, 2020.

[16] H. Zhou, Z. Wang, H. Chen, X. Wang, “A novel spectral clustering algorithm based on neighbor relation and Gaussian kernel function with only one parameter,” Soft Comput., 28(2): 981–989, 2024.

[17] S. Pourbahrami, M. A. Balafar, L. M. Khanli, “ASVMK: A novel SVMs Kernel based on Apollonius function and density peak clustering,” Eng. Appl. Artif. Intell., 126: 106704, 2023.

[18] S. Pourbahrami, “A neighborhood-based robust clustering algorithm using Apollonius function kernel,” Expert Syst. Appl., .248: 123407, 2024.

[19] S. Pourbahrami, S. Azimpour, “A new method for detection of clustering based on four zones Apollonius circle,” Iran J. Comput. Sci., 3(1): 59–64, 2020.

[20] N. Abdolmaleki, L. M. Khanli, M. Hashemzadeh, S. Pourbahrami, “ACQC: Apollonius circle-based quantum clustering,” J. Comput. Sci., 64: 101877, 2022.

[21] S. Ding, W. Du, X. Xu, T. Shi, Y. Wang, C. Li, “An improved density peaks clustering algorithm based on natural neighbor with a merging strategy,” Inf. Sci., 624: 252–276, 2023.

[22] S. Ding, C. Li, X. Xu, L. Guo, L. Ding, X. Wu, “Horizontal federated density peaks clustering,” IEEE Trans. Neural Networks Learn Syst,, 2023.

[23] J. Xie, X. Liu, M. Wang, “SFKNN-DPC: Standard deviation weighted distance based density peak clustering algorithm,” Inf. Sci., 653: 119788, 2024.

[24] S. Pourbahrami, L. M. Khanli, S. Azimpour, “A novel and efficient data point neighborhood construction algorithm based on Apollonius circle,” Expert Syst. Appl., 115: 57–67, 2019.

[25] S. Pourbahrami, L. M. Khanli, S. Azimpour, “Improving neighborhood construction with Apollonius region algorithm based on density for clustering,” Inf. Sci., 522: 227–240, 2020.

[26] S. Pourbahrami, M. Hashemzadeh, “A geometric-based clustering method using natural neighbors,” Inf. Sci., 610: 694–706, 2022.

[27] W. Durani, D. Mautz, C. Plant, C. Böhm, "DMDHC: Discovery of multi-density hierarchical cluster structures," in Proc. the 2025 SIAM International Conference on Data Mining (SDM): 261–269, 2025.

[28] M. Ankerst, M. M. Breunig, H. P. Kriegel, J. Sander, “OPTICS: Ordering points to identify the clustering structure,” ACM Sigmod Rec., 28(2): 49–60, 1999.

[29] R. J. Campello, D. Moulavi, A. Zimek, J. Sander, “Hierarchical density estimates for data clustering, visualization, and outlier detection,” ACM Trans. Knowl. Discovery Data (TKDD), 10(1): 1–51, 2015.

[30] S. Sharma, N. Batra, "Comparative study of single linkage, complete linkage, and ward method of agglomerative clustering," in Proc. 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon): 568–573, 2019.

[31] Q. Zhu, J. Feng, J. Huang, “Natural neighbor: A self-adaptive neighborhood method without parameter K,” Pattern Recognit. Lett., 80: 30–36, 2016.

[32] P. Fränti, S. Sieranoja, “Clustering accuracy,” Appl. Comput. Intell., 4(1): 24-44, 2024.

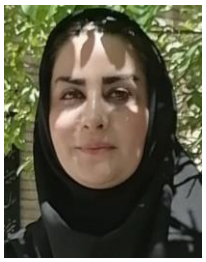
[33] M. E. Newman, G. T. Cantwell, J. G. Young, “Improved mutual information measure for clustering, classification, and community detection,” Phys. Rev. E, 101(4): 042304, 2020.

[34] S. Zhang, H. S. Wong, Y. Shen, “Generalized adjusted rand indices for cluster ensembles,” Pattern Recognit., 45(6): 2214–2226, 2012.

Biographies

Mohamad Asadpour is Associate Professor, Faculty of Electrical and Computer Engineering, university of Tabriz. His research interests include Computer Vision, Image Processing, Medical Imaging, clustering, semi-supervised Learning, Deep Learning.

- Email: m_asadpour@tabrizu.ac.ir
- ORCID: 0000-0002-1672-149X
- Web of Science Researcher ID: NA
- Scopus Author ID: 56667337000
- Homepage: https://scholar.google.com/citations?user=D8QWpmkAAAAJ&hl=en



Shahin Pourbahrami is currently an Assistant Professor in the Department of Computer Engineering at the Technical and Vocational University (TVU), Tehran, Iran. Her research interests include clustering algorithms, quantum computing, deep learning, and proteomics using deep learning approaches. Her email address is:

- Email: Shpourbahrami@tvu.ac.ir
- ORCID: 0000-0001-6950-2138
- Web of Science Researcher ID: NA
- Scopus Author ID
- Homepage: https://scholar.google.com/citations?user=jLE1e_oAAAAJ&hl=en&oi=ao