Simultaneous Clustering and Feature Selection Using Social Group Optimization With Dynamic Threshold Setting for Microarray Data

Y V Nagesh Meesala[1], Ajaya Kumar Parida[2], Anima Naik[3*]
[1] Research Scholar, School of Computer Engineering, KIIT Deemed to be University, Bhubaneswar, Odisha, India, Department of CSE, Raghu Engineering College, Visakhapatnam Andhra Pradesh, India.
[2] School of Computer Engineering, KIIT Deemed to be University, Bhubaneswar, Odisha, India.
[3] Department of CSE, Raghu Engineering College, Visakhapatnam Andhra Pradesh, India,
E-mail: 2181072@kiit.ac.in, nagesh.myv@raghuenggcollege.in, ajaya.paridafcs@kiit.ac.in, anima.naik@raghuenggcollege.in
[*]Corresponding author

Keywords: SGO, real-world data, microarray data, clustering, feature selection

Received: August 29, 2024

In this research, a unique method for automatically and simultaneously choosing significant features as well as cluster numbers from a dataset is proposed. The Social Group Optimization (SGO) algorithm is used as a metaheuristic. The SGO incorporates two new ideas for threshold setting and encoding. During the optimization phase, several features and cluster centers are encoded using the encoding scheme. The dataset variance is utilized to determine the value of the threshold for both clusters as well as features. A new clustering criterion is employed to enhance the efficiency of the search process. We compare the proposed algorithm's performance to eight newly developed clustering algorithms and evaluate it on nine well-known real-world datasets. The statistical significance of the SGO-based approach, evaluated through classification accuracy, is assessed using T-tests. Results indicate that the SGO method is extremely statistically significant in 6 cases, very statistically significant in 2 cases, and statistically significant in 1 case compared to the second-best algorithm. Additionally, this method effectively identifies the optimal number of clusters and features from the dataset without user input. Microarray data is also analyzed using this method to demonstrate the algorithm's accuracy and success.

Povzetek: Raziskava predstavlja metodo SGO_FSC, ki združuje hkratno optimizacijo izbire značilnosti in gručenja z algoritmom socialne optimizacije skupin, kar izboljšuje analizo.

1 Introduction

An unlabeled dataset is divided into clusters of related data points via clustering. Those data points which resemble one another but are distinct from those in other groups make up each group, which is referred to as a cluster [1]. Numerous fields, such as pattern recognition, image segmentation, spatial database analysis, textual document analysis, machine learning, and pattern recognition, use clustering. Numerous clustering algorithms are reported in the literature [2,3], typically categorized into two types: hierarchical and partitional. Hierarchical clustering creates a structure of divisions that include the arrangement of each level within the following level in the structure [4]. Nevertheless, there are numerous drawbacks to hierarchical clustering methods. These include the fact that data points can only be moved inside the cluster to which they have already been assigned and the fact that they cannot differentiate between clusters that overlap [5]. In contrast to this, the data is divided into a number of sets of separate clusters in partitional clustering. This study specifically concentrates on partitional clustering.

All the features are treated equally by many partitional clustering algorithms regardless of their relevance, which may only sometimes be suitable. Certain attributes may be inconsequential or duplicative, particularly in datasets with many dimensions, which can impede attaining superior outcomes. Therefore, selecting relevant features that enhance the clustering process is crucial. This selection not only improves clustering accuracy but also reduces computational time and storage space. Furthermore, the task of identifying the most suitable number of clusters is a considerable obstacle in partitional clustering. Simultaneously choosing appropriate traits and finding out the optimal number of clusters is a particularly difficult undertaking. This requires a method that can automatically and simultaneously calculate the appropriate number of clusters and relevant attributes during runtime. This study introduces a method called SGO_FSC, which combines automatic feature selection and clustering using the Social Group Optimization algorithm. While there have been some efforts in this area, none have completely utilized the statistical property of variance in datasets. As a result, we were inspired to create an algorithm that can effectively find out the number of clusters and their corresponding characteristics.

We aim to concurrently compute the number of clusters and relevant features in this research, even if we don't know the exact number of clusters in the dataset. To encode characteristics and clusters separately, a composite agent representation was devised. Furthermore, to precisely and effectively define the ideal number of clusters and features, a novel threshold concept has been presented. To calculate the thresholds, the statistical attribute, or variance, of a particular dataset is utilized. “Three clustering metrics—the number of clusters, pertinent characteristics, and classification error—are used to assess the effectiveness of the suggested method on real-world datasets. An analysis is done by contrasting the suggested method with eight popular current methods. Analysis of microarray data has also been done using the SGO_FSC. The experimental findings show how SGO_FSC is effective and efficient in finding out the ideal number of clusters with” pertinent attributes.

The key contributions and findings “of this work are summarized as follows:

- **Simultaneous optimization:** The proposed SGO_FSC technique effectively determines both the optimal number of clusters as well as relevant features” accurately and simultaneously.
- **Dynamic threshold setting:** A novel dynamic threshold setting method was introduced for finding out the features as well as the count of clusters. This method enhances the algorithm's ability to adapt to various datasets.
- **Fitness function:** We proposed a new fitness function that improves the search efficiency for optimal clusters and features, contributing to the overall effectiveness of the SGO_FSC algorithm.
- **Experimental evaluation:** The SGO_FSC algorithm was tested on different 9 real-life datasets. The results demonstrated its robustness and versatility across different types of data.
- **Statistical validation:** Statistical t-tests were conducted to validate the significance of the results achieved by SGO_FSC, confirming its superior performance compared to other clustering techniques.
- **Microarray data analysis:** The proposed algorithm was also applied to microarray datasets, where it showed improved performance over other competitive clustering techniques.

The remainder of this work is divided into as follows. The basics of clustering are explained in Sect. 2. The relevant work in the area of simultaneous clustering along with feature selection is briefly explained in Section 3. Sect. 4 presents the specifics of the suggested strategy. Section 5 discusses the experimental findings and focuses on using the” suggested method to analyze microarray data. Finally, Sect. 6 presents conclusions.

2 Scientific background

2.1 Clustering analysis

Consider a set X of N data points, denoted as $X = \{x_1, x_2, x_3, \dots\dots\dots, x_N\}$, where each data point $x_i = (x_{i1}, x_{i2}, x_{i3}, \dots\dots\dots, x_{iD})$ lies in R^D. Here, x_{ij} represents the j-th feature of x_i data point. Assume that X is divided into K clusters, $C = \{C_1, C_2, C_3, \dots\dots\dots, C_K\}$, with data points in a given cluster being identical to each other and data points in distinct clusters being dissimilar [5]. The clusters must meet the following requirements [6]:

- A minimum of one data point must be present in each clust
  $C_K \neq \emptyset$ for k=1, 2, …, K (1)
- Clusters must be mutually exclusive, meaning they do not share any data points:
  $C_K \cap C_l = \emptyset$ for k, l=1, 2, ………., K and k≠l (2)
- Every data point must be included in one of the clusters:
  $$\bigcup_{k=1}^{K} C_K = X \qquad (3)$$

Clusters are created by allocating data points according to how similar or dissimilar they are. The most widely used dissimilarity metric is the Euclidean distance, which is the separation between a data point x_i and the cluster center m_k of a cluster C_K :

$$Dist_{ki} = \left(\sum_{j=1}^{D} (x_{ij} - m_{kj})^2\right)^{1/2} \qquad (4)$$

2.2 Overview of SGO

Satapathy et al [7] proposed a population-based optimization technique named as social group optimization (SGO) algorithm which attempts to model human social behavior to solve difficult problems. In SGO, the potential answers to a problem are seen as a group of people, or social grouping. An individual's knowledge contributes to the assessment of an individual’s performance. The optimization method has two distinct “phases: the Improving Phase and the Acquiring Phase. The solution to the problem corresponds to the person, and the fitness function determines the knowledge level of the person. The procedure for SGO can be outlined as follows:

Step 1: *Person’s initialization*: We initialize the N individual at random within the specified search interval as

$$P_i = (p_{i1}, p_{i2}, p_{i3}, \dots\dots, p_{iD}), \quad \forall i = 1,2,3,4, \dots\dots, N \qquad (5)$$

Where p_{ij} represents the position of ith person in jth dimension in D dimensional space.

Step 2: *Evaluation of fitness and computation of the best (gbest) person of the group*: Calculate the fitness values for each individual at every iteration. The ‘gbest’ individual can be calculated as follows (given a minimization problem”):

$f_i = fitness(P_i)$ (6)

[minvalue, index] =min$\{f_i; i = 1,2,3 \dots\dots, N\}$
gbest=P (index, :) (7)

Step 3: *Compute the Improving phase*: Launch the improvement phase to bring people's knowledge up to date using 'gbest' in the following ways;

$newP_i = c * P_i + rand * (gbest - P_i)$ (8)

Where $rand \sim U(0,1)$, c is known as "self-introspection parameter lies in between 0 and 1.
Update $newP_i$ if it gives a better fitness than P_i.

[minvalue, index] =min$\{f_i; i = 1,2,3 \dots\dots, N\}$ (9)
gbest= P (index, :)

Step 4: *Compute the" Acquiring phase*: Start the acquisition phase by selecting a person at random from the group and going with the "gbest" in order to learn more about them.

```
Randomly select one person$P_r$, where $i \neq r$,
    If $f_i < f_r$
        $newP_i = P_i + rand * (P_i - P_r) + rand * (gbest - P_i)$
    Else
        $newP_i = P_i + rand * (P_r - P_i) + rand * (gbest - P_i)$
    End for                                                    (10)
```

Update $newP_i$ if it gives a better fitness than P_i.

["minvalue, index] =min$\{f_i; i = 1,2,3 \dots\dots, N\}$
gbest=P (index, :) (11)

Step 5: *Termination condition:* Repeat Steps 3 to step 4 until the iteration process reaches its maximum number of iterations. The gbest person will provide desired solution.

3 Related works

This section presents a concise overview of the relevant literature on the simultaneous application of feature selection and clustering,
summarized in a table Table 1 highlights the strengths and weaknesses of each method.

Table 1: Strength and weakness of the methods related to feature selection and clustering

Author(s)	Method	Strengths	Weaknesses
Vaithyanathan & Dom [8]	Bayesian method using stochastic complexity	Simultaneously calculates clusters and characteristics, efficient feature clustering	Dependent on the quality of clustering, vulnerable to noisy features
Frigui & Nasraoui [9]	Feature weighing and clustering	Learns a unique set of feature weights for each cluster, useful in image segmentation	Does not address potential feature redundancy or scalability
Kim et al. [10]	ELSA wrapper with K-Means & Gaussian mixture	Combines wrapper model with two clustering techniques, effective in clustering and feature selection	Limited scalability and potential for local optima issues
Dy & Brodley [11]	EM wrapper framework	Addresses biases in clustering using cross-projection normalization, effective feature subset selection	Sensitive to dimensionality, maximum likelihood criteria can lead to biased results
Roth & Lange [12]	Feature selection with automatic significance	Introduces feature saliency to tackle feature selection, single EM run for features and clusters	Assumes conditional independence, might overlook dependencies among features
Law et al. [13]	An expectation-maximization (EM) algorithm	Effectively selects relevant features by incorporating feature saliency, leading to simultaneous detection of significant features and clusters within a single EM run.	May struggle with datasets that contain highly correlated or noisy features, as the assumption of independence might limit the method's robustness.

Author(s)	Method	Strengths	Weaknesses
Sheng et al. [14]	NMA_CFS (niching memetic algorithm)	Avoids local optima with niching strategy, uses local search to refine clusters and feature centers	Higher computational complexity, sensitive to initialization
Maugis et al. [15]	Backward stepwise selection for Gaussian models	Integrated likelihood criterion for exploring features and clusters simultaneously	Time-consuming due to stepwise selection, may overfit to data
Zeng & Cheung [16]	Feature selection with iterative clustering	Markov blanket filter for eliminating redundant features, iterative clustering integration	May converge prematurely, requires careful tuning of parameters
Sarvari et al. [17]	Harmony search algorithm for clustering and feature selection	Harmony vector representation enhances feature and cluster selection, efficient local search	Similar limitations to NMA_CFS, potentially higher computational cost
Cobos et al. [18]	IHSK (K-Means + Harmony Search)	Combines global and local search, improves cluster positioning accuracy	HS method may require fine-tuning for optimal performance, risk of overfitting
Hamla1& Ghanem [19]	Filter-embedded hybrid feature selection method	Excels at removing redundant and irrelevant features, crucial in high-dimensional microarray data.	Performance depends on the quality of the initial feature subset chosen by the Fisher Score.
Akarsu & Karahoca [20]	FS_ACO (Ant Colony Optimization)	ACO-based clustering followed by backward selection, effective in feature exclusion	May suffer from high complexity and slow convergence
Javani et al. [21]	PSO for clustering and feature selection	Efficient simultaneous clustering and feature selection with novel fitness function	Risk of being trapped in local optima, complex parameter tuning
Guan et al. [22]	Beta-Bernoulli + Dirichlet Process Mixture Model	Hierarchical model with Bayesian formulation for feature selection and clustering	Requires high computational resources, complex to implement
Swetha & Devi [23]	Randomized PSO	Improves PSO with random feature selection, good for handling high-dimensional datasets	Random selection may lead to suboptimal feature sets, convergence issues
Du & Sheng [24]	JCFS (Joint Clustering & Feature Selection)	Combines spectral clustering and supervised feature selection, preserves manifold structure	Complex framework, potential for overfitting in small datasets
Song et al. [25]	FAST (Graph-theoretic clustering)	Effective feature clustering with least spanning tree method, suitable for large datasets	Limited to specific graph-based clustering methods, dependent on feature distribution
Naik & Satapathy [26][27]	Weighted principal components with differential evolution	Simplifies clustering of high-dimensional data, fast convergence	Risk of premature convergence, dependent on parameter tuning
Kumar et al. [28]	Enhanced NMA_CFS	Automated control parameter selection for NMA_CFS, optimizes both clustering and feature selection	Similar to NMA_CFS, potential scalability issues in large datasets

Author(s)	Method	Strengths	Weaknesses
Satapathy et al. [29][30]	Automated feature selection	Focuses on feature selection for datasets, reduces manual effort	May miss complex relationships between features and clusters
Satapathy &Naik [31]	Hybridization of Roughest and DE	Effective feature selection for datasets, reduces manual effort	Risk of premature convergence, dependent on parameter tuning

In presented related works, the primary challenge that remains is balancing the trade-off between accurate feature selection and effective clustering, particularly when dealing with high-dimensional datasets. Various methods utilize evolutionary algorithms, expectation-maximization (EM), or specific filter and wrapper models, but each approach has some limitations such as getting stuck in local optima, difficulty with large datasets, or assumptions about feature independence.

The **SGO** algorithm could be effectively applied to address these limitations in several ways:

- **Scalability and flexibility**: SGO, which models optimization processes based on group behaviors, can adaptively balance exploration and exploitation. In contrast to many other evolutionary and memetic algorithms (such as NMA_CFS, PSO, and FS_ACO), SGO can adjust group dynamics to avoid convergence to local optima. This could enhance clustering effectiveness when datasets are high-dimensional, noisy, or imbalanced.
- **Handling of feature independence**: Methods like those presented by Law et al. and Dy and Brodley assume conditional independence among features, which can be a limiting factor. SGO's stochastic nature and adaptive grouping strategies could better handle interdependencies among features without requiring strict independence assumptions.
- **Simultaneous feature selection and clustering**: Many algorithms discussed in the literature, such as ELSA and IHSK, require separate stages or multiple algorithms for feature selection and clustering. SGO can unify these processes, ensuring the simultaneous selection of optimal features and clusters in a single run, which can improve computational efficiency and the accuracy of both tasks.
- **Dimensionality reduction**: Methods like the harmony search-based algorithms or PSO often struggle with extremely high-dimensional datasets. SGO, by mimicking social behaviors, can more effectively reduce the feature space, selecting only the most relevant features while maintaining or improving classification performance.

In summary, applying SGO to the domain of simultaneous feature selection and clustering offers scope for improvement in terms of convergence behavior, flexibility in handling feature interdependencies, and enhanced scalability for high-dimensional datasets. This could result in better performance across a wide range of challenging clustering tasks compared to existing state-of-the-art methods.

4 Proposed approach

4.1 Automatic feature selection then clustering using SGO

An automatic feature selection then clustering using social group optimization (SGO_FSC) compromises the below mentioned pseudocode.

Pseudocode: SGO_FSC

```
Initialize:
      max_iterations     # Maximum number of iterations
      population_size    # Number of individuals (persons)
      K_max              # Maximum number of clusters, computed using equation (12)

   For each individual in the population:
      Initialize K_max cluster centers randomly
      Initialize K_max + D activation thresholds randomly (where D is the number of data
      dimensions)

   For iteration = 1 to max_iterations:
      For each individual:
         Evaluate activation thresholds for features and clusters (refer to Sect. 4.1.4)
      For each data point:
         Assign the data point to the closest active cluster center based on Euclidean distance
      For each empty cluster:
         Recalculate cluster center (refer to Sect. 4.1.5)
      For each individual:
         Calculate fitness value based on cluster quality and feature selection (refer to Sect. 4.1.6)
```

```
            Use SGO adaptation method to update individual parameters
        For each individual:
            Assign selection thresholds to clusters and features (refer to Sect. 4.1.2)
    For each individual:
            Determine cutoff thresholds based on updated selection thresholds (refer to Sect. 4.1.3)
      Return the best individual:
        Optimal feature subset
        Cluster centers
        Final number of clusters
```

The flowchart of the SGO_FSC algorithm is given in Fig 1.

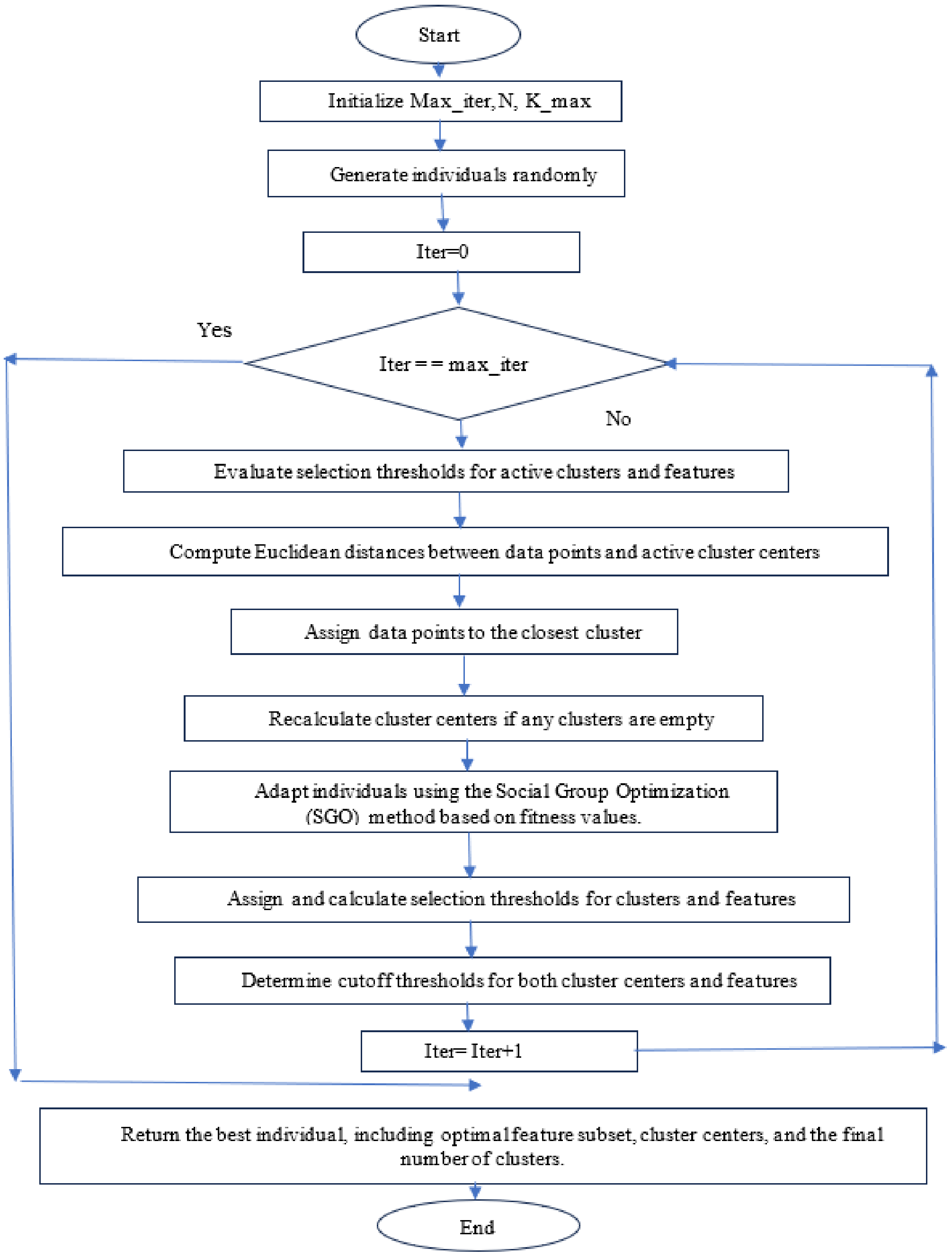
Start
Initialize Max_iter, N, K_max
Generate individuals randomly
Iter=0
Yes
Iter = = max_iter
No
Evaluate selection thresholds for active clusters and features
Compute Euclidean distances between data points and active cluster centers
Assign data points to the closest cluster
Recalculate cluster centers if any clusters are empty
Adapt individuals using the Social Group Optimization (SGO) method based on fitness values.
Assign and calculate selection thresholds for clusters and features
Determine cutoff thresholds for both cluster centers and features
Iter= Iter+1
Return the best individual, including optimal feature subset, cluster centers, and the final number of clusters.
End

Figure 1: Flowchart of the SGO_FSC algorithm

4.1.1 Person representation and initialization

In SGO_FSC, features as well as "cluster centers with varying numbers of clusters are encoded using a variable composite person representation approach. According to the suggested approach, a person a made up of a vector of a real numbers with dimensions of $(K_{max} + D)$+($K_{max} \times D)$ for a maximum of K_{max} clusters and N data points, each with D dimensions. In this case, K_{max}, the upper bound on the number of clusters, is defined as [32]:

$$K_{max} = 3\sqrt{N \times D} \quad (12)$$

Positive real numbers in the interval [0,1] make up the" part (K_{max} + D) of an individual's entries in this case. The K_{max} entry values govern which cluster during the clustering process will activate the relevant cluster and whether or not. Whether or not to activate the associated features depends on the values of the second D element. The K_{max} cluster centers of size D are indicated by Part II ($K_{max} \times D$) A person i at a specific time t is represented by their vector $V_i(t)$ as shown in Fig 2.

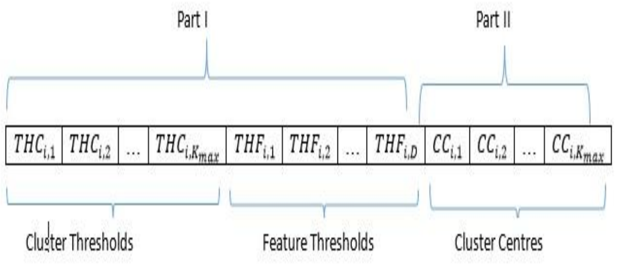
Part I
Part II
$THC_{i,1}$ $THC_{i,2}$ ... $THC_{i,K_{max}}$ $THF_{i,1}$ $THF_{i,2}$ ... $THF_{i,D}$ $CC_{i,1}$ $CC_{i,2}$ ... $CC_{i,K_{max}}$
Cluster Thresholds
Feature Thresholds
Cluster Centres

Figure 2: Illustration of vector $V_i(t)$

Where $CC_{i,j}$ represents "the jth cluster center of the ith person, and $THC_{i,j}$ represents the associated threshold value of the cluster center $CC_{i,j}$. The symbol $THF_{i,j}$ denotes the threshold value of the jth feature in each cluster of the ith individual. The $THC_{i,j}$ and $THF_{i,j}$ are the selection thresholds used to choose active cluster centers and features, respectively. To provide an illustration, please consider the following example."

Example 1

Assume that K_{max} = 4 and D = 3, meaning that there can be a maximum of four computed clusters and that the space has three dimensions. The representation can be seen in Fig. 3 below:

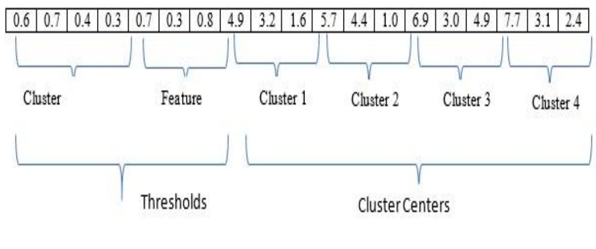

Figure 3: Person representation

For four clusters, the cluster thresholds has been represented by "the first four entries (0.6, 0.7, 0.4, and 0.8).The feature selection thresholds are listed as follows: 0.7, 0.3, and 0.8. There are three characteristics in each cluster. (4.9, 3.2, 1.6), (5.7, 4.4, 1.0), (6.9, 3.0, 4.9), and (7.7, 3.1, 2.4) are the remaining entries that correspond to the four cluster centers.

4.1.2 Threshold setting computation

This work proposed a new approach based on intra-cluster variation for computing the threshold for cluster center [33]. Therefore, the suggested criterion for selecting the cluster center is defined as:

$$THC_m = \sqrt{\frac{1}{n_m}\sum_{i=1}^{n_m}(x_i^m - C_m)}, \quad \text{where } m=2,\ldots\ldots, K_{max} \quad (13)$$

In this case, the selection threshold for cluster m is shown by THC_m. The ith data point in the cluster $C_m$is represented by the symbol x_i^m**.** The number of data points that apply to cluster $C_m$is indicated by the n_m.

The following suggested formula "is used to calculate the relevance/importance of the same feature, averaged over all clusters, corresponding to each cluster (the number of clusters being determined automatically):

$$THF_q = \frac{1}{K}\sum_{i=1}^{K}\left(\frac{v_{r_q}-v_{r_{q,i}}}{v_{r_q}}\right), \text{ where q=1,2,,,,,,D} \quad (14)$$

In this case, THF_q and v_{r_q} stand for the dataset's qth feature's threshold values and variance, respectively. The variance of the qth feature in the ith cluster is denoted by $v_{r_{q,i}}$. K is the number of clusters that were chosen to divide the dataset. The average significance value of the qth feature in the clustering structure is shown by THF_q. This value (THF_q) is close to 1 on a one-point scale, indicating that the clusters in the current solution are dispersed widely for the feature, which makes it valuable for identifying clustering structure. This" equation is essential to "our algorithm.

4.1.3 Threshold cutoff computation

The mean values, which are calculated across all dimensions for feature thresholds and all clusters for cluster thresholds, are used to define the cutoff feature threshold and cluster threshold values" respectively. The mathematical formulation of the cluster selection cutoff threshold ($TH_{c-cutoff}$) is as below:

$$TH_{c-cutoff} = \frac{1}{K}\sum_{m=1}^{K} THC_m \quad (15)$$

In the similar fashion. $TH_{f-cutoff}$ (cutoff threshold for feature selection) is transformed as:

$$TH_{f-cutoff} = \frac{1}{D}\sum_{q=1}^{D} THF_q \quad (16)$$

The cutoff thresholds i.e., $TH_{c-cutoff}$ as well as $TH_{f-cutoff}$ feature are adjusted to the value 0.5 at the start of the algorithm.

Example 2

In the agent under consideration in Example 1, "the cluster center and feature selection thresholds are (0.6, 0.7, 0.4, 0.3) and (0.7, 0.3, 0.8), respectively. The" values of $TH_{c-cutoff}$ is equal to 0.5 and $TH_{f-cutoff}$ is equal to 0.6, according to the previously described proposed notions.

4.1.4 Active cluster center and feature extraction

On the basis of $TH_{c-cutoff}$ as well as $TH_{f-cutoff}$which are corresponding cutoff threshold values, the selection of features as well as cluster centers is done. Cluster center CC_{ij} is chosen for related dataset splitting if and only if the threshold value of THC_{ij} is greater than$TH_{c-cutoff}$. To extract the center of an active cluster, follow these steps:

If $THC_{ij} > TH_{c-cutoff}$ "then the jth cluster center in ith person (CC_{ij} is active Else jth cluster center is inactive)

Here, THC_{ij} denotes the jth cluster center in ith person. $TH_{c-cutoff}$ is cutoff threshold for cluster center.

Similarly, the rule for active feature extraction is given below:

If" $TFC_{ij} > TH_{f-cutoff}$ "then the jth feature in each cluster centers for ith person is active Else jth feature is inactive

Here, TFC_{ij}denotes the jth feature in the ith person. $TH_{f-cutoff}$ is cutoff threshold for feature. The computation of $TH_{c-cutoff}$ and $TH_{f-cutoff}$is mentioned in Sect. 4.1.3.

While SGO is being updated, it is possible that none of the thresholds will be higher than the cutoff threshold value" ($TH_{c-cutoff}$ and $TH_{f-cutoff}$. Two thresholds, chosen at random, should be reset to values higher than the cutoff level in this case.

Example 3

Similar to Example 1, this example uses the agent's cluster center and feature selection criteria, which are (0.6, 0.7, 0.4, 0.3) and (0.7, 0.3, 0.8) resp. In Example 2, 0.5 and 0.6 are the cutoff thresholds utilized for cluster centers ($TH_{c-cutoff}$) and features ($TH_{f-cutoff}$)), respectively.

The number of clusters in the dataset is first ascertained using the cluster center thresholds. Only two cluster center thresholds (0.6 and 0.7) are allowed to be higher than 0.5, according the guidelines. The cluster centers that are now active are (4.9, 3.2, 1.6) and (5.7, 4.4, 1.0). The centers of these active clusters are shown below (in circles).

The significant features from the respective active clusters are then determined using the feature selection thresholds. In this case, the cutoff threshold value (0.6) is less than the two feature selection thresholds (0.7 and 0.8). As a result, each active cluster center chooses the first and third attributes (highlighted in bold).

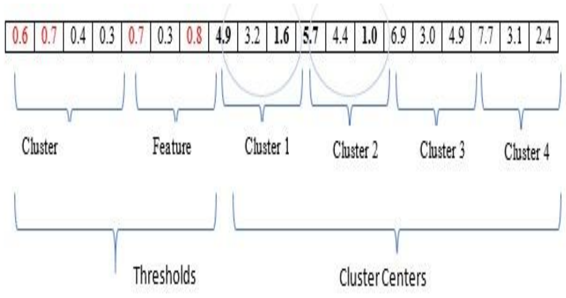

Figure 4: Person representation

4.1.5 Cluster center validation

An empty cluster could arise from a situation where no data points are included in a specific cluster. "When a cluster's center is outside the distribution points' boundaries, this circumstance arises. The cluster center for that specific cluster can be reinitialized in order to resolve this problem. Next, assign (n = K) data points to every cluster center, making sure that every data point is assigned to the cluster center that is closest to it."

Example 4

Three active clusters are present in a dataset consisting of 150 data points in a three-dimensional feature space. For a given instance, the cluster centers are (1.9, 0.52, -0.02), (5.0, 4.1, 0.9), and (7.1, 2.2, 1.8).

In this situation, the cluster center (1.9, 0.52, -0.02) does not have any data points attributed to it since it lies outside the border of the distribution of data points. By employing the average computation formula, 30 data points (n = K) are allocated to the closest cluster centers.

Consequently, the recently created cluster centers are recalculated as follows:

- (4.128, 3.269, 1.601)
- (3.900, 2.832, 1.266)
- (5.789, 3.456, 0.976)

4.1.6 Fitness function computation

The clustering criteria have a crucial role in the clustering algorithm's performance. Poor outcomes could result from selecting clustering criteria arbitrarily. We have selected the I-index following a comprehensive review of the criteria. This is how it is calculated mathematically:

$$I(K) = \left(\frac{1}{K} \times \frac{E_1}{E_K} \times DS_K\right)^P \tag{17}$$

Here, $E_K = \sum_{i=1}^{K} \sum_{j=1}^{n_K} Dist_e(x_j^K, CC_i)$ and $DS_K = max_{i,j=1}^{K} Dist_e(CC_i, CC_j)$. The value K is the number of clusters chosen to divide the dataset. As can be seen from the literature, we have chosen to use 2 as the value of P in this paper. It is crucial to note that higher the I-index value, the higher the caliber of clustering solutions.

The following concerns regarding the quantity of clusters and features have been looked at and included to the I-index.

Issue 1: Determining the ideal number of clusters for the run time is the first problem. In order to account for the impact of K_max, we have introduced the penalty function, which has the following mathematical definition:

$$Cluster\_index = \frac{K_{max}-K}{K-1} \quad (18)$$

The few clusters are favored by this penalty function.

Issue 2: Choosing the ideal feature count is the second problem. The right number of characteristics is not taken into consideration by the majority of clustering criteria. The process of choosing d features (out of a total of D features) is known as feature selection. Important aspects that support maintaining a suitable degree of performance are what we wish to keep.

We have created the penalty function to account for the influence of the number of features. It may be recast mathematically as follows:

$$Feature\_index = \frac{D-d}{D-1} \quad (19)$$

After combining the aforementioned details, the unified clustering criterion is as follows:

$$Fitness = I(K) \times Cluster\_index \times feature\_index \quad (20)$$

4.2 Complexity of algorithm

4.2.1 Time complexity

1. "Initialization of SGO_FSC needs O(personsize×stringlength) time where person size and string length designate the number of persons and the length of each encoded person in the SGO_FSC, respectively. The string length is $O((K_{max}+D)+(K_{max}\times D))$ where D is the dimension of the dataset and K_{max} is the maximum number of clusters specified by user.

2. Active cluster and feature extraction step of SGO_FSC requires O(personsize× $K_{max}\times D$) time.

3. Fitness function computation consists of three basic steps.
   (a) The assignment of data points to different clusters requires $O(n^2 \times K_{max})$ for each person.
   (b) The cluster center updating requires $O(K_{max})$.
   (c) Time complexity of fitness function is $O(n \times K_{max})$.

The third step is repeated for each person, or personsize times, and the three previously described substeps are computed" sequentially. As a result, personsize × $(n^2 \times K_{max} + K_{max} + n \times K_{max}) = O(\text{personsize} \times n^2 \times K_{max})$ will be the overall complexity of Step 3, or the fitness evaluation.

4. position update step requires 2 × O(personsize×stringlength)

The time complexity is therefore $O(n^2 \times K_{max} \times \text{personsize})$ per generation when the intricacies of all the previously listed processes are added together and the fact that string length ≥ n is taken into consideration. For a maximum number of generations, the total time complexity of SGO_FSC is $O(n^2 \times K_{max} \times \text{personsize} \times Maxgen)$. Here, Maxgen denotes the highest number of generations possible.

4.2.2 Space complexity

The number of people (personsize) has a major effect on the primary space requirement of the SGO_FSC clustering approach. Consequently, O (personsize×stringlength) is the space complexity of the SGO_FSC clustering approach.

5 Experimentation and results

5.1 Real-life dataset analysis

This section validates the SGO_FSC method on eight real-world datasets by comparing its performance with eight other competitive algorithms. Furthermore, twenty datasets are used for an independent evaluation of the SGO_FSC algorithm.

5.1.1 Datasets used

We conducted experiments using nine benchmark datasets from the UCI data repository to assess the performance of the proposed SGO_FSC algorithm in comparison to eight other methods. Table 2 details these selected datasets, including the number of features, instances, and classes in each dataset. Table 3 [34] presents the datasets on which only the SGO_FSC algorithm was evaluated. Both small and high-dimensional datasets were included in our experiments to ensure a comprehensive assessment.

Table 2: Description of datasets used

"Dataset	Data points	Features	Classes
Iris	150	4	3
Wine	178	13	3
Glass	214	9	6
Haberman	306	3	2
Bupa	345	6	2
WDBC	576	30	2
Cancer	683	9	2
Vowel	871	3	6
CMC	1473	9	3"

Table 3: List of datasets used in the experiments

Name	No. of features	No. of instances	No. of classes
Breastcancer	9	699	2
BreastEW	30	569	2
CongressEW	16	435	2
Exactly1	13	1000	2
Exactly2	13	1000	2
HeartEW	13	270	2
IonosphereEW	34	351	2
KrvskpEW	36	3196	2
Lymphography	18	148	2
M of n	13	1000	2
PenglungEW	325	73	2
Semeion	256	1593	2
Sonar	60	208	2
Spect	22	267	2
Tic-tac-toe	9	958	2
Votes	16	300	2
WaveformEW	40	5000	3
Zoo	16	101	6
Vechile	18	846	4
Dermatology	34	366	6

5.1.2 Experimental setup

The SGO_FSC algorithm is executed with varying values for control parameters, specifically the number of persons (P) and the maximum number of iterations (Maxgen). For this evaluation, P = 30 and Maxgen = 100 are set, and these parameters are fixed throughout the experiment. The algorithms are executed 40 times in a sequential fashion to evaluate the performance. MATLAB 2016a running on a Windows 10 operating system is utilized to implement SGO_FSC algorithm. These simulations are conducted on a laptop equipped with an Intel Core i5 processor and 8 GB of memory.

5.1.3 Comparison with other existing techniques

In order to test how well SGO FSC works, nine real-life data sets that were stated in Section 5.1.1 were utilized. We evaluate SGO_FSC against eight popular, the most advanced automatic clustering and feature selection algorithms. Classification accuracy, cluster count, and feature count are some of the most common cluster quality metrics used to assess and compare the results. The effectiveness of SGO_FSC is compared to eight popular clustering methods. These include K-Means, the MHSC (Modified harmony search algorithm-based clustering) [35], the simultaneous CFS_PSO (Clustering and "feature selection using particle swarm optimization) [23], the FS_ACO (Feature selection and clustering using ant colony optimization) [20], the NMA_CFS [14], the automated HS_CFS (Harmony search-based clustering technique) [17], INMA_CFS (The improved memetic algorithm-based clustering technique) [28], and GSA_CFS [36]. The parameter settings for the aforementioned methods are shown below, in accordance with the paper from which they were extracted. The selection and replacement group sizes for NMA_CFS and INMA_CFS are determined in accordance with Kumar et al. [28].

MHSC:
- Distance bandwidth: [0.001, 0.1]
- Harmony memory consideration rate: [0.5, 0.95]
- Pitch adjustment rate: 0.5

FS_ACO:
- Pheromone priori: 0.1
- Pheromone evaporation rate: 0.01
- Number of local searches: 6
- Local search threshold: 0.01

CFS_PSO:
- Velocity range: [-6, 6]
- Constant parameters (c1, c2): 1.49, 1.49
- Inertia weight: 0.71

NMA_CFS:
- Mutation (probability): Flip and Gaussian (0.01)

HS_CFS:
- Percentage of dimension: 0.30
- Harmony memory consideration rate: 0.95
- Distance bandwidth: 0.0005
- Pitch adjustment rate: 0.35

INMA_CFS:
- Mutation (probability): Flip and Gaussian (0.01)

GSA_CFS:
- Gravitational constant (G0): 100"

SGO_FSC

- Self-introspection parameter(C) : 0.2

Except for the SGO_FSC algorithms, all other algorithms have population size of 30 and maximum iterations 500. There are 100 iterations and 30 population size for SGO_FSC algorithm. Every dataset goes through the aforementioned algorithms 40 times in a sequential fashion. We use the metrics of "mean" and "standard deviation" to assess the quality of each cluster. The robustness is measured by looking at the standard deviation, which is shown in parentheses.

Table 4: “Mean and standard deviation of the cluster quality measures obtained from algorithms for different datasets

	K-Means	MHSC	FS_ACO	CFS_PSO	NMA_CFS	HS_CFS	INMA_CFS	GSA_CFS	SGO_FSC
					Iris				
No.of clusters	3	3	3.0 (0.0)	3.0 (0.0)	3.0 (0.0)	3.0(0.0)	3.0 (0.0)	3.0 (0.0)	3.0 (0.0)
No.of features	4	4	2.7 (0.9)	2.2 (0.3)	1.9 (0.2)	2.0(0.3)	1.7 (0.7)	2.0 (0.0)	2.0 (0.0)
CA(%)	84.4(6.3)	86.7 (5.7)	94.8 (2.4)	95.4 (1.9)	95.9 (1.5)	96.0(1.6)	95.0 (2.0)	96.5 (0.4)	97.1(0.2)
					Wine				
No.of clusters	3	3	3.8 (2.3)	3.2 (2.0)	3.5 (1.1)	3.4(2.5)	2.9 (0.3)	3.05 (0.6)	3.00(0.00)
No.of features	13	13	6.2 (2.6)	5.7 (2.9)	5.8 (1.5)	6.0(3.1)	6.6 (1.7)	3.8 (1.1)	3.8(1.2)
CA(%)	65.9(5.9)	64.1(5.4)	64.7(9.1)	65.0(8.7)	65.6 (8.9)	65.2(9.8)	66.2 (9.5)	66.4 (4.1)	68.5(3.1)
					Glass				
No.of clusters	6	6	6.2 (1.9)	6.0 (1.2)	5.9 (1.5)	5.7(2.0)	5.6 (0.9)	5.9 (0.8)	5.8(0.3)
No.of features	9	9	4.3 (1.6)	4.0 (0.6)	3.5 (1.1)	3.0(1.9)	3.6 (1.2)	3.8 (0.8)	3.6(0.4)
CA(%)	50.8(3.6)	49.4(3.6)	44.3 (4.7)	44.7(5.1)	43.1 (5.4)	46.4(4.9)	42.8 (6.0)	52.2 (3.2)	56.14(0.4)
					Haberman				
No.of clusters	2	2	2.9 (1.7)	2.2 (0.6)	2.5 (0.5)	2.7 (1.2)	2.0 (0.0)	2.3 (0.4)	2.00(0.00)
No.of features	3	3	2.0 (0.0)	1.9 (0.6)	1.3 (0.5)	1.7 (0.9)	1.0 (0.0)	1.0 (0.0)	1.0 (0.0)
CA(%)	50.9(1.1)	55.4(4.0)	56.7(6.4)	56.3(5.8)	55.8 (8.1)	56.9(7.5)	54.3 (4.9)	62.5 (9.7)	68.60(6.23)
					Bupa				
No.of clusters	2	2	2.9 (1.2)	2.6 (1.1)	3.0 (0.7)	2.6 (1.9)	2.2 (0.5)	2.2 (0.3)	2. (0)
No.of features	6	6	2.5 (0.4)	3.0 (0.5)	2.4 (0.8)	1.9(1.2)	2.2 (0.8)	2.2 (0.4)	2.(0)
CA(%)	53.1(0.0)	53.8(1.9)	52.4 (2.4)	51.7(3.1)	51.5 (3.9)	52.1(2.5)	52.1 (4.7)	54.7 (2.2)	58.7 (1.2)
					WDBC				
No.of clusters	2	2	2.0 (0.0)	1.9 (0.9)	2.0 (0.0)	2.0 (0.0)	2.0 (0.0)	2.0 (0.0)	2.0 (0.0)
No.of features	30	30	15.0 (0.4)	13.9 (1.0)	14.8 (0.9)	14.5 (0.5)	13.4 (2.6)	10.0 (0.0)	10.0 (0.0)
CA(%)	85.4 (0.4)	85.7 (1.9)	90.5 (0.9)	89.6 (1.3)	90.8 (0.4)	90.0 (0.8)	90.4 (0.8)	91.0 (0.0)	92.0 (1.0)
					Cancer				
No.of clusters	2	2	2.4 (1.6)	2.4 (1.3)	2.2 (0.4)	2.0 (1.1)	2.1 (0.3)	2.0 (0.0)	2.0 (0.0)
No.of features	9	9	4.3 (1.5)	3.9 (2.0)	4.1 (1.4)	4.0 (1.8)	3.4 (1.6)	3.8 (0.9)	3.8 (0.9)
CA(%)	94.0 (0.0)	94.4 (0.9)	94.3 (2.1)	93.9 (2.6)	94.6 (2.9)	94.8 (1.2)	94.3 (1.0)	95.2 (0.3)	96.5 (1.3)
					Vowel				
No.of clusters	6	6	6.2 (0.9)	6.3 (1.6)	6.0 (1.4)	6.0 (2.0)	6.3 (0.8)	6.0 (0.6)	6.0 (0.6)
No.of features	3	3	2.0 (0.0)	2.9 (0.7)	1.1 (0.3)	1.8 (0.6)	1.1 (0.3)	1.1 (0.3)	2.0 (0.4)
CA(%)	53.0 (5.0)	53.6 (5.8)	53.9 (4.5)	52.8 (3.9)	53.4 (3.6)	54.1 (4.1)	53.8 (4.1)	55.5 (2.2)	58.5 (1.8)
					CMC				
No.of clusters	3	3	3.5 (1.1)	3.2 (1.9)	2.9 (0.8)	3.2 (1.4)	3.6 (0.8)	3.0 (0.5)	3.0 (0.5)
No.of features	9	9	3.9 (1.2)	3.8 (1.5)	3.6 (1.4)	3.0 (1.0)	2.4 (1.2)	2.2 (0.4)	2.3 (0.2)
CA(%)	39.9 (0.2)	40.1 (1.4)	41.8 (3.8)	41.5 (4.3)	40.4 (3.1)	41.9 (4.2)	40.2 (2.5)	43.0 (3.1)	45.0 (4.1)

Table 5: Mean and standard deviation of the cluster quality measures obtained by SGO_FSC for different datasets independently

Sl. No.	Name	No. of features	No. of classes	CA

1	Breastcancer	4.20(0.09)	2(0.00)	96.39(0.02)
2	BreastEW	15.25(2.00)	2(0.00)	95.95(1.90)
3	CongressEW	2.20(1.01)	2(0.00)	95.89(1.67)
4	Exactly1	3.40(1.02)	2(0.00)	71.68(5.90)
5	Exactly2	1(0.008)	2(0.00)	77.60(7.89)
6	HeartEW	8.70(1.04)	2(0.00)	83.52(2.39)
7	IonosphereEW	3.65(1.1)	2(0.00)	87.95(4.54)
8	KrvskpEW	23.50(2.64)	2(0.00)	94.53(2.83)
9	Lymphography	7.75(2.01)	2(0.00)	85.41(4.45)
10	M of n	10.20(1.00)	2(0.19)	87.73(4.60)
11	PenglungEW	75.40(2.90)	2(0.00)	84.97(6.28)
12	Semeion	136.4(2.23)	2(0.00)	97.44(1.67)
13	Sonar	21(1.19)	2(0.00)	83.75(5.34)
14	Spect	10.10(2.22)	2(0.00)	86.57(2.89)
15	Tic-tac-toe	8.80(2.02)	2(0.00)	82.20(0.00)
16	Votes	3(1.00)	2(0.00)	94.80(3.89)
17	WaveformEW	25(0.22)	3(0.34)	77.33(8.7)
18	Zoo	3.60(0.02)	6(0.21)	97.47(3.61)
19	Vechile	4.75(2.89)	4(0.11)	68.40(4.71)
20	Dermatology	19.50(1.00)	6(0.01)	97.32(1.90)

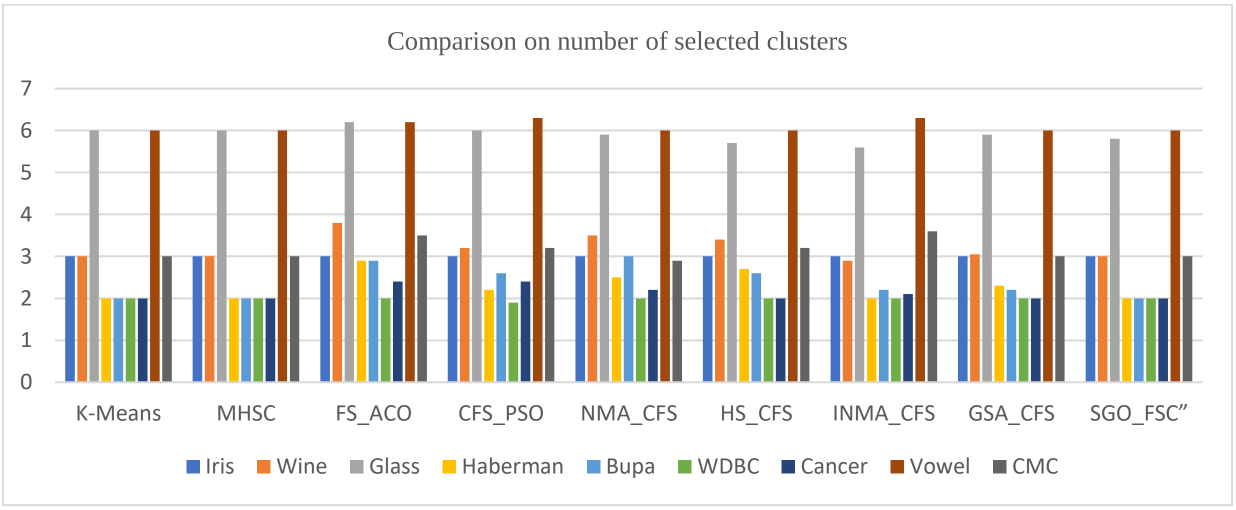

Fig 5. Comparison on number of clusters derived by different algorithms in different datasets

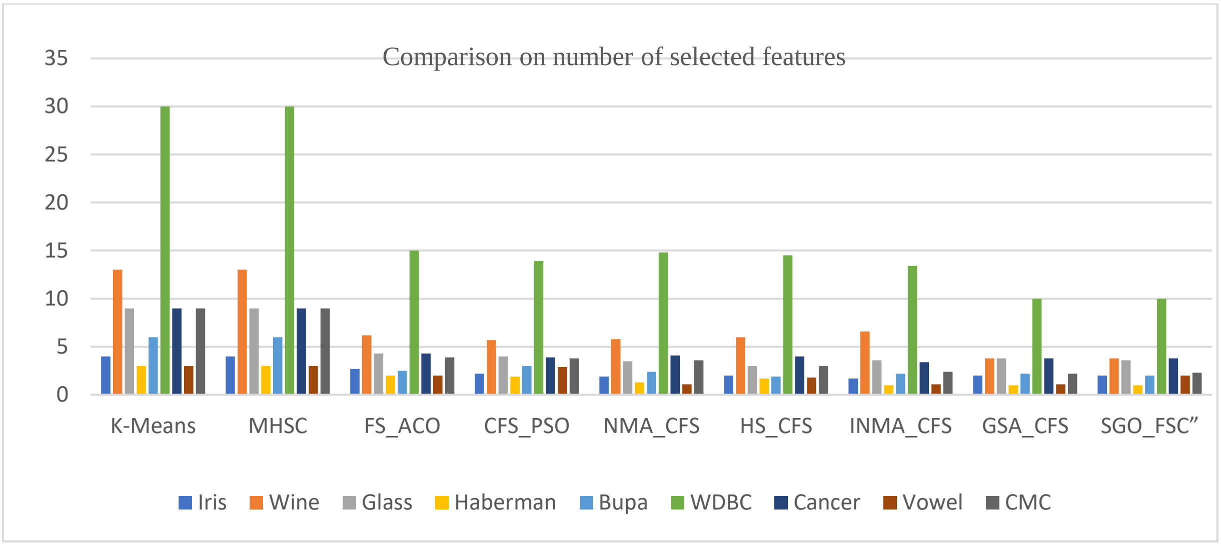

Figure 6: Comparison on number of selected features derived by different algorithms in different datasets

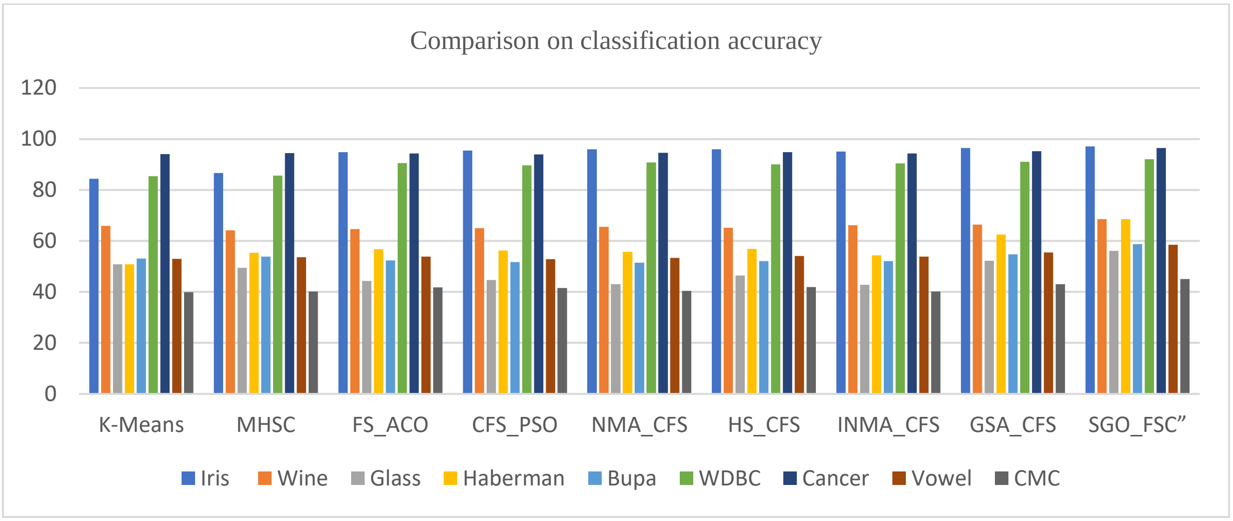

Figure 7: Comparison on classification accuracy achieved by different algorithms in different datasets

5.1.4 Performance evaluation

Cluster quality measures derived from several clustering techniques applied to real-world datasets are summarized in Tables 4 and 5, together with their respective standard deviations. Table 3 provides results for the eight competitive algorithms, while Table 5 presents results for the SGO_FSC algorithm across 20 real-life datasets. Figure 5 shows a comparison of the number of clusters derived by various algorithms, Figure 6 compares the number of features selected by different algorithms, and Figure 7 presents a comparison of classification accuracy achieved by these algorithms.

Iris dataset:

- • SGO_FSC outperformed INMA_CFS, GSA_CFS, MHSC, HS_CFS, “NMA_CFS, FS_ACO, CFS_PSO, and K-Means in terms of classification accuracy.
- The right number of clusters as well as features were found by SGO_FSC, HS_CFS, GSA_CFS, FS_ACO, INMA_CFS, CFS_PSO, and NMA_CFS.”

Wine dataset:

- SGO_FSC achieved the best classification accuracy. Both SGO_FSC and GSA_CFS identified 3 clusters.
- “NMA_CFS, HS_CFS, CFS_PSO, and FS_ACO selected six features, while SGO_FSC and GSA_CFS identified 4 features.

Glass dataset:

- All algorithms identified 6 clusters.
- SGO_FSC, CFS_PSO, GSA_CFS, NMA_CFS, INMA_CFS, and FS_ACO”, selected four features, while HS_CFS selected 3 features.
- In comparison to other algorithms, SGO_FSC showed much better classification accuracy.

Haberman dataset:

- SGO_FSC achieved a classification accuracy of 65.6%, outperforming GSA_CFS (62.5%), HS_CFS (56.9%), and NMA_CFS (55.8%).
- FS_ACO attained an accuracy of 56.7%, whereas CFS_PSO reached 56.3%.
- Three clusters were obtained for FS_ACO and HS_CFS, while CFS_PSO, HS_CFS and FS_ACO each picked two characteristics. Results for clusters and characteristics were similar for INMA_CFS, NMA_CFS, and SGO_FSC.

Bupa dataset:

- Both SGO_FSC and GSA_CFS identified 2 clusters and 2 features.
- FS_ACO, HS_CFS, CFS_PSO, and NMA_CFS all supplied erroneous cluster numbers. HS_CFS selected 2 features, while FS_ACO as well as CFS_PSO selected three features.
- SGO_FSC achieved better classification accuracy than the other algorithms.

WDBC dataset:

- The best classification accuracy was attained by SGO_FSC, which was followed by GSA_CFS, NMA_CFS, FS_ACO, INMA_CFS, HS_CFS, CFS_PSO, MHSC, and K-Means.
- FS_ACO, CFS_PSO, HS_CFS, INMA_CFS, NMA_CFS, and GSA_CFS all accurately detected two clusters.
- Both SGO_FSC and GSA_CFS selected 10 important features, whereas HS_CFS, NMA_CFS, and FS_ACO selected fifteen features.

Cancer dataset:

- All algorithms determined the number of features and clusters with accuracy.
- SGO_FSC achieved superior classification accuracy compared to the other algorithms.

Vowel and CMC datasets:

- The highest classification accuracy was shown by SGO_FSC, which was followed by K-Means, MHSC, FS_ACO, HS_CFS, NMA_CFS, INMA_CFS, and FS_ACO.
- All algorithms accurately determined the number of clusters for both datasets.
- SGO_FSC offered two features for the Vowel dataset, but NMA_CFS, INMA_CFS GSA_CFS revealed only one feature. SGO_FSC, INMA_CFS, NMA_CFS, and GSA_CFS accurately determined the number of features in the CMC dataset.

This comprehensive comparison highlights the effectiveness of SGO_FSC in both feature selection as well as clustering selection across a diverse range of real-life datasets.

5.1.5 Statistical significance

To evaluate the efficacy of the SGO_FSC algorithm, we performed an unpaired t-test for the purpose of comparing its classification accuracy against the best-performing as well as the second-best-performing algorithms from the experiments. The size of the sample for the t-test was 40, with a significance level set at 5%. For the purpose of comparing the classification accuracy of SGO_FSC with the other algorithms that are leading, the t-test was performed. The findings of the unpaired t-tests conducted using the categorization accuracy information from Table 3 are displayed in Table 5.

SGO_FSC vs. Best Algorithm: The t-test [37] results indicate that in terms of classification accuracy, SGO_FSC performs noticeably better than the best algorithm.
SGO_FSC vs. Second-Best Algorithm: Similarly, SGO_FSC shows a statistically significant improvement over the second-best algorithm.

These statistical results confirm that SGO_FSC provides a substantial improvement in classification accuracy compared to the other evaluated algorithms. This robustness is supported by the rigorous statistical testing, validating the superior performance of the SGO_FSC algorithm.

Table 5: Unpaired t-test between SGO_FSC and second-best algorithm over classification accuracy

Dataset	Standard error of difference	t	95% Confidence interval	Two-tailed P	Significance
Iris	0.071	8.4853	From 0.459 to 0.741	less than 0.0001	extremely statistically significant
Wine	0.813	2.5839	From 0.482 to 3.718	0.0116	statistically significant
Glass	7.7270	0.510	From 2.9249 to 4.9551	less than 0.0001	extremely statistically significant.
Haberman	1.823	3.3465	From 2.4711 to 9.7289	0.0013	very statistically significant
Bupa	0.396	10.0951	From 3.211 to 4.789	less than 0.0001	extremely statistically significant.

WDBC	0.158	6.3246	From 0.685 to 1.315	less than 0.0001	extremely statistically significant.
Cancer	0.211	6.1626	From 0.880 to 1.720	less than 0.0001	extremely statistically significant.
Vowel	0.449	6.6749	From 2.105 to 3.895	less than 0.0001	extremely statistically significant
CMC	0.813	2.4609	From 0.382 to 3.618	0.0161	statistically significant.

5.1.6 Overall discussion on SGO_FSC algorithm

Performance comparison

SGO_FSC demonstrated superior performance in several instances compared to other competitive algorithms like K-Means, Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE), and more complex hybrid methods such as Ant Colony Optimization (ACO) and Artificial Bee Colony (ABC) for clustering tasks. The key differences that influenced these results are outlined below:

- **Adaptation to varying datasets**: SGO_FSC exhibited better adaptability to varying dataset characteristics such as high-dimensionality, class overlap, and noise. The reason for this advantage lies in the SGO_FSC's dynamic feature selection mechanism, which adjusts based on the dataset's structure. While methods like K-Means tend to underperform on high-dimensional data due to their reliance on distance metrics that become less meaningful in higher dimensions, SGO_FSC's automatic feature selection reduces the dimensionality, thereby enhancing clustering accuracy.

- **Improved feature selection**: The newly designed fitness function in SGO_FSC plays a pivotal role in its superior performance. Unlike traditional algorithms where the fitness function focuses primarily on cluster compactness (e.g., the Sum of Squared Errors in K-Means), SGO_FSC incorporates both intra-cluster variation and feature selection criteria. This dual-objective optimization enables the algorithm to avoid irrelevant features that could distort clustering results. In comparison, algorithms like PSO or GA often struggle with feature selection in large feature spaces because their exploration is limited by their update mechanisms, which do not directly incorporate feature relevance metrics.

- **Handling of class overlap and noise**: In datasets characterized by overlapping classes and high noise, SGO_FSC outperformed algorithms such as GA and PSO. This is largely attributed to SGO_FSC's threshold-based approach, which filters out clusters and features with high variance, allowing it to concentrate on clusters with meaningful data points. Algorithms like PSO or DE tend to be more sensitive to noise and can form clusters around noisy data points, leading to degraded performance. SGO_FSC, on the other hand, dynamically adjusts cluster centers and excludes noisy data through its fitness-based cutoff mechanism.

- **Scalability and computational efficiency**: While SGO_FSC achieved notable improvements in clustering quality, it occasionally required more computational resources than simpler methods like K-Means due to its more complex encoding of individuals and its iterative optimization process. However, in comparison to algorithms such as DE or ABC, SGO_FSC maintained better computational efficiency for large-scale datasets, owing to its streamlined fitness evaluation that balances feature selection and clustering in a single pass.

Explanation of performance variability across datasets

The performance of SGO_FSC varied across different datasets, and this variability can be explained by several factors:

- **Dimensionality**: On low-dimensional datasets, simpler algorithms like K-Means often perform well because the distance metric is more reliable. However, as dimensionality increases, the curse of dimensionality affects most algorithms, leading to performance degradation. SGO_FSC mitigates this issue by automatically selecting relevant features, reducing the number of dimensions while preserving the important information for clustering. In contrast, methods like GA and PSO are not as effective in filtering out redundant features, which can lead to suboptimal clustering results.

- **Class overlap**: Datasets with significant class overlap can be challenging for many clustering algorithms. SGO_FSC outperformed the other methods in this area by using an I-index-based fitness function that rewards well-separated clusters. This allows SGO_FSC to identify clearer boundaries between overlapping classes compared to methods like DE, which may struggle to maintain cluster cohesion under these conditions.

- **Noise levels**: In noisy datasets, SGO_FSC showed resilience by filtering out noise during feature selection and cluster center updating. Other algorithms like ACO and ABC, which rely on more rigid updating mechanisms, tend to incorporate noisy data points into their clustering process, thus deteriorating performance. SGO_FSC's dynamic adjustment to thresholds based on data variability provided it with an edge in noise handling.

SGO_FSC's performance advantages stem from its novel combination of feature selection and clustering optimization, driven by a robust fitness function that adapts to the dataset's characteristics. The algorithm's ability to handle high-dimensional data, class overlap, and noise contributed to its outperformance in most benchmark datasets. However, its performance comes with a trade-off in computational complexity, especially when applied to large-scale datasets. Nevertheless, its overall performance justifies the increased complexity, especially in scenarios where clustering accuracy is critical.

5.2 Microarray data analysis

Microarray technology has progressed, enabling the observation of gene expression at different time intervals [2]. Clustering is essential in the analysis of microarray data since it allows for the grouping of genes that have resembling expression. The proposed clustering technique, SGO_FSC, has been utilized to evaluate its usefulness in analyzing a microarray dataset comprehensively.

5.2.1 Gene expression datasets

The SGO_FSC algorithm was evaluated using three prominent gene expression datasets for experimentation. The details of these datasets are provided in Table 6.

Table 6: Description of datasets used

Dataset	Number of Genes	Dimensions	Subset of Genes for experimentation	Normalization
Yeast Sporulation	6118	7	474	Each row has zero mean and unit variance
Fibroblasts Serum	8613	13 (12 time points + 1 unsynchronized)	517	Each row has zero mean and unit variance
Rat CNS	112	9	112	Each row has zero mean and unit variance

5.2.2 Algorithms for comparison

"A comparison was conducted between the performance of the SGO_FSC method and eight widely recognized clustering algorithms:

- IFCM (Iterated Fuzzy C-Means)
- INMA_CFS [28]
- CRC (Chinese Restaurant Clustering) [42]
- VSGA (Variable String Length GA-based Clustering Technique) [39]
- HS_CFS [17]
- Self-Organizing Map (SOM) [41]
- Average Linkage (AL) [40]
- SiMM-TS (Significant Multi-Class Membership-based Clustering Technique) [38]

The parameter settings for these algorithms were configured according to the specifications provided in their original papers. For SGO_FSC, the parameter settings were consistent with those described in Section 5.1.

The Silhouette Index (SI) [43] measures the level of similarity between an object and its own cluster compared to other clusters. The values span from -1 to 1. A value close to 1 indicates a high degree of similarity between the object and its assigned cluster, but a low degree of similarity with nearby clusters. For an object i, the SI(i) is defined as follows:

$$SI(i) = \frac{b(i)-a(i)}{\max\left(b(i),a(i)\right)} \tag{21}$$

where:

- a(i) represents the mean distance between object i and all other objects inside the same cluster.
- The variable b(i) represents the smallest average distance between object i and all objects in any other cluster, excluding the cluster" that i belongs to.

The SI [43] is employed to find out the effectiveness of the clustering techniques outlined above. The clustering methods have been executed separately 10 times.

5.2.3 Results and discussions

The number of clusters and SI index value are calculated and put in the Table 7. For all algorithms except SGO_FSC are taken from their respective papers. The outcomes show that six clusters were created for the Yeast Sporulation dataset by SiMM-TS, VSGA, AL, and SOM. On the other hand, seven clusters were created using

IFCM and HS_CFS. Eight clusters were retrieved from CRC and INMA_CFS. There were five clusters produced by the suggested algorithm. The suggested SGO_FSC generated the greatest SI value (0.7119) for this dataset out of all the clustering methods that were examined. As a result, we can conclude that the suggested strategy outperforms the alternative algorithms in terms of both SI and cluster count.

The Human Fibroblasts Serum dataset yielded 10 clusters according to the CRC and 8 clusters according to the IFCM. The number of clusters as reported by HS_CFS was 7. Six clusters were created via VSGA, INMA_CFS, SiMM-TS, and SOM, AL approaches. The SGO_FSC algorithm produced five clusters for this dataset. The IFCM method yields the lowest SI value of all the algorithms, 0.2995. AL's SI index is 0.3092. The clusters produced by SGO_FSC have a high SI value of 0.4394.

The VSGA, SiMM-TS, AL, and SOM algorithms determined that there were six clusters in the Rat CNS dataset. The number of clusters generated by the INMA_CFS and IFCM algorithms is 5. The number of clusters generated for the CRC is seven, according to HS_CFS. SGO_FSC as proposed produces only 3 clusters. IFCM and AL yielded values of 0.4050 and 0.3684 for SI, respectively. The suggested algorithm SGO_FSC yields a superior SI value of 0.6023 for this dataset as well.

Table 7: Number of clusters and SI Value obtained from algorithms for different datasets

	VSGA	SOM	AL	SiMM-TS	IFCM	HS_CFS	CRC	INMA_CFS	SGO_FSC
Yeast Sporulation									
Number of Clusters	6	6	6	6	7	7	8	8	5
Silhouette Index (SI)	0.58	0.5842	0.5007	0.6353	0.4717	0.5442	0.5675	0.5294	0.7119
Fibroblasts Serum									
Number of Clusters	10	8	7	6	6	6	6	6	5
Silhouette Index (SI)	0.298	0.298	0.3124	0.3184	0.3241	0.3112	0.3092	0.32	0.4394
Rat CNS									
Number of Clusters	6	6	6	6	5	5	4	7	3
Silhouette Index (SI)	0.5147	0.4134	0.3684	0.5147	0.4032	0.382	0.4455	0.4278	0.6023

The average Silhouette index values and the number of clusters produced by applying the aforementioned clustering techniques are displayed in Figures 4 and 5. The SGO_FSC method outperforms the other competitive algorithms in terms of SI value and number of clusters, as demonstrated by the findings shown in Fig. 8 and 9.

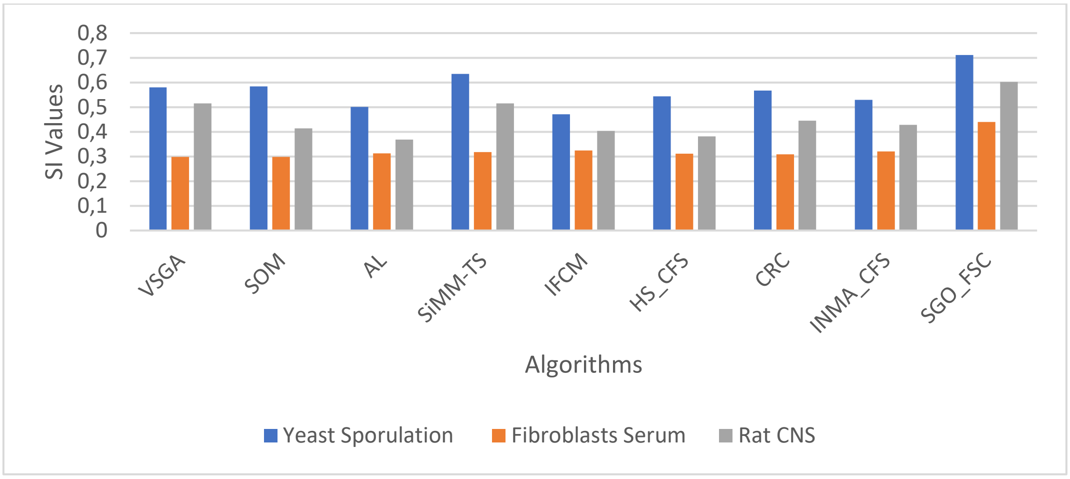

Figure 8: Comparison of proposed approach over existing clustering techniques in terms of SI value

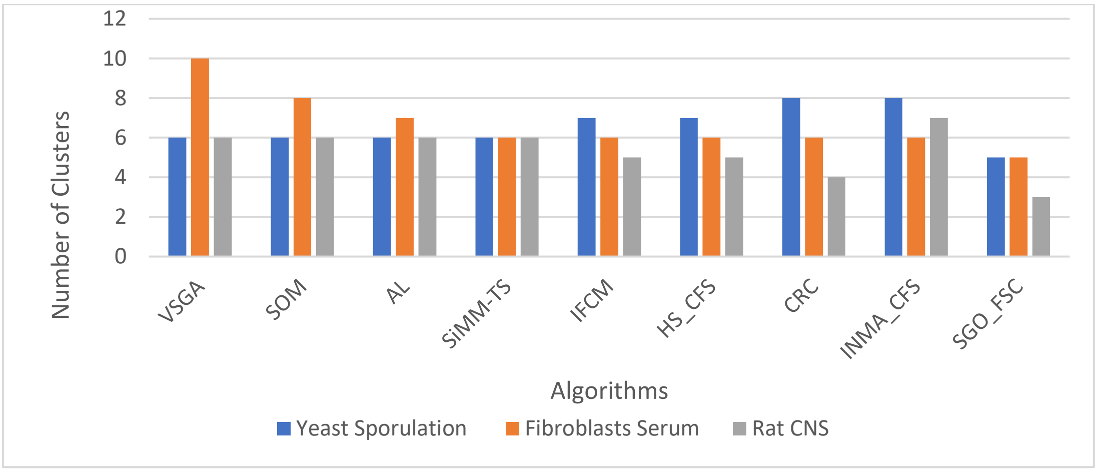

Figure 9: Comparison of proposed approach over existing clustering techniques in terms of a number of clusters

6 Conclusion

In this paper, an automatic clustering and feature selection technique using the Social Group Optimization algorithm has been proposed. The results revealed that through the proposed technique, we were able to get the optimal number of clusters and features simultaneously and accurately. The proposed approach used a novel concept of dynamic threshold setting for determining the number of clusters and features. Efficient searching of optimal clusters and features was made possible by proposing a novel fitness function. The proposed technique was investigated on nine real-life datasets with varying characteristics. Statistical t-tests have been carried out to establish the statistical significance of results produced by SGO_FSC. The proposed algorithm was further applied to Microarray data datasets for analysis. From experimental results, it has also been observed that SGO_FSC outperforms the other competitive clustering techniques.

Compliance with ethical standards

The authors declare the absence of conflict of interest.

References

[1] Das, S., & Abraham, A., & Konar, A. (2008). Automatic clustering using an improved differential evolution algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, 38*(1), 218–237. https://doi.org/10.1109/tsmca.2007.909595

[2] Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall. https://doi.org/10.2307/1268876

[3] Gams, M., & Kolenik, T. (2021). Relations between electronics, artificial intelligence and information society through information society rules. *Electronics, 10*(4), 514. https://doi.org/10.3390/electronics10040514.

[4] Sheng, W., Liu, X., & Fairhurst, M. (2008). A niching memetic algorithm for simultaneous clustering and feature selection. *IEEE Transactions on Knowledge and Data Engineering, 20*(7), 868–879. https://doi.org/10.1109/tkde.2008.33

[5] Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys, 31*(3), 264–323. https://doi.org/10.1145/331499.331504

[6] Das, S., & Konar, A. (2009). Automatic image pixel clustering with an improved differential evolution. *Applied Soft Computing, 9*(1), 226–236. https://doi.org/10.1016/j.asoc.2007.12.008

[7] Satapathy, S., & Naik, A. (2016). Social group optimization (SGO): A new population evolutionary optimization technique. *Complex & Intelligent Systems, 2*, 173–203. https://doi.org/10.1007/s40747-016-0022-8

[8] Vaithyanathan, S., & Dom, B. (1999). Generalized model selection for unsupervised learning in high dimensions. In *Advances in Neural Information Processing Systems, 12* (pp. 970–976). Cambridge.

[9] Frigui, H., & Nasraoui, O. (2000). Simultaneous clustering and attribute discrimination. In *Proceedings of IEEE International Conference on Fuzzy Systems* (pp. 158–163). San Antonio, TX. https://doi.org/10.1109/fuzzy.2000.838651

[10] Kim, Y., Street, W., & Menczer, F. (2002). Feature selection in unsupervised learning via evolutionary search. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 365–369). https://doi.org/10.1145/347090.347169

[11] Dy, J. G., & Brodley, C. E. (2004). Feature selection for unsupervised learning. *Journal of Machine Learning Research, 5*, 845–889. https://doi.org/10.1007/978-1-4419-1428-6_97

[12] Roth, R. V., & Lange, T. (2004). Feature selection in clustering problems. In *Proceedings of Advances in Neural Information Processing Systems*. Cambridge.

[13] Law, M. H. C., Figueiredo, M. A. T., & Jain, A. K. (2004). Simultaneous feature selection and clustering using mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 26*(9), 1154–1165. https://doi.org/10.1109/tpami.2004.71

[14] Sheng, W., Liu, X., & Fairhurst, M. (2008). A niching memetic algorithm for simultaneous clustering and feature selection. *IEEE Transactions on Knowledge and Data Engineering, 20*(7), 868–879. https://doi.org/10.1109/tkde.2008.33

[15] Maugis, C., Celeux, G., & Martin-Magniette, M. L. (2009). Variable selection for clustering with Gaussian mixture models. *Biometrics, 65*(3), 701–709. https://doi.org/10.1111/j.1541-0420.2008.01160.x

[16] Zeng, H., & Cheung, Y.-M. (2009). A new feature selection method for Gaussian mixture clustering. *Pattern Recognition, 42*(2), 243–250. https://doi.org/10.1016/j.patcog.2008.05.030

[17] Sarvari, H., Khairdoost, N., & Fetanat, A. (2010). Harmony search algorithm for simultaneous clustering and feature selection. In *International Conference of Soft Computing and Pattern Recognition* (pp. 202–207). Paris. https://doi.org/10.1109/socpar.2010.5686097

[18] Cobos, C., Leon, E., & Mendoza, M. (2010). A harmony search algorithm for clustering with feature selection. *Revista Facultad de Ingeniería Universidad de Antioquia, 55,* 153–164. https://doi.org/10.17533/udea.redin.14724

[19] *Hamla, H., & Ghanem, K. (2024). A hybrid feature selection based on fisher score and SVM-RFE for microarray data. Informatica, 48(1).57-68.* https://doi.org/10.31449/inf.v48i1.4759

[20] Akarsu, E., & Karahoca, A. (2011). Simultaneous feature selection and ant colony clustering. *Procedia Computer Science*, *3*, 1432–1438. https://doi.org/10.1016/j.procs.2011.01.026

[21] Javani, M., Faez, K., & Aghlmandi, D. (2011). Clustering and feature selection via PSO algorithm. In *Artificial Intelligence and Signal Processing* (pp. 71–76). https://doi.org/10.1109/aisp.2011.5960988

[22] Guan, Y., Dy, J. G., & Jordan, M. A. (2011). Unified probabilistic model for global and local unsupervised feature selection. In *Proceedings of the International Conference on Machine Learning* (pp. 509–515). Bellevue, WA.

[23] Swetha, K. P., & Devi, V. S. (2012). Simultaneous feature selection and clustering using particle swarm optimization. In *International Conference on Neural Information Processing* (pp. 509–515). Doha, Qatar. https://doi.org/10.1007/978-3-642-34475-6_61

[24] Du, L., & Shen, Y.-D. (2013). Joint clustering and feature selection. In *International Conference on Web-Age Information Management* (pp. 241–252). https://doi.org/10.1007/978-3-642-38562-9_25

[25] Song, Q., Ni, J., & Wang, G. (2013). A fast clustering-based feature subset selection algorithm for high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering*, *25*(1), 1–14. https://doi.org/10.1109/tkde.2011.181

[26] Naik, A., & Satapathy, S. C. (2014). Efficient clustering of dataset based on differential evolution. In Satapathy, S. C., Udgata, S. K., & Biswal, B. N. (Eds.), *Advances in Intelligent Systems and Computing* (pp. 217–227). Springer. https://doi.org/10.1007/978-3-319-02931-3_26

[27] Satapathy, S. C., & Naik, A. (2013). Efficient clustering of dataset based on particle swarm optimization. *International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR), 3*(1), 39–48.

[28] Kumar, V., Chhabra, J. K., & Kumar, D. (2016). An automated parameter selection approach for simultaneous clustering and feature selection. *Journal of Engineering Research, 4*(2), 65–85. https://doi.org/10.7603/s40632-016-0014-2

[29] Satapathy, S. C., Naik, A., & Parvathi, K. (2013). Rough set and teaching learning-based optimization technique for optimal feature selection. *Central European Journal of Computer Science, 3*(1), 27–42. https://doi.org/10.2478/s13537-013-0102-4

[30] Satapathy, S. C., Naik, A., & Parvathi, K. (2013). Unsupervised feature selection using rough set and

teaching learning-based optimization. *International Journal of Artificial Intelligence and Soft Computing, 3*(3), 244–256. https://doi.org/10.1504/ijaisc.2013.053401

[31] Satapathy, S.C., Naik, A. (2012). Hybridization of Rough Set and Differential Evolution Technique for Optimal Features Selection. In: Satapathy, S.C., Avadhani, P.S., Abraham, A. (eds) Proceedings of the International Conference on Information Systems Design and Intelligent Applications 2012 (INDIA 2012) held in Visakhapatnam, India, January 2012. Advances in Intelligent and Soft Computing, vol 132. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-27443-5_52

[32] Kumar, V., Chhabra, J. K., & Kumar, D. (2016). An automated parameter selection approach for simultaneous clustering and feature selection. *Journal of Engineering Research*, *4*(2), 1–21. https://doi.org/10.7603/s40632-016-0014-2

[33] Kumar, V., Kumari, R. & Kumar, S. HMOSHSSA: a novel framework for solving simultaneous clustering and feature selection problems. *Multimed Tools Appl* **83**, 82149–82175 (2024). https://doi.org/10.1007/s11042-024-18726-7

[34] Meesala, Y. V. N., Parida, A. K., & Naik, A. (2024). Optimized feature selection using modified social group optimization. *Informatica*, *48*(11), 195–220. https://doi.org/10.31449/inf.v48i11.6160

[35] Kumar, V., & Chhabra, J. K., & Kumar, D. (2014). Clustering using modified harmony search algorithm. *International Journal of Computational Intelligence Studies, 3*(2/3), 113–133. https://doi.org/10.1504/ijcistudies.2014.062726

[36] Kumar, V., & Kumar, D. (2019). Automatic clustering and feature selection using gravitational search algorithm and its application to microarray data analysis. *Neural Computing and Applications, 31*, 3647–3663. https://doi.org/10.1007/s00521-017-3321-0

[37] Flury, B. (1997). *A first course in multivariate statistics*. Springer-Verlag. http://dx.doi.org/10.1007/978-1-4757-2765-4

[38] Bandyopadhyay, S., & Mukhopadhyay, A., & Maulik, U. (2007). An improved algorithm for clustering gene expression data. *Bioinformatics, 23*(21), 2859–2865. https://doi.org/10.1093/bioinformatics/btm418

[39] Maulik, U., & Bandyopadhyay, S. (2003). Fuzzy partitioning using a real-coded variable-length genetic algorithm for pixel classification. *IEEE Transactions on Geoscience and Remote Sensing, 41*(5), 1075–1081. https://doi.org/10.1109/tgrs.2003.810924

[40] Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall. https://doi.org/10.2307/1268876

[41] Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E. S., & Golub, T. R. (1999). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proceedings of the National Academy of Sciences, 96*(6), 2907–2912. https://doi.org/10.1073/pnas.96.6.2907

[42] Qin, Z. S. (2006). Clustering microarray gene expression data using weighted Chinese restaurant process. *Bioinformatics, 22*(15), 1988–1997. https://doi.org/10.1093/bioinformatics/btl284

[43] Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics, 20*(1), 53-65. https://doi.org/10.1016/0377-0427(87)90125-7