Feature Selection using Binary Moth Flame Optimization with Time Varying Flames Strategies

Ruba Abu Khurma[1], Pedro A. Castillo[2], Ahmad Sharieh[1] and Ibrahim Aljarah[1]
[1]*King Abdullah II School for Information Technology, The University of Jordan, Amman, Jordan*
[2]*Department of Computer Architecture and Computer Technology, ETSIIT and CITIC, University of Granada, Granada, Spain*

Keywords: Moth Flame Optimization, MFO, Feature Selection, Classification, Flames Number, Optimization.

Abstract: In this paper, a new feature selection (FS) approach is proposed based on the Moth Flame Optimization (MFO) algorithm with time-varying flames number strategies. FS is a data preprocessing technique that is applied to minimize the number of features in a data set to enhance the performance of the learning algorithm (e.g classifier) and reduce the learning time. Finding the best feature subset is a challenging search process that requires exponential running time if the complete search space is generated. Meta-heuristics algorithms are promising alternative solutions that have proven their performance in finding approximated optimal solutions within a reasonable time. The MFO algorithm is a recently developed Swarm Intelligence (SI) algorithm that has demonstrated effective performance in solving various optimization problems. This is due to its spiral update strategy that enhances the convergence trends of the algorithm. The number of flames is an important parameter in the MFO algorithm that controls the balance between the exploration and exploitation phases during the optimization process. In the standard MFO, the number of flames linearly decreases throughout the iterations. This paper proposes different time-varying strategies to update the number of flames and analyzes their impact on the performance of MFO when used to solve the FS problem. Seventeen medical benchmark data sets were used to evaluate the performance of the proposed approach. The proposed approach is compared with other well-regarded meta-heuristics and the results show promising performance in tackling the FS problem.

1 INTRODUCTION

Curse of dimensionality is a challenging problem for data mining tasks (e.g classification and clustering). It means increasing the number of dimensions (features) in a data set (Khurma et al., 2020). This will lead to having noisy features that are either redundant (highly correlated with each other) or irrelevant (weakly related to the target class). The major negative consequences of this phenomenon are the degradation in the performance of the learning algorithm and the increasing of the learning time.

Feature selection (FS) is a preprocessing stage in a data mining process to minimize the number of features and improve the performance of the learning process. This is done by excluding the uninformative features and producing a smaller version of a dataset that includes only the most representative features (Aljarah et al., 2018b).

The FS process consists of two primary processes: searching for the best feature subset and evaluating the generated feature subset (Khurma. et al., 2020). Looking at the evaluation process, there are two main approaches to determine the goodness of a candidate solution: filter and wrapper approaches. Filters perform the evaluation based on the intrinsic characteristics of the features and their dependencies with each other and with the target class. In literature, there is a wide range of examples on filters such as F-score, Information Gain (IG) and Chi-Square (Tang et al., 2014). In wrapper approaches, the evaluation decision depends on the learning process. The involvement of learning in wrappers contributes to better performance and slower optimization time in comparison with filters.

Considering the search process, two main approaches can be used: complete and meta-heuristic search (Mafarja et al., 2018). Complete search approach (i.e brute force) generates the entire feature space and applies an exhaustive search to determine

Khurma, R., Castillo, P., Sharieh, A. and Aljarah, I.
Feature Selection using Binary Moth Flame Optimization with Time Varying Flames Strategies.
DOI: 10.5220/0010021700170027
In *Proceedings of the 12th International Joint Conference on Computational Intelligence (IJCCI 2020)*, pages 17-27
ISBN: 978-989-758-475-6

the best feature subset from all candidate feature subsets (possible solutions). This is time-consuming and impractical with large feature spaces. Formally speaking, for a dataset with N features, the corresponding feature space size is 2^N.

The meta-heuristic search approach mitigated the complexity problem of large feature spaces by generating random solutions. Although this approach may lose the exact optimal solution it may reach to promising (near) optimal solution in a reasonable time. Thus, it gives up the exact optimal in exchange for faster learning time.

A common taxonomy for meta-heuristic algorithms (MHs) are based on the inspiration source from nature that divided MHs into two main categories Evolutionary Algorithms (EA) and Swarm Intelligence (SI) (Khurma et al., 2020). EAs are inspired by Darwin's theory of evolution. EAs use optimization operators to evolve solutions such as crossover, mutation and elitism. The most popular EAs are Genetic Algorithm (GA) (Oliveira et al., 2010) and Differential Evolution (DE) (Khushaba et al., 2011).

SI category includes algorithms that are inspired by the social behaviour of creatures that live in groups such as schools of fish, the swarm of wolves, flocks of birds, colonies of bee, etc (Brezočnik et al., 2018). SI algorithm usually mimics the survival strategy of swarm members and how they exchange information to find the food then convert that into a mathematical methodology for the SI algorithm. There are many examples for SIs such as Gray Wolves Optimization(GWO) (Medjahed et al., 2016), Cuckoo Search (CS) (Rodrigues et al., 2013) and Bat Algorithm (BA) (Nakamura et al., 2012). SIs and EAs are similar in that they are both follow the population-based paradigm. The algorithm starts by initializing a set of solutions (e.g. population) then the population undergoes an iterative refining process that simulates either evolutionary or swarming behavior that the algorithm was originally inspired by. Updating solutions continues until a predefined stopping condition is met. In any population-based algorithm, there are two main phases for the optimization process called exploration and exploitation (Khurma. et al., 2020). These are conflicting milestones because the optimizer in exploration searches globally to explore different regions of the search space but in exploitation, the search is made locally in the most promising region where the optimal solution is likely to be found. A proper balance between exploration and exploitation will enhance the performance of the optimization process because a lot of exploration increases the time of search and may lose the optimal solution and excessive exploitation may lead to stuck in local minima. Recently, MHs have been proposed to solve various optimization problems including the FS problem. MHs have been used to provide valid optimized solutions for the FS problem in a wide range of applications such as Arabic handwritten letter recognition (Tubishat et al., 2018), facial recognition (Mistry et al., 2017), financial diagnosis (Wang and Tan, 2017; Al-Madi et al., 2018), and medical application (Wang et al., 2017). In the medical applications, the FS-MH approach has been efficiently applied to minimize the size of a medical data set without causing degradation in the performance of a learner. Moreover, FS-MH maintains the readability of the medical data set by providing the physician with the most useful features that are able to diagnose the patient's status without affecting the original meaning of those features. In literature, researchers worked to enhance the MHs by applying different modification strategies: integrating new operators (e.g chaotic maps (Khurma. et al., 2020), rough set (Inbarani et al., 2014) and Levy flight (Zhang et al., 2016)), hybridization with other algorithm (e.g filter (Yang et al., 2010), MH (Zawbaa et al., 2018) and classifier (Aljarah et al., 2018a)), using new initialization mechanism (Medjahed et al., 2016) and adopting new update strategy such as the time-varying strategy (Mafarja et al., 2017).

Moth Flame Optimization (MFO) is a recent SI algorithm that was proposed in (Mirjalili and Lewis, 2013). MFO was originally developed to solve continuous optimization problems, then a binary version called BMFO was proposed in (Reddy et al., 2018) to solve the binary optimization problem such as FS. Many modifications have been adopted to improve the BMFO as a search algorithm in the FS process including (Ewees et al., 2017; Zhang et al., 2016; Hassanien et al., 2017; Sayed and Hassanien, 2018; Sayed et al., 2016; Wang et al., 2017). For more information about MFO, a reader can refer to the reviews (Mehne and Mirjalili, 2020; Shehab et al., 2019). The MFO's success is mainly due to its spiral update strategy and the adaptive parameters that control the adaptive convergence of the algorithm and maintain the trade-off between exploration and exploitation.

In MFO, moths represent the search agents that change their positions in the search space. The best positions obtained so far are called flames. To guarantee the exploitation around the best positions, MFO applies an adaptive mechanism to decrease the number of flames over the course of iterations. Thus, there will be N flames in the initial iterations that are equal to the number of moths then due to the gradual decrements, the moths will update their positions with respect to one flame in the final iterations. In the standard MFO, the adjustment for the number of flames is performed using a linear decreasing strategy. This chapter proposes for the first time different updating strategies to control the number of flames parameter. The aim is to study the effect of using different time variant functions for the number of flames on the MFO optimization process in the feature space. The main contribution of this paper is the enhancement of the MFO capability as a wrapper FS by utilizing its spiral update strategy in combination with the time-varying flames strategy. The generated five variants of BMFO are evaluated in terms of fitness value, classification accuracy, number of selected features and processing time. The KNN classifier is used to assess the accuracy of the different approaches when applied on a seventeen benchmark medical data set. The experimental results show superior performance in tackling the FS problem.

The paper is organized as follows: Section 2 presents an overview of the MFO algorithm. Section 3, describes the proposed approach. Section 4, shows the experimental results. Finally, a conclusion and future works are provided in Section 5.

2 MOTH FLAME OPTIMIZATION

Moth Flame Optimization (MFO) is a recent SI algorithm that was developed in (Mirjalili and Lewis, 2013). The natural inspiration of MFO came from the movements of moths at night. These moths travel in a straight line and maintain a stable angle to the moon. This movement strategy is called transfer orientation. The transfer orientation is not useful if the source light is close because the moth will try to maintain the same angle with it, forcing it to move in a spiral way and eventually die. Fig 1 shows the conceptual model of the MFO algorithm.

Eq.1 describes mathematically the natural spiral motion of moths around a flame where M_i represents the i_{th} moth, F_j represents the j_{th} flame, and S is the spiral function. Eq.2 formulates the spiral motion using a standard logarithmic function where D_i is the distance between the i_{th} moth and the j_{th} flame as described in Eq.3, b is a constant value for determining the shape of the logarithmic spiral, and t is a random number in the range [-1, 1]. The parameter $t = -1$ indicates the closest position of a moth to a flame where $t = 1$ indicates the farthest position between a moth and a flame. To achieve more exploitation in the search space the t parameter is considered in the range [r,1] where r is linearly decreased throughout iterations from -1 to -2. Eq.8 shows gradual decrements of the number of flames throughout iterations. Algorithm 1 shows the entire pseudo code of the MFO algorithm.

$$M_i = S(M_i, F_j) \tag{1}$$

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

$$D_i = |M_i - F_j| \tag{3}$$

Algorithm 1: Pseudo-code of the MFO algorithm.

```
Input:$Max\_iteration$, $n$ (number of moths), $d$ (number of dimensions)
Output:Approximated global solution
Initialize the position of moths
  while $l \leq Max\_iteration$ do
    Update flame no using Eq.8
    OM = FitnessFunction(M);
    if $l$ == 1 then
      $F = sort(M)$;
      $OF = sort(OM)$;
    else
      $F = sort(M_{l-1}, M_l)$;
      $OF = sort(OM_{l-1}, OM_l)$;
    end if
    for i = 1: n do
      for j = 1: d do
        Update $r$ and $t$;
        Calculate $D$ using Eq.3 with respect to the corresponding moth;
        Update $M(i, j)$ using Eqs.1 and Eqs.2 with respect to the corresponding moth;
      end for
    end for
    $l$ = $l$ + 1;
  end while
```

2.1 Binary Moth Flame Optimization

The original MFO was designed to solve global optimization problems (Mirjalili, 2015). In such cases, the individual elements are real values. Thus, the optimizer's task is to verify that the upper and lower limits of each element are not exceeded in the initialization and update processes. However, for binary optimization problems, the case is different because each individual's elements are restricted to either "0" or "1". To enable the MFO algorithm to optimize in a binary feature space, the MFO needs to integrate some operators with it. The most common binary operator used to convert continuous optimizers into binary is the transfer function (TF) (Mirjalili and Lewis, 2013). The main reason for using TFs is that they are easy to implement without affecting the essence of the algorithm. In this paper, the used TF is the sigmoid

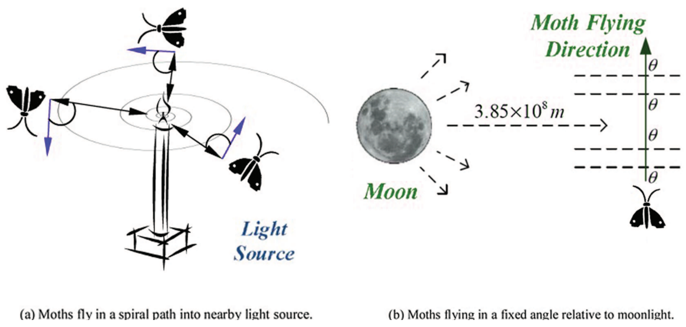

Light Source
Moon
Moth Flying Direction
$3.85\times10^8 m$
θ
(a) Moths fly in a spiral path into nearby light source.
(b) Moths flying in a fixed angle relative to moonlight.

Figure 1: The conceptual model for the movement behaviour of moths.

function which was originally used in (Kennedy and Eberhart, 1997) to generate the binary PSO (BPSO). In the MFO algorithm, the first term of Eq.2 represents the step vector which is redefined in Eq.4. The function of the sigmoid is to determine a probability value in the range [0,1] for each element of the solution. Eq.5 shows the formula of the sigmoid function. Each moth updates its position based on Eq.6 which takes the output of Eq.5 as its input.

$$\Delta M = D_i \times e^{bt} \times cos(2\pi) \tag{4}$$

$$TF(\Delta M_t) = 1/(1+e^{\Delta M_t}) \tag{5}$$

$$M_i^d(t+1) = \begin{cases} 0, & \text{if } rand < TF(\Delta M_{t+1}) \\ 1, & \text{if } rand \geqslant TF(\Delta M_{t+1}) \end{cases} \tag{6}$$

2.2 Binary Moth Flame Optimization for Feature Selection

For an FS problem, there are two important issues to consider to establish a correct optimization process: the representation of the individual and the evaluation of it. Usually, the individual in the FS problem is represented using a 1-d array. The value of each element can be assigned to two values, either ”0” or ”1”. The value ”0” indicates that the feature is not selected while the value ”1” indicates that the feature is selected. The evaluation of the individual in an FS problem depends on using a fitness function that maximizes the performance of a classifier and minimizes the number of selected features simultaneously.

Eq.7 formulates the FS problem where $\alpha\gamma_R(D)$ is the error rate of the classification produced by a classifier, $|R|$ is the number of selected features in the reduced data set, and $|C|$ is the number of features in the original data set, and $\alpha \in [0,1]$, $\beta = (1-\alpha)$ are two parameters for representing the importance of classification performance and length of feature subset based on recommendations (Faris et al., 2019).

$$Fitness = \alpha\gamma_R(D) + \beta\frac{|R|}{|C|} \tag{7}$$

3 THE PROPOSED APPROACH

The optimization process of the population-based meta-heuristic algorithms has two conflicting milestones: exploration and exploitation. In exploration, the optimizer searches globally and explores several regions of the search space to determine the most promising region where a global solution might exist. In contrast, the optimizer in exploitation searches locally within these regions to find the (near) global optima. A good balance between exploration and exploitation will enable the optimizer to achieve better convergence results in a reasonable time. Usually, a meta-heuristic algorithm has one or more parameters that control the transition between exploration and exploitation. The ideal behavior is to extend the exploration phase in the early stages of the optimization process to navigate to more regions in the search space and then seamlessly switch to the exploitation phase to focus on a specific region until it comes very close to the optimal solution. In MFO, the flames number is the parameter that controls the balance between exploration and exploitation. In the standard MFO algorithm, the number of flames is updated using a linear decreasing formula. Different update strategies are proposed in this paper for the first time to guarantee the gradual decrements of the number of flames throughout iterations.

- Flames number with linearly decreasing strategy.

  This is the standard formula used in the original MFO algorithm. As appears in Eq.8, there will be more exploration for the search space in the early optimization stages while there will be more exploitation in the final stages. The flames no is decreased linearly across iterations (Shi and Eberhart, 1999). Flames no at each iteration is defined as in Eq.8 where l is the current number of iteration, N is the maximum number of flames and T is the maximum number of iterations.

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

- Flames number with Non-linear coefficient decreasing strategy.

  The Non-linear coefficient decreasing strategy is proposed instead of the standard Linear updating strategy (Yang et al., 2015). Flames no at each iteration is defined as in Eq.9 where α is suggested by the authors to be $\alpha = 1/\pi^2$.

$$\text{Flames no} = round\left(N - \left(\frac{l}{T}\right)^{\alpha} \times \left(N-1\right)\right) \tag{9}$$

- Flames number with Decreasing strategy.

  This strategy uses a nonlinear formula to decrease the flames number over the iterations (Fan and Chiu, 2007). Flames no at each iteration is defined as in Eq.10.

$$\text{Flames no} = round\left(\frac{2}{l}\right)^{0.3} \tag{10}$$

- Flames number with Oscillating strategy.

  The Oscillating strategy uses a sinusoidal function to update flames number during the search process instead of monotonically decreasing it (Kentzoglanakis and Poole, 2009). The flames number at each iteration is defined as in Eq.11 where k is set by the authors to 7.

$$\text{Flames no} = round\left(N - cos\left(\frac{2\pi l(4k+6)}{T}\right) \times \left(N-1\right)\right) \tag{11}$$

- Flames number with Logarithmic strategy.

  The Logarithmic decreasing strategy (Gao et al., 2008) to update the flames number is shown in Eq.12 where a is a constant that set to 1 by the authors.

$$\text{Flames no} = round\left(N - \left(log_{10}\left(a + \frac{10l}{T}\right) \times \left(N-1\right)\right)\right) \tag{12}$$

4 EXPERIMENTAL SETUP AND RESULTS

In this paper, seventeen medical data sets were used to evaluate the proposed wrapper approaches. The datasets were downloaded from well-regarded data repositories: UCI, Kaggle and Keel. Table 1 lists these data sets and shows their number of features, instances, and classes. Table 2 shows the parameters setting of three well known meta-heuristic algorithms: BGWO, BCS and BBA. These wrapper based approaches were used to validate the proposed approaches by conducting fair comparisons between them. Therefore, all the experiments were executed on a personal machine with AMD Athlon Dual-Core QL-60 CPU at 1.90 GHz and memory of 2 GB running Windows7 Ultimate 64 bit operating system. The optimization algorithms have been all implemented in Python in the EvoloPy-FS framework (Khurma et al., 2020). The maximum number of iterations and the population size were set to 100 and 10 respectively. In this work, the K-NN classifier (K = 5(Aljarah et al., 2018a)) is used to assess the goodness of each solution in the wrapper FS approach. Each data set is randomly divided into two parts; 66% for training and 34% for testing. To obtain statistically significant results, this division was repeated 30 independent times. Therefore, the final statistical results were obtained over 30 independent runs. The α and β parameters in the fitness equation is set to 0.99 and 0.01, respectively (Mafarja et al., 2020). The used evaluation measures are fitness values (see Eq. 13), classification accuracy (see Eq. 14), number of selected features (see Eq. 15) and CPU time (see Eq. 16).

$$\frac{1}{M}\sum_{j=1}^{M} Fit^{i} \tag{13}$$

where M is the number of runs, Fit^i is the fitness value of the best solution in the i_{th} run.

$$\frac{1}{M}\sum_{j=1}^{M}\frac{1}{N}\sum_{i=1}^{N}(C_i = L_i) \tag{14}$$

where M is the number of runs for MFO to find the optimal subset of features, N is the number of data set instances, C_i is the predictive class, and L_i is the actual class in the labeled data.

$$\frac{1}{M}\sum_{i=1}^{M}\frac{d_i}{D} \tag{15}$$

where M is the number of runs, d_i is the number of selected features in the best solution from the $i_t h$ run, and D is the total number of features in the original dataset.

$$\frac{1}{M}\sum_{i=1}^{M} RT_i \tag{16}$$

where M is the number of runs, RT_i is the running time required for the i_{th} run to be completed.

In this section, the impact of the number of flames parameter using different time-varying update strategies is studied. Furthermore, a comparison is conducted between these strategies and three well-regarded wrapper methods (e.g, BGWO, BCS and BBA). The best results in tables are highlighted in bold.

Considering Table 3, it can be seen that using BMFO with a Linear strategy outperformed other approaches in 35% of the data sets, followed by the Decreasing strategy that got the best results in four data sets. The Decreasing strategy is a variant of the Non-decreasing strategy that reduces the number of flames during the optimization process based on the current iteration. It doesn't include the maximum number of iterations. Other strategies reduce the number of flames using the ratio of the current iteration (l) to the maximum number of iteration (T). Logarithm came third and Oscillating and Non-linear strategies came last. Inspecting Table 7 and Fig 2, it can be seen, in general, that the Linear-based BMFO approach outperformed BGOW, BCS, and BBA in terms of classification accuracy across 41% of the data sets. BGOW, BCS, and BBA obtained the best result in five, three and two data sets respectively.

Because FS is a minimization optimization problem, the target is to achieve a minimal fitness value. In other words, FS is seeking to determine the most informative features that reveal the highest classification accuracy. By comparing the BMFO strategies together in terms of fitness values, it can be seen in Table 4 that the Linear strategy was the best on six data sets then came the Logarithm and Oscillating strategies which were the best on five and four data sets respectively. Following them are the Nonlinear and Decreasing approaches which were the best only on two and one data sets respectively. From Table 8 and Fig 3, it appears that the best fitness values were obtained by the BCS which outperformed the others across 53% of data sets. The Linear-based BMFO came next and outperformed other approaches across 29% of the data sets. The decline in the fitness results of BMFO compared to BCS can be explained that they were not superior to BCS either in minimizing the number of selected features or maximizing the classification accuracy. This is because the fitness function combines the error rate and the size of the feature subset. Since Linear BMFO is superior to other methods in terms of classification accuracy, then it was not superior in terms of the number of selected features. Fig 4 illustrates the convergence behavior of all the studied wrapper approaches on all data sets. Each subfigure shows the changes in fitness value for each approach across all iterations on a specified data set. In all data sets, BMFO based approaches and BCS show the best convergence trends. This can be realized from the convergence curves that achieve the minimum fitness values in the final iterations of the optimization process. On the other hand, premature convergence and entrapment in local minima can be guessed from the convergence behavior of BGWO and BBA.

When examining the results of the number of selected features recorded in Table 5, the Decreasing-based BMFO was ranked first with the best results reported across 41% of the data sets. However, Table 9 shows that the BBA was the best approach to reduce the dimensionality of the medical data sets. BBA was superior to other approaches over 59% of the data sets.

In this section, the running time is carefully analyzed because this is a serious factor especially when the data sets become larger. In Table 6, the Decreasing-based BMFO achieved the smallest running time on eight data sets. Oscillating and linear approaches came next, and they achieved the minimum running time on five and three data sets respectively. Logarithm achieved the minimum running time over the SPECTF Heart data set only. The Non-linear approach was not the best across any of the data sets. From Table 10, It can be observed that the Decreasing BMFO approach achieved the shortest running time over twelve data sets. The BBA approach achieved the minim running time over five data sets while BGWO and BCS were not the best over any data sets.

By taking all the results together, we conclude that the update strategy of the number of flames affects the robustness and the convergence of the BMFO algorithm. In the proposed approaches, five strategies were employed to update the number of flames in the BMFO algorithm. The Linear strategy was able to outperform other strategies in terms of classification accuracy and fitness values, while the Decreasing strategy obtained the best results in terms of the number of selected features and average running time. Here, we can remark that the enhanced performance of the proposed approach is because the Linear strategy enhanced the ability of the BMFO algorithm to balance the exploration and exploitation. This is done by gradually reducing the number of flames throughout iterations. Also, allowing more exploration in the early iterations, while exploitation is emphasized in the later iterations. The results showed the competitive performance of the proposed BMFO approaches when compared to other well-regarded approaches.

Table 1: Description of the used data sets.

NO	Dataset Name	No features	No instances	No classes
1	Diagnostic	30	569	2
2	Original	9	699	2
3	Coimbra	9	115	2
4	BreastEW	30	596	2
5	Dermatology	34	366	6
6	Liver	10	583	2
7	Lymphography	18	148	4
8	Parkinsons	22	194	2
9	SPECT	22	267	2
10	HeartEW	13	270	2
11	Hepatitis	18	79	2
12	SAHeart	9	461	2
13	SPECTF	43	266	2
14	Heart	13	302	5
15	Diabetes	9	768	2
16	Colon	2000	62	2
17	Leukemia	7129	72	2

Table 2: Parameter settings.

Algorithm	Parameter	Value
GWO	α	[2,0]
BA	Qmin Frequency minimum	0
	Qmax Frequency maximum	2
	A Loudness	0.5
	r Pulse rate	0.5
CS	pa	0.25
	β	3/2

Table 3: Average classification accuracy from 30 runs for all approaches.

NO	Dataset Name	Linear	Non-linear coefficient	Decreasing	Oscillating	Logarithm
1	Diagnostic	**0.907**	0.906	0.906	0.904	**0.907**
2	Original	0.969	0.970	**0.971**	0.970	0.964
3	Coimbra	0.775	0.775	0.775	0.775	0.775
4	BreastEW	**0.935**	0.932	0.933	0.932	0.932
5	Dermatology	**0.972**	0.970	0.970	**0.972**	0.970
6	Liver	**0.774**	0.652	0.653	0.653	0.653
7	Lymphography	0.673	0.667	0.664	0.667	**0.675**
8	Parkinsons	0.776	0.776	0.776	0.776	0.776
9	SPECT	0.657	0.657	**0.675**	0.658	0.656
10	HeartEW	**0.788**	0.779	0.781	0.786	0.778
11	Hepatitis	0.824	0.821	0.818	0.815	**0.844**
12	SAHeart	0.627	0.627	0.627	0.627	0.627
13	SPECTF	0.768	0.764	**0.771**	0.766	0.764
14	Heart	0.741	**0.759**	0.742	0.754	0.758
15	Diabetes	0.725	**0.726**	**0.726**	0.725	0.725
16	Colon	**0.652**	0.645	0.644	0.638	0.632
17	Leukemia	0.852	0.845	0.849	**0.859**	0.849

Table 4: Average fitness values from 30 runs for all approaches.

NO	Dataset Name	Linear	Non-linear coefficient	Decreasing	Oscillating	Logarithm
1	Diagnostic	0.038	0.041	0.040	0.050	**0.037**
2	Original	0.049	0.049	**0.046**	0.048	0.048
3	Coimbra	0.287	0.287	0.287	0.287	0.287
4	BreastEW	**0.071**	0.072	0.075	0.075	0.075
5	Dermatology	**0.061**	0.062	0.065	0.066	0.066
6	Liver	0.348	**0.345**	**0.345**	**0.345**	**0.345**
7	Lymphography	**0.405**	0.419	0.423	0.429	0.431
8	Parkinsons	0.348	0.348	0.348	0.348	0.348
9	SPECT	0.371	**0.356**	0.362	0.372	0.366
10	HeartEW	0.196	0.185	0.184	0.190	**0.176**
11	Hepatitis	0.202	0.202	0.202	0.202	0.202
12	SAHeart	**0.406**	0.408	0.408	0.408	0.408
13	SPECTF Heart	0.303	0.301	**0.270**	0.292	0.296
14	Heart	0.243	0.214	0.240	**0.204**	0.209
15	Diabetes	**0.258**	0.260	0.260	**0.258**	**0.258**
16	Colon	0.377	0.377	0.377	0.377	0.377
17	Leukemia	**0.115**	**0.115**	0.119	**0.115**	**0.115**

Table 5: Average number of selected features from 30 runs for all approaches.

NO	Dataset Name	Linear	Non-linear coefficient	Decreasing	Oscillating	Logarithm
1	Diagnostic	16.500	15.467	**15.367**	16.167	16.533
2	Original	**4.400**	4.567	4.733	4.600	4.333
3	Coimbra	**3.667**	4.000	4.000	4.033	4.033
4	BreastEW	**19.433**	19.700	19.500	19.733	20.233
5	Dermatology	22.167	22.167	21.600	**21.567**	22.567
6	Liver	3.967	4.000	4.000	4.000	**2.467**
7	Lymphography	9.367	9.233	**8.733**	9.067	9.333
8	Parkinsons	8.233	8.067	**7.967**	8.367	8.724
9	SPECT	14.167	**13.333**	13.733	14.300	13.867
10	HeartEW	7.767	7.433	**7.167**	7.667	7.333
11	Hepatitis	7.800	7.733	**7.167**	8.067	7.900
12	SAHeart	4.033	**4.000**	**4.000**	**4.000**	**4.000**
13	SPECTF	29.500	28.500	**27.067**	28.933	29.267
14	Heart	5.700	**5.400**	5.567	5.833	5.667
15	Diabetes	6.500	4.033	4.033	**4.000**	**4.000**
16	Colon	**1082.067**	1116.033	1150.433	1152.233	1115.657
17	Leukemia	3506.367	3510.667	3497.700	**3491.767**	3492.933

Table 6: Average computational time from 30 runs for all approaches.

NO	Dataset Name	Linear	Non-linear coefficient	Decreasing	Oscillating	Logarithm
1	Diagnostic	21.476	21.429	**20.804**	20.892	21.317
2	Original	23.241	23.362	**22.498**	22.566	23.321
3	Coimbra	7.918	8.021	7.912	**7.884**	8.054
4	BreastEW	23.493	23.644	22.984	**22.922**	23.677
5	Dermatology	16.038	16.090	15.785	**15.755**	16.126
6	Liver	13.724	13.875	13.439	**13.423**	13.668
7	Lymphography	9.146	9.0655	**8.820**	9.113	9.698
8	Parkinsons	10.572	10.637	**10.387**	10.410	10.525
9	SPECT	**12.418**	12.682	12.482	12.482	16.958
10	HeartEW	**11.848**	12.313	11.954	11.990	12.154
11	Hepatitis	7.423	7.423	**7.322**	7.345	7.377
12	SAHeart	**16.354**	17.100	16.519	16.562	16.958
13	SPECTF	13.866	14.204	13.837	13.874	**13.597**
14	Heart	12.165	12.075	**11.867**	11.892	12.012
15	Diabetes	24.727	24.937	24.224	**24.216**	24.894
16	Colon	76.932	73.013	**70.149**	75.816	73.726
17	Leukemia	265.892	260.067	**251.403**	266.039	264.390

Table 7: Comparison of Linear based MFO with other algorithms based on the average accuracy values.

NO	Dataset Name	Linear	BGWO	BCS	BBA
1	Diagnostic	**0.907**	0.904	0.896	0.760
2	Original	0.969	0.963	**0.971**	0.939
3	Coimbra	0.775	0.707	**0.777**	0.583
4	BreastEW	**0.935**	0.925	0.923	0.896
5	Dermatology	**0.972**	0.956	0.961	0.739
6	Liver	**0.774**	0.591	0.653	0.582
7	Lymphography	0.673	0.682	0.690	**0.705**
8	Parkinsons	0.776	**0.799**	0.776	0.786
9	SPECT	**0.657**	0.656	0.653	0.624
10	HeartEW	**0.788**	0.770	0.778	0.709
11	Hepatitis	0.824	0.805	0.826	**0.838**
12	SAHeart	0.627	**0.647**	0.627	0.622
13	SPECTF	**0.768**	0.767	0.760	0.748
14	Heart	0.741	0.723	**0.769**	0.645
15	Diabetes	0.725	**0.750**	0.725	0.664
16	Colon	0.652	**0.655**	0.632	0.645
17	Leukemia	0.852	**0.856**	0.853	0.715

Table 8: Comparison of Linear based MFO with other algorithms based on the average fitness values.

NO	Dataset Name	Linear	BGWO	BCS	BBA
1	Diagnostic	**0.038**	0.074	0.085	0.322
2	Original	0.049	0.069	**0.042**	0.079
3	Coimbra	0.287	**0.276**	0.287	0.557
4	BreastEW	0.071	0.075	**0.068**	0.093
5	Dermatology	**0.061**	0.090	0.070	0.436
6	Liver	0.348	0.407	**0.345**	0.414
7	Lymphography	**0.405**	**0.405**	0.433	0.431
8	Parkinsons	0.348	**0.329**	0.347	0.366
9	SPECT	0.371	0.358	0.374	**0.351**
10	HeartEW	0.196	0.194	**0.154**	0.290
11	Hepatitis	0.202	0.202	**0.201**	0.204
12	SAHeart	0.406	0.400	0.404	**0.371**
13	SPECTF	0.303	0.276	**0.264**	0.272
14	Heart	0.243	0.278	**0.182**	0.456
15	Diabetes	**0.258**	0.261	**0.258**	0.386
16	Colon	0.377	**0.376**	**0.376**	**0.376**
17	Leukemia	**0.115**	**0.115**	0.118	0.328

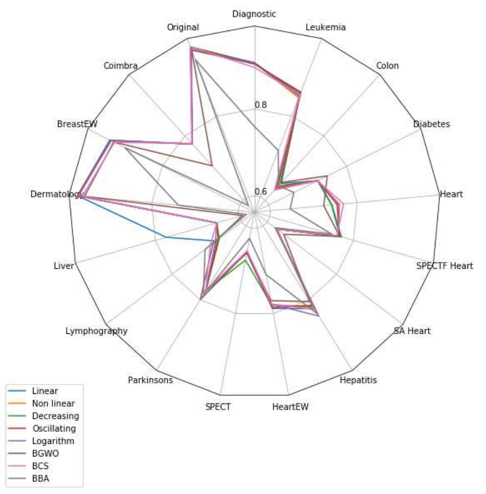


Figure 2: Comparison of time-variant BMFO approaches with other optimizers in terms of the classification accuracy.

Table 9: Comparison of Decreasing based MFO with other algorithms based on the average number of selected Features.

NO	Dataset Name	Decreasing	BGWO	BCS	BBA
1	Diagnostic	15.367	14.733	**11.533**	14.533
2	Original	4.733	5.667	3.933	**3.633**
3	Coimbra	**4.000**	4.333	4.133	4.600
4	BreastEW	19.500	15.600	14.933	**12.500**
5	Dermatology	21.600	17.267	16.667	**14.367**
6	Liver	4.000	3.533	4.000	**2.467**
7	Lymphography	8.733	9.767	7.000	**6.600**
8	Parkinsons	7.967	9.333	**5.800**	10.400
9	SPECT	13.733	10.667	10.400	**9.000**
10	HeartEW	7.167	7.100	6.200	**5.467**
11	Hepatitis	7.167	6.467	**5.300**	10.300
12	SAHeart	**4.000**	4.433	4.100	4.167
13	SPECTF	27.067	22.033	19.933	**17.900**
14	Heart	5.567	**4.767**	5.467	5.967
15	Diabetes	4.033	4.033	4.033	**3.767**
16	Colon	1150.433	970.867	**961.867**	974.233
17	Leukemia	3497.700	3495.567	3422.367	**1241.767**

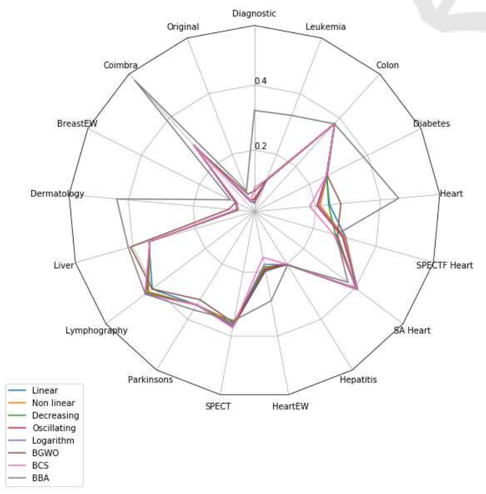


Figure 3: Comparison of time-variant BMFO approaches with other optimizers in terms of fitness values.

Table 10: Comparison of Decreasing based MFO with other algorithms based on the average computational Time.

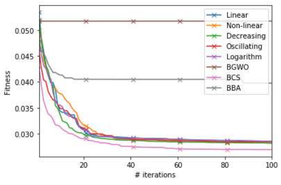
NO	Dataset Name	Decreasing	BGWO	BCS	BBA
1	Diagnostic	**20.804**	23.426	41.084	21.125
2	Original	**22.498**	24.175	45.724	23.015
3	Coimbra	**7.912**	8.690	15.852	8.274
4	BreastEW	22.984	24.955	43.112	**22.147**
5	Dermatology	15.785	18.405	30.140	**15.607**
6	Liver	**13.439**	14.249	27.131	14.001
7	Lymphography	**8.820**	10.775	17.249	9.206
8	Parkinsons	**10.387**	13.064	20.188	10.628
9	SPECT	**12.482**	14.483	24.256	12.674
10	HeartEW	**11.954**	13.625	23.590	12.395
11	Hepatitis	**7.322**	9.331	14.065	7.556
12	SAHeart	**16.519**	17.198	33.451	17.170
13	SPECTF	13.837	18.002	25.759	**13.597**
14	Heart	**11.867**	13.562	23.625	12.324
15	Diabetes	**24.224**	25.043	49.313	24.964
16	Colon	70.149	274.127	73.273	**55.723**
17	Leukemia	251.403	964.913	245.912	**189.204**

5 CONCLUSIONS

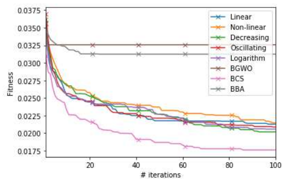
This work presents a new wrapper based FS approach based on an enhanced MFO algorithm. The enhancement step is based on adopting the time-varying functions for updating the number of flames. This parameter plays an important role in controlling the transition between exploration and exploitation phases during the optimization process. Achieving a good balance between exploration and exploitation will positively affect the performance of MFO and contributes to improved classification performance. In this paper, the impact of five different updating strategies for flames number is investigated. For evaluation purposes, seventeen medical benchmark data sets are used. The results show that the Linear update strategy that gradually decreases the flames number can improve the exploration and exploitation of the BMFO for FS tasks. The reported results show that BMFO methods achieve competitive results in comparison with other well-regarded wrapper-based approaches in terms of the classification accuracy, the fitness value, and the running time.

ACKNOWLEDGMENTS

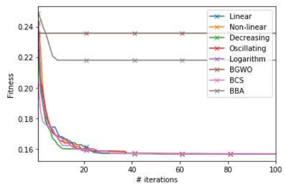
This work is supported by the Ministerio español de Economía y Competitividad under project TIN2017-85727-C4-2-P (UGR-DeepBio).



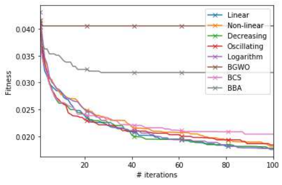
(a) Diagnostic



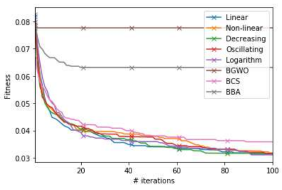
(b) Original



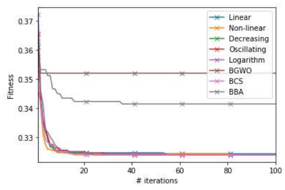
(c) Coimbra



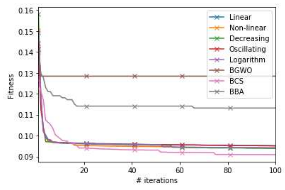
(d) BreastEW



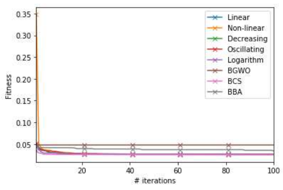
(e) Dermatology



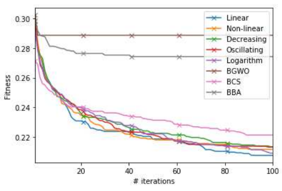
(f) Liver



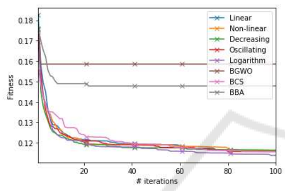
(g) Lymph



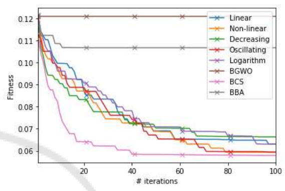
(h) Parkinsons



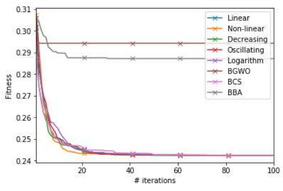
(i) SPECT



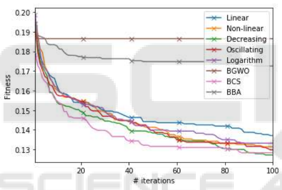
(j) HeartEW



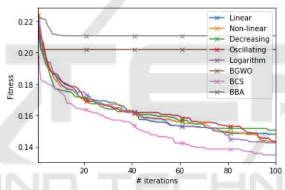
(k) Hepatitis



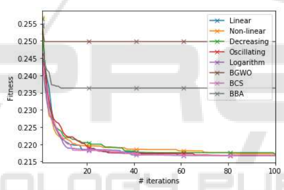
(l) SAHeart



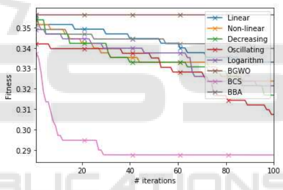
(m) SPECTF



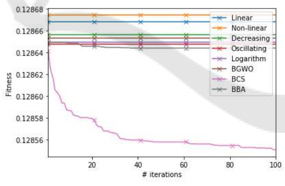
(n) Heart



(o) Diabetes



(p) Colon



(q) Leukemia

Figure 4: Convergence curves for the time-variant BMFO and other wrapper-based approaches on the used medical data sets.

REFERENCES

Al-Madi, N., Faris, H., and Abukhurma, R. (2018). Cost-sensitive genetic programming for churn prediction and identification of the influencing factors in telecommunication market. *International Journal of Advanced Science and Technology*, pages 13–28.

Aljarah, I., AlaḾ, A.-Z., Faris, H., Hassonah, M. A., Mirjalili, S., and Saadeh, H. (2018a). Simultaneous feature selection and support vector machine optimization using the grasshopper optimization algorithm. *Cognitive Computation*, pages 1–18.

Aljarah, I., Mafarja, M., Heidari, A. A., Faris, H., Zhang, Y., and Mirjalili, S. (2018b). Asynchronous accelerating multi-leader salp chains for feature selection. *Applied Soft Computing*, 71:964–979.

Brezočnik, L., Fister, I., and Podgorelec, V. (2018). Swarm intelligence algorithms for feature selection: A review. *Applied Sciences*, 8(9):1521.

Ewees, A. A., Sahlol, A. T., and Amasha, M. A. (2017). A bio-inspired moth-flame optimization algorithm for arabic handwritten letter recognition. In *Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO), 2017 International Conference on*, pages

154–159. IEEE.

Fan, S.-K. S. and Chiu, Y.-Y. (2007). A decreasing inertia weight particle swarm optimizer. *Engineering Optimization*, 39(2):203–228.

Faris, H., Ala'M, A.-Z., Heidari, A. A., Aljarah, I., Mafarja, M., Hassonah, M. A., and Fujita, H. (2019). An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks. *Information Fusion*, 48:67–83.

Gao, Y.-l., An, X.-h., and Liu, J.-m. (2008). A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation. In *2008 International Conference on Computational Intelligence and Security*, volume 1, pages 61–65. IEEE.

Hassanien, A. E., Gaber, T., Mokhtar, U., and Hefny, H. (2017). An improved moth flame optimization algorithm based on rough sets for tomato diseases detection. *Computers and Electronics in Agriculture*, 136:86–96.

Inbarani, H. H., Azar, A. T., and Jothi, G. (2014). Supervised hybrid feature selection based on pso and rough sets for medical diagnosis. *Computer methods and programs in biomedicine*, 113(1):175–185.

Kennedy, J. and Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, volume 5, pages 4104–4108. IEEE.

Kentzoglanakis, K. and Poole, M. (2009). Particle swarm optimization with an oscillating inertia weight. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 1749–1750.

Khurma., R. A., Aljarah., I., and Sharieh., A. (2020). An efficient moth flame optimization algorithm using chaotic maps for feature selection in the medical applications. In *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM,*, pages 175–182. INSTICC, SciTePress.

Khurma, R. A., Aljarah, I., Sharieh, A., and Mirjalili, S. (2020). Evolopy-fs: An open-source nature-inspired optimization framework in python for feature selection. In *Evolutionary Machine Learning Techniques*, pages 131–173. Springer.

Khushaba, R. N., Al-Ani, A., and Al-Jumaily, A. (2011). Feature subset selection using differential evolution and a statistical repair mechanism. *Expert Systems with Applications*, 38(9):11515–11526.

Mafarja, M., Aljarah, I., Heidari, A. A., Hammouri, A. I., Faris, H., AlaḾ, A.-Z., and Mirjalili, S. (2018). Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems. *Knowledge-Based Systems*, 145:25–45.

Mafarja, M., Qasem, A., Heidari, A. A., Aljarah, I., Faris, H., and Mirjalili, S. (2020). Efficient hybrid nature-inspired binary optimizers for feature selection. *Cognitive Computation*, 12(1):150–175.

Mafarja, M. M., Eleyan, D., Jaber, I., Hammouri, A., and Mirjalili, S. (2017). Binary dragonfly algorithm for feature selection. In *New Trends in Computing Sciences (ICTCS), 2017 International Conference on*, pages 12–17. IEEE.

Medjahed, S. A., Saadi, T. A., Benyettou, A., and Ouali, M. (2016). Gray wolf optimizer for hyperspectral band selection. *Applied Soft Computing*, 40:178–186.

Mehne, S. H. H. and Mirjalili, S. (2020). Moth-flame optimization algorithm: theory, literature review, and application in optimal nonlinear feedback control design. In *Nature-Inspired Optimizers*, pages 143–166. Springer.

Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, 89:228–249.

Mirjalili, S. and Lewis, A. (2013). S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation*, 9:1–14.

Mistry, K., Zhang, L., Neoh, S. C., Lim, C. P., and Fielding, B. (2017). A micro-ga embedded pso feature selection approach to intelligent facial emotion recognition. *IEEE transactions on cybernetics*, 47(6):1496–1509.

Nakamura, R. Y., Pereira, L. A., Costa, K., Rodrigues, D., Papa, J. P., and Yang, X.-S. (2012). Bba: a binary bat algorithm for feature selection. In *2012 25th SIBGRAPI conference on graphics, Patterns and Images*, pages 291–297. IEEE.

Oliveira, A. L., Braga, P. L., Lima, R. M., and Cornélio, M. L. (2010). Ga-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation. *information and Software Technology*, 52(11):1155–1166.

Reddy, S., Panwar, L. K., Panigrahi, B. K., and Kumar, R. (2018). Solution to unit commitment in power system operation planning using binary coded modified moth flame optimization algorithm (bmmfoa): a flame selection based computational technique. *Journal of Computational Science*, 25:298–317.

Rodrigues, D., Pereira, L. A., Almeida, T., Papa, J. P., Souza, A., Ramos, C. C., and Yang, X.-S. (2013). Bcs: A binary cuckoo search algorithm for feature selection. In *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*, pages 465–468. IEEE.

Sayed, G. I. and Hassanien, A. E. (2018). A hybrid samfo algorithm for function optimization and engineering design problems. *Complex & Intelligent Systems*, pages 1–18.

Sayed, G. I., Hassanien, A. E., Nassef, T. M., and Pan, J.-S. (2016). Alzheimerś disease diagnosis based on moth flame optimization. In *International Conference on Genetic and Evolutionary Computing*, pages 298–305. Springer.

Shehab, M., Abualigah, L., Al Hamad, H., Alabool, H., Alshinwan, M., and Khasawneh, A. M. (2019). Moth–flame optimization algorithm: variants and applications. *Neural Computing and Applications*, pages 1–26.

Shi, Y. and Eberhart, R. C. (1999). Empirical study of particle swarm optimization. In *Proceedings of the 1999* *congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*, volume 3, pages 1945–1950. IEEE.

Tang, J., Alelyani, S., and Liu, H. (2014). Feature selection for classification: A review. *Data classification: Algorithms and applications*, page 37.

Tubishat, M., Abushariah, M. A., Idris, N., and Aljarah, I. (2018). Improved whale optimization algorithm for feature selection in arabic sentiment analysis. *Applied Intelligence*, pages 1–20.

Wang, G.-G. and Tan, Y. (2017). Improving metaheuristic algorithms with information feedback models. *IEEE Transactions on Cybernetics*.

Wang, M., Chen, H., Yang, B., Zhao, X., Hu, L., Cai, Z., Huang, H., and Tong, C. (2017). Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses. *Neurocomputing*, 267:69–84.

Yang, C., Gao, W., Liu, N., and Song, C. (2015). Low-discrepancy sequence initialized particle swarm optimization algorithm with high-order nonlinear time-varying inertia weight. *Applied Soft Computing*, 29:386–394.

Yang, C.-H., Chuang, L.-Y., Yang, C. H., et al. (2010). Igga: a hybrid filter/wrapper method for feature selection of microarray data. *Journal of Medical and Biological Engineering*, 30(1):23–28.

Zawbaa, H. M., Emary, E., Grosan, C., and Snasel, V. (2018). Large-dimensionality small-instance set feature selection: A hybrid bio-inspired heuristic approach. *Swarm and Evolutionary Computation*, 42:29–42.

Zhang, L., Mistry, K., Neoh, S. C., and Lim, C. P. (2016). Intelligent facial emotion recognition using moth-firefly optimization. *Knowledge-Based Systems*, 111:248–267.