Research Article

Morphological Accuracy Data Clustering: A Novel Algorithm for Enhanced Cluster Analysis

Abdel Fattah Azzam,1,2 Ahmed Maghrabi,3 Eman El-Naqeeb,4 Mohammed Aldawood,1 and Hewayda ElGhawalby

1Department of Mathematics, Faculty of Science and Humanities, Prince Sattam Bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia
2Department of Mathematics, Faculty of Science, New Valley University, Elkharga 72511, Egypt
3Department of Mathematics, Faculty of Science, Kafrelsheikh University, Kafrelsheikh, Egypt
4Department of Physics and Engineering Mathematics, Faculty of Engineering, Port Said University, Port Fuad, Egypt

Correspondence should be addressed to Abdel Fattah Azzam; azzam0911@yahoo.com

Received 20 February 2024; Revised 25 April 2024; Accepted 7 May 2024; Published 22 May 2024

In today’s data-driven world, we are constantly exposed to a vast amount of information. This information is stored in various information systems and is used for analysis and management purposes. One important approach to handle these data is through the process of clustering or categorization. Clustering algorithms are powerful tools used in data analysis and machine learning to group similar data points together based on their inherent characteristics. These algorithms aim to identify patterns and structures within a dataset, allowing for the discovery of hidden relationships and insights. By partitioning data into distinct clusters, clustering algorithms enable efficient data exploration, classification, and anomaly detection. In this study, we propose a novel centroid-based clustering algorithm, namely, the morphological accuracy clustering algorithm (MAC algorithm). The proposed algorithm uses a morphological accuracy measure to define the centroid of the cluster. The empirical results demonstrate that the proposed algorithm achieves a stable clustering outcome in fewer iterations compared to several existing centroid-based clustering algorithms. Additionally, the clusters generated by these existing algorithms are highly susceptible to the initial centroid selection made by the user.

1. Introduction

Clustering algorithms play a crucial role in data analysis and machine learning by enabling the identification and grouping of similar data points. With the exponential growth of data in various domains, the need for effective data exploration and organization has become paramount. Clustering algorithms offer a solution to this challenge by partitioning data into meaningful clusters, allowing for the discovery of hidden patterns and relationships. There are various types of clustering algorithms, each with its own approach and characteristics, such as:

(1) Centroid-based clustering: in this method, the data are grouped into clusters based on the similarity of the data points to the central position or centroid of each cluster.

(2) Hierarchical clustering: this algorithm builds a hierarchy of clusters either by starting with each data point as a separate cluster and merging them iteratively or by initially considering all data points as a single cluster and recursively splitting them. The number of clusters does not need to be specified in advance.

(3) Density-based spatial clustering: this algorithm aims to group data points based on their proximity and density. It considers a data point as a core point if it has a sufficient number of other data points within a specified radius. These core points, along with their
neighboring points, form a cluster. Data points that do not meet the density requirements are considered noise or outliers.

(4) Distribution-based clustering: this is a type of clustering algorithm that assumes the data points are generated from a specific probability distribution. The goal is to estimate the parameters of the distribution to identify the underlying clusters in the data.

Centroid-based clustering algorithms have emerged as powerful tools for organizing and analyzing large volumes of data in information systems. These algorithms utilize the concept of centroids, which represent the center or average of a group of data points, to form clusters based on similarity or dissimilarity measures. The ability of centroid-based clustering algorithms to efficiently handle high-dimensional data and their interpretability makes them particularly suitable for applications in information systems. The objective of this research paper is to provide a centroid-based clustering algorithm that can be used for information systems. The proposed algorithm uses a morphological accuracy measure to define the centroid of the cluster. In literature, there is a tremendous body of research on clustering algorithms; for instance, the work done in [1–11].

The rest of this paper is organized as follows:

1. Mathematical morphology: in this section, we delve into the fundamental concepts of mathematical morphology, a theory and technique for the analysis and processing of geometrical structures. We will explore its basic definitions, including dilation, erosion, opening, and closing [12–14].

2. Neighborhood morphological operators: in this section, we explore the concept and definition of neighborhood morphological operators. These operators, including dilation, erosion, opening, and closing, form the foundation of mathematical morphology.

3. Morphological clustering algorithm: in this section, we introduce a novel clustering algorithm based on the concepts of mathematical morphology. We will discuss how morphological operations can be incorporated into clustering processes to enhance their effectiveness and interpretability. Furthermore, we will introduce the morphological accuracy measure, inspired by the principles of mathematical morphology.

4. Clustering performance evaluation: in this section, we focus on assessing the effectiveness of clustering algorithms by using the Calinski–Harabasz index. This index provides a quantitative measure of the compactness and separation of clusters. We will explain the calculation of the index and its interpretation, discussing how it can be used to compare and evaluate different clustering solutions.

5. Conclusion: in the final section of the study, we summarize the key findings and contributions of the research. We highlight the importance of mathematical morphology in understanding spatial structures and its potential for enhancing clustering algorithms. We discuss the insights gained from the survey of centroid-based clustering algorithms and the introduction of the morphological clustering algorithm. Finally, we conclude by discussing future research directions and potential applications of morphological concepts in clustering algorithms and information systems.

2. Mathematical Morphology

In the late 1960s, a distinct branch of image analysis known as “mathematical morphology” emerged. This branch focuses on the mathematical theory of describing shapes using set concepts to extract meaningful information from images. Mathematical morphology (MM) is a subset of nonlinear image processing and analysis that was initially developed by Serra [13, 14]. Its primary focus lies in the study of the geometric structure within an image [12]. As we aim to provide a comprehensive understanding of both classical and neighborhood morphological operators, we structure this section in the following manner:

(1) Classical morphological operators: in this section, we provide an overview of the concepts and definitions of classical morphological operators [12, 14]. These operators, including dilation, erosion, opening, and closing, form the foundation of mathematical morphology.

(2) Neighborhood morphological operators: in this section, we explore the concept and definition of neighborhood morphological operators. These operators extend the classical morphological operators by considering the spatial relationships between elements within a neighborhood. We explore how neighborhood morphological operators can capture more detailed information about the local structure, enabling more precise analysis and processing [15].

2.1. Classical Mathematical Morphology. In classical mathematical morphology, the two fundamental operators are dilation and erosion. In this section, we will review their definitions as presented by Heijmans [12], which are consistent with the original definitions of Minkowski addition and subtraction [14].

2.1.1. The Dilation Operator. The dilation operator is used to expand a set based on the shapes contained within the object under consideration [14]. It involves the use of a structuring element, which is a small shape or pattern that defines the neighborhood around each element. The dilation operation involves placing the structuring element at each point of the object and expanding it to cover all the neighboring points that intersect with the structuring element. The result is
a new set that encompasses the original object while incorporating the shapes and structures present in the structuring element.

**Definition 1.** For a universe set $U$ and the two subsets $A, SE \subseteq U$, the dilation of the set $A$ in relation to the structure element $SE$ is defined as follows:

$$\delta(A) = A \oplus SE = \bigcup_{b \in SE} A_b,$$

(1)

where $A_b = \{a + b \in X | a \in A, b \in SE\}$.

2.1.2. The Erosion Operator. The erosion operator is closely related to Minkowski subtraction [13]. It is used to reduce a set based on the shapes included in the object under study. Similar to the dilation operator, the erosion operator also utilizes a structuring element. The erosion operation involves placing the structuring element at each point of the object and checking if all the points within the structuring element are also present in the object. If they are, the point is retained; otherwise, it is removed from the set. This process results in a new set that represents the original object with the shapes and structures that can be completely contained within the structuring element removed.

**Definition 2.** For a universe set $U$ and the two subsets $A, SE \subseteq U$, the erosion of the set $A$ relation to the structure element $SE$ is defined as follows:

$$\epsilon(A) = A \ominus SE = \{p \in U | SE_p \subseteq A\},$$

(2)

where $SE_p = \{b + p \in U | b \in SE\}, \forall p \in U$.

2.2. Neighborhood Mathematical Morphology. By using the concept of the topological neighborhood, the definition of neighborhood-dilation and neighborhood-erosion operations; this idea is assigned to a structure element that lies in the neighborhood of each point instead of the only one structure element for the universe [15].

**Definition 3.** For any universe set $U$, where $SE_x$ is the neighborhood-structuring element and the two subsets $A, SE_x \subseteq U$, the neighborhood-erosion and neighborhood-dilation operators ($n \cdot \epsilon(A), n \cdot \delta(A)$) are defined as follows:

(i) The neighborhood-erosion is given by:

$$n \cdot \epsilon(A) = \{x : x \in U | SE_x \subseteq A\},$$

(ii) The neighborhood-dilation is given by:

$$n \cdot \delta(A) = \{x : x \in U | SE_x \cap A \neq \emptyset\}.$$

**Definition 4.** For any set $A \subseteq X$, the morphological accuracy of $A$ is defined to be the ratio between the size of the neighborhood-eroded set of $A$ and the size of the neighborhood-dilated set of $A$, namely, $\mathcal{A}_{\delta}\epsilon(A) = |n \cdot \epsilon(A)| / |n \cdot \delta(A)|$, where $|n \cdot \delta(A)| \neq 0$.

3. Centroid-Based Clustering Algorithms

The purpose of clustering is to assign similar objects into the same cluster and dissimilar objects into different clusters. In many applications, clusters may not necessarily have clear and sharp boundaries. On the other hand, we find the distances between them, and two clusters are almost the same [16, 17].

In general, clustering techniques work like this: choose a center point for some cluster, the distance of a data point from the center increases, and the probability of it being a part of that cluster decreases.

Eventually, for a universe set $U$, the clusters $C = \{C_1, C_2, \ldots, C_k\}$ should hold the following properties:

1. $C_i \neq \emptyset, \ i = 1, 2, \ldots, k$,
2. $\bigcup_{i=1}^{k} C_i = U$,
3. $C_i \cap C_j = \emptyset, \ i \neq j$.

One of the well-defined clustering techniques is the $K$-mean technique [17] and the $k$-medoids clustering [19]. The centroid-based clustering algorithms are the most known and widely used clustering tools. It is sensitive to provide the initial parameter, and it may be considered fast and efficient. One of the most used clustering algorithms is the $k$-means algorithm and the $k$-medoids clustering. The primary objective of this paper is to develop a centroid clustering algorithm that emphasizes simplicity and effectiveness, specifically focusing on the $k$-means and $k$-medoids algorithms. In Section 4, we will explore alternative approaches for selecting initial centroids and assess their performance through simulations involving student datasets. The proposed centroid clustering algorithm aims to provide a straightforward and efficient solution for partitioning data into clusters. By examining different methods for selecting initial centroids, we can evaluate their impact on the clustering results and overall algorithm performance. This analysis will be conducted using simple simulated student datasets, which will allow us to assess the algorithm’s effectiveness in real-world scenarios [20–24].

3.1. The K-Means Algorithm. K-means clustering is a well-known technique for performing nonhierarchical clustering. As a centroid-based algorithm, its primary objective is to partition data into a predetermined number of clusters, denoted as $k$, which is specified by the user. The algorithm begins by randomly selecting $k$ elements from the dataset. These elements serve as the initial centroids or center points of the $k$-clusters. Each data point in the dataset is assigned to the nearest centroid based on the Euclidean distance. This step forms the initial clusters. The centroid of each cluster is recalculated as the mean of all data points currently assigned to that cluster. This step updates the clusters’ center points to better represent their members. Continuously update and refine the centroids of each cluster until the centroids no longer change, or the changes are below a certain threshold, indicating that the algorithm has converged and the clusters have reached a steady state [10, 17, 18].
The steps of the $k$-means clustering algorithm can be summarized in Figure 1:

The $k$-means clustering algorithm is simple yet powerful, capable of efficiently partitioning data into distinct clusters. However, it does have certain limitations and disadvantages:

1. The performance of $k$-means clustering is highly sensitive to the initial selection of centroids. Different initializations can lead to different clustering results, which may not necessarily be the optimal solution. This sensitivity makes it important to carefully choose the initial centroids to ensure meaningful and accurate clustering.

2. The number of clusters, $k$, needs to be specified in advance. However, determining the optimal value of $k$ can be challenging, as it requires prior knowledge or trial-and-error experimentation. Choosing an inappropriate value of $k$ can lead to suboptimal clustering results or misinterpretation of the data structure.

3. $K$-means clustering is sensitive to outliers in the data. Outliers can significantly impact the centroid positions and distort the clustering results. Outliers may be assigned to clusters even though they do not truly belong to them, affecting the accuracy and interpretability of the clustering solution.

4. $K$-means clustering assumes that the clusters have similar sizes and variances. However, real-world data often exhibit uneven cluster sizes and varying variances, which can result in suboptimal clustering performance.

5. $K$-means clustering may struggle to handle clusters with complex shapes.

Despite these limitations, $k$-means clustering remains a widely used and effective algorithm for many applications. Careful consideration of the limitations and appropriate preprocessing of the data can help improve the performance of $k$-means clustering [6].

**Example 1.** Table 1 shows a simple information system, including eight students. $X = \{\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7, \beta_8\}$ and six subjects (Subject $a$, Subject $b$, Subject $c$, Subject $d$, Subject $e$, and Subject $f$).

To commence, we decide to divide the dataset into two clusters ($k = 2$).

Then, we choose two random students to be the center element for each cluster; let it be $\beta_4$ and $\beta_3$. Hence, we use the formula (1) to compute the Euclidean distance between the chosen clusters’ center students and all the students.

$$d = \sqrt{\sum_{i=1}^{k} \sum_{x \in X} (x_i - c_i)^2}, \quad (3)$$

where $c_i = 1/|x_i| \sum_{x \in c_i} x_i$.

Now, we are able to construct the two clusters $C_1, C_2$ based on the nearest students to each cluster $C_1 = \{\beta_1, \beta_4, \beta_5, \beta_6, \beta_7, \beta_8\}$, and $C_2 = \{\beta_3, \beta_2\}$.

With those two clusters in hand, we define the mean (centroid) of each cluster, which is shown in Table 2.

The previous steps are to be repeated until reaching a steady clustering.

Eventually, we have the final clusters to be:

$$C_1 = \{\beta_1, \beta_4, \beta_5, \beta_6, \beta_7, \beta_8\}, \quad C_2 = \{\beta_3, \beta_2\}. \quad (4)$$

Note that, if we start with different random center students, we get completely different clusters.

For instance, in Table 3, we may get the following results, when choosing the following:

3.2. The $K$-Medoids Algorithm. In $k$-medoids clustering, representative objects called medoids are used instead of centroids. The medoid is defined as the most centrally located object within each cluster. This means that the medoid is the data point that has the minimum average dissimilarity to all other points in the same cluster. By relying on the medoids as the center points of the clusters, $k$-medoids clustering demonstrates reduced sensitivity to outliers compared to $k$-means clustering. The $k$-medoids algorithm iteratively determines the optimal medoids and assigns each
The methodology of the k-medoids algorithm can be illustrated by Figure 2.

While k-medoids clustering offers advantages over k-means clustering, it also has some limitations and disadvantages:

(1) The computational complexity of k-medoids clustering is higher compared to k-means clustering. This is because, in each iteration, the algorithm needs to calculate dissimilarities between each data point and all potential medoids. As the number of data points and clusters increases, the computational cost of k-medoids clustering can become significant.

(2) The performance of k-medoids clustering can be highly dependent on the initial selection of medoids. Different initializations can lead to different clustering results. Therefore, it is crucial to carefully choose the initial medoids to ensure meaningful and accurate clustering.

(3) K-medoids clustering may face scalability issues when dealing with large datasets. As the number of data points increases, the computation of dissimilarities and the search for optimal medoids become more time-consuming and memory-intensive, making the algorithm less efficient for big data applications.

(4) Like other clustering algorithms, k-medoids clustering can struggle with high-dimensional data. In high-dimensional spaces, the concept of distance becomes less meaningful, and the dissimilarity calculations may not accurately capture the true similarities between data points. Dimensionality reduction techniques or other preprocessing methods may be necessary to address this issue.

(5) K-medoids clustering assumes that clusters have a single representative object (medoid) at their center. This assumption restricts the flexibility of cluster shapes, as the algorithm may not be able to handle complex cluster structures effectively.

Despite these limitations, k-medoids clustering remains a valuable technique, especially in scenarios where outlier robustness and interpretability are critical considerations.

Example 2. For the system given in Example 1, we apply the k-medoids cluster as follows:
To commence, we choose to divide the dataset into two clusters \((k = 2)\).

Then, we choose two random students to be the medoid for each cluster; that is, \(c_1 = \beta_3, c_2 = \beta_8\). Hence, we use the formula (5) to compute the Manhattan distance between the chosen medoid students and the students, where \(Md\) is Manhattan distance:

\[
Md = \sum_{i=1}^{k} \sum_{x \in X} |x_i - c_i|.
\]  

(5)

Now, we are able to construct the two clusters \(C_1\) and \(C_2\) based on the nearest students to each cluster. Then, we compute the total cost for each cluster

\[
\text{total cost} = \sum_{x \in X} |x_i - c_i|,
\]

(6)

where the Manhattan distance between the chosen medoid students and all the students.

1. \(Md(\beta_1, \beta_3) = |83 - 26| + |40 - 89| + |17 - 19| + |72 - 71| + |90 - 44| + |60 - 22| = 193.\)

2. \(Md(\beta_2, \beta_3) = |22 - 26| + |93 - 89| + |81 - 19| + |71 - 71| + |30 - 44| + |40 - 22| = 102.\)

3. \(Md(\beta_1, \beta_8) = |83 - 30| + |40 - 87| + |17 - 66| + |72 - 69| + |90 - 50| + |60 - 93| = 225.\)

4. \(Md(\beta_2, \beta_8) = |22 - 30| + |93 - 87| + |81 - 66| + |71 - 69| + |30 - 50| + |40 - 93| = 104.\)

From Table 4, we obtained:

\[
C_1 = \{\beta_1, \beta_2, \beta_3, \beta_5\},
\]

\[
C_2 = \{\beta_4, \beta_6, \beta_7, \beta_8\},
\]

then summation of the total cost for each cluster is

\[
\text{total cost} = \text{cost}(\beta_3, \beta_1) + \text{cost}(\beta_3, \beta_2) + \text{cost}(\beta_3, \beta_5)
\]

\[+ \text{cost}(\beta_8, \beta_4) + \text{cost}(\beta_8, \beta_6) + \text{cost}(\beta_8, \beta_7),
\]

(8)

total cost \(_1\) = 193 + 102 + 148 + 131 + 153 + 153 = 880.

Randomly select one nonmedoid student and recalculate the cost \(c_1 = \beta_3, c_3 = \beta_6\).

From Table 5, we obtained:

\[
C_1 = \{\beta_2, \beta_3, \beta_8\},
\]

\[
C_3 = \{\beta_1, \beta_4, \beta_5, \beta_6, \beta_7\},
\]

(9)

total cost \(_1\) = 102 + 132 + 166 + 152 + 83 + 76 = 711,

\[
D = \text{total cost}_1 - \text{total cost}_2 = 711 - 880 < 0.
\]

We select randomly another one nonmedoid student and recalculate the cost \(c_1 = \beta_3, c_4 = \beta_5\).

\[
C_1 = \{\beta_2, \beta_3, \beta_8\},
\]

\[
C_4 = \{\beta_1, \beta_4, \beta_5, \beta_6, \beta_7\},
\]

total cost \(_3\) = 556,

\[
D = \text{total cost}_3 - \text{total cost}_2 = 556 - 711 < 0.
\]

(10)

### 4. Morphological Accuracy Cluster (MAC)

In this section, we introduce a clustering algorithm based on the morphological concepts outlined earlier in [15], including the neighborhood-erosion operator, neighborhood-dilation operator, and the morphological accuracy measure. This proposed algorithm determines the centroid of each cluster, with the number of clusters decided upon by the user. At the heart of the proposed algorithm, the centroid for each cluster is chosen using the morphological accuracy measure for each data point. The higher the accuracy, and the more stable the set containing the point, the closer objects to each cluster are assigned based on the Euclidean distance between the centroids and the remaining points. Lastly, we apply the neighborhood-erosion operator to each cluster until we obtain a new centroid for each one. In this stage, we examine if the clusters are identical to those obtained previously. If they are not, we repeat the operation until we achieve a steady set of clusters.
Remark 5. To construct a family of neighborhood structure elements, it is necessary to assign a neighborhood radius for each object. Unlike the fuzzy $\alpha$-cut concept, where the $\alpha$-cut value is the same for the entire dataset, utilizing a different radius for each object provides more accurate results. This allows for a more precise representation of the individual characteristics and relationships within the data.

4.1. The MAC Methodology. To commence the implementation of the proposed strategy for data clustering using the neighborhood and morphological concepts, we can follow the following steps:

(1) Define the universe set $X$ and the distance function $\delta$, as well as the modified distance function $\delta^*$ based on the specific requirements of the problem. For this discussion, we will use the following formulas:

$$\delta^*(x_i, x_j) = \frac{\delta(x_i, x_j)}{\max_{i \neq n} x_i} \quad (13)$$

Where

$$\delta(x_i, x_j) = \sqrt{\sum_{i,j=1}^{n} (x_i - x_j)^2}, \quad \forall x_i, x_j \in X. \quad (14)$$

(2) Determine the neighborhood radius $R$ for each object in the dataset, using the formula:

$$R = \frac{\sum_{i=1}^{n} x_i}{n-2}. \quad (15)$$

(3) Deduce a neighborhood-structuring element for each object,

$$SE_x = \{ y \in X | \delta^*(x, y) < \xi \ast R \}. \quad (16)$$

Taking into account the adjustable parameter $\xi$.

(4) Apply the clustering algorithm, utilizing the morphological concepts presented earlier, such as the neighborhood-erosion operator, neighborhood-dilation operator, and the morphological accuracy measure.

(5) Define the centroid based on the value of the morphological accuracy.

(6) Assign objects to clusters based on their morphological accuracy and proximity to the cluster centroids.

(7) Continuously update and refine the centroids of each cluster using the neighborhood-erosion/dilation operators until a steady set of clusters is obtained. This involves checking if the clusters remain consistent with previously obtained clusters and repeating the operation if necessary.

This methodology can be summarized in the following Figure 3.

Example 3. For the system given in Example 1, we construct morphological cluster as follows:

Step 1. In Table 6, we construct modified distance matrix using formula (4).

Step 2. Using formulas (16) and (17) at $\xi = 0.8$ to compute the neighborhood radius for each object, which is shown in Table 7.

Step 3. Deduce a neighborhood structure element for each object.

$$SE_{\beta_1} = \{ \beta_1 \},$$
$$SE_{\beta_2} = \{ \beta_2, \beta_3 \},$$
$$SE_{\beta_3} = \{ \beta_3 \},$$
$$SE_{\beta_4} = \{ \beta_4 \},$$
$$SE_{\beta_5} = \{ \beta_1, \beta_5, \beta_6, \beta_7 \},$$
$$SE_{\beta_6} = \{ \beta_5, \beta_6, \beta_7 \},$$
$$SE_{\beta_7} = \{ \beta_5, \beta_6, \beta_7 \},$$
$$SE_{\beta_8} = \{ \beta_2, \beta_8 \}. \quad (17)$$
Table 6: The modified Euclidean distance matrix between objects.

<table>
<thead>
<tr>
<th>(d_{ij})</th>
<th>(\beta_1)</th>
<th>(\beta_2)</th>
<th>(\beta_3)</th>
<th>(\beta_4)</th>
<th>(\beta_5)</th>
<th>(\beta_6)</th>
<th>(\beta_7)</th>
<th>(\beta_8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\beta_1)</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.78</td>
<td>0.49</td>
<td>0.79</td>
<td>0.79</td>
<td>1</td>
</tr>
<tr>
<td>(\beta_2)</td>
<td>1</td>
<td>0.69</td>
<td>0.80</td>
<td>1</td>
<td>0.93</td>
<td>0.59</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\beta_3)</td>
<td>0.79</td>
<td>0.54</td>
<td>0</td>
<td>1</td>
<td>0.82</td>
<td>0.91</td>
<td>1</td>
<td>0.85</td>
</tr>
<tr>
<td>(\beta_4)</td>
<td>0.61</td>
<td>0.62</td>
<td>0.97</td>
<td>0</td>
<td>0.56</td>
<td>0.77</td>
<td>0.58</td>
<td>0.61</td>
</tr>
<tr>
<td>(\beta_5)</td>
<td>0.39</td>
<td>0.79</td>
<td>0.82</td>
<td>0.57</td>
<td>0</td>
<td>0.44</td>
<td>0.42</td>
<td>0.76</td>
</tr>
<tr>
<td>(\beta_6)</td>
<td>0.63</td>
<td>0.79</td>
<td>0.91</td>
<td>0.79</td>
<td>0.44</td>
<td>0</td>
<td>0.42</td>
<td>0.80</td>
</tr>
<tr>
<td>(\beta_7)</td>
<td>0.59</td>
<td>0.70</td>
<td>0.94</td>
<td>0.56</td>
<td>0.40</td>
<td>0.40</td>
<td>0</td>
<td>0.70</td>
</tr>
<tr>
<td>(\beta_8)</td>
<td>0.83</td>
<td>0.49</td>
<td>0.89</td>
<td>0.66</td>
<td>0.80</td>
<td>0.84</td>
<td>0.78</td>
<td>0</td>
</tr>
</tbody>
</table>

Step 4. Operate the neighborhood-erosion and neighborhood-dilation operators to compute the morphological accuracy for each object, which is shown in Table 8.

Step 5. To assign the clusters’ center elements, we choose the two elements with the highest and lowest accuracy values.

In this example, \(\beta_4\) has the highest accuracy values, while at the lowest value, we have five elements \(\beta_2, \beta_5, \beta_6, \beta_7, \) and \(\beta_8\).

In this case, we choose the two center elements to be \(\beta_4\) and \(\beta_2\).

That is, we assign \(\beta_4\) with clusters \(C_1\) and \(\beta_2\) with cluster \(C_2\).

Step 6. Calculate the distance between all the objects in \(X\) and the two centroids, which is shown in Table 9.

Step 7. Assign each element in \(X\) to the closest cluster:

\[
C_1 = \{\beta_1, \beta_4, \beta_5, \beta_6, \beta_7\},
\]

\[
C_2 = \{\beta_2, \beta_3, \beta_8\}.
\]

Step 8: Deduce the erosion of the two clusters:

\[
n_e(C_1) = \{\beta_1, \beta_4, \beta_5, \beta_6, \beta_7\},
\]

\[
n_e(C_2) = \{\beta_2, \beta_3, \beta_8\}.
\]

Step 9: In this step, we examine the stability of the obtained clusters, which are

\[
C_1 = \{\beta_1, \beta_4, \beta_5, \beta_6, \beta_7\},
\]

\[
C_2 = \{\beta_2, \beta_3, \beta_8\}.
\]

5. Clustering Performance Evaluation

In assessing the effectiveness of clustering algorithms, a key measure is the Calinski–Harabasz index also known as the variance ratio criterion. The index is the ratio between-cluster separation to within-cluster dispersion, each normalized by their respective degrees of freedom. A higher

Table 8: The neighborhood-erosion, neighborhood-dilation, and the morphological accuracy of some subsets of information system.

<table>
<thead>
<tr>
<th>(A)</th>
<th>(n_e(A))</th>
<th>(n_\delta(A))</th>
<th>Morphological accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\beta_1)</td>
<td>(\beta_1)</td>
<td>(\beta_1, \beta_5)</td>
<td>0.5</td>
</tr>
<tr>
<td>(\beta_2)</td>
<td>(\emptyset)</td>
<td>(\beta_2, \beta_8)</td>
<td>0</td>
</tr>
<tr>
<td>(\beta_3)</td>
<td>(\beta_3)</td>
<td>(\beta_2, \beta_3)</td>
<td>0.5</td>
</tr>
<tr>
<td>(\beta_4)</td>
<td>(\beta_4)</td>
<td>(\beta_4)</td>
<td>1</td>
</tr>
<tr>
<td>(\beta_5)</td>
<td>(\emptyset)</td>
<td>(\beta_5, \beta_6, \beta_7)</td>
<td>0</td>
</tr>
<tr>
<td>(\beta_6)</td>
<td>(\emptyset)</td>
<td>(\beta_5, \beta_6, \beta_7)</td>
<td>0</td>
</tr>
<tr>
<td>(\beta_7)</td>
<td>(\emptyset)</td>
<td>(\beta_5, \beta_6, \beta_7)</td>
<td>0</td>
</tr>
<tr>
<td>(\beta_8)</td>
<td>(\emptyset)</td>
<td>(\beta_8)</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 9: Euclidean distance between sets \(A\) and \(\{C_1, C_2\}\).

<table>
<thead>
<tr>
<th>(d_{ij})</th>
<th>(c_1 - \text{centroid})</th>
<th>(c_2 - \text{centroid})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\beta_1)</td>
<td>73.6070</td>
<td>120.9421</td>
</tr>
<tr>
<td>(\beta_2)</td>
<td>75.30604</td>
<td>0</td>
</tr>
<tr>
<td>(\beta_3)</td>
<td>93.58953</td>
<td>66.30234</td>
</tr>
<tr>
<td>(\beta_4)</td>
<td>54.17564</td>
<td>95.9479</td>
</tr>
<tr>
<td>(\beta_5)</td>
<td>74.28324</td>
<td>96.3908</td>
</tr>
<tr>
<td>(\beta_6)</td>
<td>52.87722</td>
<td>84.70478</td>
</tr>
<tr>
<td>(\beta_7)</td>
<td>61.86275</td>
<td>59.48109</td>
</tr>
</tbody>
</table>

index value suggests a model with better-defined clusters as it encapsulates the principle that an optimal clustering solution should maximize the distance between distinct clusters, while minimizing the variance within each cluster.

This concept can be expressed mathematically as follows:

Given a dataset of \(n\) elements: \(\{x_1, x_2, \ldots, x_k\}\), and the assignment of these elements to \(k\) clusters: \(C = \{C_1, C_2, \ldots, C_k\}\), the Calinski–Harabasz index \((C)\) is defined as follows:

\[
C = \frac{B/(k - 1)}{W/(n - k)},
\]

where \(B\) is the sum of squared distances from each cluster centroid to the overall data centroid, which measures how well the clusters are separated from each other (the higher the better), and \(W\) is the sum of squared distances from each data element to its assigned cluster centroid, which measures the compactness or cohesiveness of the clusters (the smaller the better), that is,

\[
B = \sum_{i=1}^{k} n_i \|c_i - c\|^2,
\]

\[
W = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - c\|^2.
\]

Now, let us use this index to evaluate the clustering algorithms presented in §3.1, §3.2, and §4.1.

5.1. Clustering Performance Evaluation (K-Means). In this section, we evaluate the k-means clustering methodology presented in §3.1, using the results obtained in Example 1.
(i) For the chosen center elements $\beta 3, \beta 4$, the final clusters were $C_1 = \{\beta 3, \beta 2\}$ and $C_2 = \{\beta 1, \beta 4, \beta 5, \beta 6, \beta 7, \beta 8\}$.

Applying the formulas (22) and (23) gives the values $B = 18697.25$ and $W = 24890$. Hence, the Calinski–Harabasz index is $C = 4.5$.

(ii) Similarly, for the center elements $\beta 1, \beta 7$, the final clusters to be: $C_1 = \{\beta 1\}, C_2 = \{\beta 2, \beta 3, \beta 4, \beta 5, \beta 6, \beta 7\}$.

Then, when using the formulas (22) and (23), we get $B = 13056.5$ and $W = 26160$. Hence, the Calinski–Harabasz index is $C = 2.99$.

5.2. Clustering Performance Evaluation (K-Medoids). In this section, we evaluate the k-medoids clustering methodology presented in §3.2, using the results obtained in Example 2.

(i) For the chosen center elements $\beta 3, \beta 8$, the final clusters were $C_1 = \{\beta 1, \beta 2, \beta 3, \beta 3\}, C_2 = \{\beta 4, \beta 6, \beta 7, \beta 8\}$.

Applying the formulas (22) and (23) gives the values $B = 26190.75$ and $W = 35320$. Hence, the Calinski–Harabasz index is $C = 4.44$.

(ii) Similarly, for the center elements $\beta 3, \beta 7$, the final clustering to be: $c_1 = \{\beta 2, \beta 3, \beta 8\}, c_4 = \{\beta 1, \beta 4, \beta 5, \beta 6, \beta 7\}$. Then, when using the formulas (22) and (23), we get $B = 18128.5, W = 22685,$ and $C = 4.79$.

5.3. Clustering Performance Evaluation (MAC). In this section, we evaluate the k-medoid clustering methodology presented in §4.1, using the results obtained in Example 3.

(i) For the chosen center elements $C_1 = \{\beta 1, \beta 4, \beta 5, \beta 6, \beta 7\}$ and $C_2 = \{\beta 2, \beta 3, \beta 8\}$. Applying the formulas (22) and (23) gives the values $B = 21070$ and $W = 25632$. Hence, the Calinski–Harabasz index is $C = 4.9$.

6. Conclusion

In conclusion, we have introduced a novel clustering algorithm called the morphological accuracy cluster algorithm (MAC algorithm). Unlike existing clustering methods, our proposed algorithm utilizes an accuracy measure to define the centroids of the clusters, eliminating the need for predefined centroids. The accuracy measure employed in our algorithm, known as the morphological accuracy, is computed using the concepts of neighborhood-erosion and neighborhood-dilation operators. In the MAC algorithm, the neighborhood-erosion operation is applied to each cluster iteratively, resulting in the determination of a new centroid for each cluster. This process is repeated until the desired clusters are obtained. Empirical results demonstrate that our proposed algorithm achieves a steady cluster state in fewer iterations compared to the traditional k-means algorithm. Additionally, the clusters produced by the k-means algorithm are highly sensitive to the initial centroid selection made by the user. The MAC algorithm offers several advantages over existing clustering methods. It eliminates the need for manual centroid initialization and provides a more robust and accurate clustering solution. By incorporating the morphological accuracy measure, our algorithm can effectively capture the structural information within the data, leading to improved clustering performance. Future research directions may include exploring the applicability of the MAC algorithm to various domains and datasets, as well as investigating its scalability and performance on large-scale datasets. Furthermore, further refinement and optimization of the algorithm could be pursued to enhance its efficiency and effectiveness.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported via funding from the Prince Sattam bin Abdulaziz University project number (PSAU/2024/R/1445).

References


