

RESEARCH NOTE

Open Access



An inversion-based clustering approach for complex clusters

Mohammad Mahdi Barati Jozan¹, Aynaz Lotfata², Howard J. Hamilton³ and Hamed Tabesh^{1*}

Abstract

Background The choice of an appropriate similarity measure plays a pivotal role in the effectiveness of clustering algorithms. However, many conventional measures rely solely on feature values to evaluate the similarity between objects to be clustered. Furthermore, the assumption of feature independence, while valid in certain scenarios, does not hold true for all real-world problems. Hence, considering alternative similarity measures that account for inter-dependencies among features can enhance the effectiveness of clustering in various applications.

Methods In this paper, we present the *Inv* measure, a novel similarity measure founded on the concept of inversion. The *Inv* measure considers the significance of features, the values of all object features, and the feature values of other objects, leading to a comprehensive and precise evaluation of similarity. To assess the performance of our proposed clustering approach that incorporates the *Inv* measure, we evaluate it on simulated data using the adjusted Rand index.

Results The simulation results strongly indicate that inversion-based clustering outperforms other methods in scenarios where clusters are complex, i.e., apparently highly overlapped. This showcases the practicality and effectiveness of the proposed approach, making it a valuable choice for applications that involve complex clusters across various domains.

Conclusions The inversion-based clustering approach may hold significant value in the healthcare industry, offering possible benefits in tasks like hospital ranking, treatment improvement, and high-risk patient identification. In social media analysis, it may prove valuable for trend detection, sentiment analysis, and user profiling. E-commerce may be able to utilize the approach for product recommendation and customer segmentation. The manufacturing sector may benefit from improved quality control, process optimization, and predictive maintenance. Additionally, the approach may be applied to traffic management and fleet optimization in the transportation domain. Its versatility and effectiveness make it a promising solution for diverse fields, providing valuable insights and optimization opportunities for complex and dynamic data analysis tasks.

Keywords Clustering algorithm, Inversion-based similarity measure, Overlapping clusters, Adjusted Rand index

*Correspondence:

Hamed Tabesh
Tabeshh@mums.ac.ir

¹ Department of Medical Informatics, Faculty of Medicine, Mashhad University of Medical Sciences, Mashhad, Iran

² Department of Pathology, Microbiology, and Immunology, School Of Veterinary Medicine, University of California, Davis, USA

³ Department of Computer Science, University of Regina, Regina, SK, Canada



© The Author(s) 2024. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>. The Creative Commons Public Domain Dedication waiver (<http://creativecommons.org/publicdomain/zero/1.0/>) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

Introduction

Clustering is a fundamental technique in data mining and machine learning, aiming to group objects into distinct clusters [1–7]. Objects within a cluster show high similarity to each other and low similarity to objects in other clusters, determined by a similarity measure [8–11].

Selecting an appropriate similarity measure is crucial for clustering algorithms. Studies indicate its significance in algorithm performance [8–12]. Various research assesses similarity measures across disciplines like web clustering [13], trajectory clustering [14], and chemical databases [15]. Some research endeavors have further examined performance variations based on the type of data being analyzed, differentiating between categorical data [16] and continuous data [8]. Consequently, some scholars [17] have advocated for the utilization of two fundamental clustering techniques: "similarity measures" for qualitative data and "distance measures" for quantitative data [17]. In this study, we'll refer to both types of measures as "similarity measures."

Table 1 lists similarity functions for qualitative data, and Table 2 shows distance functions for quantitative data.

A key limitation of similarity measures (e.g., Euclidean distance [17] and Hamming similarity [18]) is their exclusive reliance on feature values. Consequently, when two objects or entities exhibit similar feature values, they are considered more similar, regardless of any other pertinent factors. This oversimplification may overlook crucial aspects of the data.

Secondly, similarity measures often assume feature independence, neglecting their interdependence and potential influence on each other's values. This oversight may result in incomplete representations of data relationships. Moreover, most measures overlook feature prioritization, disregarding the varying importance of features in determining similarity. These assumptions do not fully align with real-world complexities, potentially limiting applicability and accuracy. To address these challenges, researchers explore inversion as a promising approach, investigating its theoretical and practical aspects [19–22].

Table 1 Some similarity functions for qualitative data [17]

Name	Function formula or measure method	Explanation
Jacard similarity	$J(A, B) = \frac{ A \cap B }{ A \cup B }$	<ol style="list-style-type: none"> 1. Measures the similarity of two sets 2. X is the number of elements of set X 3. Jacard distance = 1 – Jacard similarity
Hamming similarity	The minimum number of substitutions needed to change one data point into the other	<ol style="list-style-type: none"> 1. Smaller numbers indicate greater similarity 2. Hamming distance is the opposite of Hamming similarity
For data of mixed type	Map the feature into (0,1) Transform the feature into a dichotomous one $S_{ij} = \frac{1}{d} \sum_{l=1}^d S_{ijl}$ $S_{ij} = \left(\sum_{l=1}^d Y_{ijl} S_{ijl} \right) / \left(\sum_{l=1}^d Y_{ijl} \right)$	

Table 2 Some distance functions for quantitative data [17]

Approach	Formula	Explanation
Minkowski distance	$\left(\sum_{i=1}^d x_{ij} - x_{jl} ^n \right)^{1/n}$	A set of definitions for distance <ol style="list-style-type: none"> 1. City-block when n=1 2. Euclidean distance when n=2 3. Chebyshev distance when n → ∞
Standardized Euclidean distance	$\left(\sum_{i=1}^d \left \frac{x_{ij} - x_{jl}}{s_i} \right ^2 \right)^{1/2}$	<ol style="list-style-type: none"> 1. s stands for the standard deviation 2. A weighted Euclidean distance on the deviation
Cosine distance	$1 - \cos \alpha = \frac{\text{Cov}(x_i, x_j)}{\sqrt{D(x_i)} \sqrt{D(x_j)}}$	<ol style="list-style-type: none"> 1. Cov stands for the covariance and D stands for the variance 2. Measures the distance based on linear correlation
Pearson correlation distance	$1 - \frac{\text{Cov}(x_i, x_j)}{\sqrt{D(x_i)} \sqrt{D(x_j)}}$	<ol style="list-style-type: none"> 1. Cov stands for the covariance and D stands for the variance 2. Measures the distance based on linear correlation
Mahalanobis distance	$\sqrt{(x_i - x_j)^T S^{-1} (x_i - x_j)}$	<ol style="list-style-type: none"> 1. S is the covariance matrix inside the cluster 2. Has high computational complexity

In one study [19], a new constructive bijection connects permutations with a specific number of inversions to those with a particular major index, facilitating exploration of mathematical connections. Another work [20] introduces a probability distribution on the group of permutations of the set of integers, providing insights into inversion’s probabilistic aspects for permutation-based data analysis. Furthermore, [21] presents six bijections linking a specific type of polyominoes called deco polyominoes with permutations, establishing connections between classical statistics and permutation-related analyses. Moreover, [22] proposes an efficient solution for counting interior edge crossings in bipartite graphs, relevant for layered graph drawing and data visualization enhancement.

This study introduces "Inv," an inversion-based similarity measure addressing previous challenges. It forms the basis of a new clustering approach grouping objects by inversion-based similarity. The primary goal is to optimize clustering by maximizing intra-cluster similarity and minimizing inter-cluster similarities. By incorporating this measure, we anticipate achieving more meaningful clustering results that better reflect underlying data patterns and relationships.

Data and method

The flowchart of the activities undertaken to assess the new inversion-based clustering approach is presented in the Fig. 1.

Formulation of challenges related to similarity criteria

Several key terms used in simulation examples are defined below.

Object: Any entity that we intend to cluster is an *object*. Each object obj_i has n features and can be represented by a vector: $obj_i = \langle f_{i1}, f_{i2}, \dots, f_{in} \rangle$.

Feature space: Each feature has a valid range of values known as its *feature space*, represented by S_f for feature f .

Universe: The set of all objects we want to cluster is known as the *universe*, represented by $U = \{obj_1, obj_2, \dots, obj_k\}$.

Similarity Measure: A *similarity measure* is a measure that takes two objects as input and outputs a numeric value representing their similarity.

Clustering problem: A problem of partitioning k objects into m groups $\langle C_1, C_2, \dots, C_m \rangle$ according to a specified similarity measure, so that the objects in each group are as similar as possible to each other and as different as possible from objects in other groups. These clusters adhere to two constraints [1–7]:

$$\bigcup_{i=1}^m C_i = U \tag{1}$$

$$C_i \cap_{\substack{j=1..m \\ i \neq j}} C_j = \emptyset \tag{2}$$

In the following paragraphs, the challenges stated in the introduction section are formulated using the defined terms.

Symmetry challenge

A drawback of distance-based measures is called the *Symmetry challenge*, where adding or subtracting a value to a feature has the same distance effect. Given that Euclidean distance is widely used in clustering algorithms [23], it is used as a representative of distance-based

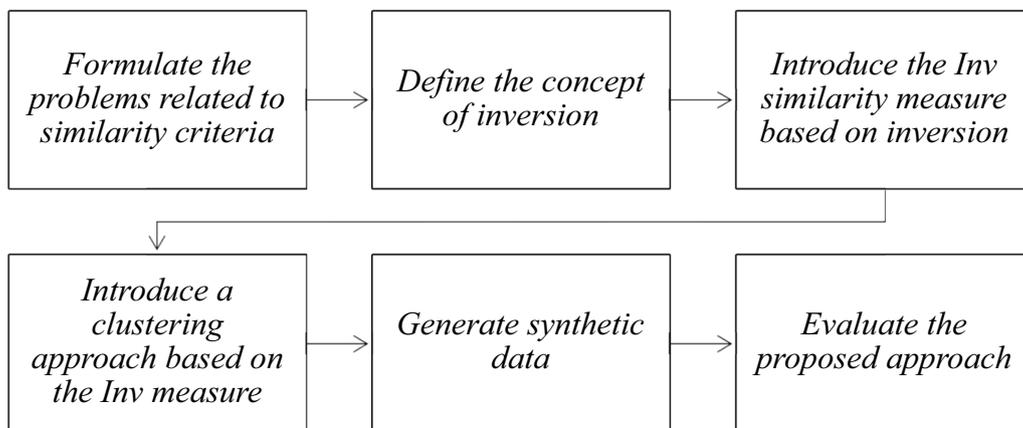


Fig. 1 Flowchart of implementation and evaluation of the inversion-based clustering approach

measures in the following practices. Practice 1 demonstrates the weakness of similarity criteria when adding or subtracting a specific value to a feature.

Practice 1: Consider the following three objects:

$$obj_1 = \langle V_1, V_2, \dots, V_n \rangle$$

$$obj_2 = \langle V_1 + v, V_2, \dots, V_n \rangle$$

$$obj_3 = \langle V_1 - v, V_2, \dots, V_n \rangle$$

where

$$V_1, V_1 + v, V_1 - v \in S_{f_1} \tag{3}$$

The first feature of obj_2 is v units less than the first feature of obj_1 , and the first feature of obj_3 is v units more than the first feature of obj_1 . All other features are equal in the three objects. The Euclidean distance E between them, the similarity for adding v to the first feature is calculated as follows:

$$\begin{aligned} E(obj_1, obj_2) &= \sqrt{(V_1 - (V_1 + v))^2 + (V_2 - V_2)^2 + \dots + (V_n - V_n)^2} \\ &= \sqrt{(V_1 - V_1 - v)^2 + (V_2 - V_2)^2 + \dots + (V_n - V_n)^2} = v \end{aligned} \tag{4}$$

and the similarity for subtracting v from the first feature is calculated as follows:

$$\begin{aligned} E(obj_1, obj_3) &= \sqrt{(V_1 - (V_1 - v))^2 + (V_2 - V_2)^2 + \dots + (V_n - V_n)^2} \\ &= \sqrt{(V_1 - V_1 + v)^2 + (V_2 - V_2)^2 + \dots + (V_n - V_n)^2} = v \end{aligned} \tag{5}$$

This challenge can arise for any feature. In general, this challenge can be expressed as follows:

$$obj_1 = \langle V_1, V_2, \dots, V_i, \dots, V_n \rangle$$

$$obj_2 = \langle V_1, V_2, \dots, V_i + v, \dots, V_n \rangle$$

$$obj_3 = \langle V_1, V_2, \dots, V_i - v, \dots, V_n \rangle$$

$$E(obj_1, obj_2) = E(obj_1, obj_3)$$

where

$$\begin{aligned} V_i, V_i + v, V_i - v &\in S_{f_i} \\ 1 \leq i \leq n \end{aligned} \tag{6}$$

The equality of the distances is mathematically correct, but in the real world, the significance of adding a specific value to a feature may be different from that

of subtracting the same amount. An example based on student scores can be found as Supplementary Example 1 (see Additional file 1).

Place symmetry challenge

The *Place Symmetry challenge* is a generalized form of the previous challenge. In this challenge, increasing or decreasing a value can occur for each feature. In Practice 2, the weakness of the similarity criteria when adding or subtracting a certain value to possibly different features of objects is shown.

Practice 2: Consider the following three objects:

$$obj_1 = \langle V_1, V_2, \dots, V_n \rangle$$

$$obj_2 = \langle V_1, V_2, \dots, V_i + v, \dots, V_n \rangle$$

$$obj_3 = \langle V_1, V_2, \dots, V_j - v, \dots, V_n \rangle$$

$$E(obj_1, obj_2) = E(obj_1, obj_3)$$

where

$$\begin{aligned} V_i, V_i + v, V_i - v &\in S_{f_i} \\ V_j, V_j - v &\in S_{f_j} \\ 1 \leq i, j \leq n \end{aligned} \tag{7}$$

if V_i represents age and V_j represents weight, both normalized to their respective feature spaces, adding v units to V_i and subtracting v units from V_j yields the same Euclidean distance between two pairs of objects. However, the implications of increasing or decreasing v units in weight differ from those in age due to varying value distributions. Hence, altering their values by v holds distinct meanings for each feature.

Feature independence challenge

The Feature Independence challenge refers to the treatment of features as if they are independent of each other when calculating distance-based measures, whereas in practice they are often interdependent.

Definition of the concept of inversion

To address the challenges defined in section "Formulation of challenges related to similarity criteria", a similarity measure named *Inv*, which is based on the concept of inversion, is proposed. The mathematical definition of inversion is as follows:

Inversion: In a sequence $S = \langle a_1, a_2, \dots, a_n \rangle$ of pairwise comparable elements $a_i (i = 1, 2, \dots, n)$, a pair (a_i, a_j) is called an *inversion* if $i < j$ and $a_i > a_j$ [22].

The concept of inversion is illustrated by Practice 3.

Practice 3: Suppose we have five movies and we ask two people to rate them on a scale of 1 to 10 according to their preferences. In this example, an object is a *person* and it is represented by a vector with 5 features [24].

$$person = \langle Score(Movie_1), Score(Movie_2), Score(Movie_3), Score(Movie_4), Score(Movie_5) \rangle \tag{8}$$

Scores are given in Table 3.

Table 3 Users' ratings of movies

Movies	Score of person 1	Score of person 2
Movie1	10	9
Movie2	6	3
Movie3	5	9
Movie4	1	6
Movie5	8	1

The definition of inversion for only one person is presented in Supplementary Example 1 (see Additional file 1). Let's now explore a more indirect application of the definition to determine the inversions between the scores of the two people. We arrange the movies for each person in a rank sequence based on preferences, with the highest-scoring movie first. Ties are resolved by ranking the movie with the lower movie number first. Person 1's sequence is $\langle Movie_1, Movie_5, Movie_2, Movie_3, Movie_4 \rangle$, while Person 2's is $\langle Movie_1, Movie_3, Movie_4, Movie_2, Movie_5 \rangle$. Renaming the movies based on Person 1's rank sequence, we get $\langle 1, 2, 3, 4, 5 \rangle$ for Person 1 and $\langle 1, 4, 5, 3, 2 \rangle$ for Person 2. Applying the inversion definition to the second sequence, we find 4 is inverted compared to 3 and 2 (two inversions), 5 is inverted compared to 3 and 2 (two inversions), and 3 is inverted compared to 2 (one inversion), totaling five inversions.

A visual method for counting inversions involves organizing movies for each person by their scores, highest to lowest, and connecting corresponding movies. The intersections of these lines indicate the number of inversions between the two sequences.

In Fig. 2, the five inversions are indicated by the five points where the lines intersect. The more different the sequences, the greater the number of the inversions. The minimum and maximum number of inversions of two vectors with n features are 0 and $\frac{n(n+1)}{2}$, respectively [25]. Supplementary Code 1 contains the inversion calculation algorithm (see Additional file 1).

Notice from *Practice3* that the way ties are broken affects the number of inversions. The proposed way of resolving ties is to consider a default sequence for features, and when the values of two or more features are the same, their order will be based on the default order. For instance, in the movies database, priority is given to

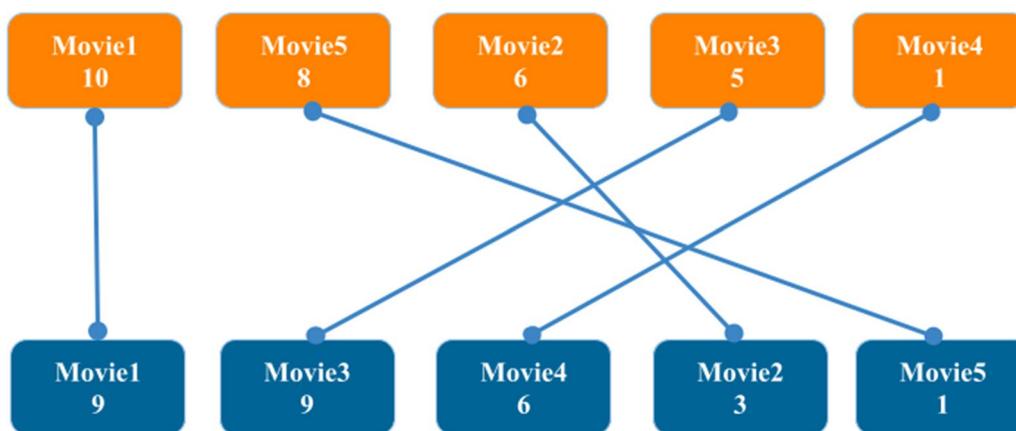


Fig. 2 Each point of intersection between lines representing an inversion (practice 3)

movies with smaller numbers. Thus, as Person 2 rated Movie 1 and Movie 3 equally at 9, Movie 1 takes precedence in the rank order.

Practice3 illustrates the importance of feature prioritization in cases where values are equal. Depending on specific requirements, certain features may need to be prioritized over others, which can have a considerable impact on the resulting number of inversions.

Introduction of the *Inv* similarity measure based on the notion of inversion

In this subsection, the *Inv* inversion-based similarity measure is introduced. The *Inv* measure of the similarity of two objects is defined as the number of inversions that exist between two objects according to the *Count_Inversions* function. Formally, the measure is defined as follows:

$$obj_1 = \langle V_{11}, V_{12}, \dots, V_{1n} \rangle$$

$$obj_2 = \langle V_{21}, V_{22}, \dots, V_{2n} \rangle$$

$$Inv(obj_1, obj_2) = Count_Inversions(obj_1, obj_2) \quad (9)$$

According to the *Inv* measure, as inversion count between two sequences rises, their similarity decreases, and vice versa. Although we refer to inversion as a similarity measure, it actually yields a measure of dissimilarity because the greater the similarity between two objects, the lower the number of inversions.

Three advantages of the inversion-based similarity measure are as follows [22]: the number of inversions is affected by the rank positions of the features in the vector (Feature Independence challenge); the number of inversions when v is added to feature f_i can be different from the number of inversions when it is subtracted from f_i (Symmetry challenge); and the number of inversions when v is added to f_i may not be the same as the number

of inversions when v is added to f_j (Place Symmetry challenge).

The Symmetry challenge in distance-based measures results from the equal impact of adding or subtracting a value from a feature on the distance. In calculating inversions, not only is the value of a particular feature considered, but also the values of other features become integral. The feature’s rank position within the object’s vector significantly affects the inversion count. Altering a feature’s rank position by adding a value may yield different outcomes compared to subtracting the same value, influenced by other features. Consequently, the inversion count varies depending on specific features, emphasizing the nuanced nature of inversion calculations.

The Place Symmetry challenge expands upon the Symmetry challenge by allowing the increase or decrease of a value for each feature individually. As previously noted, the count of inversions is influenced by both the value of a specific feature and other features, so it addresses this challenge as well as the previous challenge.

To challenge the independence of features, since determining the number of inversions requires determining the values and order of all features, hence the features cannot be independent of each other. The explanation of how the proposed measure addresses the outlined challenges are illustrated in Supplementary practice 1 (see Additional file 1).

The proposed similarity measure offers an additional advantage by allowing adjustments to consider feature importance in distance calculation. This enables the consideration of feature relevance or priority, enhancing the measure’s utility. To accommodate priorities, the inversion measure can be modified to incorporate both feature value and assigned priority. By default, all variables are assumed to have equal priority. *Practice 4* demonstrates a method for factoring feature priority into similarity calculation.

Practice 4: Consider two objects, each with three features, depicted by colored circles in Fig. 3. Let the orange

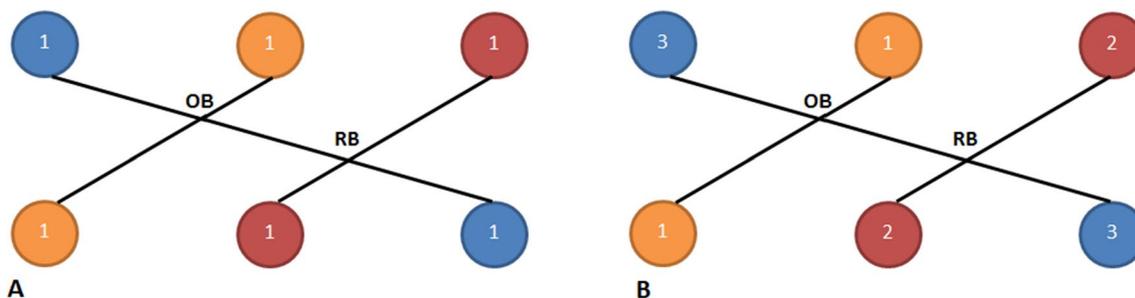


Fig. 3 Prioritizing features: **A** All features have the same priority, which is one, **B** The priority of each feature is displayed within the corresponding circle (Practice 4)

feature have priority 1 (lowest), the red feature priority 2, and the blue feature priority 3 (highest). An adjusted inversion function is employed, where each inversion detected is weighted by the product of the relevant features' priorities.

According to Fig. 3A, there are 2 inversions because there are 2 points where the lines intersect (OB and RB). Since the priority of every feature is one, the adjusted inversion function is as calculated as:

$$AdjInv(\text{Figure 3_A}) = OB + RB = 1(1) + 1(1) = 2 \tag{10}$$

To calculate the adjusted inversion for Fig. 3B, the products of the priorities of each inversion (represented by a line) are also considered.

$$AdjInv(\text{Figure 3_b}) = OB + RB = 1(3) + 2(3) = 9 \tag{11}$$

Introduction of the clustering approach based on the Inv measure

In this section, we formulate an algorithmic framework for an inversion-based clustering approach as a type of partitioning clustering method and then we instantiate the framework with different measures to specify two inversion-based clustering algorithms. The main steps of a partitioning clustering method are as follows. First, the number of desired clusters is chosen and initial centroids in the range of the feature values are randomly selected. Next, every input object is assigned to the nearest centroid based on its distance from it using the similarity measure. Every centroid is then moved to the mean location of its assigned cluster, and this process is iteratively repeated until convergence is reached, as indicated by no further changes in assignments [25]. The pseudo-code of proposed algorithmic framework can be found as Supplementary Code 2 (see Additional file 1).

Algorithmic framework for inversion-based clustering

The steps of the algorithmic framework are described below.

Input: $U = \{obj_1, obj_2, \dots, obj_k\}$, number of clusters (m)

Output: $mclusters(C_1, C_2, \dots, C_m)$

Step 1) Normalize each feature based on the Min–Max Feature scaling method. The normalized feature value F_{ia} for feature f_a in object obj_i is calculated as the following equation:

$$F_{ia} = \frac{f_{ia} - \min(f_a)}{\max(f_a) - \min(f_a)} \tag{12}$$

where f_{ia} is the original feature value for obj_i , $\min(f_a)$ is the minimum value and $\max(f_a)$ is the maximum value of the feature across all objects in universe U , and F_{ia} is the normalized value. The feature space for every normalized feature is equal to $[0, 1]$.

Step 2) Initialize the centroids c_1, c_2, \dots, c_m of clusters C_1, C_2, \dots, C_m randomly.

Step 3) Calculate the number of inversions between every combination of an object obj_i and a centroid c_j and assign every object to the cluster for the centroid with the fewest inversions.

$$\begin{aligned} & \text{Assign } obj_i \text{ to Cluster } C_b, \text{ where } b \\ & = \operatorname{argmin}_j (\text{Measure}(obj_i, c_j)) \end{aligned}$$

where

$$1 \leq i \leq k, 1 \leq j \leq m \tag{13}$$

Step 4) For each cluster C_j , calculate the new centroid c_j as the mean of the objects assigned to the cluster:

$$c_j = \frac{1}{N_j} \sum_{c \in C_j} c$$

where

$$1 \leq j \leq m \tag{14}$$

N_j is the number of objects assigned to cluster C_j , and the sum is taken over all objects in this cluster.

Step 5) Repeat Steps 3 and 4 until the clusters do not change, or the number of iterations reaches a predetermined value.

For the third step of the algorithmic framework, which is to assign an object to a cluster using a measure, two measures are proposed, which lead to the Inversion-Based Clustering Algorithm (ICA) and the Regulator Inversion Euclidean distance based Clustering Algorithm (RIECA).

Approach 1) Inversion-Based Clustering Algorithm (ICA)

In the first round of ICA, the centroids are randomly initialized, then inversions between every combination of an object and a centroid are calculated, and finally every object is assigned to the centroid with the fewest inversions. In subsequent rounds, inversions between every combination of an object and an updated centroid are calculated. In other words, the measure function is defined as follows:

$$Measure(obj_i, c_c) = Inv(obj_i, c_c) \tag{15}$$

where obj_i is the i^{th} object, C_c is c^{th} cluster (which has centroid c_c), and $Inv(obj_i, c_c)$ is the number of inversions between obj_i and c_c .

Approach 2) Regulator Inversion Euclidean distance based Clustering Algorithm (RIECA)

With RIECA, inversion is used as a regulator for Euclidean distance [17], i.e. the Euclidean distance is multiplied by the number of inversions. If the number of inversions is high, the value of the measure grows more than if the number of inversions is low. In this approach, the *Measure* function is defined as follows:

$$Measure(Obj_i, c_c) = Inv(Obj_i, c_c) * E(Obj_i, c_c)$$

where

$$1 \leq i \leq k, 1 \leq c \leq m \tag{16}$$

where Obj_i is the i^{th} object, c_c is the c^{th} cluster, $Inv(Obj_i, c_c)$ is the number of inversions between Obj_i and c_c , and $E(Obj_i, c_c)$ is the Euclidean distance between Obj_i and c_c .

Results

This section describes the generation of synthetic data and the evaluation of the proposed approach, which correspond to the final two steps of the flowchart given in section "Data and method".

Generation of synthetic data

Synthetic datasets were generated using the MixSim package [26]. The average pairwise overlap parameter (denoted as $\tilde{\omega}$) in package allows for the creation of clusters with varying degrees of complexity, ranging from well-separated ($\tilde{\omega}=0.001$) to highly overlapped ones ($\tilde{\omega}=0.4$) [27]. The summary of the package documentation can be found as Supplementary Documentation 1 (see Additional file 1).

To evaluate the performance of the ICA and REICA algorithms, we generated two types of random data

samples for each value of $\tilde{\omega}$ (0.4, 0.3, 0.2, 0.05, and 0.001). The first type of data sample (referred to as Simulated Dataset1) comprised 200 random data values with 6 features, forming 5 clusters. The second type of data sample (referred to as Simulated Dataset2) included 500 random data values with 7 features, forming 10 clusters. These datasets facilitated comprehensive evaluation of algorithm performance across diverse clustering complexities.

Evaluation of the proposed approach

The proposed ICA and REICA algorithms are compared with EM clustering [28] from the *mclust* package [29], k-means [3] and hierarchal clustering [30] from the *stats* package, and k-medoids [31] clustering from the *cluster* package [32]. For the comparison, 1000 data samples of Simulated Dataset 1 and 1000 of Simulated Dataset 2 were prepared and then each algorithm was run on each sample separately. The ICA and REICA algorithms are implemented in R version 3.4.2 [33]. This study employs the *adjusted Rand index* implemented in the *MixSim* package to evaluated algorithms [26, 34–36]. The values obtained for the adjusted Rand index for the algorithms when applied to Simulated Dataset 1 are shown in Table 4.

Table 4 shows that for complex cluster structures ($\tilde{\omega}=0.4, 0.3,$ and 0.2), REICA algorithm outperforms other algorithms in cluster formation. However, for moderately or highly separated clusters ($\tilde{\omega}=0.05,$

Table 5 The most effective algorithm in clustering 6-dimensional datasets of size 200 with 5 clusters (Simulated Dataset 1)

$\tilde{\omega}$	The most effective algorithm
$\tilde{\omega}=0.4$	REICA, ICA
$\tilde{\omega}=0.3$	REICA
$\tilde{\omega}=0.2$	REICA
$\tilde{\omega}=0.05$	k-means
$\tilde{\omega}=0.001$	k-means, EM, k-medoids, hierarchical

Table 4 Effectiveness of the algorithms in clustering 6-dimensional datasets of size 200 with 5 clusters (Simulated Dataset 1)

	0.4	0.3	0.2	0.05	0.001
EM clustering	0.055 ± 0.027	0.097±0.044	0.176 ± 0.064	0.572 ± 0.098	0.981 ± 0.032
k-means clustering	0.066±0.029	0.119 ± 0.043	0.211 ± 0.061	0.622 ± 0.087	0.983 ± 0.025
k-medoids clustering	0.061±0.028	0.107 ± 0.039	0.187 ± 0.060	0.567 ± 0.092	0.975 ± 0.023
Hierarchal clustering	0.058±0.027	0.103 ± 0.40	0.185 ± 0.060	0.574 ± 0.091	0.979 ± 0.021
ICA	0.106 ± 0.029	0.156 ± 0.040	0.229 ± 0.054	0.494 ± 0.092	0.824 ± 0.093
REICA	0.109 ± 0.030	0.167 ± 0.043	0.253 ± 0.059	0.584 ± 0.087	0.935 ± 0.046

The most effective algorithm is highlighted in bold

Table 6 The most effective algorithm in clustering 7-dimensional datasets of size 500 with 10 clusters (Simulated Dataset 2)

$\tilde{\omega}$	The most effective algorithm
$\tilde{\omega}=0.4$	REICA
$\tilde{\omega}=0.3$	REICA
$\tilde{\omega}=0.2$	REICA
$\tilde{\omega}=0.05$	k-means
$\tilde{\omega}=0.001$	EM

0.001), both ICA and REICA algorithms perform poorly compared to other algorithms. This was anticipated, as distance between objects outweighs the influence of inversions in such scenarios.

To evaluate the statistical significance of the adjusted Rand index differences among algorithms, we conducted a one-way ANOVA test. Further details are available in Supplementary statistical test result 1 (see Additional file 1).

Tables 5 and 6 reveal the optimal algorithm for each $\tilde{\omega}$ value, offering insights into algorithm performance across diverse clustering scenarios. These results elucidate the efficacy of ICA and REICA algorithms amidst varying clustering complexities and separations.

In highly or moderately separated clusters, distance becomes more influential, diminishing the role of the number of inversions. Consequently, the proposed algorithms are less effective in handling such scenarios compared to other types of clustering. REICA excelled in complex structures, while no algorithm consistently outperformed others in highly or moderately separated clusters. When $\tilde{\omega}=0.05$, k-means was most effective in both simulations. Conversely, with $\tilde{\omega}=0.001$ and a small number of clusters and samples (Simulated Dataset 1), k-means, EM, K-medoids, and Hierarchical algorithms were highly effective. However, with a large number of clusters and samples (Simulated Dataset 2), EM algorithm proved most effective. These findings highlight algorithm strengths in diverse clustering conditions, revealing performance across varying complexities and separations. The selection of the optimal algorithm depends on specific data characteristics and nature.

Discussion

Clustering techniques encompass partitioning, hierarchical, density-based, grid-based, and model-based methods [25]. The proposed algorithms fall under partitioning clustering. ICA is similar to k-means but utilizes

an inversion-based similarity measure instead of a distance measure, whereas REICA utilizes inversion as a form of regulation for the Euclidean distance. In REICA, Euclidean distance is multiplied by the number of inversions. As a result, when there are more inversions, the measure value increases more prominently compared to scenarios with fewer inversions. This innovative method enriches the algorithm’s ability for intricate cluster handling and enhances data analysis insights.

One strength of this study is that it identified three major challenges for Euclidean distance measures: the Symmetry challenge, the Place Symmetry challenge, and the Feature Independence Challenge. To address these challenges, we introduce the inversion-based measure "Inv." Our findings show the significance of both Euclidean distance and inversions for similarity measures. Particularly, REICA, which multiplies inversions by Euclidean distance, outperforms ICA, which relies solely on inversions. The effectiveness of REICA, which employs a hybrid measure that considers inversions and Euclidean distance, suggests potential for developing other hybrid measures considering both factors. Such measures could prioritize inversion for complex clusters and emphasize Euclidean distance for well-separated clusters, aligning with k-means and other clustering methods.

Another strength of this study is that it benefits from applying statistical testing techniques like one-way ANOVA to rigorously assess the effectiveness of the proposed algorithms in comparison to existing algorithms. Additionally, simulating diverse data samples covering various clustering complexities enhances the findings’ robustness and generalizability.

One strength of the proposed inversion-based algorithm is its ability to prioritize features differently. This prioritization operates at two levels. Firstly, during inversion calculation, equal feature values are sorted based on predetermined priority, influencing the number of inversions. Secondly, priority can be introduced as a coefficient in inversion calculations, where each intersection reflects the multiplication of feature priority values. As a result, the distance between two objects will increase with higher priority values, allowing more flexible and nuanced clustering.

For evaluation, the proposed algorithms’ performance was compared with classical clustering methods: k-means [3], EM clustering [28], hierarchical clustering [30], and k-medoids [31]. Results indicate REICA outperforms other algorithms with complex cluster structures [37, 38]. However, in scenarios of moderate or high cluster separation, the proposed algorithms are less effective, consistent with Berikov [39] and Cupertino [40].

A limitation of the inversion-based clustering approach is its effectiveness sensitivity to initial centroid selection, a common challenge in partition-based algorithms [25]. Mitigation strategies, such as repetition and employing the K-means++ initialization method [41], can be adapted to address this issue. In this study, we adopt the solution of running the algorithm several times with random initial centroids [25].

Another limitation is the lack of evaluation on real datasets. Assessing an algorithm's performance on actual data is challenging due to unknown cluster complexity. To address this, one could compare synthetic datasets with known complexity to real databases to gain an understanding of the complexity, then evaluate the algorithms under conditions similar to practical tasks. This approach offers insights into real-world performance, establishing their relevance and reliability for diverse data analysis scenarios.

Conclusion

This paper emphasizes the significant impact of similarity measures on clustering algorithm performance. Existing measures, often reliant on feature values and assuming feature independence, may not yield optimal results in practice. To address this, we introduced the innovative inversion-based *Inv* measure, which considers other object and feature values through inversion. We proposed two algorithms (ICA and RIECA) based on *Inv* measure, and evaluated their performance using simulated data. Results showed inversion-based clustering outperformed traditional techniques for complex cluster structures.

Future studies can explore practical applications of the *Inv* measure in real-world problems to improve clustering performance across domains. Further research could investigate other hybrid measures combining inversions and distance-based measure.

Supplementary Information

The online version contains supplementary material available at <https://doi.org/10.1186/s13104-024-06791-y>.

Additional file 1. Supplementary Figures, Tables and Codes. This file contains supplementary figures, tables and codes that provide further insights into the experimental setup and results discussed in the article.

Author contributions

M.M.B.J.: Conceptualization, methodology, formal analysis, investigation, writing—original draft, draft preparation. A.L.: Methodology, validation, formal analysis, review, supervision. H.J.H.: Methodology, validation, formal analysis, review, supervision. H.T.: Conceptualization, methodology, validation, formal analysis, review, supervision.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Availability of data and materials

The datasets used and/or analyzed during the current study are available from the first author on reasonable request.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors submitting this manuscript have no conflict of interest to declare.

Received: 8 August 2023 Accepted: 30 April 2024

Published online: 12 May 2024

References

- Jain AK, Narasimha Murty M, Flynn PJ. Data clustering: a review. *ACM Comput Surv.* 1999;31(3):264–323.
- Lingras P, Huang X. Statistical, evolutionary, and neurocomputing clustering techniques: cluster-based vs object-based approaches. *Artif Intell Rev.* 2005;23:3–29.
- MacQueen J. Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability.* 1967;1(14)
- Xu R, Wunsch D. Survey of clustering algorithms. *IEEE Trans Neural Netw.* 2005;16(3):645–78.
- Rodriguez SIR, de Carvalho FAT. Fuzzy clustering algorithms with distance metric learning and entropy regularization. *Appl Soft Comput.* 2021;113: 107922.
- Jothi R, Mohanty SK, Ojha A. Gene expression clustering using local neighborhood-based similarity measures. *Comput Electr Eng.* 2021;91: 107032.
- Nozad SAN, Haeri MA, Folino G. SDCOR: Scalable density-based clustering for local outlier detection in massive-scale datasets. *Knowl Based Syst.* 2021;228: 107256.
- Shirkhorshidi AS, Saeed A, Wah TY. A comparison study on similarity and dissimilarity measures in clustering continuous data. *PLoS ONE.* 2015;10(12): e0144059.
- Boriah S, Chandola V, Kumar V. Similarity measures for categorical data: a comparative evaluation. *Proceedings of the 2008 SIAM international conference on data mining. Society for Industrial and Applied Mathematics, 2008*
- Yan Q, et al. A discriminated similarity matrix construction based on sparse subspace clustering algorithm for hyperspectral imagery. *Cogn Syst Res.* 2019;53:98–110.
- Renjith S, Sreekumar A, Jathavedan M. Performance evaluation of clustering algorithms for varying cardinality and dimensionality of data sets. *Mater Today Proc.* 2020;27:627–33.
- Shrifan NHMM, Akbar MF, Isa NAM. An adaptive outlier removal aided k-means clustering algorithm. *J King Saud Univ Comput Inf Sci.* 2022;34(8):6365–76.
- Strehl A, Ghosh J, Mooney R. Impact of similarity measures on web-page clustering. *Workshop on artificial intelligence for web search (AAAI 2000).* 2000;58.
- Zhang Z, Huang K, Tan T. Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. *18th International Conference on Pattern Recognition (ICPR'06).* IEEE, 2006;3
- Khalifa, Aysha AI, Maciej Haranczyk, and John Holliday. "Comparison of nonbinary similarity coefficients for similarity searching, clustering and compound selection." *Journal of chemical information and modeling* 49.5 (2009): 1193–1201
- Lourenço, Fernando, Victor Lobo, and Fernando Bacao. "Binary-based similarity measures for categorical data and their application in Self-Organizing Maps." (2004): 1–18

17. Xu D, Tian Y. A comprehensive survey of clustering algorithms. *Ann Data Sci.* 2015;2:165–93.
18. Taheri R, et al. Similarity-based Android malware detection using Hamming distance of static binary features. *Future Gener Comput Syst.* 2020;105:230–47.
19. Vajnovszki V. A new Euler-Mahonian constructive bijection. *Discret Appl Math.* 2011;159(14):1453–9.
20. Gnedin A, Olshanski G. The two-sided infinite extension of the Mallows model for random permutations. *Adv Appl Math.* 2012;48(5):615–39.
21. Deutsch E, Pergola E, Pinzani R. Six bijections between deco polyominoes and permutations. 2008, arXiv preprint, [arXiv:0810.2876](https://arxiv.org/abs/0810.2876)
22. Barth W, Jünger M, Mutzel P. Simple and efficient bilayer cross counting. *Graph Drawing: 10th International Symposium, GD 2002 Irvine, CA, USA, August 26–28, 2002 Revised Papers 10.* Springer Berlin Heidelberg, 2002.
23. Grabusts P. The choice of metrics for clustering algorithms. *Environ Technol Resourc Proc Int Sci Pract Confer.* 2011;2:20.
24. Jozan MM, Taghiyareh F, Faili H. An inversion-based genetic algorithm for grouping of students. *Proc. 7th Int. Conf. Virtual Learn.* Vol. 1. No. 1. 2012
25. Han J, Pei J, Tong H. *Data mining: concepts and techniques.* Burlington: Morgan kaufmann; 2022.
26. Melnykov V, Chen W-C, Maitra R. MixSim: an R package for simulating data to study performance of clustering algorithms. *J Stat Softw.* 2012;51:1–25.
27. Maitra R, Melnykov V. Simulating data to study performance of finite mixture modeling and clustering algorithms. *J Comput Graph Stat.* 2010;19(2):354–76.
28. Dempster AP, Laird NM, Rubin DB. Maximum likelihood from incomplete data via the EM algorithm. *J Roy Stat Soc: Ser B (Methodol).* 1977;39(1):1–22.
29. Fraley C, Raftery AE. MCLUST version 3 for R: Normal mixture modeling and model-based clustering. Vol. 504. Technical report, 2006
30. Chris D, He X. Cluster merging and splitting in hierarchical clustering algorithms. 2002 IEEE International Conference on Data Mining, 2002. Proceedings.. IEEE, 2002
31. Kaufman L, Rousseeuw PJ. *Partitioning around medoids (Program PAM).* Wiley Series in Probability and Statistics, Hoboken, NJ, USA: John Wiley & Sons, Inc., 1990, pp. 68–125, <https://doi.org/10.1002/9780470316801.ch2>
32. Maechler M, Rousseeuw P, Struyf A, Hubert M, Hornik K. *Cluster: cluster analysis basics and extensions.* R package version 1.14.3. 2012.
33. R Core Team, R. R: a language and environment for statistical computing. 2013: 275–286
34. Halkidi M, Batistakis Y, Vazirgiannis M. On clustering validation techniques. *J Intell Inf Syst.* 2001;17:107–45.
35. Agustí LE, et al. A new grouping genetic algorithm for clustering problems. *Expert Syst Appl.* 2012;39(10):9695–703.
36. Rand WM. Objective criteria for the evaluation of clustering methods. *J Am Stat Assoc.* 1971;66(336):846–50.
37. Thong PH. Picture fuzzy clustering for complex data. *Eng Appl Artif Intell.* 2016;56:121–30.
38. Khan L, Luo F. Hierarchical clustering for complex data. *Int J Artif Intell Tools.* 2005;14(05):791–809.
39. Berikov V. Weighted ensemble of algorithms for complex data clustering. *Pattern Recogn Lett.* 2014;38:99–106.
40. Cupertino TH, Huertas J, Zhao L. Data clustering using controlled consensus in complex networks. *Neurocomputing.* 2013;118:132–40.
41. Hämäläinen J, Kärkkäinen T, Rossi T. Improving scalable K-means++. *Algorithms.* 2020;14(1):6.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.