IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 27, 2020, accepted October 20, 2020, date of publication October 26, 2020, date of current version November 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3033757

Dynamic Butterfly Optimization Algorithm for Feature Selection

MOHAMMAD TUBISHAT[1], MOHAMMED ALSWAITTI[2], SEYEDALI MIRJALILI[3], (Senior Member, IEEE), MOHAMMED ALI AL-GARADI[4], MA'EN TAYSEER ALRASHDAN[1], AND TOQIR A. RANA[5]

[1]School of Technology and Computing, Asia Pacific University of Technology and Innovation, Kuala Lumpur 57000, Malaysia
[2]School of Electrical and Computer Engineering (ICT), Xiamen University Malaysia, Bandar Sunsuria 43900, Malaysia
[3]Centre for Artificial Intelligence Research and Optimization, Torrens University Australia, Fortitude Valley, QLD 4006, Australia
[4]Department of Radiology, University of California at San Diego, La Jolla, CA 92093, USA
[5]Department of Computer Science and IT, The University of Lahore, Lahore 54590, Pakistan

Corresponding author: Mohammed Alswaitti (alswaitti.mohammed@xmu.edu.my)

This work was supported by the Research Management Center, Xiamen University Malaysia (XMUM), through the XMUM Research Program Cycle 4 under Grant XMUMRF/2019-C4/IECE/0012.

ABSTRACT Feature selection represents an essential pre-processing step for a wide range of Machine Learning approaches. Datasets typically contain irrelevant features that may negatively affect the classifier performance. A feature selector can reduce the number of these features and maximise the classifier accuracy. This paper proposes a Dynamic Butterfly Optimization Algorithm (DBOA) as an improved variant to Butterfly Optimization Algorithm (BOA) for feature selection problems. BOA represents one of the most recently proposed optimization algorithms. BOA has demonstrated its ability to solve different types of problems with competitive results compared to other optimization algorithms. However, the original BOA algorithm has problems when optimising high-dimensional problems. Such issues include stagnation into local optima and lacking solutions diversity during the optimization process. To alleviate these weaknesses of the original BOA, two significant improvements are introduced in the original BOA: the development of a Local Search Algorithm Based on Mutation (LSAM) operator to avoid local optima problem and the use of LSAM to improve BOA solutions diversity. To demonstrate the efficiency and superiority of the proposed DBOA algorithm, 20 benchmark datasets from the UCI repository are employed. The classification accuracy, the fitness values, the number of selected features, the statistical results, and convergence curves are reported for DBOA and its competing algorithms. These results demonstrate that DBOA significantly outperforms the comparative algorithms on the majority of the used performance metrics.

INDEX TERMS Butterfly optimization algorithm, feature selection, local search algorithm based on mutation.

I. INTRODUCTION

In the last years, feature selection received more attention from researchers due to its ability to improve the performance of machine learning classifiers and to decrease the problem dimensionality. Feature selection can remove irrelevant, redundant, and noisy features from the dataset while retaining the most informative and relevant features. Feature selection is a crucial step in many applications such as sentiment analysis, cancer detection, medical applications, image classification, intrusion detection, software fault prediction, text classification, spam classification, and many more. In these applications, the use of all features which are included in the dataset will result in performance degradation and increase the computational cost [1].

Generally speaking, there are two main types of feature selection methods: filter-based methods and wrapper-based methods. Chi-Square (Chi) and Information Gain (IG) are some examples of filter-based methods [2]. On the other hand, the wrapper-based methods mainly work based on using an optimization algorithm to select the optimal subset of features. In filter-based feature selection, the features are ranked before using a machine classifier, which means that there is no direct contact between the used filter feature selection method and the used classifier. On the other hand, in the wrapper-based method, features are selected by an optimization algorithm and the used classifier. Therefore, in wrapper mode, there is a direct interaction between the features and

The associate editor coordinating the review of this manuscript and approving it for publication was Md. Abdur Razzaque.

the used classifier. In the wrapper-based method, a fitness function will be used by the optimization algorithm to find the quality of the selected features taking into account the achieved classification accuracy and the number of selected features [2].

In the literature, there are several studies that applied optimization algorithms for feature selection problem. For example, [3], [4] improved whale optimization algorithm (WOA) and used it for feature selection, [5] improved particle swarm optimization (PSO) and used it for feature selection, [6] used Grasshopper optimization algorithm (GOA) for feature selection, [7] used Firefly algorithm (FFA) for feature selection, [8] used Differential Evolution (DE) for feature selection, [1] improved Salp Swarm Algorithm (SSA)and used it for feature selection, [9] used Genetic Algorithm (GA) for feature selection, [10] used Grey Wolf Optimization (GWO) for feature selection, [11] improved Gravitational Search algorithm (GSA) and used it for feature selection, [12] used fish swarm optimization (FSO) for feature selection, [13] improved crow search algorithm (CSA) and used it for feature selection, [14] improved Dragonfly Algorithm (DA) and used it for feature selection, [15] improved social spider algorithm (SSA) and used it for feature selection, [1], [16] improved Salp Swarm Algorithm (SSA) and used it for feature selection, [2] improved Harris hawks optimization (HHO) and used it for feature selection.

However, based on the random nature of optimization algorithms, there is no single optimization algorithm that is superior to all other optimization algorithms. In other words, one optimization algorithm can be superior over some datasets or specific types of problems but not all datasets or problems. Also, based on No-Free-Lunch (NFL) theorem, there is no one specific optimiser that is adequate to solve all types of problems, and at the same time, it outperforms all other algorithms. Therefore, based on the NFL theorem, there are many available opportunities to develop new optimization algorithms or improve the available algorithms, where these algorithms can be superior to the available algorithms [17]. Thus, the NFL theorem motivated this study to use Butterfly Optimization Algorithm (BOA) for feature selection problems. BOA represents one of the recently proposed optimization algorithms which mimics the butterflies' behaviour for food foraging [18]. Besides, BOA has been applied to different types of problems and proved its efficiency and performance compared to other state-of-the-art optimization algorithms.

The authors in [18] showed that BOA could be superior to other modern and well-known optimization algorithms on a set of benchmark functions and classical engineering problems. However, in high dimensional problems as an FS problem, the BOA algorithm may be stuck in local optima. Also, it has a solutions diversity problem. Therefore, in this paper, we propose a new DBOA for the FS problem, which works in a wrapper mode. This new algorithm encapsulates two major improvements to the original BOA algorithm to solve its weakness. Based on included improvements in DBOA, it can avoid being stuck into local optima and improve the diversity of the solutions. The main contributions of this study are summarised as follows:

1) To Introduce DBOA novel algorithm for FS in wrapper mode, where the new DBOA represents an improved version of the original BOA.
2) The main improvement to the BOA algorithm includes the development of a new LSAM algorithm, based on the mutation operator. LSAM works in two ways, including 1) improving the current best solution to solve local optima problem; 2) Selecting random solutions from other solutions and improve the selected solutions to enhance the diversity of the solutions.
3) The performance of the proposed DBOA is tested and evaluated on 20 benchmark datasets from the UCI repository.
4) The performance of DBOA is compared with six well-known optimization algorithms. Based on the conducted experiments, it reveals the superiority and outperformance of DBOA in comparison to these algorithms. Also, the performance of DBOA is compared with three variants of BOA, which were applied for the feature selection problem.

The rest of the paper is structured as follows: Section II presents related works. Section III presents the details of the BOA algorithm, LSAM algorithm and describes the details of the proposed DOBA algorithm. The conducted experiments and discussion are presented in Section IV. Finally, Section V includes the paper conclusion

II. RELATED WORK

In literature, different types of optimization algorithms or its improvements (variants) have been proposed for solving the FS problem in wrapper mode. For example, [14], [19] improved Dragonfly Algorithm (DA) for the FS problem. Reference [20] used Ant Colony Optimization (ACO) for the FS problem. Reference [21] used the Genetic Algorithm(GA) for the FS problem. References [4], [22] improved Whale Optimization Algorithm (WOA) for the FS problem. Reference [23] improved Ant Lion Optimizer for the FS problem. Reference [24] used Harris hawks optimiser (HHO) for the FS problem [25] hybridised Grey Wolf Optimizer (GWO) with WOA for the FS problem. [1], [26]–[28] improved Salp Swarm Algorithm (SSA) for the FS problem [15] used Social Spider Algorithm (SSA) for the FS problem. Reference [6] used the Grasshopper optimization algorithm (GOA) for the FS problem. Reference [29] improved Crow Search Algorithm (CSA). Reference [30] improved the Firefly Algorithm (FA) for the FS problem and many more studies.

However, based on the NFL theorem, many opportunities are still available to develop or improve variants of the existing optimization algorithms and use it for the FS problem. Therefore, this has motivated our study to improve the recent BOA algorithm and use it for the FS problem. BOA algorithm was developed by [18], where the BOA idea originates from mimicking the food search and mating behaviour of butterflies in our nature. Furthermore, the effectiveness and

simplicity of BOA has attracted many studies and confirmed its efficiency and outperformance to solve different problems. For example, [31] used BOA for the power systems. Reference [32] used improved BOA by introducing the adaptive mechanism for updating the sensory value at each iteration in the original BOA and applied it for some benchmark functions.

The authors in [33] improved BOA by using learning automata and applied it to a number of benchmark functions and engineering design problems. Reference [34] improved BOA by using the mutualism mechanism of symbiosis organisms search (SOS) and applied it for four real-life problems. Reference [35] used BOA for vehicle designs. Reference [36] used BOA for finding the optimal biases and weights of artificial neural networks (ANN) and used it for Data Classification. Reference [37] used BOA for stock forecasting. Reference [38] used GA with BOA for the optimal load flow problem, by taking the results of BOA as an initial result for the GA algorithm. Reference [39] used BOA for power point tracking of photovoltaic systems.

The authors in [40] improved BOA using a logistic chaotic map and used it for the heat recovery system. Reference [41] used BOA for optimising the parameters of the control system. Reference [42] used the BOA for regression test suite optimization problem. Reference [43] improved BOA by using a chaotic Tent map for population initialisation and adaptive weight, then applied it for finding the optimal parameter settings for the least square support vector machine (LSSVM) and used the improved algorithm for the household CO2 emissions problem. Reference [44] improved BOA to be used in Capacitated Vehicle Routing Problem (CVRP), where they applied intra route operators of CVRP, including move operator, swap operator, and inversion operator as local search operators in BOA. Reference [45] improved BOA by using a cross-entropy method and applied it to benchmark functions and engineering design problems. Reference [46] improved BOA by using the inertia weight of the PSO algorithm and random population restart strategy for optimising some benchmark functions. Reference [47] proposed an unbiased version of BOA and applied it to some functions.

In comparison to the improved variants of BOA, which were applied to the FS problem, the authors in [48] proposed a binary version of the standard BOA algorithm using two transfer functions. These functions are used to map the continuous search space of BOA to a discrete one. They transformed the continuous butterflies' positions into their corresponding binary values, where the employed function is Sigmoidal (S-shaped) transfer function in one BOA version. Also, another binary variant of BOA was introduced using a V-shaped transfer function. Asa result, they concluded that the BOA binary version using S-shaped) transfer function achieved the best results. They called the improved algorithm Binary BOA (S-bBOA). Moreover, in work by [49], the main improvements which were introduced into BOA include 1) the use of DE strategy; 2) the new initialisation strategy that works by using a greedy strategy that will add or remove features from the solution after ranking of features importance; 3) the improvement of the transfer function by adding the butterfly fragrance to it to make it more adaptive; 4) the development of evolution population dynamics (EPD) mechanism. These improvements were used to reduce the randomness in both BOA local search process and initialisation to improve the balance between exploration and exploitation through the search process. The modifications were expected to improve the global search ability of BOA and the convergence speed of BOA. They called the improved algorithm as optimization and Extension of binary BOA (OEbBOA).

Further, in work by [50], authors combined BOA with chaotic maps to improve solution diversity and exploitation ability of BOA, and they called the improved BOA as Chaotic BOA (CBOA). The improvement in their work includes the replacement of r random variable in equations 2 and 3 with chaotic map values. However, there is a clear difference between these mentioned works and our proposed work. In our proposed DBOA, the main improvement includes developing a new LSAM algorithm based on the mutation operator. This LSAM algorithm will work in two ways using a mutation operator. The first part of the LSAM will improve BOA's exploitation by improving the current best solution, while the second part of LSAM will improve BOA solutions' diversity.

III. THE PROPOSED DYNAMIC BUTTERFLY OPTIMIZATION ALGORITHM (DBOA)

In this section, the BOA algorithm is first presented. Then, the DBOA algorithm is proposed.

A. THE ORIGINAL BUTTERFLY OPTIMIZATION ALGORITHM (BOA)

BOA is one of the newly metaheuristic optimization algorithms proposed by [18]. This algorithm mimics the food foraging and finding mating partner behaviours of butterflies. Butterflies chemoreceptors that scattered on their bodies and work as sense receptors. The butterflies use these chemoreceptors for sensing/smelling the fragrance of the flowers/food. Also, the chemoreceptor helps the butterfly to find the optimal mating partner. Butterflies generate a fragrance with an intensity level while it is changing its places. This fragrance guides the search-agents (butterflies) movement in the BOA algorithm. Based on the fragrance intensity, if a given butterfly fails to sense the fragrance of any butterfly within the search space, this butterfly will do exploitation (local search) by moving to a new random selected position. In case the butterfly senses the fragrance of the best butterfly, then it will move toward that butterfly, and this is called exploration (global search).

In the BOA algorithm, the fragrance is defined as a function of the stimulus intensity and can be determined by using equation (1) as follows:

$$f = cI^a, \tag{1}$$

where f is the fragrance magnitude emitted by a given butterfly, c is the sensory modality with its values over the range

```
- Initialize n butterflies population positions x_i (i = 1, 2, ..., n)
- Set the initial values of the parameters including switch probability ρ, c sensory
  modality, the power exponent a, and the maximum number of iterations known as
  max_iter
- while not reach max_iter do
    for each butterfly bf in the population do
        Find the fragrance value f of bf using Eq. (1)
    end for
    Find the best butterfly bf
    Assign the best butterfly to g*
    for each butterfly bf in the population do
        generate a random value r over the interval [0,1]
        if (r < p)
            Update bf position by Eq. (2) (Exploration)
        else
            Update bf position by Eq. (3) (Exploitation)
        end if
        Evaluate the new butterfly
        If the new butterfly is better, update it in the population
    end for
    update the value of the power exponent variable c
    update the best global solution if find beter solution
  end while
- Return the best solution found by the BOA algorithm
```

FIGURE 1. The BOA Pseudocode.

[0, 1], I is the stimulus intensity of the emitted fragrance by butterfly, and a is the power exponent that is dependent on the sensory modality with its values over the range [0, 1]. Also, a responsibility is to control the degree of fragrance absorption by stimulus intensity. Based on f value, there are two main equations for updating butterflies' positions in BOA. The first equation is responsible for global search and represented by equation (2), while the local search is represented by equation (3) as follows:

$$x_i^{t+1} = x_i^t + \left(r^2 \times \mathrm{g}^* - x_i^t\right) \times f_i, \tag{2}$$

$$x_i^{t+1} = x_i^t + \left(r^2 \times x_j^t - x_k^t\right) \times f_i, \tag{3}$$

where g* value represents the best solution at the current iteration, f_i is the fragrance magnitude of the i$^{\text{th}}$ butterfly, r represent a random value over the interval [0, 1], x_j^t represents the j$^{\text{th}}$ butterfly, and x_k^t represents the kth butterfly of the available solutions space. The pseudocode of the BOA algorithm is shown in Figure 1.

As shown in Figure 1, the BOA algorithm contains two search phases: local and global search. The switching between these two phases is controlled by the switch probability ρ, which means that if the value of r(randomly generated value) is less than p, then BOA will do a global search (exploration); otherwise, it will make a local search (exploitation).

B. THE NOVEL LOCAL SEARCH ALGORITHM BASED ON MUTATION (LSAM) OPERATOR

In this study, a novel local search algorithm based on a mutation operator is proposed. The pseudocode of this algorithm is shown in Figure 2. As shown in Figure 2, LSAM works in two folds by either improving the current best solution or by improving the quality of other solutions. LSAM at first receives the current best solution g* from BOA at the end of each iteration. Then, LSAM will apply the mutation operator on g*, if the new mutated solution is fitter than g*, then the new solution will replace g*. Otherwise, LSAM will select a random solution from the other remaining solutions and do the same previous process. If the fitness value of the mutated solution is better than the fitness value of the selected solution, then it will replace the selected solution with the new mutated solution. LSAM will repeat these steps *Num_iterations* times. Finally, if it finds a better solution, it will assign it to g*.

The proposed LSAM algorithm, as shown in Figure 2, will iterate for a specified number of times trying to improve the best solution or the other solutions in the population. In each iteration of LSAM, LSAM will mutate the current best solution. Then, LSAM will check the fitness value of the new solution, and if it is better than the current best solution, then it will replace the current best solution with the new mutated solution position; otherwise, LSAM will select one solution randomly from the other available solutions, if the fitness of the new mutated solution is better than the selected solution, then LSAM will replace the randomly selected solution with the new mutated solution position. After that, LSAM will continue the process of improvement based on the Num_iterations variable of LSAM. Therefore, LSAM can improve the best solution to avoid local optima and improve other solutions diversity by replacing the worst solution with a fitter one.

```
- Current_Best = g*(contains the best solution at end of current iteration by BOA)
- Old_fitness = fitness (g*) (fitness value of the best solution)
- K = 1
- Num_iterations = 20
- Mutation_Rate = 0.1
- while (k < Num_iterations)
      New_Position = Mutate(Current_Best  , Mutation_Rate)
      New_Fitness = fitness (New_Position)
      if  New_Fitness < Old_fitness
              g*= New_Position
              Old_fitness = New_Fitness
              Current_Best = g*
      else
              P_Selected= Select randomly a search_agents from other
              population elements
              P_fitness= fitness(P_Selected)
              if  New_Fitness < P_fitness
                     P_Selected= New_Position
              end if
      end if
      k = k+1
  end while
- Return g*
```

FIGURE 2. Pseudocode of LSAM algorithm.

```
- Initialize n butterflies population positions x_i (i = 1, 2, ..., n)
- Set the initial values of the parameters including switch probability ρ, c sensory
  modality, the power exponent a, and the maximum number of iterations known as
  max_iter
- while not reach max_iter  do
          for each butterfly bf in the population do
               Find the fragrance value f of bf using Eq. (1)
          end for
          Find the best butterfly bf
          Assign the best buuterfly to g*
          for each butterfly bf in the population do
                 generate a random value r over the interval [0,1]
                 if (r < p)
                       Update bf position by Eq. (2) (Exploration)
                 else
                       Update bf position by Eq. (3) (Exploitation)
                 end if
                 Evaluate the new butterfly
                 If the new butterfly is better, update it in the population
          end for
          update the value of  the power exponent variable c
          Apply LSAM on the current best solution using mutation operator
          ( to generate a new solution or to improve population elements (Fig.2))
          update the best global solution if find beter solution
  end while
- Return the best solution found by the BOA algorithm
```

FIGURE 3. The DBOA Pseudocode.

C. DYNAMIC BUTTERFLY OPTIMIZATION ALGORITHM (DBOA)

Based on the previous sections, the complete pseudocode of the proposed DBOA algorithm is shown in Figure 3. As shown in the figure, the main improvement to BOA includes the development of the LSAM algorithm, which can improve the exploitation of BOA and solutions diversity. Therefore, at the end of each DBOA iteration, DBOA will call the LSAM algorithm, which in turn can improve the current best solution or improve other population elements based on selecting a solution randomly from the other solutions.

TABLE 1. Description of the experimental datasets.

#	Dataset	Number of	Number of
1	exactly	13	1000
2	credit	20	1000
3	CTG	22	2126
4	m-of-n	13	1000
5	ionosphere	34	351
6	sonar	60	208
7	spect	22	267
8	Diabetic	20	1151
9	vowel	13	990
10	Australian	14	690
11	vote	16	300
12	Hill Valley	100	606
13	OBS-Network	21	1075
14	QSAR	41	1055
15	spambase	57	4601
16	waveform	21	5000
17	libras	90	360
18	penglungew	325	73
19	TOX-171	5748	171
20	Yale	1024	165

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. DATASETS

We selected 20 datasets from the UCI datasets repository to serve as experimental benchmarking datasets to verify the effectiveness of the proposed DBOA algorithm compared to other optimization algorithms. The description of the experimental datasets is presented in Table 1.

B. PARAMETERS SETTINGS

In this subsection, the parameter settings of all experiments are represented as follows: we used 5-fold cross-validation for all experiments, where four-folds are used for training and one-fold used for testing. To conduct a comprehensive comparison to ensure the effectiveness of the proposed algorithm with recent optimization algorithms, we compared our proposed DBOA algorithm with the original BOA and other state-of-the-art optimization algorithms, such as Genetic Algorithm (GA), Grasshopper Optimization Algorithm (GOA) [51], Particle Swarm Optimization (PSO), Ant Lion Optimizer (ALO) [52], and A Sine Cosine Algorithm (SCA) [53]. All parameter settings for each of the proposed algorithm and baseline algorithms are reported in Table 2.

To compare the performance of the proposed algorithm with all baseline algorithms, we used four metrics, namely, classification accuracy, the number of selected features, fitness value, and the p-values based on the Wilcoxon test. All reported results are based on the average values over the thirty distinct runs. The general parameter settings are presented in Table 3.

C. RESULTS AND ANALYSIS

To verify the effectiveness of the proposed DBOA, we conducted a comparison between DBOA and the baseline optimization algorithms (BOA, GA, GOA, PSO, ALO, and SCA using the four-evaluation metrics. The following will discuss one by one evaluation metrics results.

TABLE 2. Parameter settings of each optimization algorithm.

Algorithm	Parameter settings
GA	Crossover_ratio = 0.9, Mutation_ratio = 0.1 are
GOA	c_Max=1, c_Min=0.00004are set identical as the
PSO	Acceleration_constants (C1=2,C2=2),
ALO	I=1 set as in [52]
SCA	a=2 set as in [53].
BOA	a=0.1 as in [18]
DBOA	a=0.1 as in [18] nm=20; mu=0.1

TABLE 3. Parameter settings of all competing algorithms for all experiments.

Parameter	Value
Population	10
Number of iterations	100
Number of runs for each Algorithm	30

Accuracy: In Table 4, we report the comparisons based on the classification accuracy between DBOA and other baseline algorithms on all 20 datasets. Table 4 demonstrates that the proposed DBOA has shown better classification accuracy than all other baseline algorithms on all 20 datasets.

DBOA outperformed all other baseline algorithms with average accuracy 7.83% higher than BOA, 4.71 % higher than GA, 8.09% higher than GOA, 3.00% higher than POS, 8.94% higher than ALO, and 7.18% higher than SCA.

The number of selected features: Table 5 shows the average number of selected features DBOA and other baseline algorithms over 30 runs. As shown in Table 5, DBOA has selected the lowest minimum number of features for 17 out of 20 datasets compared to other baseline algorithms. For the vowel dataset, the POS algorithm selected the lowest minimum number of features. For spambase, and waveform datasets, the ALO algorithm selected the lowest minimum number of features. However, the obtained classification accuracy of ALO on these two datasets (spambase, and waveform datasets) were 0.89583 and 0.79963, respectively. We may conclude that the selection of the lowest minimum using the ALO algorithm has reduced the classification accuracy on both datasets compared to the classification accuracy of the other algorithms, as reported in Table 4. This observation is evident that DBOA has a robust balance between the lowest number of selected features and classification accuracy compared to all other baseline algorithms. Therefore, this analysis can be interpreted as that the DBOA emphasises on selecting the informative features that are essential for better classification accuracy.

Fitness value: Table 6 shows the findings of fitness value measurement. Based on reported fitness values of DBOA in comparison with other baseline optimization algorithms, we can confirm the superiority of DBOA compared with other baseline optimization algorithms over all of the 20 datasets. Furthermore, the experiments are conducted using datasets of various sizes (as shown in Table 1) to ensure the competence of the proposed DBOA and to have a consistent performance regardless of the size of the dataset.

A statistical test (p-values based on the Wilcoxon test): To ensure the significance of improvement of classification

TABLE 4. The average classification accuracy of the competing algorithms over 30 runs.

Dataset	BOA	GOA	GA	DBOA	PSO	SCA	ALO
exactly	0.75117	0.72300	0.81700	**0.99850**	0.87467	0.76967	0.70400
credit	0.72300	0.72517	0.75050	**0.77717**	0.76233	0.72683	0.71033
CTG	0.93052	0.93053	0.94527	**0.98463**	0.95695	0.93492	0.93405
m-of-n	0.90200	0.85400	0.91150	**0.99767**	0.95967	0.92483	0.84200
ionosphere	0.88374	0.88853	0.91413	**0.95488**	0.93071	0.88091	0.88043
sonar	0.84202	0.84777	0.89578	**0.96138**	0.91800	0.85329	0.83479
spect	0.76597	0.75656	0.80670	**0.86546**	0.81734	0.77908	0.75093
Diabetic	0.67985	0.68549	0.70635	**0.73155**	0.71707	0.68505	0.68347
vowel	0.91801	0.90572	0.93249	**0.94731**	0.93502	0.92020	0.90118
Australian	0.78647	0.79541	0.84106	**0.88043**	0.85048	0.78913	0.79179
vote	0.95056	0.94778	0.95944	**0.98167**	0.96389	0.95056	0.94667
Hill Valley	0.55704	0.56034	0.60022	**0.63984**	0.61480	0.55730	0.55237
OBS-Network	0.93597	0.93566	0.94450	**0.97504**	0.95008	0.93891	0.93783
QSAR	0.85229	0.85134	0.87867	**0.90284**	0.88815	0.85118	0.84581
spambase	0.90601	0.90199	0.91713	**0.94136**	0.93028	0.90626	0.89583
waveform	0.82687	0.80643	0.82543	**0.84423**	0.83170	0.83093	0.79963
libras	0.75501	0.75640	0.80132	**0.83750**	0.80972	0.75733	0.75270
penglungew	0.91189	0.91247	0.93775	**0.96744**	0.93945	0.91861	0.90811
TOX-171	0.75591	0.78532	0.82148	**0.89073**	0.85361	0.76266	0.74513
Yale	0.63056	0.64080	0.68155	**0.75189**	0.72706	0.64175	0.62571

TABLE 5. The average number of the selected features of all competing algorithms over 30 runs.

Dataset	BOA	GOA	GA	DBOA	PSO	SCA	ALO
exactly	11.1333	6.6989	8.0333	**6.5**	6.7358	10.9	7.0819
credit	11.6333	9.9332	9.0333	**8.7**	9.9725	12.3667	9.441
CTG	13.0667	10.9731	9.3333	**4.5333**	10.1811	12.6667	11.1189
m-of-n	11.6667	7.228	8.5	**6.7667**	6.9745	10.3	7.1898
ionosphere	20.1333	16.6499	13.8	**9.1667**	15.8013	17.7667	15.7929
sonar	40.9667	29.2555	26.5667	**21.4**	28.5945	35.8333	28.9794
spect	14.9667	10.8672	11.0333	**9.7**	10.5004	14.0667	10.8273
Diabetic	11.4667	9.4699	8.3667	**6.5667**	9.0138	11.6	9.7155
vowel	10.5	7.2583	8.8667	8.4667	**7.0501**	10.6333	7.3103
Australian	7.0667	6.4226	5.6333	**4.5333**	6.3915	6.4333	6.6174
vote	9.9333	7.938	7.1667	**4.7**	7.5863	8.7333	7.8593
Hill Valley	60.5667	49.3863	46.3333	**43.2**	48.5704	57.0667	49.1976
OBS-Network	8.6	9.8711	5.3667	**2.7333**	8.7467	7.8333	9.3635
QSAR	24.8333	20.2472	19.2	**18.2**	19.8152	26.2333	20.2475
spambase	42.3333	28.888	28.8	33	29.0016	41.0333	**28.1641**
waveform	19.4	11.792	13.9333	15.2667	11.4777	19	**11.3216**
libras	52.0667	44.3719	40.1667	**33.9333**	42.3334	48.2333	45.0879
penglungew	162	161.4068	135.9667	**104.6333**	151.7413	178.4667	160.8653
TOX-171	3621.3	2878.4847	2825.9	**2743.8667**	2846.2309	3268.7333	2875.7921
Yale	591.2	509.808	492.9667	**466.9667**	501.2169	563.2333	509.2821

TABLE 6. The average fitness values of all competing algorithms over 30 runs.

Dataset	BOA	GOA	GA	DBOA	PSO	SCA	ALO
exactly	0.25491	0.27964	0.18735	**0.006485**	0.12921	0.23641	0.29871
credit	0.28005	0.27697	0.25152	**0.22496**	0.23974	0.27662	0.29127
CTG	0.07472	0.073658	0.05843	**0.017276**	0.045889	0.07019	0.07038
m-of-n	0.10599	0.1508	0.094153	**0.0075151**	0.045571	0.082338	0.16216
ionosphere	0.12102	0.11484	0.089069	**0.047362**	0.071979	0.12312	0.12271
sonar	0.16322	0.1554	0.10761	**0.041798**	0.084889	0.15121	0.16828
spect	0.23849	0.24583	0.19639	**0.13761**	0.1853	0.2251	0.25144
Diabetic	0.32299	0.31617	0.29511	**0.26922**	0.28382	0.3179	0.31871
vowel	0.089244	0.099513	0.073654	**0.058679**	0.070333	0.087179	0.10409
Australian	0.21644	0.20676	0.16137	**0.12161**	0.15145	0.21336	0.21077
vote	0.055158	0.056471	0.044629	**0.021088**	0.039354	0.054408	0.057675
Hill Valley	0.44458	0.44011	0.40042	**0.36088**	0.38574	0.44398	0.44801
OBS-Network	0.067486	0.067634	0.057504	**0.026013**	0.051344	0.064205	0.065517
QSAR	0.15229	0.1519	0.1248	**0.10062**	0.11479	0.15373	0.1575
spambase	0.10048	0.10216	0.087097	**0.063843**	0.074125	0.1	0.10804
waveform	0.18064	0.19795	0.17946	**0.16148**	0.17335	0.17642	0.20425
libras	0.24833	0.24593	0.20115	**0.16465**	0.1921	0.2456	0.24972
penglungew	0.092217	0.09146	0.065808	**0.035458**	0.06343	0.086069	0.095834
TOX-171	0.24795	0.2175	0.18165	**0.11295**	0.14965	0.24065	0.25732
Yale	0.37152	0.3605	0.32008	**0.25018**	0.27466	0.36017	0.37553

accuracy, the Wilcoxon statistical test was conducted as represented in Table 7. based on reported results, the P-value is less than 0.05 for 18 out of 20 datasets (except for libras and Yale) using PSO. Consequently, it is evident

TABLE 7. The p-values based on the wilcoxon test of all competing algorithms in terms of the average classification accuracy over 30 runs ($p \geq 0.05$ are bolded).

Dataset	BOA	GOA	GA	PSO	SCA	ALO
exactly	1.90E-11	6.15E-11	5.41E-10	1.28E-05	1.89E-11	1.89E-11
credit	1.94E-10	2.98E-10	3.41E-06	1.62E-02	3.42E-10	6.61E-11
CTG	3.65E-11	2.98E-11	3.13E-10	2.09E-07	3.65E-11	7.30E-11
m-of-n	1.89E-11	2.10E-11	2.97E-10	7.25E-04	1.17E-10	1.99E-11
ionosphere	5.31E-10	4.83e-10	1.86E-06	3.76E-03	9.90E-11	1.61E-10
sonar	1.77E-10	1.68E-10	1.10E-06	6.19E-04	4.60E-10	2.86E-10
spect	3.63E-10	1.32E-10	3.37E-07	3.36E-04	1.20E-08	3.46E-10
Diabetic	1.10E-08	7.88E-07	1.72E-03	3.64E-02	3.49E-08	2.95E-08
vowel	1.72E-06	3.06E-07	1.71E-03	4.82E-02	1.86E-05	9.19E-10
Australian	2.53E-09	5.41E-09	1.08E-05	7.25E-04	7.70E-09	1.68E-09
vote	2.16E-07	1.92E-07	1.66E-04	4.67E-04	2.08E-07	3.89E-07
Hill Valley	3.82E-09	1.70E-08	1.86E-03	2.15E-02	8.47E-09	1.07E-09
OBS-Network	3.09E-09	4.03E-09	4.15E-08	1.80E-06	4.41E-09	2.52E-08
QSAR	2.73E-10	7.34E-10	7.87E-05	4.52E-03	2.60E-10	1.09E-10
spambase	3.34E-11	4.97E-11	1.55E-09	1.07E-04	4.50E-11	3.02E-11
waveform	1.87E-07	3.01E-11	1.43E-06	1.40E-04	1.49E-06	3.01E-11
libras	5.38E-07	4.10E-07	2.15E-03	**5.10E-02**	1.15E-06	2.02E-07
penglungew	3.32E-06	4.42E-06	1.14E-03	2.19E-02	1.43E-05	4.41E-06
TOX-171	6.12E-10	1.85E-08	2.00E-06	3.58E-03	5.97E-09	6.12E-10
Yale	1.43E-08	7.09E-08	6.36E-05	**1.71E-01**	7.70E-08	7.12E-09

TABLE 8. The classification accuracy of DBOA in comparison to other BOA varients.

Dataset	DBOA	S-bBOA	CBOA	OEbBOA
exactly	**0.99850**	0.97242	0.82311	0.89253
credit	0.77717	0.74421	0.74351	**0.78111**
CTG	**0.98463**	0.96583	0.94211	0.97156
m-of-n	**0.99767**	0.97200	0.92668	0.96991
ionosphere	0.95488	0.90700	**0.97700**	0.96650
sonar	**0.96138**	0.93620	0.94200	0.95140
spect	**0.86546**	0.84630	0.82900	0.85160
Diabetic	**0.73155**	0.68111	0.68776	0.71332
vowel	0.94731	**0.95134**	0.91772	0.92123
Australian	**0.88043**	0.79122	0.79772	0.83221
vote	0.98167	0.96530	0.96441	**0.98442**
Hill Valley	**0.63984**	0.57881	0.56117	0.61446
OBS-Network	**0.97504**	0.94767	0.93778	0.95116
QSAR	0.90284	0.86221	0.87556	**0.91031**
spambase	**0.94136**	0.91115	0.90911	0.91400
waveform	**0.84423**	0.74290	0.80300	0.83100
libras	**0.83750**	0.76611	0.79441	0.81003
penglungew	**0.96744**	0.87750	0.91200	0.92919
TOX-171	**0.89073**	0.76110	0.77122	0.83077
Yale	**0.75189**	0.64540	0.67220	0.70156

that the improvement introduced by DBOA is statistically significant.

As the convergence rate is crucial to optimization algorithms, the convergence curves of all competing algorithms were plotted to highlight the different algorithms behaviours on all datasets. By contemplating Figure 4, two main comparison perspectives will arise. Firstly, the DBOA succeeded in obtaining lower fitness values in all datasets at the early iterations even it did not converge yet compared to other algorithms (either converged ones or not). This might be imputed to the employed initialisation method, population size, the search mechanism, etc., of the different algorithms. Secondly and generally speaking, the DBOA converged slower than all other algorithms on all datasets. However, this slow convergence was to further and better fitness values in all datasets than all other algorithms. Evidently, this highlights the importance of the employed local search strategy (LSAM) in finding the possible global optima or a closer solution to it.

To further confirm the superiority of our proposed DBOA, we compared it with the three variants of BOA, which were applied for the FS problem. The comparison results between DBOA and these works are shown in Table 8.

Based on the achieved results in Table 8, DBOA outperformed these works in 15 datasets out of 20 datasets in terms of classification accuracy results. Furthermore, for the other five datasets, DBOA achieved comparative results in comparison to other BOA variants. For example, compared to other BOA variants, on the **credit** dataset, DBOA got the 2nd rank, on **ionosphere** DBOA got the 3rd rank, on **vowel** DBOA got the 2nd rank, on the **vote** dataset DBOA got the 2nd rank, and on **QSAR** dataset DBOA got the 2nd rank. This DBOA outperformance is imputed to the new LSAM employment to the BOA. Besides, these achieved results by DBOA confirmed its ability to work efficiently over datasets of different dimensionalities ranging from low, mid, to high.

The overall performance of the proposed algorithm in all metrics - represented by getting better results compared to the state of the art algorithms - is imputed to its ability to improve both the current best solution and the other solutions in the population. These modifications increase the algorithm ability to reach unseen or hidden areas by other

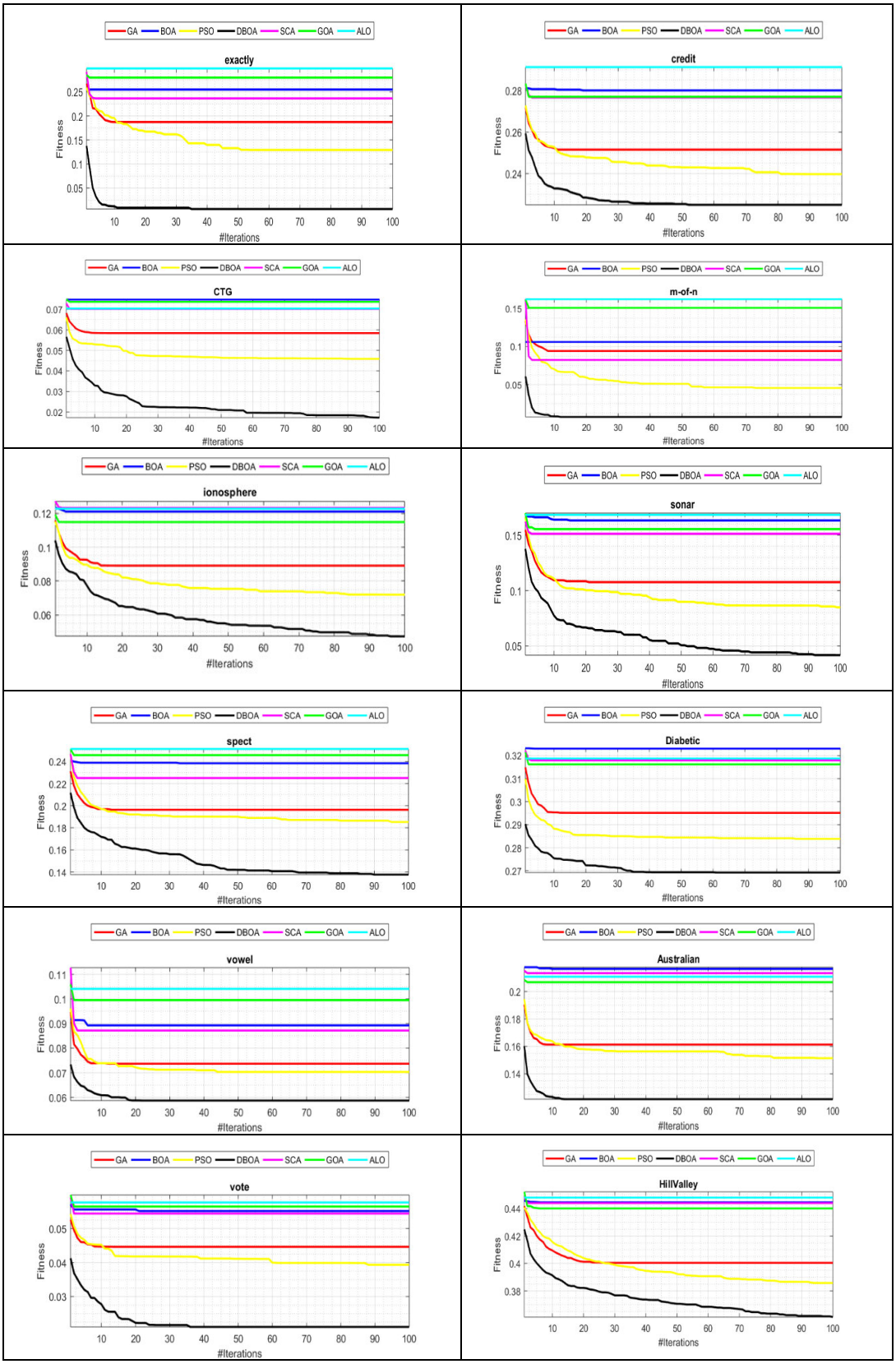

FIGURE 4. Comparison of Convergence curves of all competing algorithms over the 20 datasets.

algorithms mechanisms. In other words, the proposed DBOA algorithm has several strengths, such as the ability to improve the diversity of the solutions, avoiding the local optima problem, and working on datasets with different dimensionalities. However, the main limitation of the DBOA algorithm is the slightly high execution time compared to other

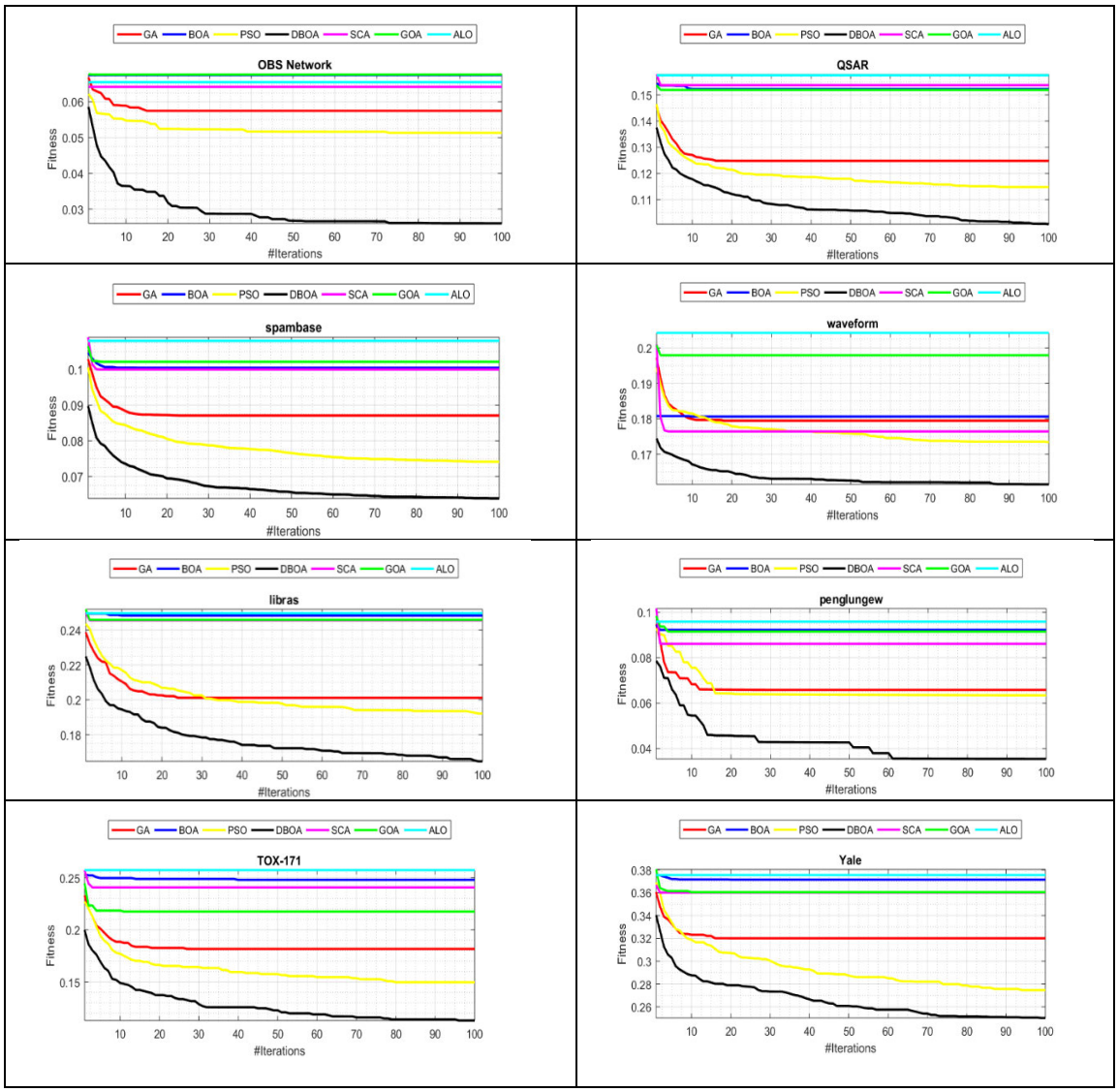

FIGURE 4. *(Continued)* **Comparison of Convergence curves of all competing algorithms over the 20 datasets.**

algorithms, which can be explored as a future work by developing a parallel version of the algorithm to reduce its time complexity.

V. CONCLUSION

This paper proposed an improved version of the BOA algorithm to solve feature selection problems. The improvements were made to alleviate two drawbacks of this algorithm when solving high-dimensional problems: local optima stagnation and low diversity of the solutions during the optimization process. The algorithm was integrated with a local search algorithm based on a mutation (LSAM) operator to avoid the local optima problem and to improve BOA solutions diversity. The algorithm was called DBOA and was designed in a way to solve binary problems.

The proposed method was tested on 20 case studies and compared with a wide range of algorithms in the literature. The obtained results -of four different metrics- demonstrated the merits of the proposed method in solving feature selection problems. Referring to the No Free Lunch theorem in optimization, this algorithm has the potential to supersede existing algorithms in similar binary problems to feature selection. Therefore, we suggest this algorithm as an alternative to solve binary problems due to its ability to select the most informative features and discard the irrelevant features. Also, the proposed algorithm is capable of exploring unseen areas by other baseline algorithms.

For future work, it is recommended to develop a binary multi-objective version of the proposed DBOA to address conflicting objectives in multi-objective problems. Besides, constrained handling methods can be integrated into DBOA to solve constrained problems. In addition, further investigation could be made on the performance of combining the LSAM algorithm with other optimization algorithms or investigating the performance of the DBOA on a broader range of real-world problems.

REFERENCES

[1] M. Tubishat, N. Idris, L. Shuib, M. A. M. Abushariah, and S. Mirjalili, "Improved salp swarm algorithm based on opposition based learning and novel local search algorithm for feature selection," *Expert Syst. Appl.*, vol. 145, May 2020, Art. no. 113122.

[2] R. Sihwail, K. Omar, K. A. Z. Ariffin, and M. Tubishat, "Improved harris hawks optimization using elite opposition-based learning and novel search mechanism for feature selection," *IEEE Access*, vol. 8, pp. 121127–121145, 2020.

[3] Y. Zheng, Y. Li, G. Wang, Y. Chen, Q. Xu, J. Fan, and X. Cui, "A novel hybrid algorithm for feature selection based on whale optimization algorithm," *IEEE Access*, vol. 7, pp. 14908–14923, 2019.

[4] M. Tubishat, M. A. M. Abushariah, N. Idris, and I. Aljarah, "Improved whale optimization algorithm for feature selection in arabic sentiment analysis," *Int. J. Speech Technol.*, vol. 49, no. 5, pp. 1688–1707, May 2019.

[5] B. Wei, W. Zhang, X. Xia, Y. Zhang, F. Yu, and Z. Zhu, "Efficient feature selection algorithm based on particle swarm optimization with learning memory," *IEEE Access*, vol. 7, pp. 166066–166078, 2019.

[6] I. Aljarah, A.-Z. Ala'M, H. Faris, M. A. Hassonah, S. Mirjalili, and H. Saadeh, "Simultaneous feature selection and support vector machine optimisation using the grasshopper optimisation algorithm," *Cognit. Comput.*, vol. 10, pp. 478–495, 2018.

[7] B Selvakumar and K. Muneeswaran, "Firefly algorithm based feature selection for network intrusion detection," *Comput. Secur.*, vol. 81, pp. 148–155, Mar. 2019.

[8] E. Hancer, "Differential evolution for feature selection: A fuzzy wrapper–filter approach," *Soft Comput.*, vol. 23, no. 13, pp. 5233–5248, Jul. 2019.

[9] C. B. Gokulnath and S. P. Shantharajah, "An optimized feature selection based on genetic approach and support vector machine for heart disease," *Cluster Comput.*, vol. 22, no. S6, pp. 14777–14787, Nov. 2019.

[10] P. Sharma, S. Sundaram, M. Sharma, A. Sharma, and D. Gupta, "Diagnosis of Parkinson's disease using modified grey wolf optimization," *Cognit. Syst. Res.*, vol. 54, pp. 100–115, May 2019.

[11] M. Taradeh, M. Mafarja, A. A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, and H. Fujita, "An evolutionary gravitational search-based feature selection," *Inf. Sci.*, vol. 497, pp. 219–239, Sep. 2019.

[12] R. P. S. Manikandan and A. M. Kalpana, "Feature selection using fish swarm optimization in big data," *Cluster Comput.*, vol. 22, no. S5, pp. 10825–10837, Sep. 2019.

[13] A. M. Anter and M. Ali, "Feature selection strategy based on hybrid crow search optimization algorithm integrated with chaos theory and fuzzy c-means algorithm for medical diagnosis problems," *Soft Comput.*, vol. 24, no. 3, pp. 1565–1584, Feb. 2020.

[14] G. I. Sayed, A. Tharwat, and A. E. Hassanien, "Chaotic dragonfly algorithm: An improved metaheuristic algorithm for feature selection," *Int. J. Speech Technol.*, vol. 49, no. 1, pp. 188–205, Jan. 2019.

[15] E. Baş and E. Ülker, "An efficient binary social spider algorithm for feature selection problem," *Expert Syst. Appl.*, vol. 146, May 2020, Art. no. 113185.

[16] M. Tubishat, S. Ja'afar, M. Alswaitti, S. Mirjalili, N. Idris, M. A. Ismail, and M. S. Omar, "Dynamic salp swarm algorithm for feature selection," *Expert Syst. Appl.*, vol. 164, Feb. 2021, Art. no. 113873.

[17] D. H. Wolpert and W. G. Macready, "No free lunch theorems for search," Santa Fe Inst., Santa Fe, NM, USA, Tech. Rep. SFI-TR-95-02-010, 1995.

[18] S. Arora and S. Singh, "Butterfly optimisation algorithm: A novel approach for global optimisation," *Soft Comput.*, vol. 23, no. 3, pp. 715–734, 2019.

[19] M. Mafarja, I. Aljarah, A. A. Heidari, H. Faris, P. Fournier-Viger, X. Li, and S. Mirjalili, "Binary dragonfly optimization for feature selection using time-varying transfer functions," *Knowl.-Based Syst.*, vol. 161, pp. 185–204, Dec. 2018.

[20] M. Ghosh, R. Guha, R. Sarkar, and A. Abraham, "A wrapper-filter feature selection technique based on ant colony optimisation," *Neural Comput. Appl.*, vol. 11, pp. 1–19, Apr. 2019.

[21] Z. Tao, L. Huiling, W. Wenwen, and Y. Xia, "GA-SVM based feature selection and parameter optimization in hospitalization expense modeling," *Appl. Soft Comput.*, vol. 75, pp. 323–332, Feb. 2019.

[22] M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," *Neurocomputing*, vol. 260, pp. 302–312, Oct. 2017.

[23] M. M. Mafarja and S. Mirjalili, "Hybrid binary ant lion optimizer with rough set and approximate entropy reducts for feature selection," *Soft Comput.*, vol. 23, no. 15, pp. 6249–6265, Aug. 2019.

[24] T. Thaher, A. A. Heidari, M. Mafarja, J. S. Dong, and S. Mirjalili, "Binary Harris hawks optimizer for high-dimensional, low sample size feature selection," in *Evolutionary Machine Learning Techniques*. Springer, 2020, pp. 251–272.

[25] M. Mafarja, A. Qasem, A. A. Heidari, I. Aljarah, H. Faris, and S. Mirjalili, "Efficient hybrid nature-inspired binary optimizers for feature selection," *Cognit. Comput.*, vol. 12, no. 1, pp. 150–175, Jan. 2020.

[26] G. I. Sayed, G. Khoriba, and M. H. Haggag, "A novel chaotic salp swarm algorithm for global optimization and feature selection," *Int. J. Speech Technol.*, vol. 48, no. 10, pp. 3462–3481, Oct. 2018.

[27] N. Neggaz, A. A. Ewees, M. A. Elaziz, and M. Mafarja, "Boosting salp swarm algorithm by sine cosine algorithm and disrupt operator for feature selection," *Expert Syst. Appl.*, vol. 145, May 2020, Art. no. 113103.

[28] H. Faris, A. A. Heidari, A. M. Al-Zoubi, M. Mafarja, I. Aljarah, M. Eshtay, and S. Mirjalili, "Time-varying hierarchical chains of salps with random weight networks for feature selection," *Expert Syst. Appl.*, vol. 140, Feb. 2020, Art. no. 112898.

[29] G. I. Sayed, A. E. Hassanien, and A. T. Azar, "Feature selection via a novel chaotic crow search algorithm," *Neural Comput. Appl.*, vol. 31, no. 1, pp. 171–188, Jan. 2019.

[30] L. Zhang, K. Mistry, C. P. Lim, and S. C. Neoh, "Feature selection using firefly optimization for classification and regression models," *Decis. Support Syst.*, vol. 106, pp. 64–85, Feb. 2018.

[31] D. K. Lal, A. Barisal, and S. D. Madasu, "AGC of a two area nonlinear power system using BOA optimized FOPID+PI multistage controller," in *Proc. 2nd Int. Conf. Adv. Comput. Commun. Paradigms (ICACCP)*, Feb. 2019, pp. 1–6.

[32] B. Singh and P. Anand, "A novel adaptive butterfly optimisation algorithm," *Int. J. Comput. Mater. Sci. Eng.*, vol. 7, Dec. 2018, Art. no. 1850026.

[33] S. Arora and P. Anand, "Learning automata-based butterfly optimization algorithm for engineering design problems," *Int. J. Comput. Mater. Sci. Eng.*, vol. 07, no. 04, Dec. 2018, Art. no. 1850021.

[34] S. Sharma and A. K. Saha, "M-MBOA: A novel butterfly optimization algorithm enhanced with mutualism scheme," *Soft Comput.*, vol. 24, no. 7, pp. 4809–4827, Apr. 2020.

[35] B. S. Yıldız, A. R. Yıldız, E. İ. Albak, H. Abderazek, S. M. Sait, and S. Bureerat, "Butterfly optimization algorithm for optimum shape design of automobile suspension components," *Mater. Test.*, vol. 62, no. 4, pp. 365–370, Apr. 2020.

[36] S. M. J. Jalali, S. Ahmadian, P. M. Kebria, A. Khosravi, C. P. Lim, and S. Nahavandi, "Evolving artificial neural networks using butterfly optimisation algorithm for data classification," in *Proc. Int. Conf. Neural Inf. Process.*, 2019, pp. 596–607.

[37] M. Ghanbari and H. Arian, "Forecasting stock market with support vector regression and butterfly optimization algorithm," 2019, *arXiv:1905.11462*. [Online]. Available: http://arxiv.org/abs/1905.11462

[38] M. G. Megahed, R. A. Sweif, and T. S. Abdel-Salam, "Optimal allocation for photovoltaic/ wind turbine applying a hybrid butterfly genetic algorithm," in *Proc. 21st Int. Middle East Power Syst. Conf. (MEPCON)*, Dec. 2019, pp. 994–999.

[39] K. Aygül, M. Cikan, T. Demirdelen, and M. Tumay, "Butterfly optimization algorithm based maximum power point tracking of photovoltaic systems under partial shading condition," *Energy Sources A, Recovery, Utilization, Environ. Effects*, pp. 1–19, Jul. 2019.

[40] Y. Zhi, W. Weiqing, W. Haiyun, and H. Khodaei, "Improved butterfly optimization algorithm for CCHP driven by PEMFC," *Appl. Thermal Eng.*, vol. 173, Jun. 2020, Art. no. 114766.

[41] R. Abdul-Rashid and B. O. Alawode, "Robustness evaluation of the butterfly optimization algorithm on a control system," 2019, *arXiv:1912.00185*. [Online]. Available: http://arxiv.org/abs/1912.00185

[42] A. S. Verma, A. Choudhary, and S. Tiwari, "Test case optimization using butterfly optimization algorithm," in *Proc. 10th Int. Conf. Cloud Comput., Data Sci. Eng. (Confluence)*, Jan. 2020, pp. 704–709.

[43] L. Wen and Y. Cao, "A hybrid intelligent predicting model for exploring household CO_2 emissions mitigation strategies derived from butterfly optimization algorithm," *Sci. Total Environ.*, vol. 727, Jul. 2020, Art. no. 138572.

[44] P. Priyadharshini, K. Prakashraj, and S. Padmapriya, "Improved butterfly optimisation algorithm using local search operator for capacitated vehicle routing problem," *Int. Res. J. Eng. Technol.*, vol. 6, no. 4, pp. 983–988, Apr. 2019.

[45] G. Li, F. Shuang, P. Zhao, and C. Le, "An improved butterfly optimization algorithm for engineering design problems using the cross-entropy method," *Symmetry*, vol. 11, no. 8, p. 1049, Aug. 2019.

[46] Y. Guo, X. Liu, and L. Chen, "Improved butterfly optimisation algorithm based on guiding weight and population restart," *J. Experim. Theor. Artif. Intell.*, pp. 1–19, Feb. 2020.

[47] G. A. Bahgat, A.-A. Fawzy, and H. M. Emara, "An unbiased butterfly optimisation algorithm," in *Proc. Int. Conf. Bio-Inspired Comput. Theories Appl.*, 2019, pp. 506–516.
[48] S. Arora and P. Anand, "Binary butterfly optimization approaches for feature selection," *Expert Syst. Appl.*, vol. 116, pp. 147–160, Feb. 2019.
[49] B. Zhang, X. Yang, B. Hu, Z. Liu, and Z. Li, "OEbBOA: A novel improved binary butterfly optimization approaches with various strategies for feature selection," *IEEE Access*, vol. 8, pp. 67799–67812, 2020.
[50] A. A. Awad, A. F. Ali, and T. Gaber, "Feature selection method based on chaotic maps and butterfly optimisation algorithm," in *Proc. Joint Eur.-US Workshop Appl. Invariance Comput. Vis.*, 2020, pp. 159–169.
[51] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: Theory and application," *Adv. Eng. Softw.*, vol. 105, pp. 30–47, Mar. 2017.
[52] S. Mirjalili, "The ant lion optimiser," *Adv. Eng. Softw.*, vol. 83, pp. 80–98, May 2015.
[53] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimisation problems," *Knowl.-Based Syst.*, vol. 96, pp. 120–133, Mar. 2016.
[54] A. E. Hegazy, M. Makhlouf, and G. S. El-Tawel, "Improved salp swarm algorithm for feature selection," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 32, no. 2, pp. 335–344, 2018.
[55] A. E. Hegazy, M. A. Makhlouf, and G. S. El-Tawel, "Feature selection using chaotic salp swarm algorithm for data classification," *Arabian J. Sci. Eng.*, vol. 44, no. 4, pp. 3801–3816, Apr. 2019.
[56] S. Arora, H. Singh, M. Sharma, S. Sharma, and P. Anand, "A new hybrid algorithm based on Grey wolf optimisation and crow search algorithm for unconstrained function optimisation and feature selection," *IEEE Access*, vol. 7, pp. 26343–26361, 2019.
[57] R. A. Ibrahim, A. A. Ewees, D. Oliva, M. A. Elaziz, and S. Lu, "Improved salp swarm algorithm based on particle swarm optimisation for feature selection," *J. Ambient Intell. Humanized Comput.*, vol. 10, no. 8, pp. 3155–3169, 2019.

MOHAMMAD TUBISHAT received the B.Sc. degree in computer science and the M.Sc. degree in computer and information sciences from Yarmouk University, Jordan, in 2002 and 2002, respectively, and the Ph.D. degree in computer science (artificial intelligence–natural language processing) from the University of Malaya, Malaysia, in 2019. He is currently working as a Lecturer with the Asia Pacific University of Technology and Innovation, Kuala Lumpur, Malaysia. He has published several articles in top ranked academic journals. His research interests include natural language processing, data mining, artificial intelligence, cybersecurity, machine learning, optimization algorithms, data science, and sentiment analysis.

MOHAMMED ALSWAITTI received the B.Eng. degree in computer engineering from The Islamic University of Gaza (IUG), Palestine, in 2010, the M.Sc. degree in electronic systems design engineering from Universiti Sains Malaysia (USM), Malaysia, in 2011, and the Ph.D. degree from the Electrical and Electronic Engineering Department, USM, under the Malaysian International Scholarship (MIS) Scheme. He worked as a Lecturer with the University of Palestine, and an Instructor with the Educational Technology Centre, IUG. He worked as a Graduate Assistant with the Electrical and Electronic Engineering Department, USM. He is currently an Assistant Professor with the Electrical and Computer Engineering Department, XMUM. His research interests include nature-inspired optimization-based clustering techniques, machine learning, and pattern recognition.

SEYEDALI (ALI) MIRJALILI (Senior Member, IEEE) is currently the Director of the Centre for Artificial Intelligence Research and Optimization, Torrens University Australia at Brisbane. He is internationally recognized for his advances in swarm intelligence and optimization, including the first set of algorithms from a synthetic intelligence standpoint–a radical departure from how natural systems are typically understood, and a systematic design framework to reliably benchmark, evaluate, and propose computationally cheap robust optimization algorithms. He has published over 200 publications with over 18 000 citations and an index of 45. As the most cited researcher in robust optimization, he is in the list of 1% highly cited researchers and named as one of the most influential researchers in the world by the Web of Science. He is working on the applications of multi-objective and robust meta-heuristic optimization techniques. His research interests include robust optimization, engineering optimization, multi-objective optimization, swarm intelligence, evolutionary algorithms, and artificial neural networks.

Dr. Mirjalili is an associate editor of several journals, including *Neurocomputing*, *Applied Soft Computing*, *Advances in Engineering Software*, *Applied Intelligence*, and IEEE Access.

MOHAMMED ALI AL-GARADI received the Ph.D. degree in computer science from the University of Malaya, Malaysia, in 2017. He is currently a Postdoctoral Scholar with the University of California at San Diego. He has published several research articles in refereed journals and conferences. His research interests include big data analytics, machine learning, deep learning, AI in healthcare, and medical IoT systems. He served as a reviewer for several journals, including *Journal of Biomedical Informatics*, *IEEE Communications Magazine*, the IEEE Transactions on Knowledge and Data Engineering, IEEE Access, *Future Generation Computing Systems*, *Computers & Electrical Engineering*, and the *Journal of Network and Computer Applications*.

MA'EN TAYSEER ALRASHDAN received the B.Eng. degree in electronic engineering from Yarmouk University, Jordan, in 2001, the M.Sc. degree in computer and communication engineering from the National University of Malaysia (UKM), Malaysia, in 2004, and the Ph.D. degree in computer and communication engineering from the Electrical, Electronic and System Engineering Department, UKM, in 2011. He is currently working as a Senior Lecturer with the Asia Pacific University of Technology and Innovation (APU). He has published several articles in top journals and international conferences. His research interests include smart selection hierarchical mobile internet protocol version six with load balancing and mobility anchor points queue management.

TOQIR A. RANA received the Ph.D. degree from Universiti Sains Malaysia. He is currently working as an Assistant Professor with the Computer Science and IT Department, The University of Lahore, Pakistan. He has published several articles in top journals and international conferences. His research interests include data mining, text mining, sentiment analysis, and NLP.