Received: August 10, 2022. Revised: November 13, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A Discrete Moth Flame Algorithm for Feature Selection

Nitesh Sureja[1]* Priya Patel[1] Mohit Rathod[2] Monika Labana[2]

[1]*Department of Computer Science and Engineering, KSET, KPGU, Vadodara, Gujarat, India*
[2]*Department of Computer Engineering, PIT, Parul University, Vadodara, Gujarat, India*
* Corresponding author's Email: nmsureja@gmail.com

Abstract: A crucial and important task in machine learning is feature selection (FS). The primary goal of the feature selection task is to minimize the dimension of the feature set while preserving performance accuracy. In order to address the FS task, a discrete moth flame algorithm that is combined with levy flights (DL-MFA) is presented in this research. The proposed DL-MFA imitates the natural navigational patterns of moths. The moths move along a straight line at a constant angle in the direction of the true light source (the moon) known as transverse orientation. Additionally, moths are drawn to artificial lights like fires and because of the close proximity; they constantly adjust their flying angles, creating a spiral path. In order to maintain healthy population diversity and increase the global search capabilities of the algorithm, the levy flight search technique is also used as a regulator of the moth position updating mechanism. The five swarm intelligence algorithms (SIAs) are contrasted with the proposed algorithm using measures such as entropy, purity, completeness score (CS), and homogeneity score (HS). For evaluating fitness, the SSE fitness function is utilised. The outcomes have shown that the proposed algorithm achieved purity values in the range 90% to 100%, and entropy 10% to 50%. Proposed DL-MFA has also achieved homogeneity score and completeness score up to 50%. These results prove that the proposed algorithm is better than its state-of-the-art competitors.

Keywords: Feature selection, Levy flight, Moth flame algorithm, Swarm intelligence, Classification.

1. Introduction

The process of choosing the most crucial features to feed into machine learning algorithms is known as feature selection (FS) [1]. Feature selection (FS) eliminates the majority of the dataset's pointless features, which simplifies our machine learning model. Additionally, it decreases the problem's dimensionality while increasing prediction accuracy and processing efficiency. Filter methods, wrapper methods, and embedded methods are the three categories of feature selection techniques, and they are detailed below.

Filter methods: These methods forecast the relationship between the features and our aim using some statistical techniques. Each feature receives a score based on how essential it is. This score is used by filter algorithms to determine if a characteristic is pertinent or not. There is no machine learning algorithm used in this strategy.

Wrapper Methods: Using the classifier's performance evaluation, wrapper methods rate the subset of features' quality. Here, a feature subset selection machine learning approach on a given dataset is applied. A greedy technique is used to assess all potential feature combinations against the evaluation criterion. When compared to filter approaches, the wrapper feature selection typically produces good prediction accuracy.

Embedded Methods: Methods that incorporate feature selection into the learning algorithm are referred to as embedded methods. The feature selection and classification operations are carried out simultaneously by embedded algorithms. Typically, during training, the characteristics that have contributed the most across all iterations are extracted. Examples of embedded approaches include feature selection using decision trees and random forests.

The filter methods ignore the interaction with the classifier and each feature is considered independently thus ignoring feature dependencies. In addition, it is not clear how to determine the threshold point for rankings. Wrapper methods are computationally expensive as most of the execution time is spent in training the predictor. They are also prone to overfitting. Embedded methods suffer from the loss of a large part of the information contained in the data set due to the elimination of most of the features. These methods also ignore interactions and correlations between variables.

Generally speaking, any FS problem is an NP-hard problem. The solutions to this problem are found via optimization algorithms. Most of the optimization algorithms have the ability to address the problems faced by the different FS methods mentioned above.

Swarm intelligence algorithms (SIAs) are one of the types of the optimization algorithms. SIAs are based on the collective intelligence shown by the objects like animals, insects and others. A number of SIAs are proposed to solve the optimization algorithms. An SIA that is based on the natural navigational patterns of moths is called the moth-flame algorithm (MFA) [2]. We have proposed a discrete version of the moth flame algorithm (DL-MFA) hybridized with levy flights in this paper. We have applied this discrete MFA to feature selection (FS) problem.

Below we emphasize this study's contribution.

1. The introduction of a swarm intelligence algorithm for feature selection based on MFA.
2. The newly presented algorithm incorporates the ideas of levy flights.
3. Twelve medical datasets that are available at UCI are used to evaluate the introduced algorithm.
4. Five cutting-edge SIAs are contrasted with the outcomes produced by the newly introduced MFA.
5. The effectiveness of the newly implemented algorithm is assessed using four external evaluation measures.
6. Based on the evaluation, it is determined that the introduced algorithm's time, convergence, and solution quality are acceptable.

There are seven sections in the paper. Section two presents a review of the literature. Section three provides an overview of the fundamental moth flame algorithm. Section four and five presents levy flights and Proposed DL-MFA. Fitness function is discussed in section 6. The results and their comparison with other known SIAs are covered in section seven. Conclusion is followed section seven.

2. Related work

In this research, we propose a discrete moth flame algorithm (DL-MFA) that is integrated with levy flights. DL-MFA is a swarm based algorithm (SIA). A brief review of the literature based on the SIAs for feature selection (FS) is as follows.

Xie et al. [3] tried to overcome the poor exploitation and premature convergence of particle swarm optimization (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. A unique binary gray wolf algorithm (BGWO) is introduced by Emery et al. [4] for FS. Hu et al. [5] introduced an enhanced update equation for balancing the exploration and exploitation in the searching. The introduced method gave some new transfer functions to be used in the new update equation. A grey wolf algorithm with multiple search strategies (MEGWO) was introduced by Q. Tu et al. [6]. Guo et al. hybridized the whale optimization algorithm (WOA) with new mutation and adaptive neighborhoods strategies for the FS problem. Using this, the algorithm chooses solutions from the neighborhoods of the current best solution. The new mutation strategy helps the algorithm balance exploration and exploitation to overcome the local optima problem. R. K. Agrawal et al. [7] integrated the quantum concept with a whale optimization algorithm (QWOA) to improve the convergence of the basic WOA for the FS problem. Chantar et al. [8] have integrated a 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. Ibrahim et al. [9] improved the social spider algorithm (SSO) by using opposition-based learning to increase the search area's exploration. They did this to save the SSO from falling into the local optima.

Hichem et al. [10] 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

Received: August 10, 2022. Revised: November 13, 2022.

them. Ahmed and colleagues [11] improved SSA by integrating a new local search and a method for re-positioning the search agents (Sparrows) into the search space that are wandering beyond the search space. This improvement was carried out to enhance the searching efficiency of the original SSA. Arora et al. presented [12] two variants of the butterfly algorithm (BBOA) by using V-shaped and S-shaped transfer functions, respectively. The authors tested both variants with the 21 different UCI datasets.

Wang et al. [13] improved the 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 to avoid local optima. Population preservation strategies are used in the next step to select the fittest feature from each iteration. Finally, levy flight is applied with the new mutation strategy to avoid convergence issues while working with a large search space. Naseer et al. [14] presented a hybridized filter-based feature selection algorithm in which ACO is combined with the gain ratio. A gain ratio is used here to normalize preferences between information gain and mutual information. The 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. Hu et al. presented [15] an improved grey wolf optimizer for feature selection. They have analyzed the 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 analyzing the range of values of A and D under binary conditions.

Zawbaa et al. [16] presented a chaotic antlion optimizer integrated with random walks and a new controlling parameter 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. Too et al. [17] presented two variants of binary harris hawk algorithms which use different types of transfer functions for converting a continuous version into a discrete one. Hegazy et al. [18] improved the convergence rate, consistency, and accuracy of the salp swarm algorithm by using a new control parameter, inertia weight. They have combined this new algorithm with the KNN classifier for feature selection. Sureja et al. [19] improved the shuffle frog leaping algorithm (SFLA) by integrating simulating annealing (SA) with it. SA is used to exploit more and more near-optimal solutions for the enhancement of the solution quality. Uzer et al. [20] improved classification accuracy by combining the artificial bee colony (ABC) algorithm with support vector machines. A 10-fold cross-validation is applied to obtain the classification accuracy of the proposed approach. Salima et al. [21] presented an improved version of a wrapper-based crow search algorithm (CSA) to extract the finest feature subsets. They made improvements by integrating adaptive awareness probability to enhance the balance between exploration and exploitation, selecting crow by the dynamic local neighborhoods to follow, and by developing new searching techniques to improve global exploration.

All of the above work suffers from the one or more drawbacks like computational complexity, execution time, stuck in local optima, loss of information, and noise.

There is a scope to propose a new, modified, and hybrid algorithm for feature selection (FS) problem based on no free lunch (NFL) theorem. This motivates us to modify and present a discrete version of the moth flame algorithm (DL-MFA) for feature selection (FS). The performance of any optimization algorithm depends on how efficiently it explores and exploit the search space and how quickly finds global optimal solution without getting trapped in local optimal solution. So, we have hybridized levy flights with DL-MFA to improve the exploration and exploitation of the search space.

3. Basic moth flame algorithm

The moth flame algorithm (MFA) is a population based approach [2]. MFA imitates moths' natural navigational strategies. The moths move in a straight line at a constant angle in the direction of the moon. Transverse orientation is the term used to describe this navigation. Typically, artificial lighting, such as flames, has a strong attraction to moths. Due to the near proximity, moths continuously alter their flight angle, which causes them to spiral. In order to solve NP-hard problems, the MFA algorithm simulates the aforementioned moth behaviors.

The set of moths is represented by a matrix M in the fundamental moth flame algorithm [2]. The fitness value of each moth in the population is kept in an array called OM. The flames are represented by a matrix F, which is very similar to the moths. The fitness value of each flame is kept in an array OF [2, 22].

The Moth Flame algorithm generates a three-tuple that represents an approximation of the global ideal for every problem. It is made in accordance with 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 modeled methodically as under.

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

The P function is developed for moving the moths about the search region. The updated matrix M of moths is returned by the P function in the output.

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

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

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

The function P performs itself repeatedly until it produces a true value. We update each moth's position in relation to the flames for mathematically simulating moth behavior using Eq. (5).

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

Here, M_i and F_j represents *ith* moth and *jth* flame respectively. S is 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 is the distance between *ith* moth and *jth* flame, b is a constant defines spiral's shape, and t represents a number (random) in [-1, 1].

Now we calculate D using Eq. (7).

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

Here, M_i and F_j *are ith* moth and *jth* flame, and D_i is the distance between *ith* moth and *jth* flame.

Eq. (6) is used to determine a moth's next position in relation to a flame. The t parameter in Eq. (6) determines how close the moth will be to the flame in its subsequent position. (The closest is represented by t = -1, and the farthest by t = 1).

Only moths travelling in the direction of the flame are considered position updates according to Eq. (6) MFA could become swiftly trapped in local optima under this circumstance. The exploitation of the most useful solutions is equally decreased when n distinct locations are used for moths' position updates. Eq. (8) is used to solve this problem.

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

Here, l is a current iteration, N is maximum flames and T represents maximum iterations. The basic moth flam algorithm is given in algorithm 1.

4. Levy flight

The concepts of Levy-flight were introduced by Paul Levy in 1937 [23]. 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

```
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
```

Figure. 1 Basic MFA algorithm

flights are composed of clusters of multiple short steps connected by longer relocations.

5. Proposed moth flame algorithm

To solve the local optima problem, this research suggests the discrete levy flight moth-flame algorithm (DL-MFA). Additionally, we work to increase the diversity of the moth population. Excellent qualities of Levy flight contribute to increasing population diversity. Due to this, the suggested method can very easily jump out from the local optimum. It is also possible to achieve a good balance between the MFA's exploitation and exploration capabilities. Therefore, following the update of positions [24, 25], we permit each moth to perform its flight in accordance with Eq. (9).

$$X_i^{t+1} = X_i^t + u\ sign[rand - 0.5]\ \oplus levy\ (\beta) \quad (9)$$

Here, X_i^t is a solution vector, *u and rand are* uniformly distributed random numbers, and $\oplus$ s represents entry-wise multiplications.

There are only three possible values for the signed random integer u (rand): 1, 0, and 1. Levy-flight and u are combined to improve the random walks of the moth, which then aids the MFA in avoiding local maxima. According to Eq. (10) [24, 25], levy flight is considered to be a random walk in which the step lengths determine the steps and a levy distribution determines the leaps. Levy random numbers are provided by Eq. (11).

$$levy\ (\beta) \sim \mu = t^{-1-\beta}, (0\ \leq 2) \quad (10)$$

$$levy\ (\beta) \sim \frac{\emptyset \times \mu}{|v|^{1/\beta,}} \quad (11)$$

Here, u and v are normal distributions, Γ is a gamma function, and β=1.5. ∅ is defined using Eq. (12).

$$\emptyset = \left[\frac{\Gamma(1+\beta) \times sin\ (\pi \times \frac{\beta}{2})}{\Gamma((\frac{1+\beta}{2}) \times \beta \times 2^{\frac{\beta-1}{2}})}\right]^{1/\beta} \quad (12)$$

6. Fitness function

To assess the fitness of the suggested DL MFA solutions, we employ a fitness function. In this study, evaluation is conducted using the Sum of Squared Error (SSE) [26-28] fitness function provided in Eq. (13).

$$SSE = \sum_{n=1}^{N} dist_{nc}^2 \quad (13)$$

```
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, M_t);
OF = sorting (M_t−1, M_t);
end
for i = 1 : n
for j = 1 : d
  • Perform Updating of r and t, and 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-flight
end
Iter = Iter + 1;
End
```

Figure. 2 Proposed DL-MFA algorithm

$Dist_{nc}$ in this case refers to the (Euclidean) distance between a point and the centroid. By reducing the sum of the squared error (SSE) function, we can get the improved outcomes.

7. Results and discussions

The effectiveness of the DL-MFA algorithm in feature selection is examined in this section. To make an accurate comparison, we present the outcomes of utilizing the DL-MFA method on 12 well-known medical data sets. A fitness function sum of squared error (SSE) is also employed to demonstrate the algorithm's strength. We also compare the DL-MFA to five other popular algorithms that have been applied to feature selection in the past: BDASA [8], OBSSO [9], BGHO [10], IBSSA [11], and BBOA [12].

7.1 Datasets and environment

A computer system with an Intel i3 processor, 8 GB of RAM, and the 64-bit Windows 10 operating system is used to carry out the research. Table 2 displays the parameter settings for the proposed DL-MFA. Twelve medical data sets were utilized [29] to evaluate the proposed MFA in Table 1 based on their characteristics.

7.2 Evaluation measures

Homogeneity score (HS), purity, completeness

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

score (CS), and entropy measures are used to evaluate the performance of the DL-MFA [26, 30, 31]. The entropy is calculated Using Eq. (14) [26, 31], and provides information about the distribution of the semantic classes inside the cluster.

$$Entropy = \sum_{j=1}^{k} \frac{(|P_j|)}{n} E(P_j) \tag{14}$$

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

$$E(P_j) = \frac{1}{logk} \sum_{i=1}^{k} \frac{(|P_j \cap T_i|)}{P_j} \; log\,(\frac{(|P_j \cap T_i|)}{P_j} \tag{15}$$

The purity is computed using Eq. (16) [26, 31].

$$Purity = \frac{1}{n} \sum_{j=1}^{k} max_i \left(\left|T_i \cap P_j\right|\right) \tag{16}$$

Here, P_j represents points allotted to cluster j, k represents all clusters, and T_i represents points actually allotted to cluster I.

We compute homogeneity score (HS) using Eq. (17) [26, 30, 31].

$$HS = 1 - \frac{H(T/P)}{H(T)} \tag{17}$$

Here $H(T)$ and $H(T|P)$ are entropy and conditional entropy of the classes and computed using Eqs. (18) and (19).

$$H(T) = -\sum_{t=1}^{|T|} \frac{n_t}{N} . log \left(\frac{n_t}{N}\right) \tag{18}$$

$$H(T|P) = -\sum_{p=1}^{|P|} \sum_{t=1}^{|T|} \frac{n_{pt}}{N} log \left(\frac{n_{pt}}{n_p}\right) \tag{19}$$

Here, p is the anticipated cluster, while n_t and n_p stand for the points that a true class t holds. Additionally, n_{pt} denotes the quantity of points that are grouped together into a true class (t) of a predicted cluster (p). Eq. (20) is used to get the completeness score (CS) [26, 30, 31].

$$CS = 1 - \frac{H(P/T)}{H(P)} \tag{20}$$

Table 2. Experimental settings of DL-MFA

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

Table 3. Comparison of results (Brest Cancer)

Criteria	DL-MFA	OB SSO	BDA -SA	B-BOA	B-GHO	IB-SSA
SSE						
Best	209	374	216	331	274	359
Worst	699	695	670	653	647	742
Avg	321	429	253	378	384	496
Std.	0.3	0.3	0.3	0.3	0.3	0.3
Evaluation Measures						
Purity	1.0	0.8	0.9	0.8	0.9	0.8
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 (BrestEW)

Criteria	DL-MFA	OB SSO	BDA -SA	B-BOA	B-GHO	IB-SSA
SSE						
Best	292	578	400	630	610	887
Worst	1622	1773	1187	1537	1610	1716
Avg	653	675	500	700	1094	1178
Std.	0.3	0.2	0.2	0.2	0.3	0.2
Evaluation Measures						
Purity	0.9	0.8	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	0.4	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
SSE						
Best	1010	1222	1116	1429	1067	1254
Worst	1685	1756	1620	1685	1767	1743
Avg	1181	1296	1171	1477	1300	1469
Std.	0.4	0.3	0.3	0.3	0.4	0.3
Evaluation Measures						
Purity	0.9	0.8	0.9	0.7	0.9	0.8
Entropy	0.2	0.3	0.3	0.3	0.3	0.3
HS	0.6	0.5	0.5	0.5	0.5	0.5
CS	0.7	0.5	0.5	0.5	0.5	0.5
Time	17.9	20.5	17.8	17.7	29.2	34.9

Here, $H(P)$ and $H(P|T)$ stand for the entropy and the conditional entropy of the clusters and computed using Eqs. (21) and (22) [26, 30, 31].

$$H(P) = -\sum_{p=1}^{|P|} \frac{n_p}{N} . log\left(\frac{n_p}{N}\right) \quad (21)$$

$$H(P|T) = -\sum_{t=1}^{|T|} \sum_{p=1}^{|P|} \frac{n_{pt}}{N} log\left(\frac{n_{pt}}{n_p}\right) \quad (22)$$

Table 6. Comparison of results (Exactly)

Criteria	DL-MFA	OB-SSO	BDA-SA	B-BOA	B-GHO	IB-SSA
SSE						
Best	3044	3097	3123	3126	3051	3147
Worst	3191	3437	3432	3373	3420	3437
Avg	3009	3131	3163	3207	3166	3212
Std.	0.1	0.1	0.1	0.1	0.1	0.1
Evaluation Measures						
Purity	1.0	0.9	0.9	0.9	1.0	0.9
Entropy	0.5	0.6	0.6	0.6	0.6	0.6
HS	0.6	0.6	0.6	0.6	0.6	0.6
CS	0.7	0.7	0.6	0.6	0.7	0.6
Time	51.9	51.8	59.6	53.3	52.6	60.7

Table 7. Comparison of results (KrVsKpEW)

Criteria	DL-MFA	OB-SSO	BDA-SA	B-BOA	B-GHO	IB-SSA
SSE						
Best	4083	6839	5097	5671	4565	7099
Worst	11721	10449	11452	10425	7605	11067
Avg	6714	8463	5784	6318	4907	8477
Std.	0.2	0.2	0.1	0.1	0.2	0.2
Evaluation Measures						
Purity	0.9	0.7	0.8	0.8	0.9	0.7
Entropy	0.6	0.6	0.6	0.6	0.6	0.6
HS	0.2	0.1	0.1	0.1	0.2	0.3
CS	0.2	0.1	0.1	0.1	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
Best	127	153	157	159	131	200
Worst	290	278	276	260	232	288
Avg	168	168	202	169	141	235
Std.	0.2	0.2	0.2	0.1	0.2	0.2
Evaluation Measures						
Purity	0.9	0.8	0.8	0.8	0.8	07
Entropy	0.5	0.5	0.5	0.5	0.5	0.5
HS	0.3	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
SSE						
Best	3033	3127	3107	3143	3074	3162
Worst	3474	3406	3414	3462	3161	3379
Avg	3170	3201	3141	3171	3027	3224
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	0.2	0.1	0.2	0.1	0.2	0.0
CS	0.3	0.1	0.2	0.1	0.3	0.1
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
SSE						
Best	2457	3337	3096	3755	2612	4050
Worst	4137	4095	3995	4131	3004	4090
Avg	2620	3387	3366	4008	2585	3996
Std.	0.4	0.4	0.5	0.3	0.5	0.3
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
SSE						
Best	1035	1193	1162	1198	1149	1244
Worst	1471	1461	1485	1467	1412	1480
Avg	1170	1228	1270	1222	1164	1345
Std.	0.5	0.4	0.5	0.5	0.4	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
Best	538	581	551	579	543	585
Worst	823	775	720	817	667	816
Avg	603	604	606	601	545	642
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
			SSE			
Best	376	429	410	396	398	468
Worst	567	659	682	608	664	653
Avg	390	455	481	419	459	533
Std.	0.2	0.2	0.2	0.3	0.2	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	0.4	0.1	0.1	0.2	0.3	0.2
CS	0.3	0.1	0.0	0.1	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
			SSE			
Best	15823	26960	28242	24808	20691	33647
Worst	47114	46086	39762	44239	42891	46271
Avg	16963	28975	29640	32247	29261	38264
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	0.3	0.2	0.2	0.3	0.3	0.2
CS	0.2	0.1	0.1	0.1	0.1	0.1
Time	1194	2239	1709	18206	2069	2922

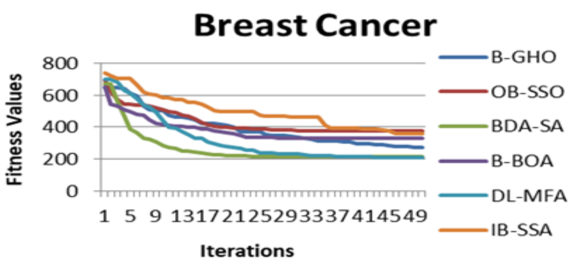

Figure. 3 Result for fitness function SSE (Breast Cancer dataset)

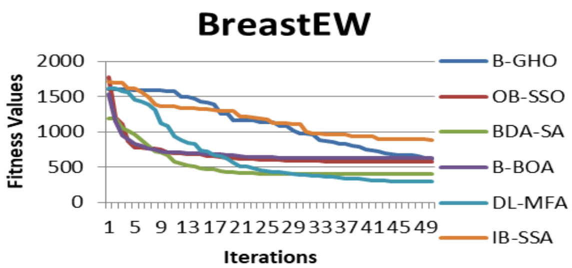

Figure. 4 Result for fitness function SSE (BreastEW dataset)

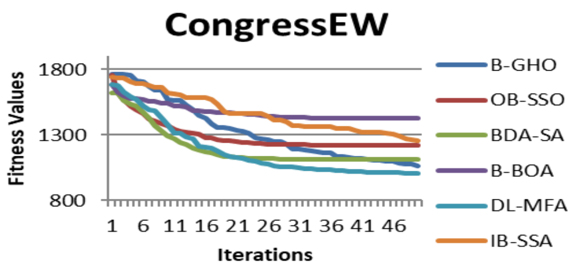

Figure. 5 Result for fitness function SSE (CongressEW dataset)

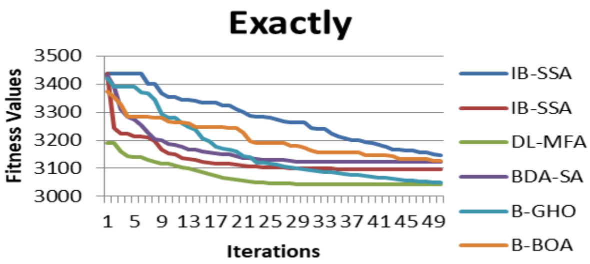

Figure. 6 Result for fitness function SSE (Exactly dataset)

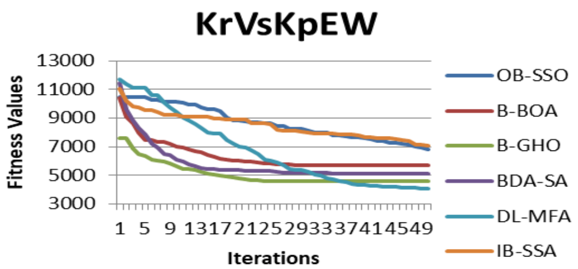

Figure. 7 Result for fitness function SSE (KrVsKpEW dataset)

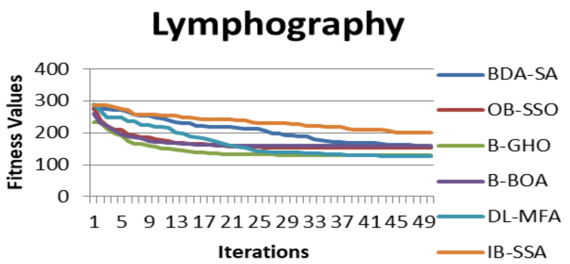

Figure. 8 Result for fitness function SSE (Lymphography dataset)

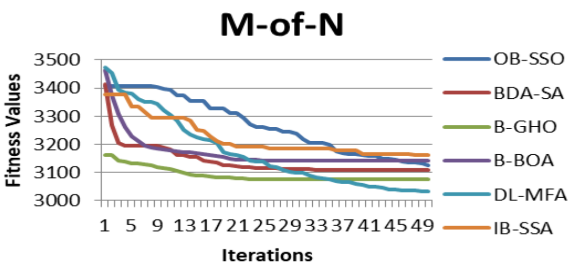

Figure. 9 Result for fitness function SSE (M-of-N dataset)

Received: August 10, 2022. Revised: November 13, 2022.

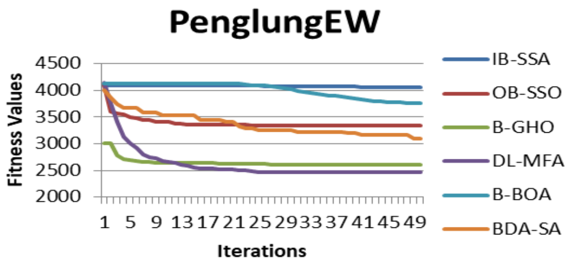

Figure. 10 Result for fitness function SSE (PenglungEW dataset)

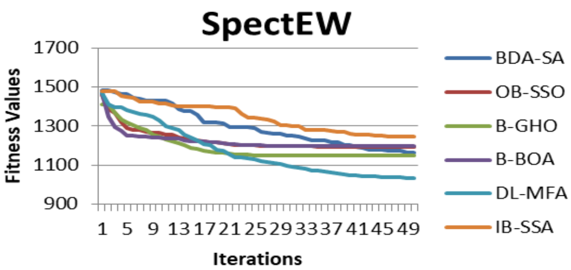

Figure. 11 Result for fitness function SSE (SpectEW dataset)

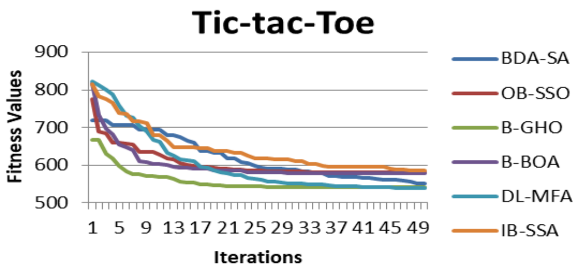

Figure. 12 Result for fitness function SSE (Tic-tac-Toe dataset)

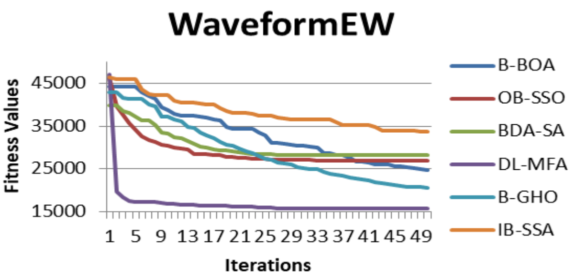

Figure. 13 Result for fitness function SSE (Vote dataset)

8. Conclusion

This paper proposes a discrete moth flame algorithm (DL-MFA) for feature selection that has been improved with levy flights. Levy flights are used in this algorithm to further balance the exploration and exploitation of MFA. Along with presenting the DL-MFA, the experimental analysis and outcomes were also covered. Purity, entropy, homogeneity score, and completeness score evaluation measures are used to evaluate the performance of the algorithm. This algorithm (DL-MFA) produces purity in the range of 90% to100%, very less entropy values up to 10%, good homogeneity score up to 70%, and completeness score up to 70%. Based on the results achieved, it was determined that DL-MFA produces good performance in terms of quality, consistency, and convergence when compared to the other SIA algorithms. Future applications of DL-MFA to various real-world issues can be made by combining classifiers such as neural networks (NN) and support vector machines (SVM).

Conflicts of Interest

The authors declare no conflict of interest

Author Contributions

Conceptualization, NMS, PDP, MJR and ML; methodology and software, NMS, and PDP; validation, and formal analysis, NMS, PDP and MJR; writing—review and editing, NMS; visualization, NMS, PDP, MJR; supervision, NMS investigation, NMS; resources, PDP,MJR and ML; data collection, PDP, MJR and ML; writing—original draft preparation, NMS, PDP, MJP and ML.

Acknowledgments

We thank all of our supporters.

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.*, Vol. 89, pp. 228-249, 2015.

[3] H. Xie, L. Zhang, C. Lim, Y. Yu, and H. Liu, "Feature Selection Using Enhanced Particle Swarm Optimisation for Classification Models", *Sensors*, Vol. 21, p. 1816, 2021.

[4] E. Emary, H. Zawbaa, and A. Hassanien, "Binary grey wolf optimization approaches for feature selection", *Neurocomputing*, Vol. 172, pp. 371-381, 2016.

[5] P. Hu, J. Pan, and S. Chu, "Improved binary grey wolf optimizer and its application for feature selection", *Knowl.-Based Syst.*, Vol. 195, 105746, 2020.

[6] 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.

[7] R. Agrawal, B. Kaur, and S. Sharma, "Quantum based whale optimization algorithm for wrapper feature selection", *Appl. Soft Comput.*, Vol. 89, 106092, 2020.

[8] H. Chantar, M. Tubishat, M. Essgaer, and S. Mirjalili, "Hybrid Binary Dragonfly Algorithm with Simulated Annealing for Feature Selection", *SN Computer Science*, Vol. 2, No. 295, 2021, doi:10.1007/s42979-021-00687-5

[9] R. Ibrahim, A. Elaziz, D. Oliva, E. Cuevas, and S. Lu, "An opposition-based social spider optimization for feature selection", *Soft Computing*, 2019.

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

[11] 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

[12] 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

[13] 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, Article ID 77966, 2021, doi: 10.1155/2021/7796696.

[14] 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*, Vol. 9, No. 1, doi: 10.14569/IJACSA.2018.090142.

[15] P. Hu, J. Pan, and S. Chu, "Improved Binary Grey Wolf Optimizer and Its application for feature selection", *Knowledge-Based Systems*, 105746, 2020.

[16] H. Zawbaa, E. Emary, and C. Grosan, "Feature Selection via Chaotic Antlion Optimization", *PLoS ONE*, Vol. 11, No. 3, e0150652, 2020, doi: 10.1371/journal.pone.0150652

[17] J. Too, A. Abdullah, and N. Saad, "A New Quadratic Binary Harris Hawk Optimization for Feature Selection", *Electronics*, Vol. 8, No. 10, 1130, 2019, doi: 10.3390/electronics8101130

[18] A. Hegazy, M. Makhlouf, and G. E. 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

[19] 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

[20] 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.

[21] S. Ouadfel and M. A. Elaziz, "Enhanced Crow Search Algorithm for Feature Selection", *Expert Systems with Applications*, 159, 113572, doi: 10.1016/j.eswa.2020.113572

[22] 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, p. 22, 2016, https://doi.org/10.1155/2016/1423930

[23] A. Kamaruzaman, A. Zain, S. Yusuf, and A. Udin, "Lévy flight algorithm for optimization problems—a literature review", *Applied Mechanics and Materials*, Vol. 421, pp. 496-501, 2013.

[24] X. S. Yang and S. Deb, "Cuckoo search via Lévy flights", In: *Proc. of the IEEE World Congress on Nature & Biologically Inspired Computing (NaBIC '09)*, pp. 210-214, 2009.

[25] X. Yang, "Appendix a: test problems in optimization", *Engineering Optimization*, pp. 261-266, 2010.

[26] R. Qaddoura, H. Faris, I. Aljarah, and P. A. Castillo, "EvoCluster: An Open-Source Nature-Inspired optimization Clustering Framework in Python", *Applications of Evolutionary Computation. EvoApplications*, Vol. 12104, pp. 20-36, 2020.

[27] D. Chang, X. Zhang, and C. Zheng, "A genetic algorithm with gene rearrangement for K-means clustering", *Pattern Recognition*, Vol. 42, No. 7, pp. 1210–1222, 2009.

[28] C. Lee and E. Antonsson E, "Dynamic partitional clustering using evolution strategies", In: *Proc. 2000 26th Annual Conference of the IEEE Industrial Electronics Society. IECON 2000. 2000 IEEE International Conference on*

Industrial Electronics, Control and Instrumentation. 21st Century Technologies, vol.4, pp. 2716-2721, 2000, doi: 10.1109/IECON.2000.972427.

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

[30] A. Rosenberg and J. Hirschberg, "V-measure: conditional entropy based external cluster evaluation measure", In: *Proc. of 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 410–420, 2007.

[31] I. Aljarah and S. Ludwig, "A new clustering approach based on Glowworm Swarm Optimization", In: *Proc. of IEEE Congress on Evolutionary Computation*, pp. 2642-2649, 2013.