International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

OL-MFA: An Improved Moth Flame Algorithm for Feature Selection

Nitesh Sureja*[1], Priya Patel[2], Prerna Kadia[3], Nilamba Vala[4]

Submitted: 22/07/2022 **Accepted**: 25/09/2022

Abstract: Feature selection (FS) is an important and crucial task in machine learning. The goal of the feature selection problem is to reduce the dimension of the feature set and maintaining the accuracy of the performance at the same time. This paper presents an improved moth flame algorithm (MFA) to solve the FS problem. The algorithm is improved by integrating opposition based learning (OBL) and levy flights with the original moth flame algorithm (MFA). This improvisation is done to overcome the premature convergence and local optima problem of MFA. The proposed algorithm (OL-MFA) is a swarm intelligent algorithm (SIA) that mimics moths' navigation behavior in nature. The moths navigate toward the real-light source (moon) with a straight path and a fixed angle which is called transverse orientation. Moreover, moths are highly attracted to artificial lights such as flames, and because of the close distance, they change their flight angles continuously, which forms a spiral path. Opposition based learning (OBL) is used to address the premature convergence problem. The search strategy of levy flight works as a regulator of the moth position update to maintain decent population diversity and expand the algorithm's global search capability. The proposed approach is compared against the five swarm intelligence algorithms (SIAs) in terms of metrics, including entropy, purity, completeness score (CS), and homogeneity score (HS). The SSE fitness function is used for fitness evaluation. The results have established that the proposed algorithm is superior over the other state of the art counterparts.

Keywords: feature selection, levy flight, moth flame algorithm, swarm intelligence

1. Introduction

Feature selection (FS) is a task of extracting the most relevant features from a dataset [1]. Feature selection (FS) makes the task of a machine learning model very simple by removing the irrelevant features from the dataset. FS reduces dimensionality of the problem and computational time while increasing predicting accuracy. Feature selection methods are classified as filter methods, wrapper methods, and embedded methods.

Filter methods use some statistical procedures to predict the relations between the features and the target. In this method, a fitness score is assigned to every feature based on its relevance. This fitness score is used by the filter methods to decide the relevance of the respective feature. This method doesn't use any machine learning algorithm. Wrapper methods use different classification algorithms to evaluate the relevance of the extracted subset of the features. All potential combinations of the features are evaluated based on the criterion of evaluation. These methods usually produces good prediction accuracy than filter methods. In the embedded methods, learning algorithms are integrated with the FS methods. Embedded methods do feature selection and classification simultaneously. The features that contribute more in all iterations during training are normally extracted.

Proper selection of the feature selection method is very important. We can solve the FS problems using search methods such as random search, greedy algorithms, and exhaustive search. Most of these methods face the issues of local optima and premature convergence due to their limitations. Typically, all FS problem falls into the category of the NP-hard problems. So, Different optimization algorithms are used to solve FS problems. The exceptional quality of the Swarm intelligence algorithms (SIAs) in finding optimal solutions has attracted numerous authors to use them for solving FS problems.

Moth-flame algorithm (MFA) [2] is an SIA which is based on the navigation behaviour of the moths in nature. This algorithm has proved itself very good through its outstanding performance in solving various NP-hard problems [2]. This algorithm is in its initial state of the research so far. So, we can further improve the performance of MFA by applying some improvements to it. An improved MFA that is integrated with opposition based learning (OBL) and levy flights (OL-MFA) is proposed in this paper.

The contribution of this research is highlighted below.

- An improved MFA is introduced for feature selection.
- The concepts of opposition based learning (OBL) and levy flights are integrated in the introduced algorithm OL-MFA.
- The introduced OL-MFA is assessed using 12 medical datasets available in UCI.
- The results attained with introduced OL-MFA are compared with five state of the art SIAs.
- Four external evaluation measures are used to evaluate the performance of the OL-MFA.
- Based on the assessment, it is found that the time, convergence, and solution quality of the introduced OL-MFA is justified.

The paper is organized into nine sections. A literature review is given in section two. The basic moth flame algorithm, opposition based learning (OBL) and levy flights are presented in section

[1] *Krishna School of Emerging Tech. & Applied Research, Vadodara – 391243, INDIA, ORCID ID: 0000-0002-4429-1597*
[2] *Krishna School of Emerging Tech. & Applied Research, Vadodara – 391243, INDIA, ORCID ID: 0000-0003-2271-7216*
[3] *Gujarat Technological University, Ahmedabad – 382424, INDIA ORCID ID: 0000-0002-9204-6780*
[4] *Dharmsinh Desai University, Nadiad – 387001, INDIA ORCID ID: 0000-0002-7896-1528*
* *Corresponding Author Email: nmsureja@gmail.com*

three, four, and five. The proposed MFA (OL-MFA) and fitness function are described in Section six and seven. Section eight discusses the results and their comparison with the other known SIAs. We conclude in Section nine**.**

2. Related Work

In this research, we have proposed an improved moth flame algorithm (OL-MFA) that is integrated with opposition based learning (OBL) and levy flights. A brief review of the literature based on the SIAs, opposition based learning (OBL) and levy flights for feature selection (FS) is presented in the following section.

Elaziz et. al. [3] presented an opposition based moth flame algorithm for feature selection problem. This algorithm was enhanced by differential evolution. Authors have integrated opposition based learning (OBL) and differential evolution with basic MFA to improve its convergence and exploitation capability. The performance of the proposed algorithm is assessed against some CEC2005 benchmark functions, UCI datasets, and a real dataset comprises of the galaxy images. Tubishat et. al.[4] presented a salp swarm algorithm (SSA) integrated with opposition based learning (OBL) and a unique local search algorithm for feature selection. Opposition based learning (OBL) is used at initialization of SSA for improving the population diversity in the search area. A new local search algorithm is used to improve the exploitation of the SSA. Performance of the proposed algorithm is assessed against 18 UCI datasets. Authors have also compared results with four other SIAs and found their approach superior. Rahul et. al. [5] presented a grey wolf algorithm for the feature selection. This algorithm is improved by integrating opposition based learning (OBL) with it. The goal of this algorithm was to improve the classification results by providing it an optimal subset of the features. The authors have evaluated the performance of the proposed algorithm against 13 bench marked functions and breast cancer dataset. Kelidari et. al. [6] presented a chaotic cuckoo search algorithm enhanced with opposition based learning (OBL), a disruption operator and levy flights. This algorithm was developed for solving feature selection problem. This algorithm addresses the local optima issue by reducing randomization in selecting the features. The authors have evaluated the performance of the proposed algorithm against 20 bench marked UCI datasets. Sihwail et.al. [7] presented an improved harris hawks optimization algorithm for feature selection. They have improved basic Harris Hawks Optimization algorithm by integrating opposition based learning (OBL) and a new search algorithm. Reducing the dimensionality of the problem, computational cost, improving classification accuracy, improving population diversity, and accelerating convergence were the goals of this algorithm. This algorithm was tested against twenty datasets available in the UCI repository.

Xie et. al. [8] tried to overcome the poor exploitation and premature convergence of particle swarm optimisation (PSO) by presenting two new variants of PSO. In the first variant they integrated global best signals, rectified personal, swarm leader enhancement with gaussian distribution, local exploitation using spiral search, mutation operations, and mirroring for solution improvement. The authors tried to improve the first approach by using search coefficients, scattering schemes, adaptive breeding mechanism, and multiple optimal signals. Hu et. al. presented [21] an improved grey wolf optimizer for feature selection. They have analysed A and D parameters which are controlled by parameter a in the position updating. These parameters influence the exploration and exploitation process. A new position update function for balancing global and local search is proposed by analysing the range of values of A and D under binary conditions. Agrawal et al. [10] integrated the quantum concept with a whale optimization algorithm (QWOA) for improving the convergence of the basic WOA for the FS problem. A unique binary gray wolf algorithm (BGWO) is introduced by Emery et. al. [11] for feature selection. Guo et. al. [12] hybridized whale optimization algorithm (WOA) with new mutation and adaptive neighborhood strategies for FS problem. Using this, the algorithm chooses the solutions from the neighborhood of the current best solution. The new mutation strategy helps the algorithm to balance exploration and exploitation to overcome the local optima problem. Chantar et. al. [13] have integrated simulated annealing (SA) algorithm with a binary dragonfly algorithm (BDA) to improve the classification accuracy of BDA in feature selection. The best solution found with the BDA was given to SA for further improvement of the search results. Ahmed et. al. [14] Improved SSA by integrating a new local search and a method for re-positioning of the search agents (Sparrows), into the search space which are wandering beyond the search space. This improvement is carried out to enhance the searching efficiency of the original SSA. Ibrahim et. al. [15] improved social spider algorithm (SSO) by using opposition-based learning for increasing the exploration of the search area. They did this to save the SSO from falling into the local optima. Arora et. al. presented [16] two variants of butterfly algorithm (BOA) by using V-shaped and S-shaped transfer functions respectively. The authors tested both variants with the 21 different UCI datasets. Naseer et. al. [17] presented a hybridised filter based feature selection algorithm in which ACO is combined with the gain ratio. Gain ratio is used here for normalizing preferences between information gain and mutual information. Gain ratio penalizes some high split information as a part of the classifier and uses it over different convergence thresholds for final feature subset selection.

Hichem et. al. [18] presented a binary grasshopper algorithm (BGHO) for the feature selection problem. The results obtained are compared with the other five known approaches and BGHO has proved itself the best among all of them. Wang et. al. [19] improved cuckoo search algorithm by integrating the chaotic maps, two population preservation strategies, levy flights, and a mutation strategy. Initially, chaotic maps are used to improve the initialization diversity of the algorithm for avoiding local optima. Population preservation strategies are used in the next step to select the fittest feature from each of iteration. Finally, levy flight is applied with the new mutation strategy for avoiding convergence issues while working with a large search space.

Zawbaa et. al. [20] presented a chaotic antlion optimizer integrated with random walks and a new controlling parameter I for balancing exploration of the search space and exploitation of the best solutions. The parameter I is used to control the range of the random walks. The chaotic approach is used in this algorithm to improve the tradeoff between exploitation and exploration. Sureja et. al. [21] improved shuffle frog leaping algorithm (SFLA) by integrating simulating annealing (SA) with it. SA is used to exploit more and more near to optimal solutions for the enhancement of the solution quality. Uzer et. al. [22] hybridized the artificial bee colony (ABC) algorithm with support vector machines for improving classification accuracy. 10-fold cross-validation is applied to obtain the classification accuracy of the proposed approach. Too et. el. [23] presented two variants of binary harris hawk algorithms which use different types of the transfer functions for converting continuous version into a discrete one. Salima et. al.

[24] presented an improved version of a wrapper based crow search algorithm (CSA) to extract the finest feature subsets. They did improvements by integrating adaptive awareness probability to enhance the balance between exploration and exploitation, selecting crow by the dynamic local neighborhood to follow, and new searching technique to improve global exploration. Hegazy et. al. [25] improved convergence rate, consistency, and accuracy of salp swarm algorithm by using a new control parameter, inertia weight. They have combined this new algorithm with the KNN classier for feature selection.

3. Basic Moth Flame Algorithm

The moth flame algorithm (MFA) is a population based algorithm [2]. MFA mimics the navigation behaviours of the moths in the nature. The moths travel towards to the moon in a straight path with a fixed angle. This navigation is defined as a transverse orientation. Usually, artificial lights like flame attract the moths very heavily. Moths keep changing their angles of flight due to the close distance and in turn form a spiral. The MFA algorithm models the above behaviours of moths for solving NP-hard problems.

In the basic moth flame algorithm [2], a matrix *M* is used to represent the set of moths. An array *OM* is used to store the fitness value of each moth in the population. Very similar to the moths, a matrix *F* is used to represent the flames. An array *OF* is used to store the fitness value of each flame [2, 29].

The Moth Flame algorithm approximates the global optimal for any problem in the form of a three-tuple. It is formulated as per Eq. (1).

$$MFO = (I, P, T) \tag{1}$$

Here *I* represent a function that is used to generate a population of moths randomly with the respective values of fitness. The function can be modelled methodically as under:

$$I: \emptyset \rightarrow \{M, OM) \tag{2}$$

The *P* function is developed for moving the moths around the search area. In the output, *P* function returns an updated matrix *M* of moths.

$$P : M \rightarrow M \tag{3}$$

If the termination criterion is satisfied then function *T* returns true and false otherwise:

$$T : M \rightarrow \{true, false) \tag{4}$$

The function *P* executes itself iteratively till it returns a true value. We update the position of each moth concerning flames for simulating the behaviour of moths mathematically using Eq. (5).

$$M_i = S(M_i, F_j) \tag{5}$$

Here,

- M_i = i^{th} moth and F_j = j^{th} flame and
- S = the spiral function.

We follow the following conditions to use any type of spirals.

1. Both, starting and ending points for the spiral would be a moth.
2. The position of the flame should be the ending point of the spiral.
3. Variation in the range of the spiral would be restricted to the search area.

For a moth flame algorithm, we can define a logarithmic spiral by considering the above conditions as per Eq. (6).

$$S(M_i, F_j) = D_i \cdot e^{bt} \cdot cos(2\pi t) + F_j \tag{6}$$

Here,

- D_i = distance between ith moth and jth flame
- b = is a constant defines spiral's shape
- t = a number (random) in [-1, 1].

Now we calculate *D* using Eq. (7).

$$D_i = |F_j - M_i| \tag{7}$$

Here,

- M_i = i^{th} moth
- F_j = j^{th} flame
- D_i = distance between ith moth and jth flame.

Concerning a flame, moths next position can be found using Eq. (6). In Eq. (6), the *t* parameter defines how near the next position of the moth is to the flame. (*t* = -1 represent the nearest and *t* = 1 represents the farthermost). Moths moving towards the flame are the only requirement for position updating of moths as per Eq. (6). This situation may trap MFA in local optima very quickly. Using *n* different locations for the position updating of moths also reduces the exploitation of the top favourable solutions. To resolve this problem, we use Eq. (8) in this regard.

$$flame\ no = round\left(N - l * \frac{N-1}{T}\right) \tag{8}$$

Here,

- l = current iteration
- N = maximum flames
- T = maximum iterations

The basic moth flam algorithm (MFA) is described in algorithm 1.

4. Opposition Based Learning (OBL)

Opposition based learning (OBL) is introduced by [26-28] and used to improve search space exploration abilities of the SIAs to jump out of the local optima. OBL selects a number of individuals as solutions from the population and generates an opposite population corresponding to the selected solutions. Usually, OBL helps in selecting the best solutions according to their fitness function by avoiding some local points during the exploration of search space. The opposition based learning (OBL) method in *D*-dimensional space is as per Eq. 9 [27].

$$\bar{x}_j = u_j + l_j - x_j \quad j = 1,2,\ldots,D \tag{9}$$

Here, *x* represents the solution in search space and $\bar{x}_j$ represents the opposite sides of the solution which are generated within the interval [*u*, l]. An opposite number represented by *j* is calculated for each dimension. Finally, $\bar{x}$ represents the opposite solution in the *D* dimension. Here, *u* represents a random number.

5. Levy Flights

The concepts of Levy-flight were introduced by Paul Levy in 1937 [29]. Levy-flight is a random walk with the particular heavy jumps using step lengths which are derived from a probability distribution. We can define term flight as maximum distance in a straight line between two points that an entity in motion covers without directional variation.

Levy flight is integrated with basic MFA to improve the diversity of the population of moths to jump out of the local optima. Specifically, levy flights are composed of clusters of multiple short steps connected by longer relocations.

```
Initialize moths (M) randomly in dimension (D)
While (Iter<=Maxiter)
Perform Updating of flame number using Eq. (8)
$OM$ = Fitness Function ($M$);
if Iter = = 1
    $F$ = sorting ($M$);
    $OF$ = sorting($OM$);
else
    $F$ = sorting($M_t - 1$, $Mt$);
    $OF$ = sorting ($M_t$–1, $Mt$);
end
for $i$ = 1 : $n$
        for $j$ = 1 : $d$
            • Perform Updating of $r$ and $t$
            • Compute $D$ using Eq. (7) concerning the respective moth
            • Updating $M(i, j)$ using Eq. (5) and Eq. (6) concerning the respective moth
        end
end
```

Fig. 1 Basic MFA Algorithm

6. Proposed Moth Flame Algorithm

This paper proposes an improved moth flame algorithm (OL-MFA) to improve convergence, reducing computational cost, and to overcome the local optima problem. We also try to improve the population diversity of the moths. Levy flight possesses excellent characteristics which help in improving the diversity of the population. This makes it very easy for the proposed algorithm to jump out of the local optima and a good balance between the exploitation and exploration capacity of the MFA can be achieved. So, we allow every moth to do levy flight as per Eq. (10) after the updating of positions [31- 32].

$$X_i^{t+1} = X_i^t + u\, sign[rand - 0.5] \oplus levy\,(\beta) \tag{10}$$

Here,

- X_i^t = solution vector
- u = uniformly distributed random number
- $\oplus$ = entry-wise multiplications
- $rand$ = a random number

The signed random number u (rand) can take three values only (1, 0, and −1). The blending of levy-flight and u improves the random walks of moth which in turn helps the MFA to avoid the local optima. Levy flight is considered to be a random walk in which step lengths determined the steps and a levy distribution defines the jumps as per Eq. (11) [31-32]. Eq. (12) provides Levy random numbers.

$$\text{levy}\ (\beta) \sim \mu = t^{-1-\beta}, (0\ \leq 2) \tag{11}$$

$$\text{levy}\ (\beta) \sim \frac{\emptyset \times \mu}{|v|^{1/\beta,}} \tag{12}$$

Here,

- μ and v = normal distributions
- Γ = gamma function,
- β = 1.5 and

ϕ is defined using Eq. (12).

$$\emptyset = \left[\frac{\Gamma(1+\beta)\times sin\ (\pi\times\frac{\beta}{2})}{\Gamma((\frac{1+\beta}{2})\times\beta\times2^{\frac{\beta-1}{2}})}\right]^{1/\beta} \tag{13}$$

```
Initialize moths (M) randomly in dimension (D)
Calculate the opposite moths ($\bar{M}$) using OBL
Select the N fittest moths from ($M \cup \bar{M}$) as initial population using levy flights
While (Iter <= Maxiter)
Perform Updating of flame number using Eq. (8)
$ON$ = Fitness Function (N);
If Iter ==1
$F$ = sorting (N);
$OF$ = sorting (ON);
else
$F$ = sorting ($N_t$– 1, $N_t$);
$OF$ = sorting ($N_t$–1, $N_t$);
end
for $i$ = 1 : $n$
for $j$ = 1 : $d$
  • Perform Updating of $r$ and $t$
  • Compute $D$ using Eq. (7) concerning the respective moth
  • Update ($i$, $j$) using Eq. (5) and Eq. (6) concerning the respective moth
end
for each moth (search agent)
  • Perform Updating the position of the current moth (search agent) using levy flights
end
Iter = Iter + 1;
End
```

Fig. 2 Proposed OL-MFA Algorithm

7. Fitness Function

We use a fitness function to evaluate the fitness of the solutions of the proposed OL-MFA. The Sum of squared error (SSE) [33 - 35] fitness function given in Eq. (13) is used in this work for evaluation.

$$SSE = \sum_{n=1}^{N} dist_{nc}^2 \tag{14}$$

Here, *dist*$_{nc}$ is the distance (Euclidean) between a point and the centroid. We can achieve the enhanced results by minimizing the sum of the squared error (SSE) function.

8. Results and Discussion

This section investigates the efficiency of the OL-MFA algorithm in feature selection. We present the results obtained using the OL-MFA algorithm on well-known 12 medical data sets to conduct a reliable comparison. Furthermore, a fitness function sum of squared error (SSE) is used to show the algorithm's strength. We also present the comparison of the OL-MFA with other five well-known algorithms: BDASA [13], IBSSA [14], OBSSO [15], BBOA [16], and BGHO [18] which have been used previously for feature selection.

8.1. Datasets and Environment

For experiments, a system with an Intel i3 processor, 8 GB RAM, and 64-bit Windows 10 OS is used. The parameter setting for OL-MFA is shown in Table 2. The properties of data sets [36] used to evaluate the proposed OL-MFA is given in Table 1.

Table 1. Data set Properties

Dataset	*Instances*	*Attributes*
Exactly	1000	13
KrVsKpEW	3196	36
M-of-N	1000	13
Vote	300	16
BrestEW	569	30
CongressEW	435	16
Lymphography	148	18
Tic-tac-Toe	958	9
PenglungEW	73	325
Breast cancer	569	569
SpectEW	267	22
WaveformEW	5000	40

8.2. Evaluation Measures

We use four evaluation measures known as homogeneity score (HS), completeness score (CS), purity, and entropy [33, 37, 38] for the performance evaluation of the OL-MFA.

Entropy measure gives the idea about the distribution of the semantic classes inside the cluster and is computed using Eq. (15) [33, 38].

$$Entropy = \sum_{j=1}^{k} \frac{(|P_j|)}{n} E(P_j) \tag{15}$$

Here, $E(P_j)$ represents the individual cluster entropy.

$$E(P_j) = \frac{1}{logk} \sum_{i=1}^{k} \frac{(|P_j \cap T_i|)}{P_j} log\ (\frac{(|P_j \cap T_i|)}{P_j} \tag{16}$$

The purity is computed using Eq. (17) [26, 31].

$$Purity = \frac{1}{n} \sum_{j=1}^{k} max_i \left(|T_i \cap P_j|\right) \tag{17}$$

Here, P_j represents points assigned to cluster j, k represents the total clusters, and truly allocated points in cluster I is represented by T_i,

We use Eq. (18) to compute Homogeneity score (HS) [33, 37, 38].

$$HS = 1 - \frac{H(T/P)}{H(T)} \tag{18}$$

Here $H(T)$ and $H(T|P)$ are entropy and conditional entropy of the classes and they are computed using Eq. (19) and (20).

$$H(T) = -\sum_{t=1}^{|T|} \frac{n_t}{N} . log\left(\frac{n_t}{N}\right) \tag{19}$$

$$H(T|P) = -\sum_{p=1}^{|P|} \sum_{t=1}^{|T|} \frac{n_{pt}}{N} log\left(\frac{n_{pt}}{n_p}\right) \tag{20}$$

Here, n_t and n_p represent the points a true class t has and p is the predicted cluster. And, n_{pt} represents number of points which are clustered in a true class t of a predicted cluster p.

Completeness score (CS) is computed using Eq. (21) [33, 37, 38].

$$CS = 1 - \frac{H(P/T)}{H(P)} \tag{21}$$

Here, $H(P)$ and $H(P|T)$ represent the entropy and the conditional entropy of the clusters and computed using Eq. (22) and (23) [33, 37, 38].

$$H(P) = -\sum_{p=1}^{|P|} \frac{n_p}{N} . log\left(\frac{n_p}{N}\right) \tag{22}$$

$$H(P|T) = -\sum_{t=1}^{|T|} \sum_{p=1}^{|P|} \frac{n_{pt}}{N} log\left(\frac{n_{pt}}{n_p}\right) \tag{23}$$

Table 2. Experimental settings of DL-MFO

Parameters	**Value**
Population size of moths	40
Maximum Iterations	500
Search agents	50

Table 3. Comparison of results (Breast Cancer)

Criteria	***DL-MFA***	***OB SSO***	***BDA SA***	***B-BOA***	***B-GHO***	***IB-SSA***
Fitness Function (SSE)						
Best	199	364	206	321	263	348
Worst	689	684	659	642	637	731
Avg	310	418	242	368	374	486
Std.	0.3	0.3	0.3	0.3	0.3	0.3
Evaluation Measures						
Purity	1.0	0.9	1.0	1.0	1.0	1.0
Entropy	0.1	0.2	0.1	0.1	0.1	0.1
HS	0.8	0.6	0.7	0.8	0.7	0.7
CS	0.8	0.6	0.7	0.8	0.7	0.7
Time	30.0	37.0	31.4	31.2	55.4	69.9

Table 4. Comparison of results (Breast EW)

Criteria	***DL-MFA***	***OB SSO***	***BDA SA***	***B-BOA***	***B-GHO***	***IB-SSA***
Fitness Function (SSE)						
Best	281	567	389	620	599	876
Worst	1611	1762	1177	1527	1599	1705
Avg	642	665	489	690	1083	1167
Std.	0.3	0.2	0.2	0.2	0.3	0.2
Evaluation Measures						
Purity	0.7	0.6	0.6	0.7	0.6	0.6
Entropy	0.5	0.6	0.6	0.5	0.6	0.6
HS	0.2	0.0	0.0	0.2	0.0	0.0
CS	0.2	0.1	0.1	0.3	0.2	0.1
Time	34.2	43.8	38.3	37.4	60.7	70.4

Table 5. Comparison of results (CongressEW)

Criteria	***DL-MFA***	***OB SSO***	***BDA SA***	***B-BOA***	***B-GHO***	***IB-SSA***
Fitness Function (SSE)						
Best	999	1211	1106	1418	1057	1243
Worst	1674	1745	1610	1675	1756	1732
Avg	1171	1285	1161	1466	1289	1459
Std.	0.4	0.3	0.3	0.3	0.4	0.3
Evaluation Measures						
Purity	0.9	0.9	0.9	0.9	0.9	0.9
Entropy	0.3	0.3	0.3	0.3	0.3	0.3
HS	0.5	0.5	0.5	0.5	0.5	0.5
CS	0.5	0.5	0.5	0.5	0.5	0.5
Time	17.9	20.5	17.8	17.7	29.2	34.9

Table 6. Comparison of results (Exactly)

Criteria	***DL-MFA***	***OB SSO***	***BDA SA***	***B-BOA***	***B-GHO***	***IB-SSA***
Fitness Function (SSE)						
Best	3033	3086	3136	3113	3040	3116
Worst	3180	3427	3426	3421	3409	3363
Avg	2999	3120	3202	3152	3155	3197
Std.	3033	3086	3136	3113	3040	3116
Evaluation Measures						
Purity	0.1	0.1	0.1	0.1	0.1	0.1
Entropy	0.7	0.7	0.7	0.7	0.7	0.7
HS	0.6	0.6	0.6	0.6	0.6	0.6
CS	0.0	0.0	0.0	0.0	0.0	0.0
Time	0.0	0.0	0.0	0.0	0.0	0.0

Table 7. Comparison of results (KrVsKpEW)

Criteria	*DL-MFA*	*OB SSO*	*BDA SA*	*B-BOA*	*B-GHO*	*IB-SSA*
Fitness Function (SSE)						
Best	4073	5660	4555	5087	6829	7089
Worst	9171	8041	7594	9844	9438	956
Avg	6703	6308	4897	5774	8452	8467
Std.	4073	5660	4555	5087	6829	7089
Evaluation Measures						
Purity	0.5	0.5	0.5	0.6	0.5	0.5
Entropy	0.6	0.6	0.6	0.6	0.6	0.6
HS	0.0	0.0	0.0	0.0	0.0	0.0
CS	0.1	0.1	0.1	0.0	0.1	0.1
Time	703	2719	781	609	1120	1297

Table 8. Comparison of results (Lymphography)

Criteria	*DL-MFA*	*OB SSO*	*BDA SA*	*B-BOA*	*B-GHO*	*IB-SSA*
Fitness Function (SSE)						
Best	116	143	121	148	146	190
Worst	279	267	222	249	265	277
Avg	158	157	130	158	192	225
Std.	116	143	121	148	146	190
Evaluation Measures						
Purity	0.6	0.6	0.6	0.6	0.6	0.6
Entropy	0.5	0.5	0.5	0.5	0.5	0.5
HS	0.1	0.1	0.1	0.2	0.1	0.1
CS	0.5	0.3	0.4	0.3	0.4	0.2
Time	6.1	6.6	6.2	7.2	11.3	11.9

Table 9. Comparison of results (M-of-N)

Criteria	*DL-MFA*	*OB SSO*	*BDA SA*	*B-BOA*	*B-GHO*	*IB-SSA*
Fitness Function (SSE)						
Best	3022	3097	3064	3132	3116	3151
Worst	3463	3404	3150	3452	3395	3368
Avg	3159	3130	3017	3160	3191	3214
Std.	0.1	0.1	0.1	0.1	0.1	0.1
Evaluation Measures						
Purity	0.6	0.6	0.6	0.6	0.6	0.6
Entropy	0.6	0.6	0.6	0.6	0.6	0.6
HS	0.0	0.0	0.0	0.0	0.0	0.0
CS	0.0	0.0	0.0	0.0	0.0	0.0
Time	52.4	52.7	52.0	52.7	87.9	105.2

Table 10. Comparison of results (PenglungEW)

Criteria	*DL-MFA*	*OB SSO*	*BDA SA*	*B-BOA*	*B-GHO*	*IB-SSA*
Fitness Function (SSE)						
Best	2447	3327	2602	4039	3744	3085
Worst	4126	4085	2994	4079	4121	3984
Avg	2609	3377	2575	3985	3997	3356
Std.	0.4	0.4	0.5	0.3	0.3	0.5
Evaluation Measures						
Purity	0.6	0.5	0.5	0.5	0.5	0.5
Entropy	0.5	0.6	0.5	0.6	0.6	0.6
HS	0.4	0.3	0.4	0.3	0.3	0.3
CS	0.4	0.5	0.5	0.4	0.5	0.3
Time	11.9	7.1	6.6	22.9	9.6	11.3

Table 11. Comparison of results (SpectEW)

Criteria	*DL-MFA*	*OB SSO*	*BDA SA*	*B-BOA*	*B-GHO*	*IB-SSA*
Fitness Function (SSE)						
Best	1025	1183	1138	1187	1152	1234
Worst	1461	1451	1402	1456	1475	1470
Avg	1160	1217	1154	1211	1259	1334
Std.	0.5	0.4	0.4	0.5	0.5	0.3
Evaluation Measures						
Purity	0.8	0.8	0.8	0.8	0.8	0.8
Entropy	0.4	0.4	0.4	0.4	0.4	0.4
HS	0.1	0.1	0.1	0.2	0.1	0.1
CS	0.1	0.1	0.1	0.1	0.1	0.1
Time	12.2	12.8	11.7	11.9	17.9	20.8

Table 12. Comparison of results (Tic-tac-Toe)

Criteria	*DL-MFA*	*OB SSO*	*BDA SA*	*B-BOA*	*B-GHO*	*IB-SSA*
Fitness Function (SSE)						
Best	528	570	532	569	541	575
Worst	813	765	657	806	709	806
Avg	592	594	534	590	596	631
Std.	0.1	0.1	0.1	0.1	0.1	0.1
Evaluation Measures						
Purity	0.7	0.7	0.7	0.7	0.7	0.7
Entropy	0.6	0.6	0.6	0.6	0.6	0.6
HS	0.0	0.0	0.0	0.0	0.0	0.0
CS	0.0	0.0	0.0	0.0	0.0	0.0
Time	47.6	54.9	46.5	51.3	83.0	102.2

Table 13. Comparison of results (Vote)

Criteria	*DL-MFA*	*OB SSO*	*BDA SA*	*B-BOA*	*B-GHO*	*IB-SSA*
Fitness Function (SSE)						
Best	365	418	388	400	386	457
Worst	556	649	653	672	597	643
Avg	380	445	449	470	408	522
Std.	0.2	0.2	0.2	0.2	0.3	0.2
Evaluation Measures						
Purity	0.7	0.6	0.6	0.6	0.8	0.7
Entropy	0.5	0.6	0.6	0.6	0.4	0.5
HS	0.1	0.0	0.0	0.0	0.3	0.2
CS	0.1	0.1	0.0	0.0	0.3	0.2
Time	12.2	13.0	13.6	22.2	12.2	23.7

Table 14. Comparison of results (WaveformEW)

Criteria	*DL-MFA*	*OB SSO*	*BDA SA*	*B-BOA*	*B-GHO*	*IB-SSA*
Fitness Function (SSE)						
Best	5813	6949	8232	7479	7068	8963
Worst	8103	8075	9751	9422	8981	9260
Avg	6953	7464	8630	8237	8251	7893
Std.	0.2	0.2	0.2	0.3	0.2	0.2
Evaluation Measures						
Purity	0.5	0.3	0.3	0.3	0.3	0.4
Entropy	0.7	0.8	0.8	0.8	0.8	0.8
HS	0.1	0.0	0.0	0.0	0.0	0.0
CS	0.2	0.1	0.1	0.1	0.1	0.1
Time						

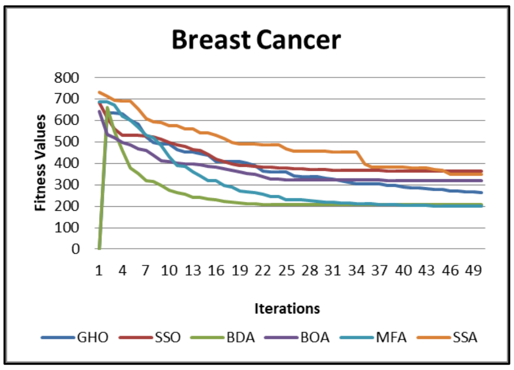

Fig. 3 Result for fitness function SSE
(Breast Cancer Dataset)

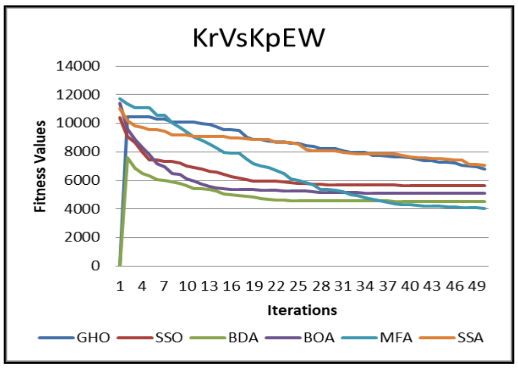

Fig. 7 Result for fitness function SSE
(KrVsKpEW Dataset)

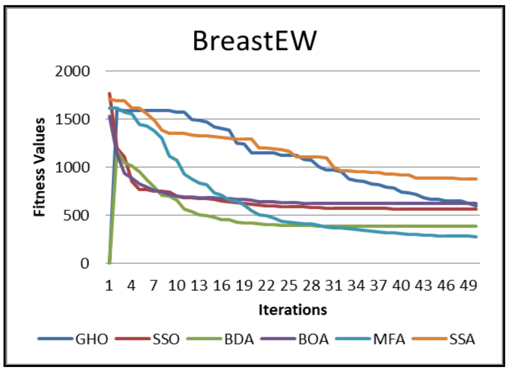

Fig. 4 Result for fitness function SSE
(BreastEW Dataset)

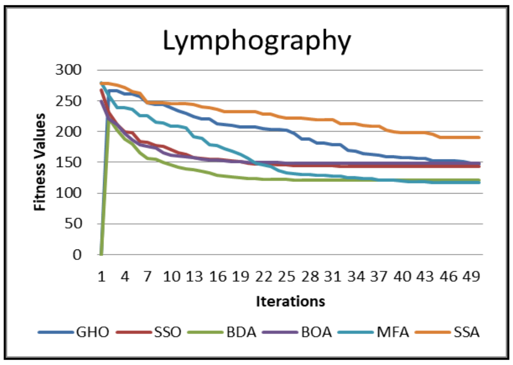

Fig. 8 Result for fitness function SSE
(Lymphography Dataset)

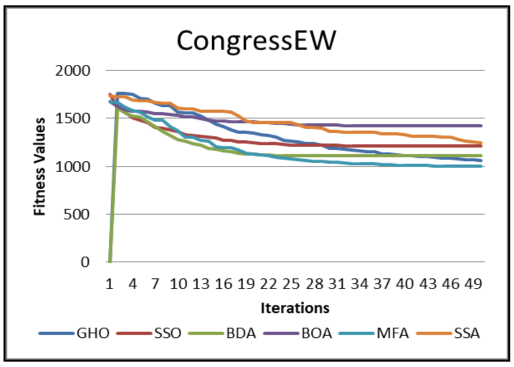

Fig. 5 Result for fitness function SSE
(CongressEW Dataset)

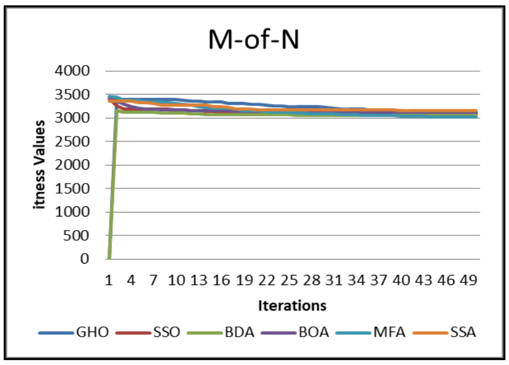

Fig. 9 Result for fitness function SSE
(M-of-N Dataset)

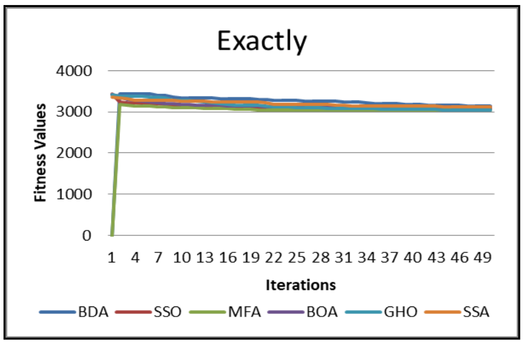

Fig. 6 Result for fitness function SSE
(Exactly Dataset)

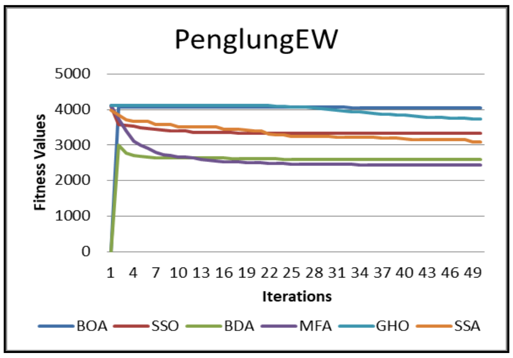

Fig. 10 Result for fitness function SSE
(PenglungEW Dataset)

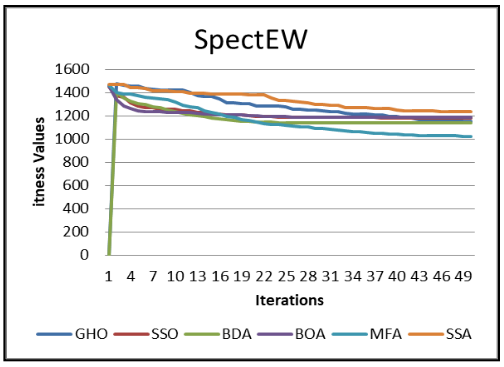

Fig. 11 Result for fitness function SSE (SpectEW Dataset)

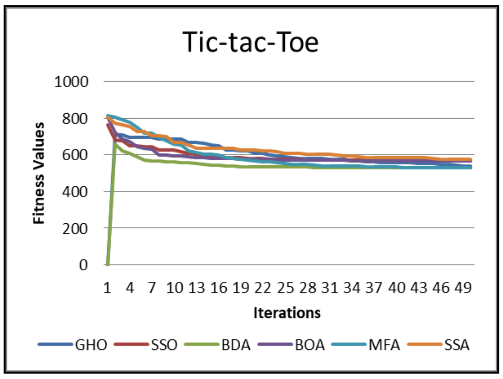

Fig. 12 Result for fitness function SSE (Tic-tac-Toe Dataset)

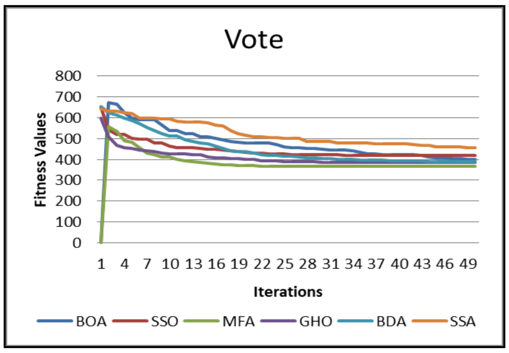

Fig. 13 Result for fitness function SSE (Vote Dataset)

9. Conclusion

The paper introduces a moth flame algorithm enhanced with levy flights (OL-MFA) for feature selection. This approach uses levy fights to further balancing of exploration and exploitation of the MFA. Besides the explanation of the OL-MFA, the experimental evaluation and results attained were also discussed. It was established that OL-MFA produces excellent performance in comparison with the other SIA algorithms, in terms of quality, consistency, and convergence. In the future, OL-MFA can be applied to other real-world problems by hybridizing the classifiers like neural network (NN), support vector machine (SVM).

Acknowledgements

We thank all who provided insight and expertise that greatly assisted the research.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] https://www.heavy.ai/technical-glossary/feature-selection

[2] S. Mirjalili, “Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm”, Knowl. Based Syst., 89, pp. 228–249, 2015.

[3] M. Tubishat, N. Idris, L. Shuib, M. Abushariah and S. Mirjalili, “Improved salp swarm algorithm based on opposition based learning and novel local search algorithm for feature selection”, Expert Systems with Applications, 113122. doi:10.1016/j.eswa.2019.113122

[4] M. kelidari and J. Hamidzadeh, “Feature selection by using chaotic cuckoo optimization algorithm with levy flight, opposition-based learning and disruption operator”, soft Computing. (2020), doi: 10.1007/s00500-020-05349-x

[5] Bulla, P. . “Traffic Sign Detection and Recognition Based on Convolutional Neural Network”. International Journal on Recent and Innovation Trends in Computing and Communication, vol. 10, no. 4, Apr. 2022, pp. 43-53, doi:10.17762/ijritcc.v10i4.5533.

[6] M. Elaziz, A. Ewees, R. Ibrahim and S. Lu,(2019), “ Opposition-based moth-flame optimization improved by differential evolution for feature selection”, Mathematics and Computers in Simulation. 2019. doi:10.1016/j.matcom.2019.06.017

[7] R. Hans, and H. Kaur, “Opposition-based enhanced grey wolf optimization algorithm for feature selection in breast density classification”, Int J Mach Learn Comput, 10(3), 458-464, 2020.

[8] R. Sihwail, K. Omar, K. Ariffin, and M. Tubishat, “Improved Harris Hawks Optimization Using Elite Opposition-Based Learning and Novel Search Mechanism for Feature Selection”, IEEE Access, 1–1,2020. doi:10.1109/access.2020.3006473

[9] H. Xie, L. Zhang, CP. Lim, Y. Yu and H. Liu, “Feature Selection Using Enhanced Particle Swarm Optimization for Classification Models”, Sensors, 21, pp. 1816, 2021.

[10] https://doi.org/10.3390/s21051816

[11] P. Hu, JS. Pan, and SC. Chu, “Improved Binary Grey Wolf Optimizer and Its application for feature selection”, Knowledge-Based Systems, 105746,2 020.

[12] R. Agrawal, B. Kaur and S. Sharma, “Quantum based whale optimization algorithm for wrapper feature selection”, Appl. Soft Comput., 89, 106092, 2020.

[13] E. Emary, H. Zawbaa and A. Hassanien, “Binary grey wolf optimization approaches for feature selection”, Neurocomputing, 172, pp. 371–381, 2016.

[14] W. Guo, T. Liu, F. Dai and P. Xu, "An improved whale optimization algorithm for feature selection," Computers, Materials & Continua, vol. 62, no.1, pp. 337–354, 2020.

[15] H. Chantar, M. Tubishat, M. Essgaer and S. Mirjalili, “Hybrid Binary Dragonfly Algorithm with Simulated Annealing for Feature Selection”, SN Computer Science, 2:295, 2021. doi:10.1007/s42979-021-00687-5

[16] A. Gad, K. Sallam, R. Chakrabortty and M. Ryan, “An improved binary sparrow search algorithm for feature selection in data classification”, Neural Comput & Applic., 2022. doi:10.1007/s00521-022-07203-7

[17] R. Ibrahim, A. Elaziz, D. Oliva, E. Cuevas and S. Lu, “An opposition-based social spider optimization for feature selection”, Soft Computing, 2019. doi:10.1007/s00500-019-03891-x

[18] S. Arora and P. Anand, “Binary butterfly optimization approaches for feature selection”, Expert Systems with Applications, 2018. doi:10.1016/j.eswa.2018.08.051

[19] A. Naseer, W. Shahzad and A. Ellahi, “A Hybrid Approach for Feature Subset Selection using Ant Colony Optimization and Multi-Classifier Ensemble”, International Journal of Advanced Computer Science and Applications, 9(1), doi: 10.14569/IJACSA.2018.090142.

[20] H. Hichem, M. Elkamel, M. Rafik, MT. Mesaaoud and C. Ouahiba, “A new binary grasshopper optimization algorithm for feature selection problem”, Journal of King Saud University - Computer and Information Sciences, 2019.

[21] doi:10.1016/j.jksuci.2019.11.007

[22] L. Wang, Y. Gao, J. Li and X. Wang, "A Feature Selection Method by using Chaotic Cuckoo Search Optimization Algorithm with Elitist Preservation and Uniform Mutation for Data Classification", Discrete Dynamics in Nature and Society, vol. 2021. doi:10.1155/2021/7796696.

[23] H. Zawbaa, E. Emary and C. Grosan, “Feature Selection via Chaotic Antlion Optimization”, PLoS ONE 11(3): e0150652, 2020. doi:10.1371/journal.pone.0150652

[24] Gupta, D. J. . (2022). A Study on Various Cloud Computing Technologies, Implementation Process, Categories and Application Use in Organisation. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(1), 09–12. https://doi.org/10.17762/ijfrcsce.v8i1.2064

[25] N. Sureja, A. Vasant and N. Chaudhari, “Hybrid Shuffled Frog-Simulated Annealing Algorithm for Clustering”, International Journal of Intelligent Engineering and Systems, Vol.14, No.4, 2021. doi: 10.22266/ijies2021.0831.43

[26] M. Uzer, N. Yilmaz and O. Inan, “Feature Selection Method Based on Artificial Bee Colony Algorithm and Support Vector Machines for Medical Datasets Classification”, The Scientific World Journal, pp. 1–10, 2013.

[27] J. Too, AR. Abdullah and NM. Saad, “A New Quadratic Binary Harris Hawk Optimization for Feature Selection”, Electronics, 8(10), 1130, 2019. doi:10.3390/electronics8101130

[28] S. Ouadfel and M. Abd Elaziz, “Enhanced Crow Search Algorithm for Feature Selection”, Expert Systems with Applications, 159, 113572. doi:10.1016/j.eswa.2020.113572

[29] A. Hegazy, MA. Makhlouf and GS. El-Tawel, “Improved salp swarm algorithm for feature selection”, Journal of King Saud University - Computer and Information Sciences, 2018. doi:10.1016/j.jksuci.2018.06.003

[30] H. Tizhoosh, “Opposition-Based Learning: A New Scheme for Machine Intelligence”, In: International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC’06). doi:10.1109/cimca.2005.1631345

[31] D. Oliva, and M. Elaziz, “An improved brainstorm optimization using chaotic opposite-based learning with disruption operator for global optimization and feature selection”, Soft Computing. doi:10.1007/s00500-020-04781-3

[32] M. Aladeemy, L. Adwan, A. Booth, M. Khasawneh, and S. Poranki, “New feature selection methods based on opposition-based learning and self-adaptive cohort intelligence for predicting patient no-shows”, Applied Soft Computing, 86, 105866. doi:10.1016/j.asoc.2019.105866

[33] AF. Kamaruzaman, AM. Zain, SM. Yusuf and A. Udin, “Lévy flight algorithm for optimization problems—a literature review”, Applied Mechanics and Materials, vol. 421, pp. 496–501, 2013.

[34] L. Zhiming, Z. Yongquan, Z. Sen and S. Junmin, “Lévy-Flight Moth-Flame Algorithm for Function Optimization and Engineering Design Problems", Mathematical Problems in Engineering, vol. 2016, ID 1423930, 22 pages, 2016. https://doi.org/10.1155/2016/1423930

[35] X. Yang and S. Deb, “Cuckoo search via Lévy flights”, In: Proceedings of the IEEE World Congress on Nature & Biologically Inspired Computing (NaBIC '09), pp. 210–214, Coimbatore, India, 2009.

[36] Sehirli, E., & Alesmaeil, A. (2022). Detecting Face-Touch Hand Moves Using Smartwatch Inertial Sensors and Convolutional Neural Networks. International Journal of Intelligent Systems and Applications in Engineering, 10(1), 122–128. https://doi.org/10.18201/ijisae.2022.275

[37] X. Yang, “Appendix a: test problems in optimization,” Engineering Optimization, pp. 261–266, 2010.

[38] R. Qaddoura, H. Faris, I. Aljarah and PA. Castillo, “EvoCluster: An Open-Source Nature-Inspired optimization Clustering Framework in Python”, In: Applications of Evolutionary Computation. EvoApplications, Vol. 12104. Springer, Cham, pp.20-36, 2020.

[39] D. Chang, X. Zhang and C. Zheng, “A genetic algorithm with gene rearrangement for K-means clustering”, Pattern Recognition, 42(7), 1210–1222.

[40] C. Lee and E. Antonsson E, “Dynamic partitional clustering using evolution strategies”, In: Industrial Electronics Society, 2000. IECON 2000. 26th Annual Conference of the IEEE, IEEE, vol 4, pp 2716–2721.

[41] Sally Fouad Shady. (2021). Approaches to Teaching a Biomaterials Laboratory Course Online. Journal of Online Engineering Education, 12(1), 01–05. Retrieved from http://onlineengineeringeducation.com/index.php/joee/article/view/43

[42] https://archi ve. ics. uci. edu/ ml

[43] A. Rosenberg and J. Hirschberg, “V-measure: conditional entropy based external cluster evaluation measure”, EMNLP-CoNLL. 2007;7:410–20.

[44] I. Aljarah and S. Ludwig, “A new clustering approach based on Glowworm Swarm Optimization”, In: IEEE congress on evolutionary computation, cancun, Mexico, pp. 2642–2649, 2013.