

## RESEARCH ARTICLE



# Automatic Clustering by Spider Monkey Optimisation with Tabu Search Algorithm

**OPEN ACCESS****Received:** 02-06-2023**Accepted:** 03-07-2023**Published:** 01-09-2023**Vaishali P Patel<sup>1\*</sup>, L K Vishwamitra<sup>2</sup>****1** Research Scholar, Department of Computer Science and Engineering, Oriental University, Indore, Madhya Pradesh, India**2** Professor, Department of Computer Science and Engineering, Oriental University, Indore, Madhya Pradesh, India

**Citation:** Patel VP, Vishwamitra LK (2023) Automatic Clustering by Spider Monkey Optimisation with Tabu Search Algorithm. Indian Journal of Science and Technology 16(33): 2589-2600. <https://doi.org/10.17485/IJST/V16i33.1357>

\* **Corresponding author.**

[vaishalipatel02@yahoo.com](mailto:vaishalipatel02@yahoo.com)

**Funding:** None

**Competing Interests:** None

**Copyright:** © 2023 Patel & Vishwamitra. This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Published By Indian Society for Education and Environment ([iSee](#))

**ISSN**

Print: 0974-6846

Electronic: 0974-5645

## Abstract

**Objective:** To design automatic data clustering algorithm that find number of clusters automatically with balance between exploration and exploitation search space. **Methods:** This work proposes Spider monkey optimisation with tabu search algorithm named as SMOTS for automatic data clustering. In this algorithm, the local search of spider monkey is improved with tabu search algorithm. For better results, compact separated index with Gaussian kernel distribution is introduced as a fitness function. The experiments are performed on Vowel, Iris, Wine, Seed, *E.coli* and Thyroid data sets. The results are validated with cluster optimality, inter and intra cluster distances with 5 well known and 7 recently published algorithms like DE, GA, GWO, WOA, PSO, AHPSON, AC-ICA, ACDCSA, AC-MeanABC, TMKGSO, Black Hole k-Means, and EOAK-means. To test the statistical significance of the proposed algorithm an unpaired t-test is performed between SMOTS and second best algorithms on mean inter cluster distance. **Findings:** In comparison with well-known clustering algorithm on six data set SMOTS produced 100%, 33.33%, 83.33% accurate results on cluster optimality, intra and inter cluster distance respectively. In comparison with recently published algorithms on six data set SMOTS produced 50%, 66.66%, 50% accurate results on cluster optimality, intra and inter cluster distance respectively. The hypothesis testing results shows that p-value of the t-test is less than 1% except vowel data set means SMOTS is highly statistically significant compare to second best algorithms. **Novelty:** In real life data set information about number of cluster is rarely available and this produced faulty results. Proposed method can process data without any prior information of number of clusters and data distribution with accurate results.

**Keywords:** Spider Monkey Optimisation; Tabu Search; Automatic Clustering; Neighbour Search; Swarm Intelligence 2

## 1 Introduction

The major drawback of clustering application is the prerequisite of number of cluster at initialisation of algorithm but it is not possible to get correct number of clusters from real life problems<sup>(1)</sup>. An incorrect cluster number at initialisation of algorithm affect the final results<sup>(1)</sup>. The algorithms used to find clusters have limitation of unbalance between exploration and exploitation<sup>(2)</sup>.

Many researchers try to solve said problem with different approach in automatic data clustering literature. In this section we identify most relevant work to automatic data clustering and their limitation.

Manju Sharma<sup>(3)</sup> proposed hybridizations of particle swarm optimization with mutation operator and extended for automatic clustering. To find optimum solution CS index is used as a fitness function. The performance of the algorithm is validated with F measure and inter, intra cluster distance. To improved local search they introduced mutation operator but memory component and data handling strategy is not studied. Mohamed Abd Elaziz<sup>(4)</sup> present atomic search optimization in that local search is improved with sine cosine algorithm. Use CS index as a fitness function. Moyinoluwa B. Agbaje et al<sup>(5)</sup> design hybridization of firefly and particle swarm optimization. They used fire fly to optimise the local solution and PSO for the global search solution. Rajesh Ranjan<sup>(6)</sup> present automatic clustering using crow search algorithm. It automatically update awareness probability and flight length based on new fitness value. In this work CVNN index is used a fitness function. In this index nearest neighbour based separation strategy is used instead of distance based function. In this work they try to address the problem of unbalance but neighbour search, data distribution effect is not shown. Ayat Alrosan<sup>(7)</sup> present mean best ABC for automatic data clustering in that the search equation of ABC is modified with previous mean value and global best of position. In this work VI index is applied as a fitness function that is function of intra, inter cluster distance and Gaussian function. In this work author try to solve complexity of data distribution with Gaussian distribution but memory component and neighbour search is not addressed. Sarah Ghanim<sup>(8)</sup> present equilibrium optimizer algorithm with k means for automatic data clustering. The algorithm is derived from dynamic mass balance of a control volume. Total within-cluster variance is used as the fitness function. Luciano<sup>(9)</sup> present group search optimization with k means for data clustering of real life data set. It try to balance between global search of group search and local search of k means algorithm. A within- cluster sum of squares function is used as a fitness function. S. S. Pal<sup>(10)</sup> present the black hole algorithm with k means for clustering problem. In this work sum of distance of data points from cluster center is considered as a fitness function. It<sup>(3,4,7-9)</sup> is hybridisation of algorithm and application to clustering but complexity of search mechanism and effect of data distribution is not explained. Vaishali Patel<sup>(1)</sup> presented locally neighbour spider monkey, in this work researcher try to solve unbalance of search process but its required number of cluster at initial stage.

In most of the methods listed above, researchers try to solve the problem of unbalance between exploration and exploitation by addition of operators, hybridization or change in search equation. But found results will trapped to local optimum due to lack of search mechanism near current solution and memory component. Further due to insufficiency of the fitness function to handle data distributions the clustering results may not accurate.

## 2 Methodology

The current work proposes SMOTS, Spider monkey optimisation algorithm with tabu search to find number of cluster automatically. In this work to balance between exploration and exploitation we introduced neighbour search mechanism with memory component of tabu search to update the local leader stage of SMO. Further to improved fitness function, the Gaussian kernel distribution is implemented as punishment function to compact separated index (CS-Index). Finally to improve dynamic nature of fitness function with data distribution it is made function of intra to inter cluster distance.

### 2.1 Clustering

In clustering N data points are grouped in to  $K_{max}$  clusters, so that intra cluster distance will be minimum and inter-cluster distance will be maximum.

It is derived as,

N Data points presented as  $Y = \{y_1, y_2, y_3 \dots, y_N\}$

Individual with D attribute written as  $y_i = (y_{i1}, y_{i2}, y_{i3}, \dots, y_{iD})$

Which to be grouped in  $K_{max}$  cluster as  $(K_1, K_2, K_3, \dots, K_{max})$

Subjected to constraint,

$$K_i \neq \phi, \forall i \in \{1, 2, \dots, K_{max}\} \quad (1)$$

$$K_i \cap K_j = \phi, \forall i \neq j \text{ And } i, j \in \{1, 2, \dots, K_{\max}\} \tag{2}$$

$$U_{i=1}^k K_i = Y \tag{3}$$

Data points are assigned to respective group based on Euclidian distance.

## 2.2 Spider monkey optimization algorithm

Spider monkey optimization algorithm is designed by Jagdish Chand Bansal<sup>(11)</sup>. It is derived based on the structure of fission-fusion social behaviour of spider monkey. During food searching task monkeys form a group of 40 to 50 candidates. During food search it is further divided into subgroup. Each subgroup is globally directed by global leader and individual group is directed by local leader.

The SMO algorithm is execute with six stages to complete the food searching task.

Initially the population is generated randomly.

Let,  $S_{ij}$  represent the individual monkey with  $i^{th}$  position and  $j^{th}$  dimension. Then  $S_{ij}$  written as

$$S_{ij} = S_{\min j} + \text{rand}(0, 1) \times (S_{\max j} - S_{\min j}), \tag{4}$$

Where, the lower and upper limit is written as  $S_{\min j}$ ,  $S_{\max j}$  respectively. r and (0,1) is a random number.

### 2.2.1 Local Leader Phase

To update the position of each individual Eq.5 is used in which position values of local leader and individual is used. Further to check position fitness function is used.

$$S_{\text{new}ij} = S_{ij} + \text{rand}(0, 1) \times (L_{nj} - S_{ij}) + \text{rand}(-1, 1) \times (S_{rj} - S_{ij}) \tag{5}$$

Where, the  $j^{th}$  attribute of the  $n^{th}$  local group leader is written as  $L_{nj}$  and  $S_{rj}$  represent  $j^{th}$  attribute of spider monkey selected randomly, Such that  $r \neq i$ . Pr (perturbation rate) is selected between 0.1 to 0.8. Algorithm 1 shows the steps of LLP (Figure 1 ).

```

for each individual  $S_i \in n^{th}$  group do
  for each  $j \in \{1, \dots, D\}$  do
    If  $\text{rand}(0,1) \geq Pr$  then
       $S_{\text{new}ij} = S_{ij} + \text{rand}(0,1) \times (L_{nj} - S_{ij}) + \text{rand}(-1,1) \times (S_{rj} - S_{ij})$ 
    else
       $S_{\text{new}ij} = S_{ij}$ 
    end if
  end for
end for
    
```

Fig 1. Algorithm 1 shows the steps of LLP

### 2.2.2 Global Leader Phase

The position is updated with position value of global leader and other individual of the group as per Eq.6

$$S_{\text{new}ij} = S_{ij} + \text{rand}(0, 1) \times (G_j - S_{ij}) + \text{rand}(-1, 1) \times (S_{rj} - S_{ij}), \tag{6}$$

The  $j^{th}$  attribute of the global leader is written as  $G_j$ . The probability ( $P_i$ ) is calculated as per (2). Algorithm 2 shows the detail steps (Figure 2 ).

$$P_i = \frac{fit_i}{\sum_{i=1}^N fit_i} \tag{7}$$

Where,

$$f_i \text{ ti} = \begin{cases} 1/1 + f'_i, & f_i \geq 0 \\ 1 + \text{abs}(f_i), & f_i < 0 \end{cases}$$

---

```

c = 0;
while group size > count do
  for  $\forall Y_i \in \text{group}$  do
    If  $\text{rand}(0,1) < P_i$  then
      c = c + 1
      randomly choose  $j \in \{1, \dots, D\}$ 
      randomly choose  $S_r \in \text{group}$  with  $i \neq r$ 
       $S_{\text{new}ij} = S_{ij} + \text{rand}(0,1) \times (G_j - S_{ij}) + \text{rand}(-1,1) \times (S_{rj} - S_{ij})$ 
    end if
  end for
end while

```

---

Fig 2. Algorithm 2: Global Leader Phase Algorithm

### 2.2.3 Global Leader Learning Step

The position value of global leader is change based on best fitness value derived from population. The counter of global limit is increment by one if position is not updated.

### 2.2.4 Local Leader Learning Step

The position value of local leader is change based on best fitness value derived from particular group. The counter of local limit is increment by one if position is not updated.

### 2.2.5 Local Leader Decision Step

After reaching to predefine number of count if local leader position is not change then each member of that group is change randomly or by Eq.8 with probability,  $P_i$ .

$$S_{\text{new}ij} = S_{ij} + \text{rand}(0, 1) \times (G_j - S_{ij}) + \text{rand}(0, 1) \times (S_{ij} - L_{nj}). \tag{8}$$

---

```

If Local Leader Limit < Local Limit Count then
  Local Limit Count = 0
  for each  $j \in \{1, \dots, D\}$  do
    If  $\text{rand}(0,1) \geq P_i$  then
       $S_{\text{new}ij} = S_{\text{min}j} + \text{rand}(0,1) \times (S_{\text{max}j} - S_{\text{min}j})$ 
    else
       $S_{\text{new}ij} = S_{ij} + \text{rand}(0,1) \times (G_j - S_{ij}) + \text{rand}(0,1) \times (S_{ij} - L_{nj})$ 
    end if
  end for
end if

```

---

Fig 3. Algorithm 3: Local Leader Decision

### 2.2.6 Global Leader Decision Step

After reaching to the predefined number of count if global leader is not then, entire group is form subgroup with minimum two candidates and with maximum of group size by increment of one. Again same process is repeated, but if still no improvement then all subgroups are merged and form single group.

---

```

Initialization: Local leader limit (LLL), Global leader limit (GLL), Probability ( $P_i$ ), Population.
Find fitness
Find a local and global leader by greedy approach.
While maximum iterations not reach do
    1) The LLP algorithm.
    2) Greedy selection.
    3) Find  $P_i$  by Eq.7.
    4) Apply GLP algorithm.
    5) Global leader learning algorithm.
    6) Local leader learning algorithm.
    7) The LLD algorithm
    8) The GLD algorithm
end while

```

---

Fig 4. Algorithm 4: SMO Algorithm

## 2.3 The proposed algorithm (SMOTS)

In SMO the position is updated randomly and its result in slow convergence, rapid breaking and merging of group<sup>(12)</sup>. Further the local search ability of algorithm is lacking of neighbour search. To defeat the said limitation we introduced Tabu Search algorithm to the local leader learning phase of SMO.

### 2.3.1 The Tabu Search Algorithm

Tabu search is derived by Fred Glover<sup>(13)</sup>. It's have adaptive memory features called a tabu list which prevents visiting repeated solution. It is local search algorithm but it also process non improving solution to free from trapping to local search. The tabu search explores all possible solution around neighbour and best solution is carried out for next step. The steps of the tabu search is presented in algorithm 5 (Figure 5).

### 2.3.2 Modified Local Leader Step

To improve local search we modified local leader search strategy by tabu search algorithm. The detail steps are given in following algorithm (Figure 6).

### 2.3.3 Proposed SMOTS Algorithm

The SMOTS start by initialising input parameters and data set. The initial positions are generated randomly based on input data set which consist cluster centres and activation threshold. First fitness is calculated and then SMO update the initial position iteration by iteration. Further explanation is given in algorithm 7 (Figure 7).

### 2.3.4 Encoding Scheme

The particle position is represented as,  $(K_{max} \times D) + K_{max}$  where the first term presents maximum cluster with attributes and second term present activation threshold between (0, 1). The activation threshold activates or deactivates the corresponding cluster.

In following example, two cluster centres with three attributes can be shown as

### 2.3.5 Active Cluster Selection

The activation threshold value decides whether current cluster is activated or not. If the activation threshold value is grater then cut off threshold then cluster is activated otherwise deactivated and it is fix to 0.5. It may be chances of zero cluster is activated in that cases two clusters with maximum threshold are selected by applying this step minimum two clusters are activated.

```

Tbest = Sij % Initial solution
Best solution = Tbest
Tabu List = [ ]
while maximum iterations not reach do
    generate neighbour position of Tbest
    set Tposition as first position in Tbest neighbourhood
    for Tposition in Tbest neighbourhood do
        if (Tposition NOT in Tabu list AND fitness (Tposition) > fitness (Best solution)) then
            Best solution = Tposition
        end if
    end for
    if fitness (Best solution) > fitness (Tbest) then
        Tbest = Best solution
    end if
    Update tabu memory and promote Best Solution
    if Tabu List size > Max Tabu List Size then
        Delete first position of Tabu list
    end if
end while
return Tbest

```

Fig 5. Tabu search Algorithm

```

for each member, Si ∈ nth group do
    for each j ∈ {1, ... D} do
        If rand(0,1) ≥ Pr then
            Snewj = Sj + rand(0,1) × (Lnj - Sj) + rand(-1,1) × (Sj - Sj)
            Calculate fitness: f(Snewj)
            P = rand(0,1) × (Lnj - Sj) + rand(-1,1) × (Sj - Sj)
            Tbest = Tabu_search (P, fitness, Data_set) % Apply Tabu search Algorithm
            Smj = Sj + Tbest
            Calculate fitness: f(Smj)
            if f(Smj) ≤ f(Snewj)
                Snewj = Smj
                f(Snewj) = f(Smj)
            else
                Snewj = Sj
            end if
        end if
    end for
end for
end for

```

Fig 6. Modified Local Leader Step

**Input parameters:**  
SMO parameters, Maximum cluster ( $K_{max}$ ).

**Output:**  
Clusters, Intra cluster distance, Inter cluster distance.

**Begin:**  
Initialize each particle position with  $K_{max}$  cluster centers and  $K_{max}$  activation thresholds.  
Repeat steps 1-5 up to maximum iterations.

- 1) Select activated cluster center based on threshold.
- 2) Calculate Euclidian distance between activated cluster center and each data point from data set.
- 3) Assign data points to activated cluster that has minimum Euclidian distance.
- 4) Clusters are reinitialized with mean of data set if they contain less than 2 data points.
- 5) Cluster center is updated by SMO algorithm with modified LLP and fitness function.

Fig 7. SMOTS Algorithm

4.3	1.5	2	3.6	5.3	2.8	0.8	0.2
Cluster 1			Cluster 2			Threshold	

Fig 8. Encoding Scheme

### 2.3.6 Fitness Function

In this work compact separated index (CS Index) is applied as a fitness function. For better clustering result smaller value of CS Index is preferred.

The CS measure is written as,

$$CSI = \frac{\sum_{i=1}^{K_{max}} \left( \frac{1}{N_i} \sum_{y_i \in K_i} \max_{y_j \in K_i} (d(y_i, y_j)) \right)}{\sum_{i=1}^{K_{max}} \left( \min_{j \in K_{max}, j \neq i} (d(z_i, z_j)) \right)} \tag{9}$$

Where,  $d(y_i, y_j)$  represent Euclidian distance between two data points'  $x_i$  and  $x_j$ .

Where,  $z_i$  and  $z_j$  are the centres of the clusters  $i$  and  $j$ .

For better clustering results we combine Gaussian kernel function in fitness function and it is presented as,

$$K(y_i, y_j) = \exp \left( - \frac{\|y_i - y_j\|^2}{2\sigma^2} \right) \tag{10}$$

Where,  $\sigma$  is the standard deviation. After Gaussian kernel function, The CS measures modified as, <sup>(14)</sup>

$$CSI_{kernel} = \frac{\sum_{i=1}^{K_{max}} \left( \frac{1}{N_i} \sum_{y_i \in K_i} \max_{y_j \in K_i} (2(1 - K(y_i, y_j))) \right)}{\sum_{i=1}^{K_{max}} \left( \min_{j \in K_{max}, j \neq i} (2(1 - K(z_i, z_j))) \right)} \tag{11}$$

The automatic clustering has tendency to grouped in to two clusters, to remove this limitation Turi <sup>(15)</sup> added punishment function to Gaussian kernel function. It is written as

$$K(\mu, \sigma) = \exp \left( - \frac{(x - \mu)^2}{2\sigma^2} \right) \tag{12}$$

Where, the k is used instead of x. The mean ( $\mu$ ) and standard deviation ( $\sigma$ ) values are initialised during experiments.

Finally, the fitness function is made function of Intra and inter-cluster distance.

$$fit = CSI_{kernal} \cdot \frac{Intra}{Inter} \tag{13}$$

$$Intra = \frac{1}{N} \sum_{i=1}^{K_{max}} \sum_{y \in K_i} \|y - z_i\|^2 \tag{14}$$

$$Inter = \min \left( \|z_i - z_j\|^2 \right), i = 1, 2, \dots, K_{max} - 1, j = i + 1, \dots, K_{max} \tag{15}$$

### 3 Results and Discussion

The proposed SMOTS algorithm is implemented in MATLAB. In this part SMOTS is validated with well-known as well as seven previously published algorithms.

#### 3.1 Dataset

The proposed SMOTS algorithm is tasted on real-life data set taken from UCI repository<sup>(16)</sup>. The detail of datasets is given in Table 1.

Table 1. Dataset

Name	Classes	Attributes	Instances
Vowel	6	3	871
Iris	3	4	150
Wine	3	13	178
Seed	3	7	210
Ecoli	8	7	336
Thyroid	3	5	215

#### 3.2 Experimental Parameters

In SMOTS and other comparison algorithms parameters are set as, maximum iterations equal to 100. The number of runs is 20. Population size is 30. Table 2 presents the maximum clusters; mean and standard deviation of Gaussian distribution.

Table 2. Parameter setting for SMOTS

Dataset	Kmax	Standard deviation ( $\sigma$ )	Mean ( $\mu$ )
Iris	10	0	0
Ecoli	30	0.5	0
Vowel	30	0	0
Wine	10	1	0.5
Thyroid	10	0.5	0
Seed	10	0	0

#### 3.3 Performance based on optimum number of cluster

As shown in Table 3 proposed SMOTS algorithm produced correct number of cluster compared to all well-known algorithms with small standard deviation except seed data set. Table 4 shows the comparison with previously published algorithms. It shows that for iris data set SMOTS produced same results compare to AC-ICA, ACDCSA, and AC-MeanABC which are correct number of clusters. For vowel data set AHPsOM produced better results but SMOTS produce nearer results to it. For thyroid dataset SMOTS and AHPsOM produced better results compared to other algorithms. For wine data set AHPsOM produced better result compared to all algorithms and ACDCSA produced second best result. ACDCSA produced better results for seed data set. For *E.coli* data set SMOTS produced better results compare to all algorithms. From the results of comparison with well-known algorithms, proposed algorithm produced 6 (100%) accurate results on six data sets on mean value of cluster obtain. In comparison with previously published algorithm SMOTS produced 3 (50%) accurate results from six data sets.



**Table 3.** Cluster optimality compared to well-known meta-heuristic algorithms. Bold face indicates best results

Algorithms	Clusters	Iris	Vowel	Thyroid	Wine	<i>E. coli</i>	SEED
Differential Evolution	Mean	4.0500	5.7500	3.5000	3.8500	5.9000	2.0000
	Std	0.2236	2.4252	0.8885	1.3485	2.1740	<b>0.0000</b>
Genetic Algorithm	Mean	3.7000	7.6500	3.5500	3.6500	8.8000	2.3000
	Std	0.4702	2.4979	0.5104	1.2587	2.2850	1.1286
Grey wolf Optimisation	Mean	3.1500	12.0000	3.7500	5.4500	2.5500	3.4500
	Std	0.5871	2.6754	1.4096	1.1910	1.0501	0.9445
Whale Optimisation Algorithm	Mean	4.3500	9.9500	3.6000	3.5400	9.8000	2.2000
	Std	0.7452	2.4810	0.5525	1.5832	2.0417	0.4104
Particle Swarm Optimisation	Mean	3.9000	8.7000	3.4500	4.1000	8.7000	2.1400
	Std	0.7182	2.7357	0.6387	1.5927	1.9762	1.0990
SMOTS	Mean	<b>3.0000</b>	<b>5.9000</b>	<b>3.0000</b>	<b>3.1500</b>	<b>8.0000</b>	<b>3.1500</b>
	Std	<b>0.0000</b>	<b>1.1400</b>	<b>0.0000</b>	<b>0.4300</b>	<b>0.2272</b>	0.3660

**Table 4.** Cluster optimality compared to previously published algorithms. Bold face indicates best results

Algorithms	Clusters	Iris	Vowel	Thyroid	Wine	SEED	<i>E.coli</i>
AHP SOM <sup>(3)</sup>	Mean	3.0250	<b>5.9500</b>	<b>3.0000</b>	<b>3.0750</b>	—	—
	Std	0.0196	0.4640	<b>0.0000</b>	<b>0.0938</b>	—	—
AC-ICA <sup>(17)</sup>	Mean	<b>3.0000</b>	5.7800	—	5.7800	—	—
	Std	<b>0.0000</b>	<b>0.2300</b>	—	0.2300	—	—
ACDCSA <sup>(6)</sup>	Mean	<b>3.0000</b>	5.2000	2.9000	2.95	<b>3.0000</b>	—
	Std	<b>0.0000</b>	0.951	0.307	0.223	<b>0.0000</b>	—
AC-MeanABC <sup>(7)</sup>	Mean	<b>3.0000</b>	—	—	3.1000	—	5.0240
	Std	<b>0.0000</b>	—	—	0.2510	—	0.3880
SMOTS	Mean	<b>3.0000</b>	5.9000	<b>3.0000</b>	3.1500	3.1500	<b>8.0000</b>
	Std	<b>0.0000</b>	1.1400	<b>0.0000</b>	0.4300	0.3660	<b>0.2272</b>

### 3.4 Performance based on internal validity index

Table 5 shows the performance based on intra and inter cluster distance compared to well-known meta-heuristic algorithms.

**Intra cluster distance:** For iris data set WOA produced better results, but SMOTS produce result with zero standard deviation. In vowel data set WOA produce better result compare to all algorithms but with incorrect number of clusters. GWO produced better intra cluster distance for wine data set but with incorrect number of clusters. For *E.coli* and thyroid data set SMOTS produce better result with correct number of clusters. For seed data set GWO produce better result but incorrect number of cluster compared with SMOTS. In intra cluster distance SMOTS produced 2 (33.33%) correct results from six data set in well-known clustering algorithms.

TMKGSO produced better results but SMOTS give zero standard deviation in iris data set with exact number of clusters. For vowel data set SMOTS produce very less value compare to previously published algorithms. In seed data set proposed algorithm give better results compared to previously published algorithms. For thyroid data set SMOTS produced minimum of intra cluster distance compare to other algorithms. ACDCSA produced better results in wine data set. Proposed algorithm perform better for *E.coli* data set. In intra cluster distance SMOTS produced 4 (66.66%) correct results from six data sets in previously published algorithms.

**Inter cluster distance:** Proposed SMOTS algorithm produced better results compared to all well-known algorithms except *E. coli* data set with correct number of clusters. In inter cluster distance SMOTS produced 5(83.33%) correct results from six data set in well-known clustering algorithms.

SMOTS produced maximum of inter cluster distance compare to previously published algorithms with exact number of clusters in iris data set. In vowel data set AHP SOM produce better results. In seed data set TMKGSO give better results and SMOTS produced second best result. SMOTS produced significant results compare to other algorithm in thyroid data set. For wine data set TMKGSO produced better result. In *E. coli* data set proposed algorithm produced significant result. In inter cluster distance SMOTS produced 3 (50%) correct results from six data sets in previously published algorithms.

**Table 5.** Performance based on internal validity indices with well-known meta-heuristic algorithms. Bold face indicates best results

Algorithms	Criteria	Iris	Vowel	Wine	<i>E. coli</i>	Thyroid	SEED
Differential Evolution (DE)	Intra cluster (Mean)	0.3982	48.0902	53.1425	0.0860	5.1778	0.8341
	Std	0.0913	17.6914	30.6873	0.0268	0.4289	<b>0.0711</b>
	Inter cluster (Mean)	4.7895	1261.7495	897.7026	1.2461	79.3529	10.1480
	Std	0.0858	311.5889	288.8303	0.1126	4.5882	<b>0.0422</b>
Genetic Algorithm (GA)	Intra cluster (Mean)	0.4224	37.5000	75.0906	0.0689	6.4918	0.8231
	Std	0.1360	13.7171	<b>0.0000</b>	0.0248	3.6653	0.1825
	Inter cluster (Mean)	4.5957	1094.7462	1097.3939	1.1239	81.8145	10.3337
	Std	0.5913	157.3490	<b>0.0000</b>	0.1274	1.9317	1.3644
Grey wolf Optimisation (GWO)	Intra cluster (Mean)	0.4616	37.0000	<b>9.1776</b>	0.1447	5.1608	<b>0.3247</b>
	Std	0.5170	10.8094	3.2551	0.0413	3.5234	0.1202
	Inter cluster (Mean)	4.3229	946.5393	462.5302	<b>1.3146</b>	76.9858	8.8522
	Std	0.8377	<b>113.8480</b>	142.9202	<b>0.0741</b>	9.4289	1.5236
Whale Optimisation Algorithm (WOA)	Intra cluster (Mean)	<b>0.3001</b>	<b>36.0000</b>	72.8941	0.0569	7.2322	0.7700
	Std	0.1317	13.5336	9.8233	0.0157	3.1553	0.2119
	Inter cluster (Mean)	4.0508	1044.9741	1102.6403	1.0826	81.3685	9.1789
	Std	0.8308	181.5115	23.4626	0.1109	2.1913	1.2962
Particle Swarm Optimisation (PSO)	Intra cluster (Mean)	0.4036	47.5000	63.6901	0.0778	5.4996	0.6521
	Std	0.0735	<b>6.3867</b>	23.4802	0.0271	0.9212	0.2131
	Inter cluster (Mean)	4.6951	1064.0085	947.5021	1.1611	80.9664	10.3074
	Std	0.3145	175.5770	317.1571	0.1670	2.7881	1.6285
SMOTS	Intra cluster (Mean)	1.6401	51.8271	73.2543	<b>0.0429</b>	<b>4.3009</b>	0.9101
	Std	<b>0.0000</b>	28.8180	24.5846	<b>0.0118</b>	<b>0.0000</b>	0.2901
	Inter cluster (Mean)	<b>5.8394</b>	<b>1303.3494</b>	<b>1402.352</b>	1.1792	<b>83.8822</b>	<b>11.4064</b>
	Std	<b>0.0000</b>	375.9623	89.275	0.0012	<b>1.1051</b>	0.5470

**Table 6.** Performance based on internal validity indices with previously published algorithms. Bold face indicates best results

Algorithms	Criteria	Iris	Vowel	SEED	Thyroid	Wine	<i>E. coli</i>
AHPSON <sup>(3)</sup>	Intracluster(Mean)	0.644	171.6	—	—	—	—
	(Std)	1.17E-16	<b>0.564</b>	—	—	—	—
	Intercluster (Mean)	5.648	<b>3108.752</b>	—	—	—	—
	(Std)	1.939	<b>130.03</b>	—	—	—	—
ACDCSA <sup>(6)</sup>	Intracluster(Mean)	1.1642	251.592	2.1115	32.1426	<b>0.6321</b>	—
	(Std)	0.4166	47.6612	0.3557	3.8762	0.0802	—
	Intercluster (Mean)	4.9521	2358.521	7.3545	47.7413	1.6298	—
	(Std)	1.4926	472.9768	2.3824	12.9715	0.6031	—
TMKGSO <sup>(9)</sup>	Intracluster (Mean)	<b>0.6423</b>	—	3.0182	—	28956.4	0.1330
	Std	<b>0.0000</b>	—	<b>0.0000</b>	—	<b>0.0000</b>	<b>0.0000</b>
	Intercluster (Mean)	3.2299	—	<b>13.3530</b>	—	<b>73087.8</b>	0.0516
	Std	<b>0.0000</b>	—	<b>0.0000</b>	—	<b>0.0000</b>	0.0040
Black Hole k-Means <sup>(10)</sup>	Intracluster (Mean)	6.998	—	—	—	48.954	—
EOAK-means <sup>(8)</sup>	Intracluster (Mean)	33.8800	—	—	—	—	—
	Std	41.2310	—	—	—	—	—
	Intracluster(Mean)	1.6401	<b>51.8271</b>	<b>0.9101</b>	<b>4.3009</b>	73.2543	<b>0.0429</b>
	(Std)	<b>0.0000</b>	28.8179	0.2901	<b>0.0000</b>	24.5846	0.0118
SMOTS	Intercluster (Mean)	<b>5.8394</b>	1303.349	11.4064	<b>83.8822</b>	1402.352	<b>1.1792</b>
	(Std)	<b>0.0000</b>	375.9623	0.5470	<b>1.1051</b>	89.275	<b>0.0012</b>

### 3.5 Statistical Analysis of SMOTS algorithm

**Table 7.** An Unpaired T-Test between Best and Second-Best Algorithm (SMOTS)

Second Best	Data set	SE	t	CI	P (two-tailed)	Significance
DE	Iris	0.0196	53.3381	1.01005 to 1.08975	< 0.0001	HSS
DE	Vowel	112.02	0.37135	-185.179 to 268.379	> 0.10	NSS
WOA	Wine	21.1766	14.15	256.842 to 342.581	< 0.0001	HSS
GWO	Ecoli	0.0170	23.797	0.370181 to 0.439019	< 0.0001	HSS
GA	Thyroid	0.5105	4.0498	1.03413 to 3.10127	< 0.001	HSS
GA	Seed	0.3372	3.1808	0.390007 to 1.75539	< 0.01	HSS

To validate the performance of the SMOTS algorithm, an unpaired t-test is performed on the obtained value of the inter cluster distance. An unpaired t-test is applied to find the best algorithm based on the mean inter cluster distance between the best and second-best algorithms. To calculate Confidence Interval (CI), 20 data sizes and 95% confidence level are considered for both algorithms. The significance level of SMOTS compared to the second-best algorithm can be predicted by the confidence interval and two-tailed p-value of the t-test. If  $P \leq 0.01$  It shows HSS: Highly Statistically Significant results, If  $P \leq 0.05$ , It shows SS: Statistically Significant results. If  $P > 0.10$ , it shows NSS: Not Statistically Significant results. From Table 7 it is concluded that proposed SMOTS algorithm produced highly statistically results compare to all second best algorithm except vowel data set.

## 4 Conclusion

In the proposed work to improve the balance between exploration and exploitation the local leader phase of SMO is updated with tabu search algorithm. Further, the compact separated index with Gaussian punishment function is used as a fitness function and applied to automatic clustering. To validate the proposed method, it is compared with 5 well-known and 7 recently published algorithms. Results show that SMOTS algorithm produced 100%, 33.33%, 83.33% accurate results on cluster optimality, intra and inter cluster distance respectively in comparison with well-known algorithms. In comparison with recently published algorithms on six data set SMOTS produced 50%, 66.66%, 50% accurate results on cluster optimality, intra and inter cluster distance respectively. The hypothesis testing results shows that p-value of the t-test is less than 1% except vowel data set means SMOTS is highly statistical significant compare to second best algorithms. In future proposed algorithm can be modified with new search strategies, parallel computing, and multi-objective function for more accurate results. It may be applied to image segmentation, optimisation problems, medical datasets, etc.

## References

- 1) Patel VP, Rawat MK, Patel AS. Local neighbour spider monkey optimization algorithm for data clustering. *Evolutionary Intelligence*. 2023;16(1):133–151. Available from: <https://doi.org/10.1007/s12065-021-00647-1>.
- 2) Patel VP, Rawat MK, Patel AS. Analysis of Search Space in the Domain of Swarm Intelligence. In: M P, TP S, T C, HM P, N GN, editors. Algorithms for Intelligent Systems. Springer Singapore. 2021;p. 99–109. Available from: [https://doi.org/10.1007/978-981-33-4087-9\\_8](https://doi.org/10.1007/978-981-33-4087-9_8).
- 3) Sharma M, Chhabra JK. Sustainable automatic data clustering using hybrid PSO algorithm with mutation. *Sustainable Computing: Informatics and Systems*. 2019;23:144–157. Available from: <https://doi.org/10.1016/j.suscom.2019.07.009>.
- 4) Elaziz MA, Nabil N, Ewees AA, Lu S. Automatic Data Clustering based on Hybrid Atom Search Optimization and Sine-Cosine Algorithm. *2019 IEEE Congress on Evolutionary Computation (CEC)*. 2019;p. 2315–2337. Available from: <https://doi.org/10.1109/CEC.2019.8790361>.
- 5) Agbaje MB, Ezugwu AE, Els R. Automatic Data Clustering Using Hybrid Firefly Particle Swarm Optimization Algorithm. *IEEE Access*. 2019;7:184963–184984. Available from: <https://doi.org/10.1109/ACCESS.2019.2960925>.
- 6) Ranjan R, Chhabra J. Automatic Data Clustering using Dynamic Crow Search Algorithm. 2022. Available from: <http://dx.doi.org/10.4108/eai.17-5-2022.173982>.
- 7) Alomoush MAAW. Automatic Data Clustering Based Mean Best Artificial Bee Colony Algorithm. *Comput Mater Contin*. 2021;68(2):1575–1593. Available from: <https://doi.org/10.32604/cmc.2021.015925>.
- 8) Sgm AK, Algamil ZY, Qasim OS. Enhancement of K-means clustering in big data based on equilibrium optimizer algorithm. *Journal of Intelligent Systems*. 2023;32(1). Available from: <https://doi.org/10.1515/jisys-2022-0230>.
- 9) Pacifico LDS, Ludermir TB. An evaluation of k-means as a local search operator in hybrid memetic group search optimization for data clustering. *Natural Computing*. 2021;20(3):611–636. Available from: <https://doi.org/10.1007/s11047-020-09809-z>.
- 10) Pal SS, Pal SS. Black Hole and k-Means Hybrid Clustering Algorithm. In: HS B, J N, B N, D P, editors. Advances in Intelligent Systems and Computing;vol. 2020. Springer Singapore. 2020;p. 403–413. Available from: [https://doi.org/10.1007/978-981-13-8676-3\\_35](https://doi.org/10.1007/978-981-13-8676-3_35).
- 11) Bansal JC, Sharma H, Jadon SS, Clerc M. Spider Monkey Optimization algorithm for numerical optimization. *Memetic Computing*. 2014;6(1):31–47. Available from: <https://doi.org/10.1007/s12293-013-0128-0>.

- 12) Dhal KG, Das A, Ray S, Das S. A Clustering Based Classification Approach Based on Modified Cuckoo Search Algorithm. *Pattern Recognition and Image Analysis*. 2019;29(3):344–359. Available from: <https://doi.org/10.1134/S1054661819030052>.
- 13) Glover F. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*. 1986;13(5):533–549. Available from: [https://doi.org/10.1016/0305-0548\(86\)90048-1](https://doi.org/10.1016/0305-0548(86)90048-1).
- 14) Das S, Abraham A, Konar A. Automatic kernel clustering with a Multi-Elitist Particle Swarm Optimization Algorithm. *Pattern Recognition Letters*. 2008;29(5):688–699. Available from: <https://doi.org/10.1016/j.patrec.2007.12.002>.
- 15) Turi RH. Clustering-based color image segmentation. Monash University, Australia. 2001. Available from: <https://users.monash.edu/~roset/thesis/thesis.pdf>.
- 16) Dua D, Graff C. UC Irvine Machine Learning Repository. 2017. Available from: <http://archive.ics.uci.edu/ml>.
- 17) Aliniya Z, Mirroshandel SA. A novel combinatorial merge-split approach for automatic clustering using imperialist competitive algorithm. *Expert Systems with Applications*. 2019;117:243–266. Available from: <https://doi.org/10.1016/j.eswa.2018.09.050>.