Applied Intelligence (2021) 51:4824–4859
https://doi.org/10.1007/s10489-020-02038-y

A novel binary farmland fertility algorithm for feature selection in analysis of the text psychology

Ali Hosseinalipour[1] · Farhad Soleimanian Gharehchopogh[1] · Mohammad Masdari[1] · Ali Khademi[2]

Accepted: 24 October 2020 / Published online: 6 January 2021

Abstract
Feature selection plays a key role in data mining and machine learning algorithms to reduce the processing time and increase the accuracy of classification of high dimensional datasets. One of the most common feature selection methods is the wrapper method that works on the feature set to reduce the number of features while improving the accuracy of the classification. In this paper, two different wrapper feature selection approaches are proposed based on Farmland Fertility Algorithm (FFA). Two binary versions of the FFA algorithm are proposed, denoted as BFFAS and BFFAG. The first version is based on the sigmoid function. In the second version, new operators called Binary Global Memory Update (BGMU) and Binary Local Memory Update (BLMU) and a dynamic mutation (DM) operator are used for binarization. Furthermore, the new approach (BFFAG) reduces the three parameters of the base algorithm (FFA) that are dynamically adjusted to maintain exploration and efficiency. Two proposed approaches have been compared with the basic meta-heuristic algorithms used in feature selection on 18 standard datasets. The results show better performance of the proposed approaches compared with the competing methods in terms of objective function value, the average number of selected features, and the classification accuracy. Also, the experiments on the emotion analysis dataset demonstrate the satisfactory results.

Keywords Farmland fertility algorithm · Feature selection · Optimization · Genetic operators · Text psychology

1 Introduction

Nowadays, due to the large increase in the amount of information, feature selection becomes an essential stage when using machine learning and data mining. Feature selection is an effective and fundamental step applied as the prerequisite of classification methods [1, 2]. Classification is the process of classifying data using class labels. The high-dimensional datasets require high processing time for learning and testing the model [3]. Applying feature selection to the dataset before the learning process improves the performance of the classification task [3]. Also, feature selection reduces the dimension of the dataset and maintains more appropriate features so that the speed of operation on the data set is high and the data set is understandable [4]. The various benefits of feature selection are summarized as follows:

- Feature selection reduces the training time and improves the performance of the classification methods.
- It provides a better visualization and understanding of the data in the dataset and also enhances users' ability to use the data.
- Feature selection reduces computational complexity effectively. On the other hand, it makes it easier to work on larger datasets.

Feature selection algorithms are generally divided into two main categories: wrapper and filter approaches [5–7]. The filtering approach is based on the inherent features of the data, not on a specific grouping. The nature of filters is to search for dependent features and remove non-dependent ones. The wrapper methods use a learning algorithm to analyze different feature subsets. Appropriateness of the selected subset is determined by a classification algorithm in these methods. According to the wrapper-based methods, the objective of feature selection is to minimize the number of selected features while maintaining the maximum classification accuracy,

✉ Farhad Soleimanian Gharehchopogh
bonab.farhad@gmail.com

[1] Department of Computer Engineering, Urmia Branch, Islamic Azad University, Urmia, Iran

[2] Department of Psychology Science, Urmia Branch, Islamic Azad University, Urmia, Iran

which can be considered as an optimization problem. This is where choosing an appropriate optimization algorithm to solve the feature selection problem becomes a challenge. On the other hand, due to the high dimensionality of the search space, selecting feature subsets using traditional optimization methods is not efficient [8]. Unlike traditional methods, meta-heuristic algorithms have shown superior performance in solving various optimization problems and can be useful for the feature selection problem [2, 9–12].

Because of the stochastic nature as well as the balance between the two important principles of exploration and efficiency, meta-heuristic algorithms are one of the important methods in solving optimization problems including feature selection. Until now, various meta-heuristic algorithms have been proposed which most of them have been inspired by natural phenomena and behavior of organisms [13–15]. Although meta-heuristic algorithms provide acceptable solutions in a reasonable time, they do not guarantee optimal solutions [16]. A new meta-heuristic algorithm called Farmland Fertility Algorithm (FFA) has been proposed in 2018 [15]. The algorithm is inspired by the phenomenon of farmland divisions and how each part of the farmland is improved. The simulation and different testing results of this algorithm show that it outperforms other meta-heuristic algorithms in solving optimization problems. Given the novelty and superior results of this algorithm in solving optimization problems, we are encouraged to present two different binary versions of this algorithm in this paper. Then, the proposed algorithms are used for feature selection. As an empirical example, we use the proposed algorithms in analyzing text psychology.

There are two approaches for emotion analysis, namely Lexicon-based and machine learning approaches [17]. In the Lexicon-based approach, points are given according to a dictionary that contains all the positive and negative words. This method is very simple in that it considers words such as "good", "excellent", "bad", "ugly", etc. that have a positive or negative connotation in the text and collects their points. The final result of the sentence score is obtained. If the sum of positive points is more, the sentence is considered positive and if the sum of negative points is more, the sentence is considered negative. Despite its simplicity, this method is rarely used due to the complexity of the linguistic structure. Because a sentence can have a lot of negative words provide a positive meaning.

Machine learning approaches are divided into two main categories: Unsupervised and supervised learning [7]. Unsupervised methods (e.g. clustering) are used to infer patterns from a dataset in which only the value of inputs is known and no information about the correct output is available. In contrast, supervised methods (e.g. classification) are the techniques that learn a model from a dataset consisting of input data and the correct output. Supervised learning includes various algorithms such as neural networks, decision trees, vector machines, etc. The supervised algorithms consist of two stages, namely the training stage and the evaluation stage. In the training phase, the model is constructed using the training dataset. The shape of the built model depends on the type of learning algorithm. In the evaluation phase, the experimental data set is used to validate and calculate the accuracy of the constructed model. The test dataset is not used by the algorithm in the model training phase.

The rest of the paper is organized as follows: Section (2) surveys related works and binary versions of various meta-initiative algorithms. In Section (3), the theory and formulation of the FFA are described. In Section (4), the details of two proposed approaches are presented. In Section (5), the performances of the proposed approaches are assessed and the results are presented in graphs and tables. The final section of the paper includes the conclusion and future works.

2 Related works

In this section, we will review state-of-the-art methods and binary versions of a variety of meta-heuristic algorithms in solving the feature selection problem. The details of the review are given in Table 1. In this table, the binary version of the meta-heuristic algorithms and the results of them are described and compared with the other algorithms. Also a column of the table is considered for the comparative algorithms.

As can be seen from Table 1, some researchers have used the original version of each meta-heuristic algorithm to provide the binary versions [1, 2, 5, 21, 24, 25, 29–31, 36, 40]. In these versions, it is attempted to modify the processes of each meta-heuristic algorithm to generate only 0 and 1 for feature selection. In some approaches, researchers have tried to first improve the meta-heuristic algorithms and then provide the binary version of it for feature selection [11, 18–20, 22, 26, 38, 39]. These versions try to use processes such as Ashup to improve and deliver the binary version of the meta-heuristic algorithms. In some binary versions, a combination of two meta-heuristic algorithms has been used to provide the binary versions [27, 32, 37, 38, 41]. The purpose of combining the two meta-heuristic algorithms is to be able to use the processes of both algorithms to solve the feature selection problem. Finally, some researchers have used two functions namely S-shaped and V-shaped, to provide the binary versions of the meta-heuristic algorithm [1, 5, 6, 9, 19, 21, 24, 25, 30, 35, 39]. Both the S-shaped and the V-shaped functions have been designed so that they can convert continuous solutions of the meta-heuristic algorithms into the binary solutions for feature selection.

Researchers have investigated some feature selection methods. The disadvantages of the methods are uninformative features, high-dimensional space, term weighting scheme,

Table 1 Types of binary versions of meta-heuristic algorithms for the feature selection problem

Ref.	Authors	Proposed approach	Comparative algorithms	Results
[18]	(Zhang, Wang, et al. 2014)	Mutation binary particle swarm optimization (MBPSO)	ILS, GA, RSA, ACO, PSO, BPSO, MBPSO	The proposed approach and other methods used on the spam e-mail dataset including 6000 e-mails showed that the proposed approach increased the accuracy, sensitivity, and precision of the decision tree by the effective feature selection.
[19]	(Xiang, Han et al. 2015)	Binary improved gravitational search algorithm(BIGSA)	GA, BPSO, QBPSO, BGSA, BIGSA	Six datasets from the UCI repository were used to evaluate and compare the proposed and other methods. The results show that the proposed hybrid approach performs better than the other methods.
[20]	(Hancer, Xue, et al. 2015)	Modified discrete binary artificial bee colony (MDisABC)	Binary PSO, new velocity based binary PSO, quantum-inspired binary PSO, discrete ABC, modification rate based ABC, angle modulated ABC and genetic algorithms	Experiments performed on 10 valid datasets showed that the proposed approach can achieve higher classification accuracy in the training and test datasets and remove unrelated and redundant features more effectively than the other methods.
[21]	(Emary, Zawbaa et al.2016)	A binary version of Grey Wolf Optimizer (GWO) (bGWO)	The proposed two binary versions of GWO (bGWO1 and bGWO2) are compared against two of the very common optimizers namely PSO and GA. random initialization, small initialization, and large initialization	18 different datasets from the UCI repository were used to evaluate and compare the proposed and other methods. The results prove the ability of the proposed binary version.
[2]	(Sayed, Nabil, et al. 2016)	Binary Clonal Flower Pollination Algorithm (FPA) (BCFPA)	Binary Clonal FPA (BCFPA) BCSA, BBA, Binary Differential Evolution Algorithm (BDEA), and Binary FPA (BFPA).	The results of the experiments on the 3 datasets show that the proposed hybrid algorithm performs significantly better than other known algorithms.
[5]	(Emary, Zawbaa et al.2016)	Binary Ant Lion Optimizer (BALO)	PSO, GA, BBA	Experiments were performed on 21 datasets, and the results show that the proposed approach yields acceptable results.
[22]	(Wan, Wang, et al. 2016)	Modified Binary Coded ACO (MBACO)	GA, binary-coded ACO (BACO), advanced BACO, BPSO, BDEA, and a hybrid GA-ACO algorithm	The proposed approach was applied to 10 datasets, the results of which show that the proposed approach is robust and consistent and shows better performance than other methods mentioned in the paper.
[23]	(LM Abualigah et al. 2017)	Text feature selection with a robust weight scheme and dynamic dimension reduction to text document clustering	HS, GA, PSO	Experiments performed on 8 valid datasets have shown that the proposed approach has a significant ability to generate low-sized subsets of the best features as well as better classification accuracy.
[24]	(Pashaei and Aydin 2017)	a binary version of the Black Hole Algorithm (BBHA)	BPSO-RF, GA-RF, SA-RF, CFS, BBHA-RF	The performance of the proposed approach was evaluated on eight biological datasets. The results show that the proposed approach is superior to other algorithms in all criteria.
[25]	(Mafarja, Eleyan et al.2017)	Binary Dragonfly Algorithm (BDA)	PSO, GA	The proposed approach was applied to 18 datasets, and the results show the better ability of the proposed approach in searching the feature space and feature selection.
[11]	(Sayed, Hassanien et al.2017)	Chaotic crow search algorithm (CSA) (CCSA)	PSO, ABC, chicken swarm optimization(CSO), FPA, moth-flame optimization(MFO), GWO, Whale Optimization Algorithm(WOA), sine cosine algorithm (SCA)	The sinusoidal random map is a suitable map to increase CSA performance and has also been able to maximize the classification accuracy and minimize the number of features selected.
[26]	(Chen, Li et al.2017)	improved swarming and elimination-dispersal bacterial	PSO, GA,SA,ALO,BBA,CS, ACBFO, ISEDBFO	The proposed approaches were tested on 10 well-known UCI datasets and compared with other algorithms. The experimental results show that the classification accuracy of the proposed

Table 1 (continued)

Ref.	Authors	Proposed approach	Comparative algorithms	Results
		foraging optimization algorithm (ISEDBFO)		approach is approximately 3% higher than the other tested methods. Also, the improved algorithms reduce the feature subset length by approximately 3% compared to the other methods.
[27, 28]	(LM Abualigah et al. 2018)	Hybrid clustering analysis using improved krill herd algorithm	HS, GA, PSO, K-Means	The results of the experiments show that the improved krill herd algorithm produced the best results compared to other methods of classification for 8 datasets with 6 criteria used.
[29]	(Allam and Nandhini 2018)	binary teaching-learning based optimization (FS-BTLBO)	FS-BTLBO, GA	The proposed feature selection algorithm was tested on the WDBC (Wisconsin Breast Cancer Diagnostic dataset), which results showed that the proposed approach improves the accuracy of different classification models. It also performed better than GA and reduced the number of features.
[30]	(Mafarja, Aljarah et al.2018)	Binary Dragonfly Algorithm (BDA)	binary grey wolf optimizer (bGWO), binary gravitational search algorithm (BGSA), binary bat algorithm (BBA), particle swarm optimization (PSO), and GA	The proposed approaches were implemented on 18 well-known UCI datasets and compared with the other algorithms. The results showed that the time-varying S-shaped BDA approach performed better than others.
[31]	(Papa, Rosa, et al. 2018)	Binary brain storm optimization	ABC, AIWPSO BA, BHA, BSO, CS, FA, FPA, GP, GSGP, HS,HIS, PSF-HS, PSO, WCA	The proposed approaches were implemented on 26 well-known UCI datasets and compared with the other algorithms. The results showed that the proposed approaches performed better than others.
[9]	(Faris, Mafarja, et al. 2018)	Binary Salp Swarm Algorithm(BSSA)	Binary GWO (BGWO), BGSA, BBA, BPSO, GA	The proposed approaches are evaluated on 22 valid UCI datasets and the results are compared with 5 FS methods. The results show that the proposed approach significantly improves the performance in more than 90% of the datasets compared to other methods.
[32]	(Mafarja and Mirjalili 2018)	Hybrid BALO (HBALO)	Balo-QR, Balos-QR, bALOV-QR, ALO, BALO-1, BALO-S, BALO-V	18 different datasets from the UCI repository were used to evaluate and compare the proposed and other methods. The results confirm the superiority of the proposed hybrid approaches over other methods.
[1]	(Mafarja and Mirjalili 2018)	WOA approaches for wrapper feature selection	PSO, GA	18 different datasets from the UCI repository were used to evaluate and compare the proposed and other methods. The results confirm the superiority of the proposed hybrid approaches over other methods.
[33]	(De Souza, dos Santos Coelho et al. 2018)	New wrapper based in a “v-shaped” binarization of the CSA(BCSA)	BBA, BPSO, SFS, SBS	The proposed and the other methods in the paper are tested on 5 datasets and the proposed approach has achieved a very good result in terms of classification accuracy as well as the selected subgroups.
[34]	(Arora and Anand 2019)	Butterfly Optimization Algorithm (BOA) S-shaped (S-BBOA) BOA V-shaped(V-BBOA)	ALO, BSO, GA, GWO, PSO, S-BBOA, V-BBOA	The 21 datasets from the UCI repository were used to evaluate and compare the performance of the algorithms. The experimental results show the effectiveness of the proposed approaches in improving the classification accuracy compared to other wrapper-based as well as the ability of the proposed approach to search the feature space and feature selection.

Table 1 (continued)

Ref.	Authors	Proposed approach	Comparative algorithms	Results
[35]	(Hussien, Hassanien et al.2019)	Binary WOA base S-shaped (BWOA-S)	WOA, BWOA-s	More than 11 datasets from the UCI repository were used to evaluate and compare the performance of the algorithms. The results show that the proposed approach has a significant performance in finding the optimal features.
[36]	(Santana Jr., Macedo et al.2019)	novel binary artificial bee colony (NBABC)	NBABC,BGA, BFSS, MBPSO, BCSO	The proposed approach was implemented on the backpack problem and 8 UCI datasets for feature selection and outperformed the others.
[37]	(Yan, Ma et al.2019)	Hybrid Binary Coral Reefs Optimization Algorithm with Simulated Annealing(BCROSAT)	BCROSAT, WOASAT, ISFLA, IGA, MPSO	Experimental results through applying the proposed approach on 13 medical datasets show that the proposed approach performs better than the other methods.
[6]	(Mafarja, Aljarah et al.2019)	Binary Grasshopper Optimization Algorithm (BGOA)	BGOA-S, BGOA-V, BGOA-M, bGWO, BGSA, BPSO, BBA, GA, and six filter based methods	The proposed approaches were evaluated using 25 UCI datasets and then compared with 8 meta-heuristic-based and 6 well-known filter-based approaches. The results of the experiments show that BGOA and BGOA-M methods are superior to other methods.
[38]	(Zhang, Gong, et al. 2020)	Binary Differential Evolution with self-learning (MOFS-BDE)	MOFS-BDE, DEMOFS, MOPSOFS, NSGAFS, B-MOABCFS, MOEA/D-2TMFI	Experimental results on 20 UCI datasets show that the balance between local exploitation and global exploration is well maintained. The proposed approach performs better than other comparative methods.
[39]	(Abdel-Basset, El-Shahat et al.2020)	Grey Wolf Optimizer algorithm with a Two-phase Mutation(TMGWO)	BA, BCSA, bGWOA, bWOA, DPSO, FA, MVO, NLPSO, and PSO	The proposed approaches were evaluated using 35 UCI datasets. Statistical analysis proved the superiority of the ultimately proposed approach, TMGWO.

trapped in the local search, premature convergence, a balance between local and global search, and more [7].

Considerable effort is needed to improve the complexity of optimization algorithms for the feature selection problem. Choosing an optimization algorithm for feature selection is a time-consuming process. Most researchers have focused on supervised methods to eliminate uninformative features. Other researchers have used metaheuristic algorithms and unsupervised feature selection techniques to reduce the dimension. Smaller dimensions lead to better performance of the method. Researchers do not use all of the features, but rather weigh in on the feature selection application and use the high-weight features. Researchers use algorithms with these properties to speed up convergence, eliminate premature convergence, avoid local collapse, and balance local and global search [7]. But most of them could not solve all the problems. The FFA has been investigated in terms of different application areas and solving optimization problems. In this paper, we improve it and use it for feature selection.

3 Farmland fertility algorithm

The FFA is a new meta-heuristic algorithm inspired by farmland fertility in nature and was introduced in 2018 [15]. This algorithm attempts to optimize the solutions in each segment by dividing farmland into the segments and using both local and global memory. This algorithm assumes that an environment of farmland is divided into different segments based on soil quality. Each segment of farmland has some soil with a certain quality and the soil quality of each segment is different. An example of segmented farmland, local and global memories are shown in Fig. 1.

In the FFA, the worst-performing segment of the farmland is combined with existing solutions in the global memory. Also the solutions found in other segments of the farmland are combined with all the available solutions in the search space. After each segment of the farmland changed its solution

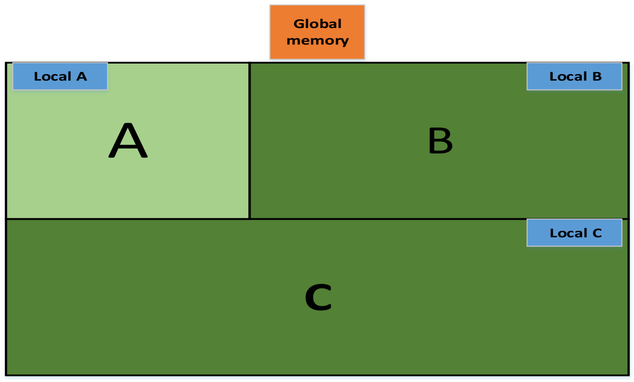


Fig. 1 A segmented example of a farmland and local and global memories[15]

with the global memory and random solutions in the search space, farmers decide to combine each soil in each segment of the farmland based on the best available solution in their local memory. Of course, the condition for combining with the best solution in the local memory is that not all the solutions are combined with their local memories, and at this stage, some solutions are combined with the best global solution to improve the quality of the existing solutions. Following the general theory of the FFA, Fig. 2 illustrates the flowchart of this algorithm and then the formulation and steps of this algorithm are shown according to [15].

3.1 Initial population production and parameter adjustments

At this stage, the initial population is randomly generated and the initial parameters are also adjusted. Of course, unlike other algorithms, this algorithm divides solutions. The initial population number is determined by Eq. (1):

$$N = k*n \tag{1}$$

In Eq. (1), N denotes the total number of solutions available in the search space, k shows the number of segments of the land or space, and n indicates the number of solutions available in each segment of the farmland. The standard number of segments of the farmland can be determined based on the optimization problem. As a result, the entire search space is divided into k segments, each segment having a certain number of solutions. Also, n is an integer variable. At this stage, the solutions available throughout the search space are also evaluated according to the objective function. In [15], a separate segment is specify to determine the value of k, which determines the optimum value of k as $2 \leq k \leq 8$. The value expressed for k can be changed according to the optimization problem.

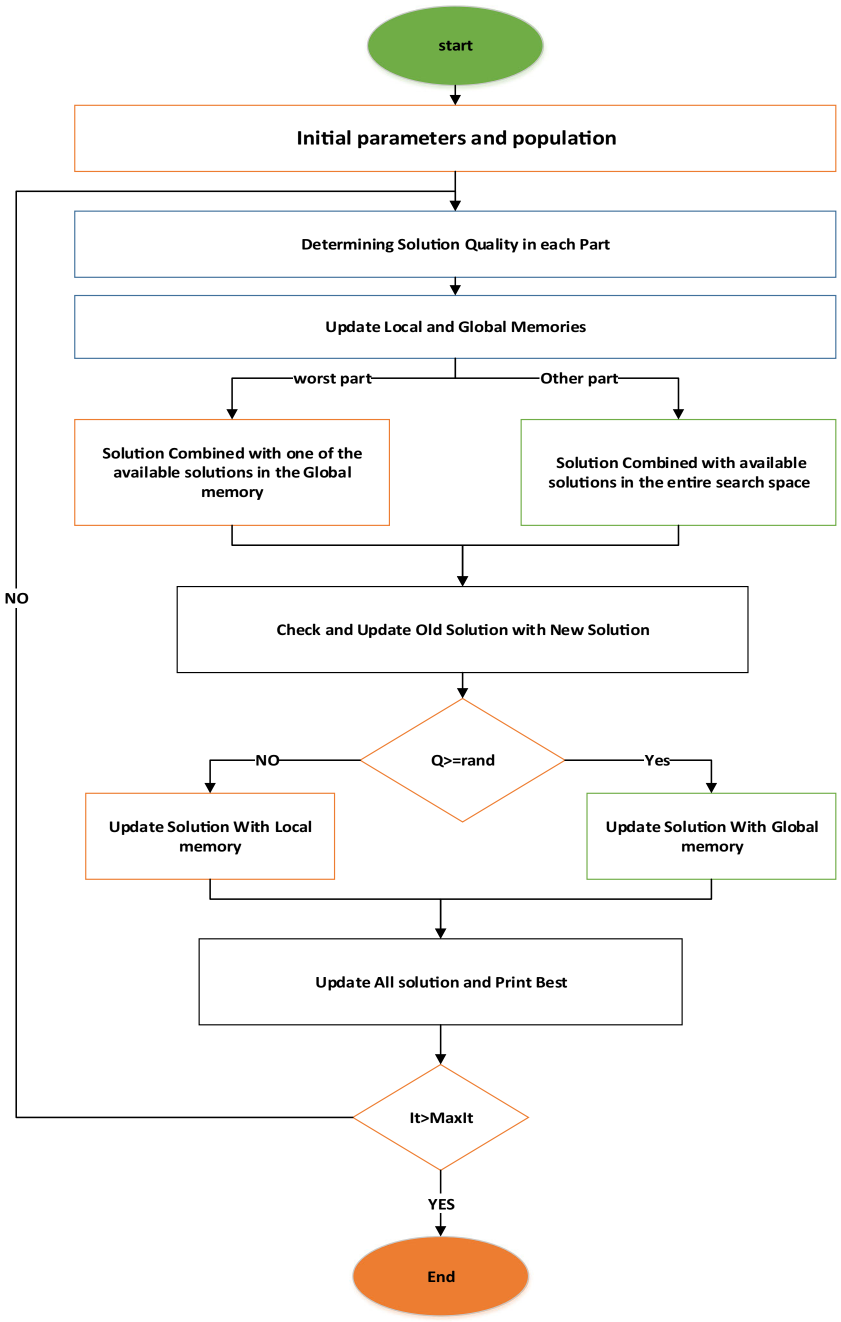
3.2 The quality determination of each segment

In this algorithm, the quality of each segment of farmland is obtained by averaging the solutions available in each segment of farmland. Initially, the solutions are assigned to different segments by Eq. (2).

$$Section_s = x\left(a_j\right)\mathrm{a} = \mathrm{n}*(\mathrm{s}-1) : n*s \tag{2}$$

Equation (2) simply separates the solutions of each segment to calculate the average of each. In this equation, x shows all the solutions in the search space, $\mathrm{s} = \{1, 2, \ldots. k\}$ represents the segment number, and $j = \{1, 2, \ldots. D\}$ shows the dimension of the variable x. According to the value of s, the index of each solution is stored as a member of the variable a_j. Then, the quality of each segment is determined by Eq. (3).

Fig. 2 FFA [15]


start
Initial parameters and population
Determining Solution Quality in each Part
Update Local and Global Memories
worst part
Other part
Solution Combined with one of the available solutions in the Global memory
Solution Combined with available solutions in the entire search space
Check and Update Old Solution with New Solution
Q>=rand
NO
Yes
Update Solution With Local memory
Update Solution With Global memory
Update All solution and Print Best
It>MaxIt
NO
YES
End

$$Fit\_Section_s = Mean\left(\text{ all Fit}\left(x_{ji}\right) in\ Section_s\ \right) \tag{3}$$

In Eq. (3), $Fit\_Section_s$ determines the quality of the solutions in each segment of the farmland, which in the search space is the average fitness or competence of all the solutions available in each segment. Therefore, for each segment of farmland, the average of the total solutions within that segment is obtained and stored in $Fit\_Section_s$. Here, Fit represents the objective function or level of competence, and x_{ji} represents all solutions available in $Section_s$. Also, $i = \{1.2. \ldots .\ n\}$ shows the dimension of the variable x.

3.3 Updating the local and global memories

At this point, the local and global memories of each segment will be updated. The local memory stores some of the best solutions of each segment and the global memory stores the best solutions of all segments, which are determined by the number of the best solutions in local and global memories via Eqs. (4) and (5).

$$M_{local} = round(\mathrm{t}*n); 0.1 < \mathrm{t} < 1 \tag{4}$$

$$M_{Global} = round(\mathrm{t}*N); 0.1 < \mathrm{t} < 1 \tag{5}$$

In Eqs. (4) and (5), M_{Global} and M_{local} show the number of solutions in the global and local memories, respectively. The solutions fall into these memories based on their fitness and competence, and at this point, both memories are updated. Also, t is a vector with random values of [0.1, 1]. For example, if the value of t is 0.1, the number of solutions will be one-tenth of the total population. N represents the total number of solutions available in the search space and n represents the number of solutions available in each segment of farmland. The *round* function is used to round a number of its Count has a decimal point.

3.4 Changing the soil quality in each segment

At this point, the worst-performing segment is most likely to change. The matter about the segment with the worst quality in the farmland is that all the solutions in the worst segment of the farmland are combined with one of the solutions in global memory according to Eqs. (6) and (7).

$$h = \alpha * rand[-1, 1] \tag{6}$$

$$Xnew = h*\left(X_{ij} - X_{MGlobal}\right) + X_{ij} \tag{7}$$

In Eq. (6), α is a number between −1 and 1 that is initialized at the beginning of the algorithm, and *rand*[−1, 1] is assumed to generate random numbers between −1 and 1. In Eq. (7), $X_{MGlobal}$ is a random solution of the global memory solutions, X_{ij} is the solution selected from the worst segment of the farmland for the modifications, and h is a decimal number that can be calculated by Eq. (6). The solutions in the other segments are changed according to Eqs. (8) and (9).

$$h = \beta * rand[0, 1] \tag{8}$$

$$Xnew = h*\left(X_{ij} - X_{uj}\right) + X_{ij} \tag{9}$$

In Eq. (8), β is a number between −1 and 1 that is initialized at the beginning of the algorithm, and *rand*[0, 1] is assumed to generate random numbers between 0 and 1. In Eq. (9), X_{uj} is a random solution from the solutions available in the whole search space (a random solution is chosen from all the solutions found in the segments), X_{ij} is the selected solution from the segments (except the worst segment) for the modifications, and h is a decimal number that can be calculated via Eq. (9).

3.5 Updating solutions based on local and global memories

In this step, the farmers decide to combine each soil within the farmland segments based on the best available cases in their local memory ($Best_{Local}$) at the last step. Of course, the condition for combining with the best in the local memory is that not all solutions in all segments are combined with their local memories, and at this point, some of the solutions available in all the segments will be combined with the best solution found so far ($Best_{Global}$) to improve the quality of the available solutions. The combination of the desired solution with $Best_{Local}$ or $Best_{Global}$ is determined by Eq. (10).

$$H = \begin{cases} X_{new} = X_{ij} + \omega_1 * \left(X_{ij} - Best_{Global}(b)\right) & .Q > rand \\ X_{new} = X_{ij} + rand[0.1] * \left(X_{ij} - Best_{Local}(b)\right) & .else \end{cases} \tag{10}$$

In Eq. (10), the new solution might be created in two ways. In this equation, Q is a parameter between 0 and 1 that must be specified at the beginning of the algorithm. This parameter determines the extent of the solution's combination with $Best_{Global}$. The rand function generates a random number between 0 and 1, and b represents an integer to select one of the solutions available in global or local memory. Also, ω_1 is an integer that must be specified at the beginning of the algorithm and then its value will be gradually reduced by the algorithm (Eq. 11).

$$\omega_1 = \omega_1 * R_v .0 < R_v < 1 \tag{11}$$

In Eq. (11), R_v is a random number between zero and one.

4 The proposed approach

The FFA in continuous form was fully described in Section (3). In this section, we will present two different versions of the algorithm to solve the feature selection problem according to the basic FFA. In Section (4.1), we present a simple binary version of FFA based on the sigmoid function. The purpose of this approach is to provide a simple and more understandable binary version with the least changes. In Section (4.2), a binary version based on the crossover and mutation operators of the genetic algorithm (GA) will be presented. The purpose of this approach is to produce an advanced version of FFA by changing its processes with processes similar to the GA. It will help us to provide a more comprehensive version of FFA for feature selection. Finally, in Section (4.3), a multi-objective function is introduced to reduce the features and increase the classification accuracy.

4.1 BFFA based on the sigmoid function (BFFAS: S-shaped)

In this subsection, we introduce a new binary version of the FFA based on the sigmoid function. As stated in Section (3), the process of FFA moves in a continuous space and therefore all solutions in the population of this algorithm are continuous numbers. Given that selecting a feature or not is a matter, the new binary solution must contain the numbers 0 and 1, where 1 represents the selection of a feature for the new dataset, and 0 indicates not choosing a feature. To this end, we used the sigmoid or S-shaped function [34, 35] to move the processes of the FFA in a binary space. Therefore, in this proposed model, the sigmoid function is used to change the continuous positioning of the solutions into a binary-state in the FFA using Eq. (12):

$$sg\left(FFA_i^d(t)\right) = \frac{1}{1 + e^{-FFA_i^d(t)}} \tag{12}$$

In Eq. (12), $FFA_i^d(t)$ shows the continuous value of the ith solution in the population of the FFA in the dth dimension of the iteration t. The sigmoid function is presented in Fig. 3. Here in the following, we will analyze the output of this function and how to incorporate it into the FFA.

According to Fig. 3, the output of the sigmoid transfer function is still in a continuous state between 0 and 1, and therefore, to convert it into a binary value, a threshold has to be set. A random threshold is presented in Eq. (13) to convert the solution into the binary value for feature selection using the sigmoid function:

$$FFA_i^d(t+1) = \begin{cases} 0 & if\ rand < sg\left(FFA_i^d(t)\right) \\ 1 & if\ rand \geq sg\left(FFA_i^d(t)\right) \end{cases} \tag{13}$$

In Eq. (13), FFA_i^d shows the position of the ith solution in the population of the FFA in the dth dimension of the iteration t. Also, rand is a function that produces numbers between 0 and 1 in a uniform distribution. Thus the solutions available in the population of the FFA are forced to move in a binary search space using Eqs. (12) and (13). Then, these relationships are embedded more precisely into the FFA. To better understand, the proposed method is described as pseudocode, illustrated in Algorithm 1.

Algorithm 1: BFFA based on the sigmoid function

```
BFFAS: S-Shaped Algorithm
01: Initialize parameters: K,$\beta$,$\alpha$, Q, N
02: Determine the number of sections of Farmland according to Equation (1)
03: Generate initial population $X_i$ (i=1... N) at random $X_i \in$ random 0,1
04:  Calculate Objective function f($X_i$ )
05: While (t < maximum number of iterations)
06:       Determine quality in each of the sections according to Equations (2) and (3)
07:       Update Global Memory And Local Memory Related to each of Sections according to Equations (4) and (5)
08:       Worst Sections: Changes on Considered Sections according to Equations (6) and (7)
09:       Other sections: Changes on Considered Sections according to Equations (8) and (9)
10:       For i=1: N
11:           Convert Continuous($X_{inew}$) to Binary($BX_{inew}$)  using transfer function according to Equations (12) and (13)
12:       End for
13:       Evaluation of All-New Solutions
14:       For all solutions
15:           If (Q > rand)
16:               Changes of Solution according to Equation (10).{part: Q > rand }
17:           else
18:               Changes of Solution according to Equation (10).{part: else}
19:           End if
20:       End for
21:       for i=1: N
22:           Convert Continuous($X_{inew}$) to Binary($BX_{inew}$)  use transfer function according to Equations (12) and (13)
23:       End for
24:       Evaluation of All-New Solutions
25: End while
26: Print Best Solution
```

This algorithm starts by adjusting the parameters (line 01) and then determining the farmland segments (line 02). In line (03), unlike the continuous FFA, the population is randomly created from 0 and 1. Lines 04–09 show the main stages of the continuous FFA including segmentation, quality determination of each segment, updating local and global memories, and soil quality changes in each segment. Lines 09–12 include the new step to convert the solutions produced in the previous steps to binary by the sigmoid transfer function based on Eqs. (12) and (13). At this point, all the continuous solutions are converted into binary solutions. Lines 13–20 perform the stage of evaluating new solutions and updating them based

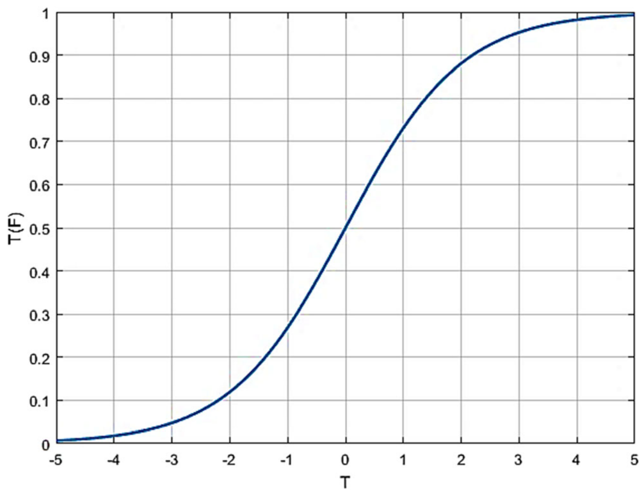


Fig. 3 Overview of the sigmoid transfer function

on local and global memories to generate new solutions. Lines 21–23 show the new step to convert the new solutions produced in the previous steps to binary by the sigmoid transfer function based on the Eqs. (12) and (13). At this point, all the continuous solutions are converted into binary solutions. Finally, lines 24–26 evaluate the new binary solutions. If the algorithm has reached its end condition, it will show the best solution. Thus, in this approach, we developed a new binary method using the sigmoid transfer function by placing lines 13–20 and 21–23 in two separate parts of the FFA. The reason of using the sigmoid transfer function in two parts is that the FFA evaluates and modifies the solutions in two stages.

4.2 BFFA based on GA (BFFAG: GA-Base)

In this subsection, BFFA based on the GA process called BFFAG is presented. This algorithm uses new operators called BGMU and BLMU and also a dynamic mutation operator. The BGMU operator is utilized to update the binary solutions based on the global memory and the entire search space of the FFA algorithm and the BLMU operator is used to update the binary solutions based on the local and global memories of the FFA algorithm. These operators increase the efficiency and exploration of the FFA algorithm. Furthermore, in the final part of the FFA algorithm, the dynamic mutation operator is used to increase the exploration of this algorithm. The BFFAG approach also defines genetic and novel operators to maintain a balance between exploration and efficiency. The genetic algorithm involves mutation and crossover operators that are designed and implemented in BGMU and BLMU operators to generate binary solutions. As shown in Section (3), the steps of the FFA are done in continuous form. But to use this algorithm in feature selection, one must use operators proportional to the binary space. In the following, the BFFAG approach is described step by step.

4.2.1 Initial population production and parameters adjustment

At this stage, the initial population is randomly generated, and the initial parameters are adjusted. The initial population is determined according to Eq. (1). Unlike the continuous version of the FFA, at this stage, binary solutions are generated according to Eq. (14).

$$X_{ij} = \mathrm{GIBP}(0,1); i = 1 : N \text{ and } j = 1 : D \tag{14}$$

In Eq. (14), N represents the total number of solutions available in the search space, D shows the number of dimensions of each problem, and X_{ij} represents a random binary solution. Therefore, according to the FFA algorithm based on k and n, where the initial population is represented by N, a random initial population can be generated according to the pseudo-code in Algorithm 2.

Algorithm 2: Generating a binary initial population for the BFFAG approach

```
Generate Initial Binary Population(GIBP) :
01: For i=1: N
02:     For j=1: D
03:         If(rand>0.5)
04:             $X_{ij}$=1
05:         Else
06:             $X_{ij}$=0
07:         Endif
08:     EndForj
09: Calculate Objective function f($x_{ij}$ )
10: EndFori
```

4.2.2 Quality determination of each segment and updating the local and the global memories

At this stage, such as continuous FFA, segmentation and quality determination of each segment is performed. It is not necessary to apply any changes to this stage, as only the computation, segmentation, and quality determination are performed for each segment. According to the FFA algorithm, some of the best solutions of each segment are stored in local memory, and the best solutions of all segments are stored in global memory, which are determined by the number of the best local memory and the number of the best global memory based on Eqs.(4) and (5). This stage does not need to be changed.

4.3 Changing the soil quality in each segment

In the BFFAG approach, soil quality changes in each segment will be done based on the new BGMU operator. In this operator, the solutions will be based on the genetic crossover operator. Also, some modifications are considered based on the basic concepts of the FFA algorithm. Fig. 4 shows the new BGMU operator performance. Furthermore, the pseudocode of this operator is illustrated in Algorithm 3. One of the advantages of this operator is the reduction of α and β

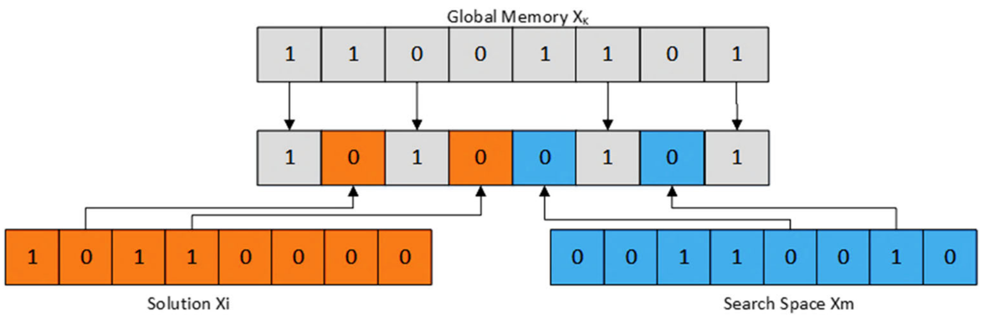


Fig. 4 An example of a BGMU operator performance

parameters. This operator selects a random solution from the best solutions in global memory and a random solution from the entire search space. Finally, the new solution is generated from the current solution (X_i) and the combination of two solutions X_k and X_m.

```
Algorithm 3: BGMU operator
Function BGMU($X_i, w$)
01: $Xnew = []$
02: $X_k$=The best randomly selected solution from Global Memory
03: $X_m$= The best randomly selected solution from All search space(All Selection)
04:   For j=1: Nvar
05:     r=rand
06:     If(r<w)
07:       $Xnew_{ij}$=$X_{kj}$
08:     Elseif(rand<0.5)
09:       $Xnew_{ij}$=$X_{mj}$
10:     Else
11:       $Xnew_{ij}$=$X_{ij}$
12:     Endif
13:   EndForj
14: Return $Xnew$
```

By the BGMU, the changes in soil quality in each section are converted into a binary format, which would increase the efficiency of the binary FFA. This operator has two parameters. The first parameter (X_i) represents the same current solution. The second parameter (w) is a random number between zero and one. This parameter (w) will have a great impact on the efficiency of the algorithm because its large value causes the use of the solutions available in the global memory. Therefore, according to the law of equilibrium of the meta-heuristic algorithms, the efficiency of the algorithm should be initially low and then can be improved by increasing the number of iterations. In BGMU operator, this parameter is dynamically set through the Eq. (15):

$$w = \text{rand}\left[0, \left(\frac{\text{It}}{\text{MaxIt}}\right)\right] \tag{15}$$

In Eq. (15), It denotes the current iteration, MaxIt denotes the maximum iteration, and rand generates a random number between 0 and $\frac{It}{MaxIt}$. Thus, according to the pseudo-code in Algorithm 3, the parameter w makes use of the current solution (X_i), the best global memory solutions (X_k), and the solutions in the search space (X_m). Fig. 5 shows how to use these solutions and increase the efficiency using the BGMU operator.

Figure 5a shows that increasing the number of iterations, the parameter w gets closer to 1 and we will see an increase in efficiency. The higher the value of the parameter w, the greater the probability of using global memory. Fig. 5b shows that the population using global memory is initially small and increases with increasing the number of iterations. Fig. 5c shows that the BGMU operator initially seeks to explore and makes the most of the entire search space, but when the number of iterations increases, the exploration reaches its minimum. Finally, Fig. 5d shows that it almost uses the current solution correctly to generate a new solution during the exploration.

4.4 Updating the solutions based on the local and global memories

In the continuous FFA, the solutions in each segment are combined based on the best available solution in their local memory ($Best_{Local}$). At this point, some of the solutions available everywhere are combined with the best solution ever found ($Best_{Global}$) to improve the quality of the available solutions. The binary version also uses a process to combine the desired solution ($Best_{Local}$) or ($Best_{Global}$). In this section, the operation is defined by the new BLMU operator whose pseudo-code is shown in Algorithm 4.

```
Algorithm 4: BLMU operator
Function BLMU($X_i$, $X_k$)
01: $Xnew = []$
02: $X_i$ is now a solution
03: $X_k$select random solution from local memory or global memory
04:   For j=1: Nvar
05:     If(rand<0.5)
06:       $Xnew_{ij}$=$X_{kj}$
07:     Else
08:       $Xnew_{ij}$=$X_{ij}$
09:     Endif
10:   EndForj
11: Return $Xnew$
```

According to Algorithm 4, the BLMU operator is combines multiple points and the probability of combining each point is considered to be 50%. This value is considered to use local and global memories equally. The combination of the solution with the solutions in local and global memories is determined by the value of parameter Q defined as Eq. (16):

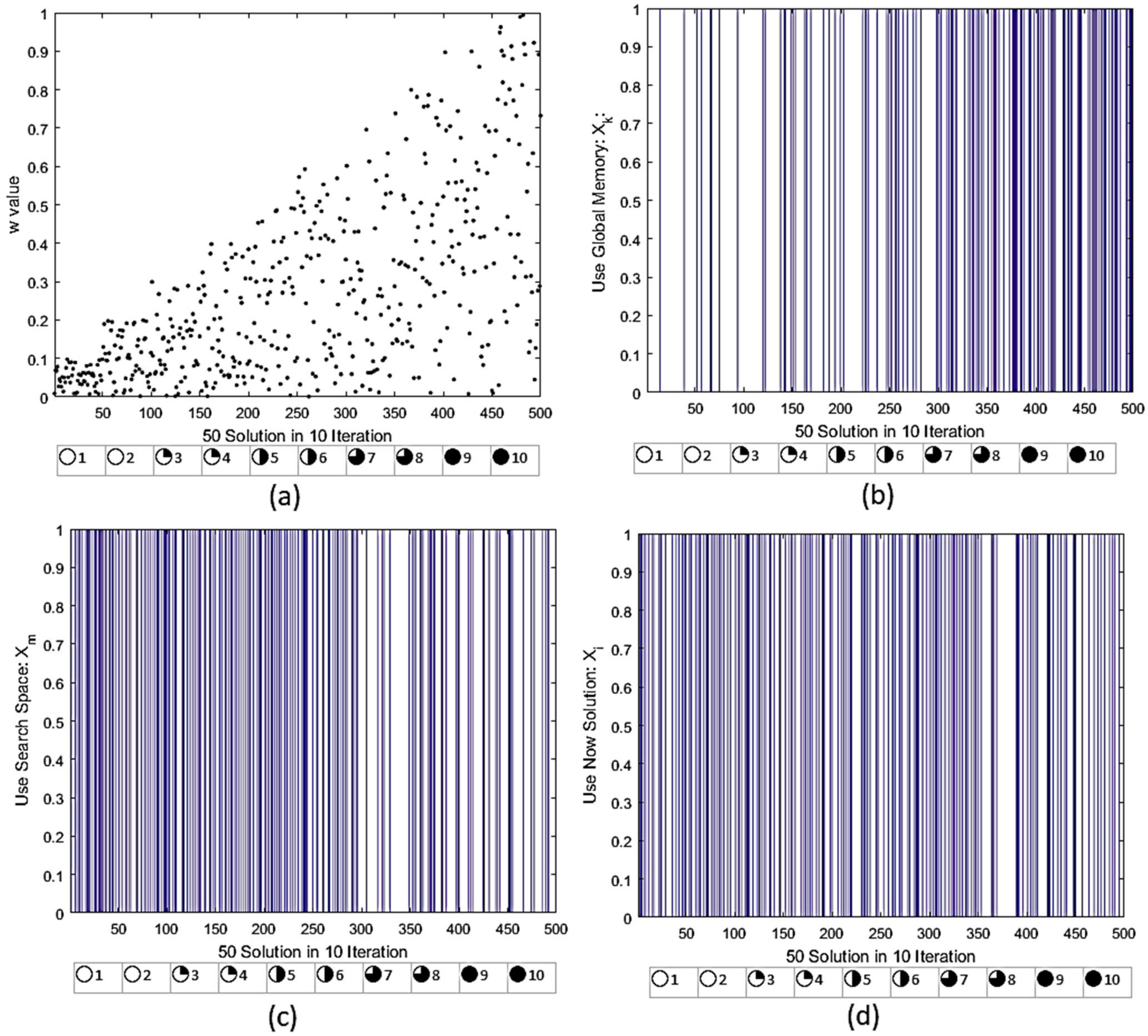


Fig 5 How to use BGMU solutions and increase efficiency

$$X_{inew} = \begin{cases} \mathrm{BLMU}(X_i, \mathrm{Best}_{Global}(b)) & .Q > \mathrm{rand} \\ \\ \mathrm{BLMU}(X_i, \mathrm{Best}_{Local}(b)) & .\ \mathrm{else} \end{cases} \tag{16}$$

In Eq. (16), the combination of a new solution in the binary version is based on the Q parameter, but we used the genetic crossover process instead of the continuous combination. It should be noted that the parameter ω_1 is considered as changes in the continuous version. This parameter is removed in the binary version and the changes are controlled by the dynamic mutation operator. So, in the binary version, a mutation operator is required to apply changes. The mutation operator enhances exploration in meta-heuristic algorithms. In the proposed approach, this operator is dynamically controlled, so the parameter ω_1 is eliminated. The pseudo-code of the dynamic mutation operator is shown in Algorithm 5.

Algorithm 5: Dynamic mutation operator to maintain exploration

```
Function DM($X_i$)
01: $Xnew = []$
02: h=$\left(\frac{It}{MaxIt}\right)$
04:   For j=1:Nvar
05:      If(r<h)
06:         $Xnew_{ij}$=~$(X_{ij})$
07:      Else
08:         $Xnew_{ij}$=$X_{ij}$
09:      Endif
10:   EndForj
11: Return $Xnew$
```

As seen in Algorithm 5, the dynamic mutation operator is intended to maintain the exploration by the number of current generations and the maximum generation. Thus, according to the meta-heuristic algorithms, the number of mutations should be large at first and gradually decrease. In this section, to test the dynamic mutation operator, population size is set to 50, the number of iterations is set to 100, and number of dimensions is set to 10. The results of 3 independents trials are shown in Fig. 6.

Figure 6 shows that the exploration reduction using the dynamic mutation operator is exactly as expected. The number of dimensions covered by the mutation operator is initially large and decreases with gradually increasing the number of iterations, so that it reaches the lowest possible value in the last iterations. Thus, we presented a binary FFA based on the genetic algorithm using the BGMU, the BLMU, and the dynamic mutation operators. The pseudo-code of the proposed approach is illustrated in Algorithm 6.

Algorithm 6: BFFA based on the GA

```
01: Initialize parameters: K,Q,N,R
02: Determining the number of sections of Farmland according to Equation(1)
03: Generate Initial Binary Population(GIBP)
04: While (t < maximum number of iterations)
05:   Determining quality: $Fit\_Section_s = Mean($ all Fit$(x_{ji}) in\ Section_s)$   $s = \{1.2. \dots .k\}. i = \{1.2. \dots .n\}$
06:   Update Global Memory And Local Memory
07:   $w = rand[0, (\frac{It}{MaxIt})]$
08:   For i=1: N
02:   $X_k$=The best randomly selected solution from Global Memory
03:   $X_m$= The best randomly selected solution from All search space(All Selection)
04:   For j=1: Nvar
05:       r=rand
06:       If(r<w)
07:           $Xnew_{ij}$=$X_{kj}$
08:       Else If (rand<0.5)
09:           $Xnew_{ij}$=$X_{mj}$
10:       Else
11:           $Xnew_{ij}$=$X_{ij}$
12:       End If
10:   End for
10:   Evaluation of All-New Solutions with Fitness function Equation(17)
11:   For i=1: N
12:     If (Q > rand)
13:         b=randi[1,Count_Global];
14:         Changes of Solution with BLMU($X_i$, $Best_{Global}$(b))
15:     else
16:         b=randi[1,Count_local];
17:         Changes of Solution BLMU($X_i$, $Best_{Local}$(b))
18:     End if
19:     Changes of Solution with DM($X_i$, R)
20:   End for
21:   Evaluation of All-New Solutions with Fitness function Equation(17)
22: End while
23: Print Best Solution
```

4.5 The objective function

In this sub-section, we describe the objective function for feature selection in the proposed approach. Feature selection can be considered as a multi-objective optimization problem in which two conflicting goals including the minimum number of the selected features and higher classification accuracy are achieved. Therefore, we need a classification algorithm to define the objective function of the feature selection problem. In this paper, we use the simplest classification method, namely the K-Nearest Neighbors (KNN) classifier [42, 43] used by most researchers [1, 7, 8, 16, 28, 44, 45]. The KNN classifier is used to more accurately evaluate the features selected by the proposed approach and other algorithms. Each solution is evaluated based on the proposed multi-objective function which depends on the KNN classifier. In the proposed

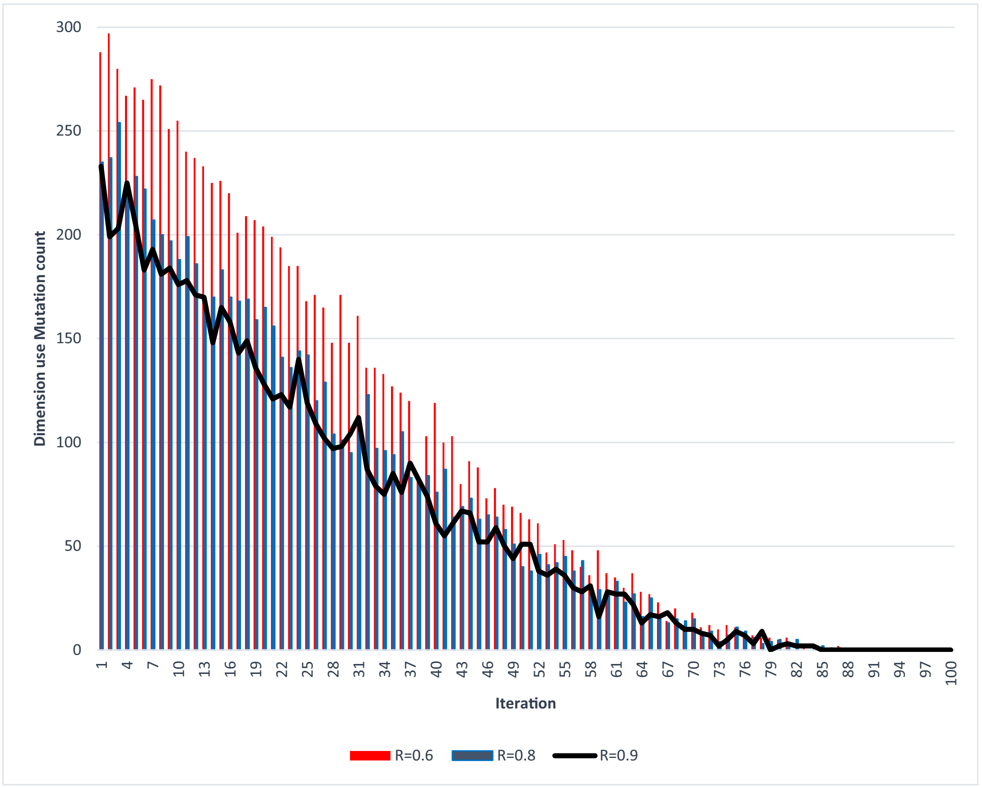


Fig. 6 The results of 3 independents trials

multi-objective function, to balance the number of features selected in each solution (minimum) and the classification accuracy (maximum), the objective function in Eq. (17) is used to evaluate a solution in each meta-heuristic algorithm.

$$Fitness = \alpha\gamma_R(D) + \beta\frac{|R|}{|N|} \quad (17)$$

In Eq. (17), $\alpha\gamma_R(D)$ represents the error rate of the classification method, | R | shows the multilinearity of the selected subset, and | N | shows the total number of features available in the dataset. Also, α represents the importance of classification quality and β shows the length of the subset. The values of these two parameters are taken as α ∈ [0, 1] and β = (1-α) from [5]. In this paper, the initial value of α is assumed to be 0.99 and thus the value of β will be equal to 0.01.

Here, the important point is the percentage of testing and training, which is typically considered to be 30 to 70 or 20 to 80. But since the goal of feature selection is to find effective features, the training percentage should be higher. In this case, the algorithm can better find the effective features. That's why 80% of the data is for training and 20% for testing.

5 Results and discussions

In this section, the evaluation and results of the proposed approaches will be discussed. All experiments are performed using MATLAB software on a Core i5 processor with 8 GB of RAM. Also for the KNN algorithm, k is set to be 5, which is the most appropriate value according to references [1, 14, 35]. To evaluate the performance of the two proposed approaches and other competing algorithms, various experiments have been conducted on 18 feature selection datasets from the UCI data repository [46] . Furthermore, we apply our proposed approaches to the emotion analysis dataset. The experimental results for the UCI datasets are presented in Section 5.1 and the results for the emotion analysis datasets are reported in Section 5.2. The results show that the proposed approaches perform better than the other algorithms in terms of the average number of the selected features, classification accuracy, and objective function value on the UCI datasets, as well as emotion recognition accuracy on the emotion analysis datasets.

5.1 Experiments on the UCI dataset

In this sub-section, the performance of the two proposed approaches compared to other meta-heuristic algorithms in the feature selection problem is evaluated using 18 datasets taken from the UCI learning database site. Each dataset has a specified number of features so that by running the two proposed approaches on each dataset, their performance can be investigated on the number of selected features. Table 2 lists the number of features, the number of samples and the number of classes in each dataset.

Table 2 Dataset description

ID	Dataset	No. of features	No. of instances	No. of classes	Missing values	Type
D1	Abalone	8	4177	29	No	Life
D2	Breast Cancer Wisconsin	9	699	2	Yes	Life
D3	BreastEW	30	569	2	No	Life
D4	Dermatology	34	366	6	Yes	Life
D5	Germen credit	24	1000	2	No	Business
D6	Glass identification	10	214	6	No	Physical
D7	Hepatitis	19	55	2	Yes	Clinical
D8	Indian Liver Patient Dataset	10	583	2	No	Clinical
D9	IonosphereEW	34	351	2	No	Physical
D10	Lung Cancer	56	32	3	Yes	Clinical
D11	Lymphography	18	148	4	No	Life
D12	SPECT	22	267	2	No	Clinical
D13	Statlog (heart)	13	270	2	No	Clinical
D14	Steel Plates Faults	33	1941	7	NO	Physical
D15	Thoracic surgery	17	470	2	No	Clinical
D16	Waveform	21	5000	3	No	Physical
D17	WineEW	13	178	3	No	Physical
D18	Zoo	17	101	2	No	Life

We will use the 18 datasets listed in Table 2 with various feature numbers (between 8 and 56) to test the algorithms in this section. We first evaluate the two proposed approaches and compare them with the original FFA. Then we select one of the proposed approaches with the best results as the main method and compare it with other important meta-heuristic algorithms including Genetic Algorithm (GA) [47], Binary Bats Algorithm (BBA) [48], V-shaped Binary PSO (BPSO) [49], Binary Flower Pollination Algorithm (BFPA) [10], Binary Gray Wolf Optimizer (BGWO) [21] and Binary Dragonfly Algorithm (BDA) [14] and Binary Chaotic Crow Search Algorithm (BCCSA) [11]. Table 3 shows the parameters of the proposed approaches and the other comparative algorithms.

5.1.1 Performance evaluation of proposed approaches (BFFAG, BFFAS) and comparison with the original FFA

The purpose of this section is to evaluate the performance of proposed BFFAG and BFFAS approaches in terms of different criteria including the average number of features, the classification accuracy and the convergence of objective function, and compare the results with the original FFA. For all experiments in this section, we set the parameters according to

Table 3 Parameters setting of the proposed approaches and Other Algorithms

Algorithm	Variable parameters	Fixed parameters
GA [47]	PC = 0.6,PM = 0.4	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
BBA [48]	A = 0.5,r = 0.5,Qmin = 0,Qmax = 2	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
BPSO [49]	C1 = 2.05, C2 = 2.05,W = 2	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
BFPA [10]	$P = 0.7$	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
BGWO [21]	–	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
BDA [14]	–	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
BCCSA [11, 50]	AP = 0.1, fl = 2	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
FFA [15]	$\alpha = 0.6, \beta = 0.4$,W = 1,Q = .7	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
BFFAS	$\alpha = 0.6, \beta = 0.4$,W = 1,Q = .7	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
BFFAG	W = 1,Q = .7, R = 0.9	MaxIt = 50,Npop = 10,MinVar = 0,MaxVar = 1
Knn in Fitness Eq. (17)	$\alpha = 0.99, \beta = 0.01$,k = 5	

Table 4 The average number of features of two proposed approaches (BFFAG and BFFAS) and original FFA

Dataset	FFA	BFFAS	BFFAG	BEST Approaches
D1	3.99	3.81	4.1	☐ FFA ☒ BFFAS ☐ BFFAG
D2	4.56	4.3	4.3	☐ FFA ☒ BFFAS ☒ BFFAG
D3	14.52	14.52	14.4	☐ FFA ☐ BFFAS ☒ BFFAG
D4	17.23	16.4	16.02	☐ FFA ☐ BFFAS ☒ BFFAG
D5	12.67	12.46	11.7	☐ FFA ☐ BFFAS ☒ BFFAG
D6	4.64	4.84	4	☐ FFA ☐ BFFAS ☒ BFFAG
D7	9.41	9.79	7.5	☐ FFA ☐ BFFAS ☒ BFFAG
D8	4.84	4.6	4.6	☐ FFA ☒ BFFAS ☒ BFFAG
D9	16.23	16.04	13.9	☐ FFA ☐ BFFAS ☒ BFFAG
D10	28.03	27.77	22.1	☐ FFA ☐ BFFAS ☒ BFFAG
D11	9.63	9.4	8	☐ FFA ☐ BFFAS ☒ BFFAG
D12	10.95	10.54	10.2	☐ FFA ☐ BFFAS ☒ BFFAG
D13	6.36	6.06	5.8	☐ FFA ☐ BFFAS ☒ BFFAG
D14	14.9	15.83	14.9	☒ FFA ☐ BFFAS ☒ BFFAG
D15	7.57	8.65	7.5	☒ FFA ☐ BFFAS ☒ BFFAG
D16	11.24	10.27	11.3	☐ FFA ☒ BFFAS ☐ BFFAG
D17	6.31	6.28	5.9	☐ FFA ☐ BFFAS ☒ BFFAG
D18	8.6	7.91	6.1	☐ FFA ☐ BFFAS ☒ BFFAG

Table 3. Table 4 shows the average number of features obtained by the proposed BFFAG and BFFAS approaches and the basic FFA basic.

Table 5 Classification accuracy of two proposed approaches (BFFAG and BFFAS) and original FFA basic

Dataset	FFA	BFFAS	BFFAG	BEST Approaches
D1	0.2333	0.2362	0.2101	☒ FFA ☐ BFFAS ☐ BFFAG
D2	0.9571	0.9257	0.9571	☒ FFA ☐ BFFAS ☒ BFFAG
D3	0.9158	0.9228	0.9298	☐ FFA ☐ BFFAS ☐ BFFAG
D4	0.6612	0.6831	0.8197	☐ FFA ☐ BFFAS ☒ BFFAG
D5	0.6720	0.7060	0.6440	☐ FFA ☒ BFFAS ☐ BFFAG
D6	0.5607	0.5794	0.9533	☐ FFA ☐ BFFAS ☒ BFFAG
D7	0.5385	0.5641	0.6410	☐ FFA ☐ BFFAS ☒ BFFAG
D8	0.5959	0.6986	0.6712	☐ FFA ☒ BFFAS ☐ BFFAG
D9	0.8295	0.8636	0. 8295	☐ FFA ☒ BFFAS ☐ BFFAG
D10	0.8125	0.6875	0.8125	☒ FFA ☐ BFFAS ☒ BFFAG
D11	0.5811	0.6757	0.5676	☐ FFA ☒ BFFAS ☐ BFFAG
D12	0.6791	0.6791	0.7164	☐ FFA ☐ BFFAS ☒ BFFAG
D13	0.6519	0.6012	0.5926	☒ FFA ☐ BFFAS ☐ BFFAG
D14	0.6076	0.6540	0.6540	☐ FFA ☒ BFFAS ☒ BFFAG
D15	0.8468	0.8426	0.8511	☐ FFA ☐ BFFAS ☒ BFFAG
D16	0.8084	0.7452	0.8084	☒ FFA ☐ BFFAS ☒ BFFAG
D17	0.7753	0.9663	0.9663	☐ FFA ☒ BFFAS ☒ BFFAG
D18	0.8039	0.7451	0.8431	☐ FFA ☐ BFFAS ☒ BFFAG

From Table 4, it can be seen that the BFFAG approach obtains the best performance in terms of feature selection. The BFFAG approach is about 89% more successful than the BFFAS and FFA on 18 datasets. In addition, the classification accuracy results obtained by the proposed BFFAG and BFFAS approaches and the original FFA are shown in Table 5.

The results in Table 5 show that the BFFAG approach obtains the best performance. This approach is about 67% more successful than the BFFAS and FFA approaches on 18 datasets. The results are shown in Figs. 7, 8 and 9.

From Figs. 7, 8 and 9, it is seen that the BFFAG approach achieves better convergence rate of objective function compared to the other methods. All the experiments in this section demonstrate the superiority of the proposed BFFAG approach over the BFFAS and the original FFA in various criteria including the average number of features, the classification accuracy, and the convergence rate of objective function. Therefore, this approach is selected as the main method and is compared with the existing meta-heuristic algorithms, in the Section 5.1.2. To further analysis, more experiments are performed to prove the superiority of the proposed BFFAG method, in the following section.

5.1.2 Further discussion

As shown in Section (5.1.1), the BFFAG approach is able to provide acceptable results in terms of convergence of the objective function, classification accuracy and average number of features compared to the BFFAS and FFA methods. In terms of feature selection, this algorithm also performs better in the high-dimensional dataset than the other algorithms. For example, in the D10 suite of 56 attributes, it is able to select an average of 22 attributes, and it also performs more successfully than other methods in the high-dimensional in accuracy criteria[21, 33, 34, 56]. In this sub-section, the stability of the proposed BFFAG and BFFAS approaches and original FFA is evaluated in different implementations. To this end, algorithms are implemented 5 times, and then the best objective function of each algorithm is obtained. The implementations are performed according to the parameters in Table 3. The results are shown in Figs. 10, 11 and 12.

The results of different implementations of the FFA in Fig. 10 show that the FFA achieves better stability in some datasets. On the other hand, some datasets, such as D14 and D10, show different results in each implementation. This experiment proves that the FFA has relatively moderate stability in five different implementations.

The results of the different implementation of the BFFAS algorithm in Fig. 11 show that the BFFAS algorithm achieves better stability in some datasets. On the other hand, some datasets, such as D14 and D18, show different results. Of course, the BFFAS algorithm performs much better than the

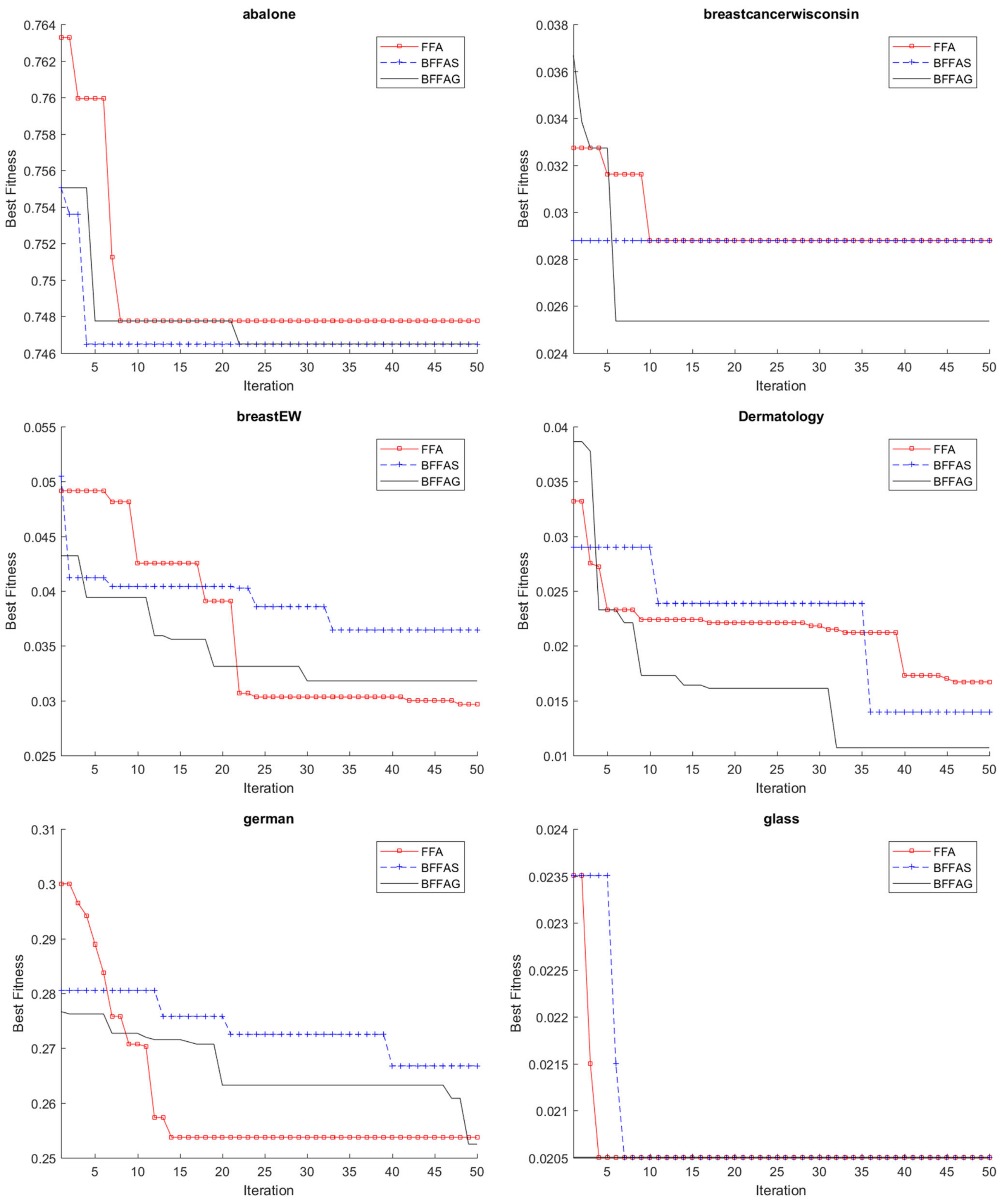


Fig. 7 Convergence rate of the objective function of two proposed approaches (BFFAG and BFFAS) and original FFA on D1: D6 dataset

original FFA. This experiment demonstrates that the BFFAS algorithm performs relatively well in five different implementations.

The results of different implementation of the BFFAG algorithm in Fig. 12 show that the BFFAG algorithm achieves better stability in most datasets. Only in the second run, the

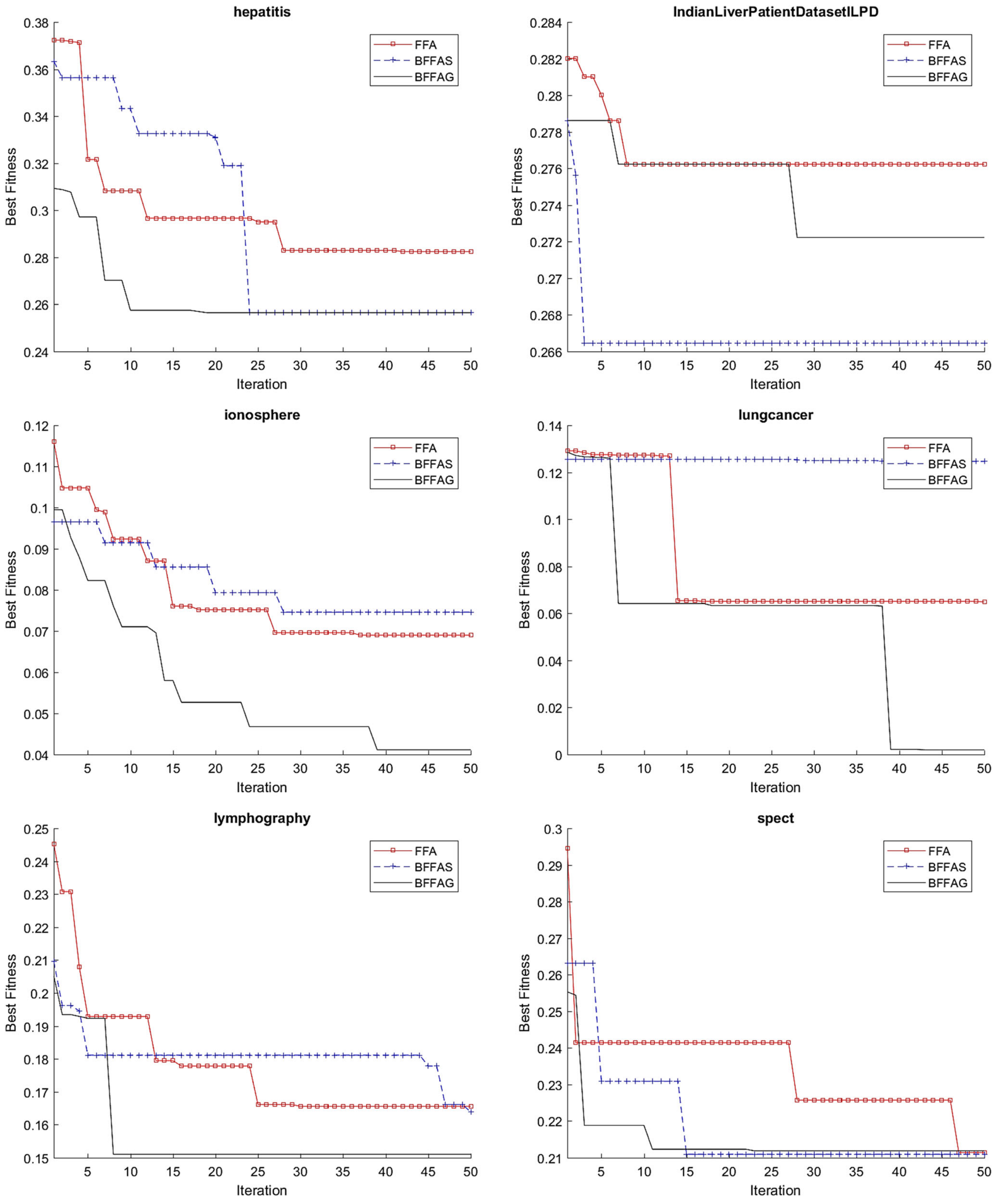


Fig. 8 Convergence of the objective function of two proposed approaches (BFFAG and BFFAS) and original FFA on D7: D12 dataset

BFFAG has a slightly different performance. The results of this experiment show the remarkable superiority of the BFFAG approach over the BFFAS and FFA approaches. The BFFAG approach also proved to be very stable. In the

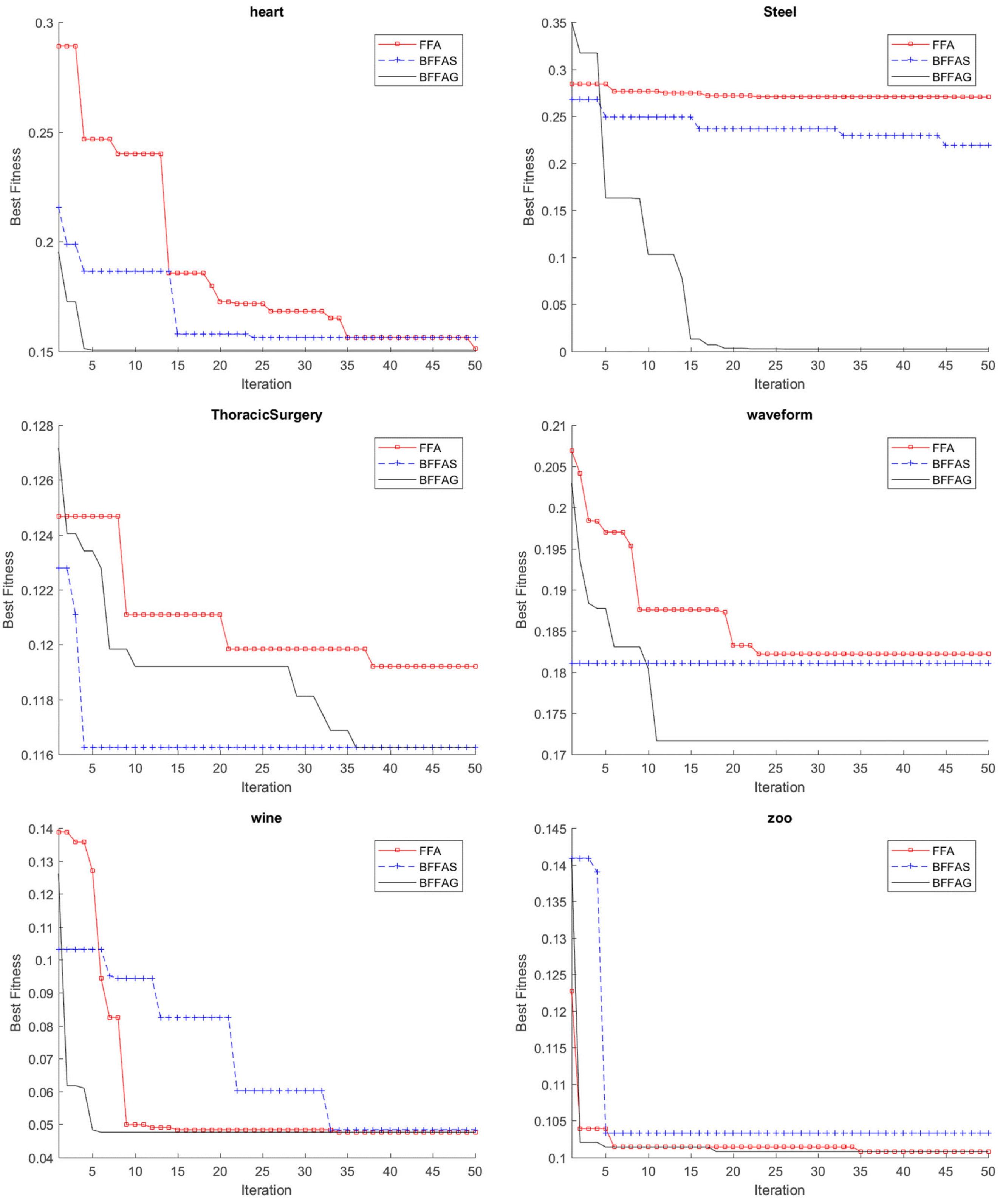


Fig. 9 Convergence rate of the objective function of two proposed approaches (BFFAG and BFFAS) and original FFA on D13: D18 dataset

following, two proposed BFFAG and BFFAS approaches, and original FFA are statistically investigated (best, worst, average, and standard deviation). The results of this investigation are shown in Table 6.

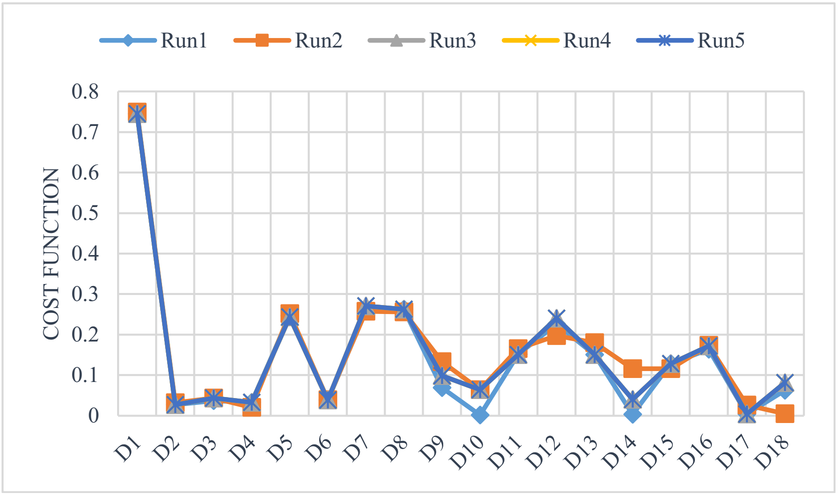


Fig. 10 Results of different implementation of the original FFA

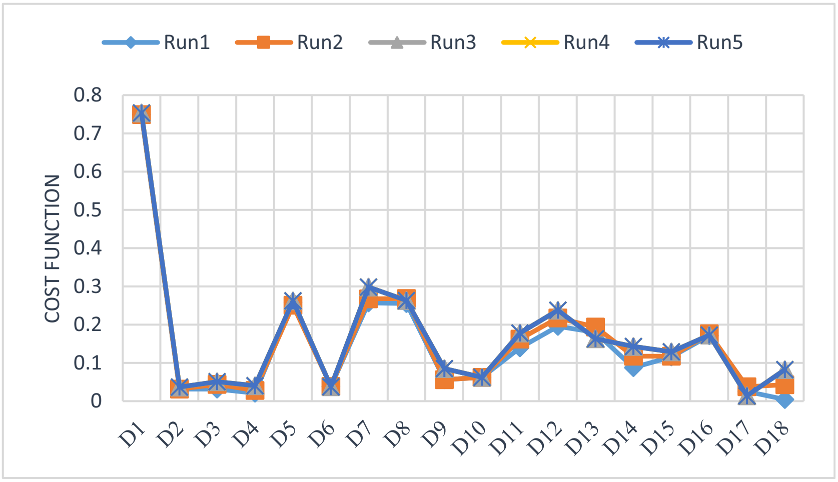


Fig. 11 Results of different implementation of BFFAS algorithm

From Table 6, it is seen that the BFFAG approach performs better than the BFFAS and FFA. The experiment proved that the proposed BFFAG approach is the best method in terms of statistical criteria.

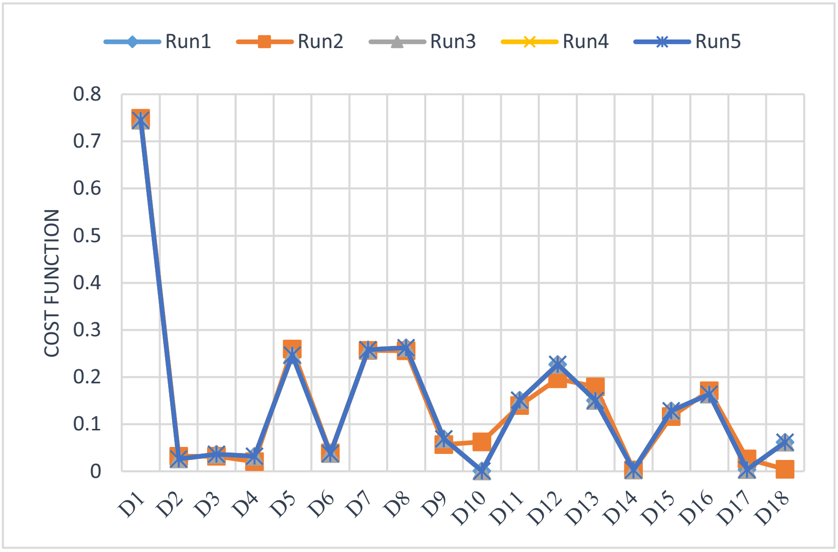


Fig. 12 Results of different implementation of BFFAG algorithm

Table 6 Statistical Comparison of proposed BFFAG and BFFAS Approaches and original FFA

Dataset	Algorithm	Best	Mean	Worst	STD
D1	FFA	**0.7452**	0.7747	**0.7929**	**0.0154**
	BFFAS	0.7492	0.7774	0.8023	0.0179
	BFFAG	**0.7452**	**0.7733**	0.8087	0.0215
D2	FFA	**0.0265**	0.0643	0.2019	0.0513
	BFFAS	0.0372	**0.0504**	0.0757	**0.0122**
	BFFAG	**0.0265**	0.0518	**0.0752**	0.0174
D3	FFA	0.0429	0.0846	0.1079	0.0193
	BFFAS	0.0510	0.0850	0.1074	0.0170
	BFFAG	**0.0366**	**0.0834**	**0.1046**	**0.0150**
D4	FFA	**0.0323**	0.1546	**0.2635**	**0.0777**
	BFFAS	0.0404	0.1754	0.2960	0.0827
	BFFAG	**0.0323**	**0.1448**	0.3512	0.0999
D5	FFA	0.2434	0.2969	0.3376	0.0323
	BFFAS	0.2623	0.3151	0.3498	0.0236
	BFFAG	**0.2419**	**0.2912**	**0.3268**	**0.0222**
D6	FFA	**0.038**	0.2124	0.5036	0.2145
	BFFAS	**0.038**	0.1946	0.4399	0.1824
	BFFAG	**0.038**	**0.1504**	**0.4101**	**0.1630**
D7	FFA	0.2707	0.4186	0.5394	0.0823
	BFFAS	0.2982	0.4126	0.5383	**0.0621**
	BFFAG	**0.2581**	**0.3859**	**0.4606**	0.0744
D8	FFA	**0.2627**	0.3186	0.3590	0.0290
	BFFAS	**0.2627**	0.3284	0.3678	0.0311
	BFFAG	**0.2627**	**0.3154**	**0.3566**	**0.0328**
D9	FFA	0.0986	0.1673	0.1897	**0.0201**
	BFFAS	0.0853	0.1596	**0.1801**	0.0212
	BFFAG	**0.0690**	**0.1519**	0.1965	0.0412
D10	FFA	0.0640	0.1775	**0.3147**	0.0822
	BFFAS	0.0624	0.1593	0.2529	**0.0608**
	BFFAG	**0.0011**	**0.1761**	**0.3147**	0.0971
D11	FFA	0.1505	0.3055	0.4855	0.1036
	BFFAS	0.1784	0.2802	**0.3802**	**0.0595**
	BFFAG	**0.1501**	**0.2659**	0.4181	0.0936
D12	FFA	0.2405	0.3472	0.4474	0.0679
	BFFAS	0.2378	0.3255	0.4048	0.0534
	BFFAG	**0.2266**	**0.3208**	**0.4019**	**0.0501**
D13	FFA	**0.1505**	0.3123	0.4334	0.0985
	BFFAS	0.1636	0.3172	0.3639	**0.0659**
	BFFAG	**0.1505**	**0.3018**	**0.4145**	0.0966
D14	FFA	0.0403	0.3472	0.3931	0.1089
	BFFAS	0.1428	0.3320	0.3917	**0.1068**
	BFFAG	**0.0030**	**0.2885**	**0.3866**	0.1515
D15	FFA	**0.1283**	0.1538	**0.1705**	**0.0139**
	BFFAS	0.1295	0.1524	0.1838	0.0179
	BFFAG	**0.1283**	**0.1516**	0.1735	0.0167
D16	FFA	0.1717	0.2408	0.2955	0.0417
	BFFAS	0.1732	0.2410	0.2974	0.0387
	BFFAG	**0.1638**	**0.2255**	**0.2693**	**0.0359**
D17	FFA	**0.0038**	0.2459	0.3836	0.1458
	BFFAS	0.0142	0.2391	0.4235	0.1353
	BFFAG	**0.0038**	**0.1815**	**0.4051**	**0.1292**
D18	FFA	0.0808	0.1984	0.3325	0.0750
	BFFAS	0.0826	0.1888	0.2361	0.0296
	BFFAG	**0.0620**	**0.1505**	**0.1810**	**0.0284**
RESULT($\frac{number\ of\ best}{total\ of\ Data}$)	FFA	$\frac{8}{18}$	$\frac{0}{18}$	$\frac{4}{18}$	$\frac{4}{18}$
	BFFAS	$\frac{2}{18}$	$\frac{1}{18}$	$\frac{2}{18}$	$\frac{6}{18}$
	BFFAG	$\frac{18}{18}$	$\frac{17}{18}$	$\frac{13}{18}$	$\frac{8}{18}$

5.1.3 Comparison with other approaches

In this sub-section, the proposed BFFAG approach is compared with the other binary meta-heuristic algorithms including Genetic Algorithm (GA) [47], Binary Bats Algorithm (BBA) [48], V-shaped Binary PSO (BPSO) [49], Binary Flower Pollination Algorithm (BFPA) [10], Binary Gray Wolf Optimizer (BGWO) [21], Binary Dragonfly Algorithm

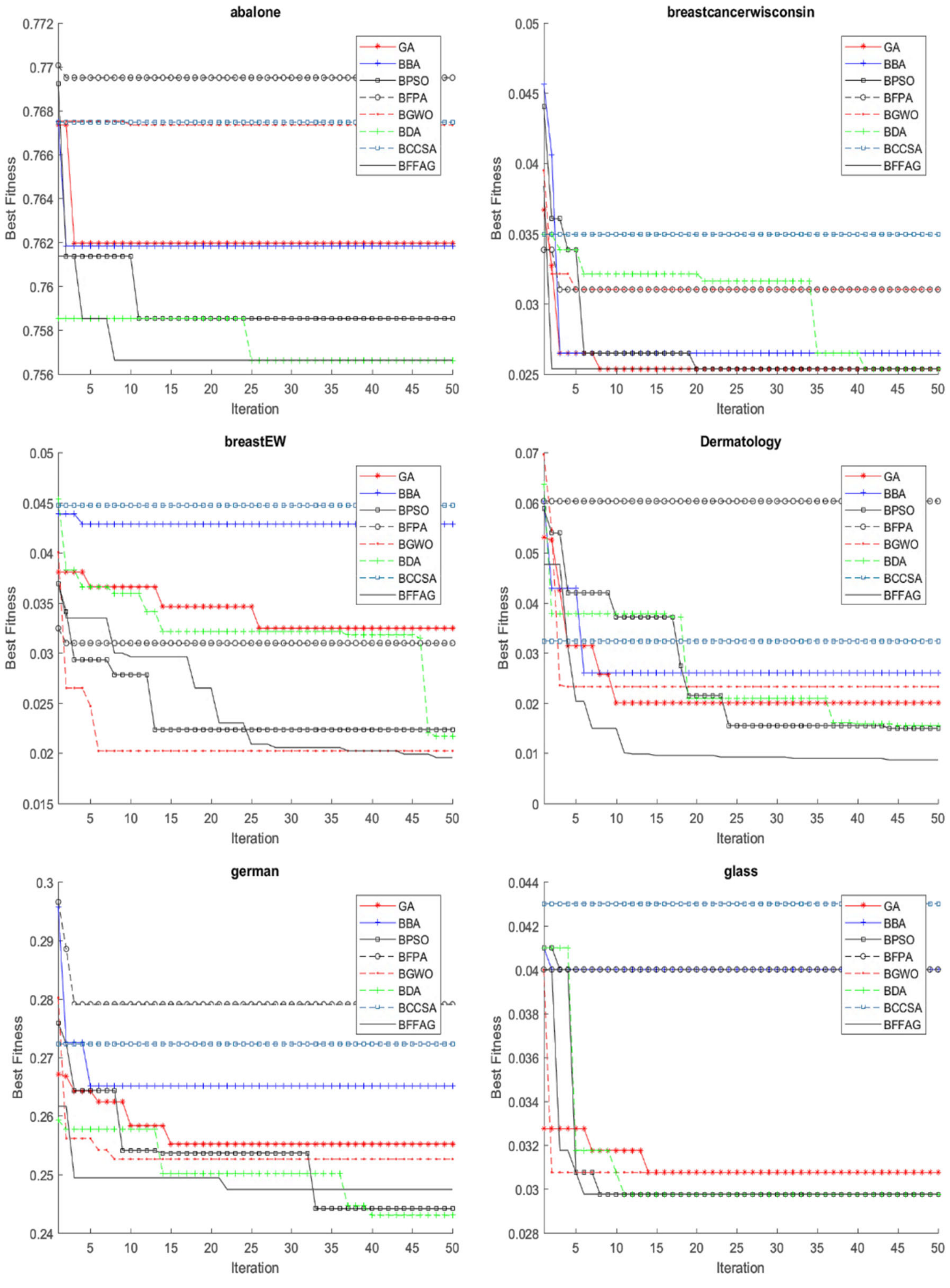


Fig. 13 Convergence of objective function of the proposed BFFAG and other algorithms on D1: D6 dataset

(BDA) [14] and Binary Chaotic Crow Search Algorithm (BCCSA) on 18 datasets. All experiments are performed according to the parameters in Table 3. Therefore, the number of iterations is set to 50 for all algorithms. The convergence results of the proposed BFFAG and other comparative algorithms on 18 datasets are shown in Figs. 13, 14 and 15.

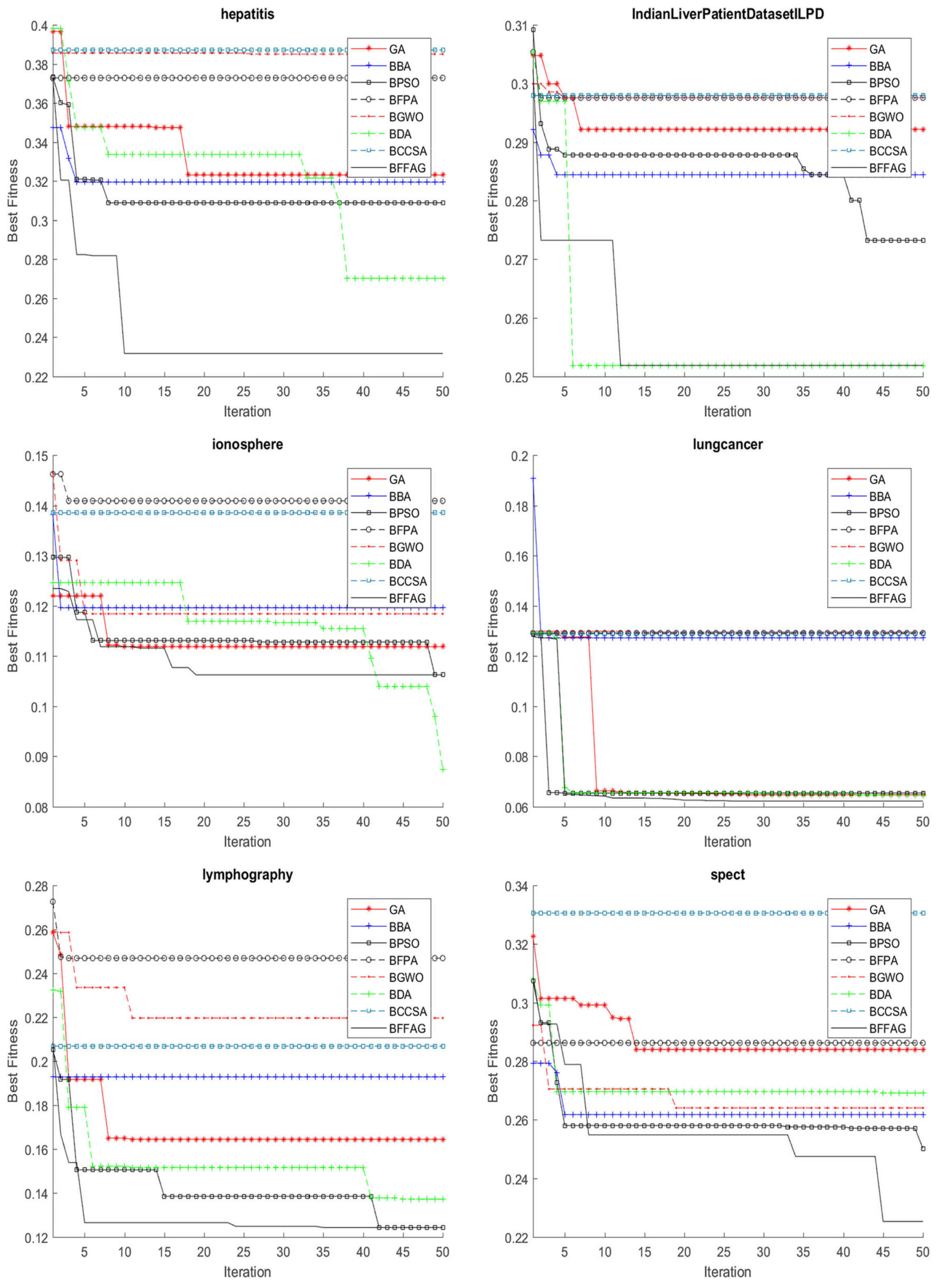


Fig. 14 Convergence of objective function of the proposed BFFAG and other algorithms on D7: D12 dataset with

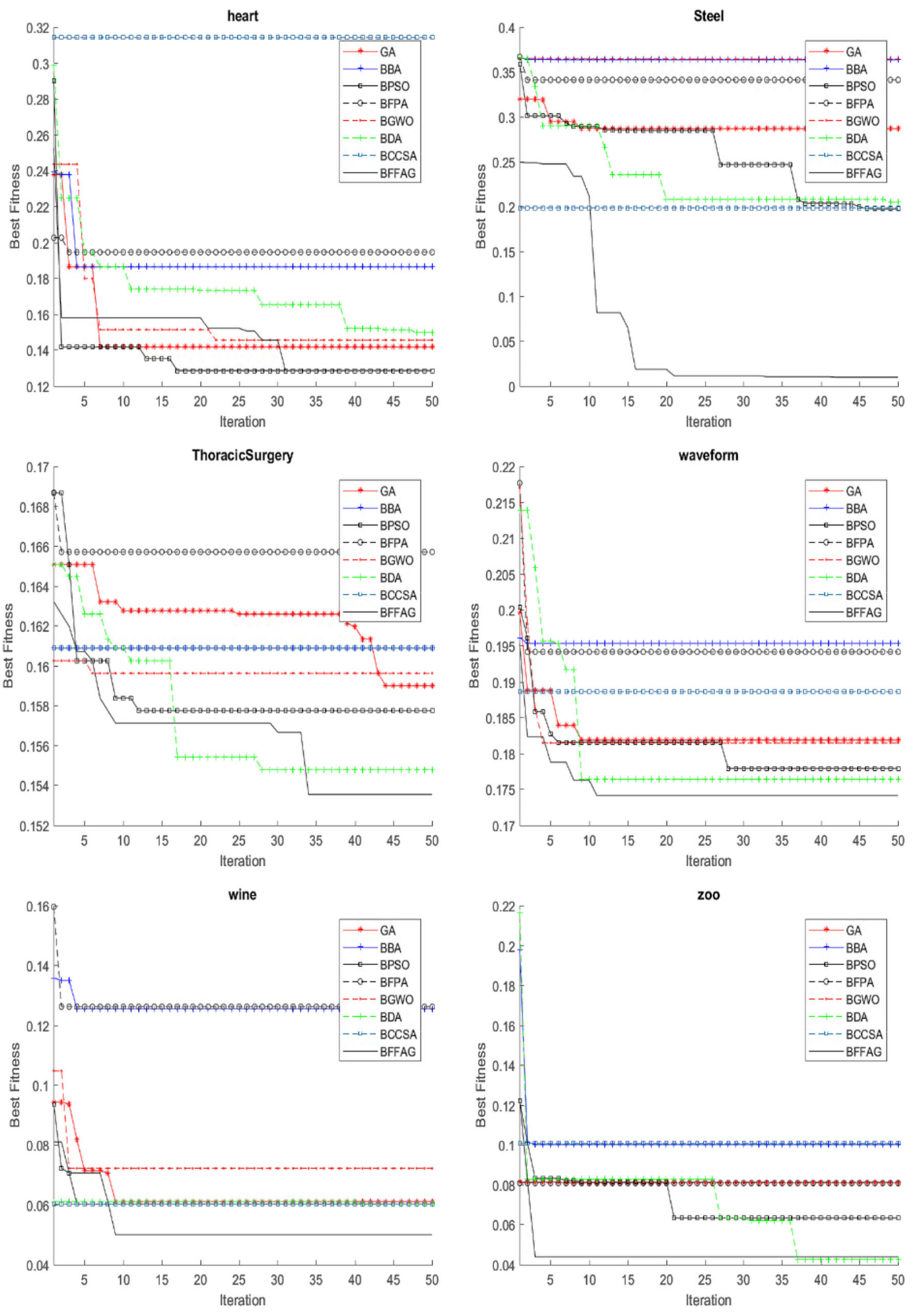


Fig. 15 Convergence of objective function of the proposed BFFAG and other algorithms on D13: D18 dataset with

From Figs. 13, 14 and 15, it can be seen that the proposed BFFAG approach is able to perform better overall. The BFFAG approach outperforms all algorithms in 15 datasets. The comparison among the algorithm show that the proposed

Table 7 Statistical comparison of the proposed BFFAG and other algorithms

Dataset	Criteria	GA	BBA	BPSO	BFPA	BGWO	BDA	BCCSA	BFFAG
D1	Best	0.762	0.7618	0.7585	0.7695	0.7674	**0.7566**	0.7675	**0.7566**
	Mean	0.762	0.7927	0.7736	0.7876	0.769	0.7577	0.7675	0.7817
	Worst	0.762	0.8112	0.7947	0.8026	0.7823	0.7674	0.7675	0.8022
	Std	0	0.0127	0.0143	0.0101	0.0047	0.0034	0	0.017
D2	Best	**0.0254**	0.0265	**0.0254**	0.0310	0.0310	**0.0254**	0.0350	**0.0254**
	Mean	0.0254	0.0538	0.0335	0.0487	0.0339	0.0307	0.0350	0.0458
	Worst	0.0254	0.0582	0.0486	0.0769	0.0378	0.0384	0.0350	0.0667
	Std	0	0.0049	0.0086	0.0116	0.0027	0.0058	0	0.0132
D3	Best	0.0325	0.0429	0.0224	0.0310	0.0202	0.0217	0.0447	**0.0196**
	Mean	0.0344	0.0613	0.0381	0.0528	0.0204	0.0409	0.0447	0.0594
	Worst	0.0346	0.0829	0.0533	0.0755	0.0206	0.0697	0.0447	0.1148
	Std	0.0007	0.0089	0.0067	0.0145	0.0002	0.0129	0	0.0289
D4	Best	0.0201	0.0261	0.0149	0.0603	0.0233	0.0155	0.0323	**0.0086**
	Mean	0.0201	0.1775	0.0658	0.1759	0.0246	0.0217	0.0323	0.1774
	Worst	0.0201	0.2416	0.329	0.2941	0.029	0.0474	0.0323	0.3563
	Std	0	0.0501	0.093	0.102	0.0023	0.0111	0	0.1437
D5	Best	0.2552	0.2651	0.2442	0.2791	0.2526	**0.2430**	0.2723	0.2474
	Mean	0.2552	0.3387	0.2937	0.3019	0.2556	0.2704	0.2723	0.3063
	Worst	0.2552	0.3586	0.3364	0.3219	0.2629	0.2976	0.2723	0.3559
	Std	0	0.0183	0.0311	0.0164	0.004	0.0125	0	0.0371
D6	Best	0.0308	0.0400	**0.0298**	0.0400	0.0308	**0.0298**	0.043	**0.0298**
	Mean	0.0308	0.3561	0.0331	0.1993	0.0363	0.0318	0.043	0.2124
	Worst	0.0308	0.5026	0.041	0.3668	0.042	0.0400	0.043	0.4924
	Std	0	0.1281	0.0052	0.1656	0.0056	0.0043	0	0.1888
D7	Best	0.3231	0.3194	0.3088	0.3728	0.3850	0.2702	0.3871	**0.2316**
	Mean	0.3231	0.4469	0.3571	0.4728	0.3895	0.3401	0.3871	0.4398
	Worst	0.3231	0.5368	0.421	0.5272	0.4119	0.4622	0.3871	0.5881
	Std	0	0.0454	0.0413	0.0464	0.0089	0.0691	0	0.1267
D8	Best	0.2922	0.2844	0.2732	0.2976	0.2976	**0.2519**	0.298	**0.2519**
	Mean	0.2922	10.3009	0.3144	0.3267	0.2985	0.2587	0.298	0.3171
	Worst	0.2922	100	0.3488	0.3586	0.3000	0.3197	0.298	0.3566
	Std	0	31.5171	0.0324	0.0197	0.0011	0.0214	0	0.0364
D9	Best	0.1119	0.1196	0.1063	0.1409	0.1184	**0.0873**	0.1385	0.1063
	Mean	0.1119	0.1699	0.1330	0.1695	0.1278	0.1306	0.1385	0.1542
	Worst	0.1119	0.2105	0.1628	0.198	0.1415	0.1664	0.1385	0.2128
	Std	0	0.0183	0.0207	0.0161	0.0091	0.0253	0	0.0302
D10	Best	0.0647	0.1271	0.0654	0.1291	0.1295	0.0646	0.1288	**0.0622**
	Mean	0.0647	0.2823	0.2144	0.2539	0.1297	0.1699	0.1287	0.1835
	Worst	0.0647	0.4986	0.3750	0.3165	0.1300	0.3126	0.1288	0.2532
	Std	0	0.1019	0.0885	0.0503	0.0001	0.0719	0	0.0699
D11	Best	0.1644	0.1929	**0.1243**	0.2469	0.2196	0.1371	0.2068	**0.1243**
	Mean	0.1644	0.3907	0.1978	0.3005	0.2202	0.156	0.2068	0.2862
	Worst	0.1644	0.5245	0.3378	0.3813	0.2207	0.2447	0.2068	0.3913
	Std	0	0.067	0.0697	0.0521	0.0004	0.0343	0	0.0949
D12	Best	0.2839	0.2618	0.2502	0.2862	0.2640	0.2692	0.3305	**0.2253**
	Mean	0.2839	0.3553	0.309	0.3708	0.2842	0.282	0.3305	0.332
	Worst	0.2839	0.4303	0.3541	0.4331	0.3227	0.3139	0.3305	0.3887
	Std	0	0.0294	0.026	0.0302	0.0183	0.0173	0	0.0532
D13	Best	0.1416	0.1864	**0.1285**	0.1945	0.1455	0.1497	0.3142	**0.1285**

Table 7 (continued)

Dataset	Criteria	GA	BBA	BPSO	BFPA	BGWO	BDA	BCCSA	BFFAG
	Mean	0.1416	0.3508	0.202	0.371	0.1521	0.1751	0.3142	0.2993
	Worst	0.1416	0.4863	0.4527	0.4292	0.1594	0.2906	0.3142	0.3875
	Std	0	0.0798	0.1103	0.0284	0.0047	0.0456	0	0.0948
D14	Best	0.2867	0.3635	0.1971	0.3411	0.3641	0.2046	0.1984	**0.0099**
	Mean	0.2867	0.3825	0.2993	0.3786	0.3645	0.2506	0.1984	0.3166
	Worst	0.2867	0.4132	0.4049	0.4165	0.3648	0.4101	0.1984	0.4138
	Std	0	0.0226	0.0645	0.0236	0.0001	0.0734	0	0.1636
D15	Best	0.159	0.1609	0.1577	0.1657	0.1596	0.1548	0.1609	**0.1535**
	Mean	0.1591	0.179	0.163	0.1699	0.1616	0.1638	0.1609	0.1682
	Worst	0.1596	0.1975	0.1716	0.1766	0.1638	0.1788	0.1609	0.1826
	Std	0.0002	0.0115	0.0046	0.0054	0.002	0.0091	0	0.0092
D16	Best	0.1819	0.1953	0.1779	0.1941	0.1815	0.1764	0.1886	**0.1741**
	Mean	0.1819	0.2455	0.1973	0.2147	0.1853	0.1893	0.1886	0.2241
	Worst	0.1819	0.307	0.2169	0.2509	0.191	0.2091	0.1886	0.2645
	Std	0	0.0276	0.0134	0.0186	0.003	0.0119	0	0.0308
D17	Best	0.061	0.1254	0.0602	0.1262	0.0721	0.0602	0.0602	**0.0499**
	Mean	0.061	0.2637	0.0777	0.2776	0.073	0.0949	0.0602	0.2309
	Worst	0.061	0.3694	0.1922	0.3479	0.0737	0.3168	0.0602	0.3176
	Std	0	0.0752	0.0412	0.0569	0.0004	0.0801	0	0.1033
D18	Best	0.0808	0.1002	0.0632	0.0808	0.0814	**0.0426**	0.1008	0.0438
	Mean	0.0808	0.2422	0.1817	0.2429	0.0996	0.093	0.1008	0.247
	Worst	0.0808	0.3507	0.355	0.355	0.1215	0.1979	0.1008	0.355
	Std	0	0.0809	0.1018	0.1055	0.0145	0.0511	0	0.1103
RESULT	Best	1 .05%	0 0.0%	4 22%	0 0.0%	0 0.0%	7 38%	0 0.0%	15 83%

BFFAG approach is 83% more convergent than other algorithms. However, to further evaluate the results of the proposed BFFAG and other comparative algorithms on 18 datasets, the statistical criteria are given in Table 7.

Table 7 shows that the "Best" criterion of the proposed BFFAG method performs 83% better than other algorithms. However, the proposed BFFAG approach is failed to minimize the parameters such as standard deviation, due to in-process exploration. In some datasets such as D14, the proposed BFFAG approach with a "Best" value of 0.0099 works much better, and certainly, this algorithm cannot reduce other parameters simultaneously. Table 8 presents the results of the proposed approach and other algorithms in terms of the average number of features.

Table 8 shows that the BFFAG approach performs very well, and proves its superiority on 9 datasets. It should be noted that both the number of features and the classification accuracy are taken into account in the objective function. Therefore the minimum number of features cannot be obtained in all datasets. The classification accuracy of BFFAG and other comparative algorithms are given in Table 9.

From Table 9, it is seen that the BFFAG approach shows high efficiency in terms of the classification accuracy. Our approach performed better in terms of classification accuracy on 9 datasets and is close to the best results in the other datasets. This is because that both the feature selection and the classification accuracy are considered in the objective function. Therefore, we cannot content with classification accuracy. Experiments in this section prove that the proposed approach has good binary space exploration and efficiency. Thus, the superiority of the BFFAG in feature selection and classification accuracy over other methods is obvious.

5.1.4 Further discussion

In this section, the results of the proposed BFFAG and other algorithms are discussed with further experiments. To this end, the proposed BFFAG and other comparative algorithms are implemented 5 times and then the best objective function of each algorithm is obtained. The implementations are performed according to the parameters in Table 3. The purpose of this experiment is to evaluate the robustness of the proposed BFFAG and other comparative algorithms in different

Table 8 Average number of features of the proposed BFFAG and other algorithms

Dataset	GA	BBA	BPSO	BFPA	BGWO	BDA	BCCSA	BFFAG
D1	6	4	4.6	4.8	4.6	4.1	7	**3.5**
D2	5	5.1	**4.8**	5.6	5.6	5	6	**4.8**
D3	**10.4**	14	16.4	18.7	19.6	13.2	30	13.5
D4	13	13.1	14.1	20.5	24.8	16.6	18	13.6
D5	9	**7.8**	11.1	15.6	17	12.7	12	11.8
D6	5	3.6	3.6	6.9	3.9	**3.2**	6	**3.2**
D7	11	7.4	8	13.2	9.4	**7.1**	12	**7.1**
D8	4	3.2	3.8	6.3	5.9	5	3	**2.2**
D9	17	**9.7**	18.2	21.7	21.3	9.8	12	15.4
D10	16.1	21.7	22.5	36.1	33.3	16.1	28	**16**
D11	7	8.4	**6.6**	11.1	11	7.3	11	8.7
D12	10.2	9.5	13.4	14.4	10.9	9.6	12	**8.3**
D13	**3**	4.2	5.2	8.5	7	4.5	8	6.8
D14	14	12.3	13.1	20.2	15.1	9.6	12	**11.2**
D15	5.1	5.6	4.7	9.6	5.8	**4.6**	8	6.2
D16	16	**10.7**	15.3	14	16.6	16.1	21	12
D17	7	**4.4**	5.6	8	8.1	6.3	6	6.6
D18	5	5.5	8.1	9.8	7.2	6	6	**5.1**

implementations. The results of the different implementations of the GA are shown in Fig. 16.

From Fig. 16, it can be seen that the GA achieves better stability in the datasets of {D1: D9, D12, D15, D16, D17}. But it is less stable in the data sets of {D10, D11, D13, D14, and D18}. These results prove that the GA has moderate stability and the algorithm is more influenced by two parameters of mutation and compound. Fig. 17 shows the results of different implementations of the BBA.

Fig. 17 shows that the BBA achieves better stability in the datasets of {D1: D9, D15, D16}, but in datasets of {D10, D11, D12, D13, D14, D18} exhibits less stability. These results

Table 9 Comparison of the proposed approach and other algorithms in terms of classification accuracy

Dataset	GA	BBA	BPSO	BFPA	BGWO	BDA	BCCSA	BFFAG
D1	0.2231	0.2165	0.2389	0.2068	0.2312	0.2408	0.2336	**0.2379**
D2	**0.9800**	0.9486	**0.9800**	0.9514	0.9743	**0.9800**	0.9714	**0.9800**
D3	0.9719	0.9544	0.9825	0.9544	**0.9860**	0.9825	0.9649	0.9614
D4	0.9836	0.8907	0.9891	0.9399	0.9836	0.9891	0.9727	**0.9836**
D5	0.746	0.642	0.758	0.704	0.752	0.76	0.73	0.74
D6	**0.972**	0.6542	**0.972**	0.6729	**0.972**	**0.972**	0.9626	**0.972**
D7	0.6795	0.5641	0.6923	**0.7436**	0.6154	0.7308	0.6154	0.7051
D8	**0.7500**	0.6678	0.726	0.661	0.7055	0.7466	0.7021	0.6849
D9	0.892	0.8466	0.8977	0.8125	0.8864	0.9034	0.8636	**0. 9148**
D10	0.9375	0.8125	0.9375	0.6875	0.875	**0.9375**	0.875	0.875
D11	0.8378	0.5946	0.8784	0.6622	0.7838	0.8649	0.7973	0.726
D12	0.7164	0.6791	0.7537	0.694	0.7388	0.7313	0.6716	**0.7612**
D13	0.8593	0.7333	0.8741	0.6444	0.8593	0.8519	0.6889	0.8444
D14	0.7147	0.6365	**0.8043**	0.6354	0.6365	0.7961	0.8033	0.8033
D15	0.8383	0.8298	0.8426	0.8213	0.8426	0.8468	0.8426	**0.8426**
D16	0.8240	0.7728	0.828	0.8036	0.8244	**0.8300**	0.8196	0.8260
D17	**0.9438**	0.7303	**0.9438**	0.6742	0.9326	**0.9438**	**0.9438**	**0.9438**
D18	0.9216	0.8824	0.9412	0.6667	0.9216	0.9608	0.902	**0.9804**

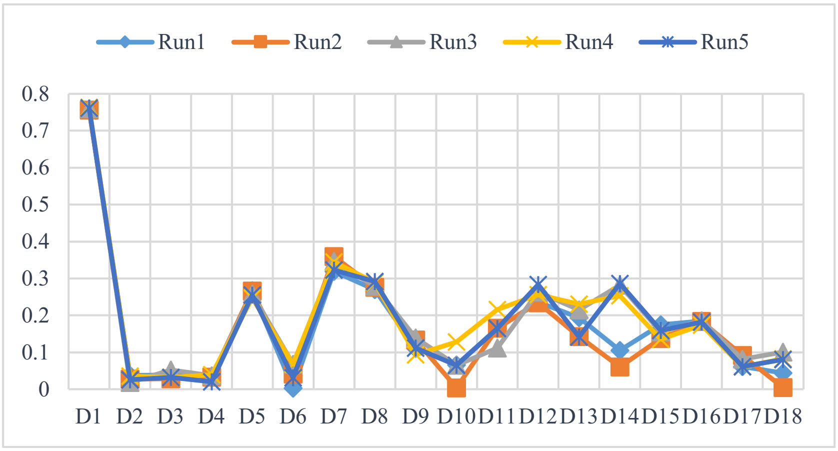


Fig. 16 Results of different implementations of GA

prove that the BBA approach has relatively moderate stability, and its results are less reliable. Fig. 18 shows the results of the different implementations of the BPSO algorithm.

From Fig. 18, it can be seen that the BPSO achieves better stability in the datasets of {D1: D9, D12, D15, D17}. But it is less stable in the datasets of {D1, D10, D11, D13, and D14}. However, in the D14 dataset, this algorithm performs the worst possible. These results prove that the BPSO has moderate stability and results in some datasets are less reliable. Fig. 19 shows the results of different implementations of the BFPA.

From Fig. 19, it can be seen that the BFPA achieves better stability in the datasets of {D1: D9, D11, D12, D14, D16}. But it is less stable in the datasets of {D10, D13, D17, and D18}. However, in the D10 dataset, this algorithm performs the worst possible. These results prove that the BFPA has better stability than BBA and BPSO. Fig. 20 shows the results of different implementations of the BGWO.

From Fig. 20, it can be seen that the BGWO achieves better stability in the datasets of {D1: D6, D8, D9, D12, D15, D18}. But it is less stable in the datasets of {D10, D11, D13, and D14}. These results prove that the BGWO has better stability than GA, BPSO, and BBA and its results are more acceptable. Fig. 21 shows the results of different implementations of the BDA.

From Fig. 21, it can be seen that the BDA achieves better stability in the datasets of {D1: D6, D8, D9, D12, D15, D17}. But it is less stable in the datasets of {D10, D11, D13, D14 and D18}. These results prove that the BDA has better stability than GA, BPSO, and BBA and its results are more acceptable. Fig. 22 shows the results of different implementations of the BCCSA.

From Fig. 22, it can be seen that the BCCSA achieves better stability in the datasets of {D1: D6, D8:D12, D15, D16, and D17}. But it is less stable in the datasets of {D7, D13, D14 and D18}. These results prove that the BCCSA has relatively good stability compared to other algorithms. Fig. 23 shows the results of different implementations of the BFFAG. From Fig. 23, it can be seen that the BFFAG achieves better stability in the datasets of {D1: D6, D8, D9, D11, and D18}. But it is less stable in the datasets of {D7, D10}. These results prove that the BFFAG has relatively stronger stability than other algorithms. Fig. 24 shows the results of the average of the five performances.

From Fig. 24, it can be seen that the proposed BFFAG approach is a reliable and high-performance method in feature

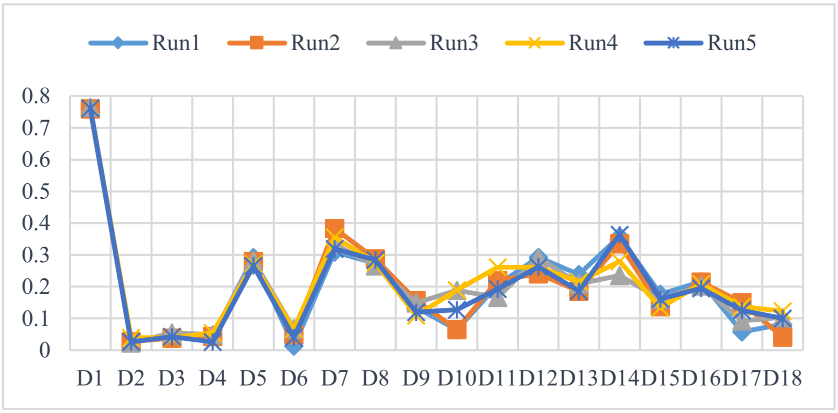


Fig. 17 Results of different implementations of the BBA

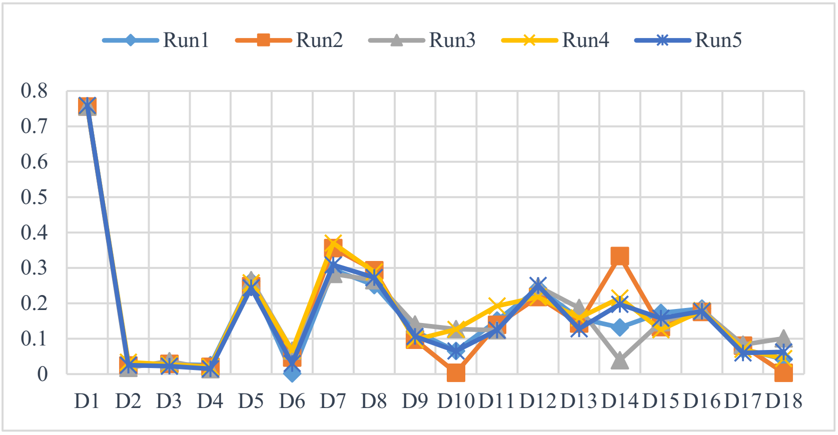


Fig. 18 Results of different implementations of the BPSO algorithm

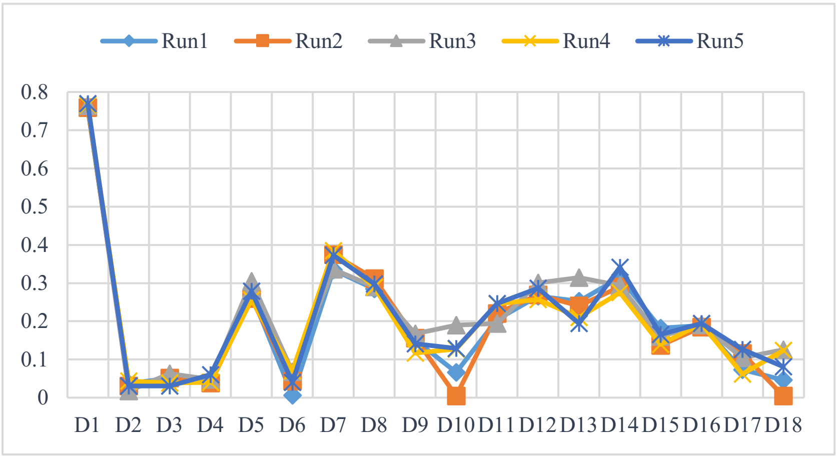


Fig. 19 Results of different implementation of BFPA algorithm

selection compared to other methods.This algorithm has a significant advantage over other methods.

5.2 T-test

The t-test as a type of inferential statistics is used to determine the significant difference between the mean of a group with a default value or the means of two groups. There are different types of t-tests, the three most common of which are:

1. One-Sample t-test (This test is used to examine the average of a community.)
2. Paired-samples t-test (This test is used to examine two means of a community.)
3. Independent-Samples t-test (This test is used to examine the mean of two independent communities.)

In this article, we utilize the Paired-samples t-test in SPSS software. Our purpose is to show the effect of the proposed algorithm on improving the evaluation parameters. To this

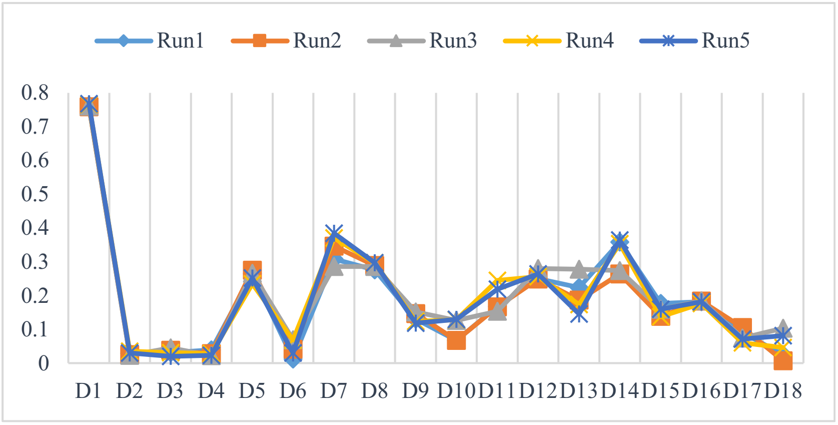


Fig. 20 Results of different implementation of BGWO algorithm

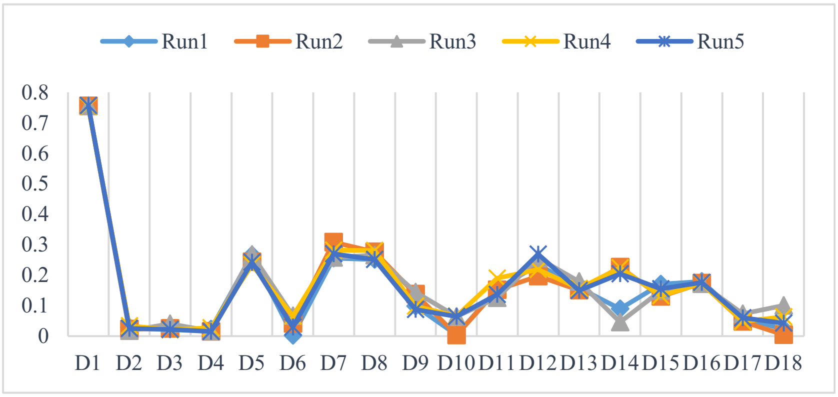


Fig. 21 Results of different implementation of BDA algorithm

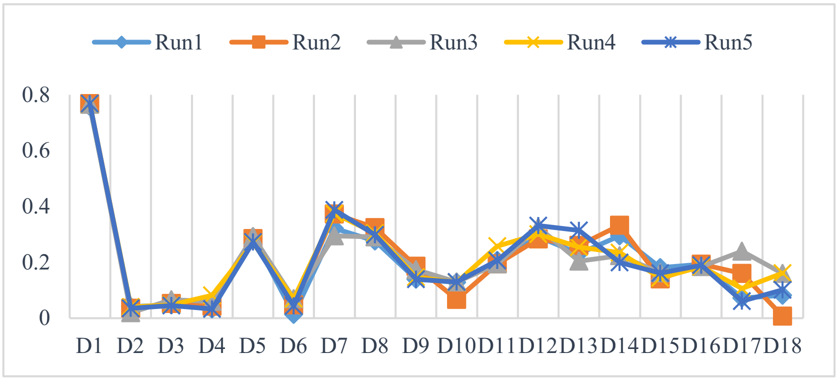


Fig. 22 Results of different implementation of the BCCSA algorithm

end, two related groups are considered, one is the proposed method and the other is one of the comparable algorithms. The optimal mean value of the proposed methods (BFFAG and BFFAS) and the other competing methods are extracted based on the Tables 6 and 7, and then their mean difference are obtained in pairs. The results of the t-test are presented in Table 10.

Moreover, the average number of selected features of the proposed method (BFFAG) and several other algorithms are extracted from Table 8 and then their mean difference are obtained in pairs. The results of the t-test related to the evaluation of the average number of selected features are shown in Table 11.

From Tables 10 and 11, we can find that the significance level (Sig.) is less than the test level of 0.05 in most cases, and so the null hypothesis is rejected. In other words, the results show that there is a significant difference in the mean value of the objective function (Table 10) as well as the average number of selected features (Table 11) of the proposed method in different datasets compared to the other methods. According

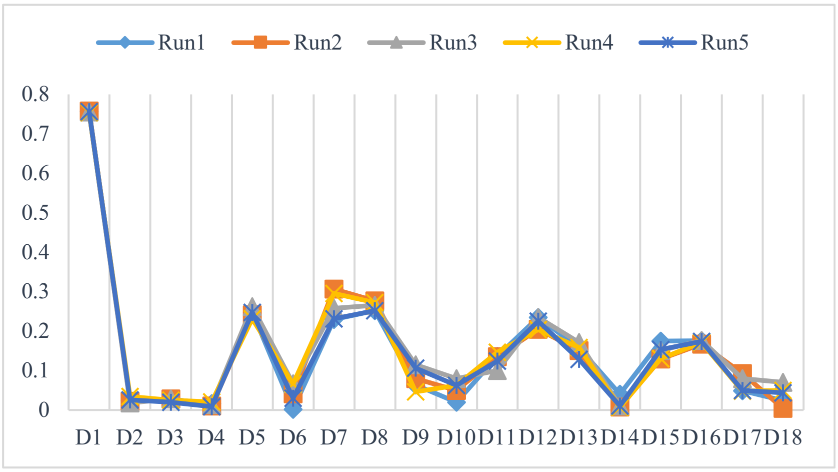
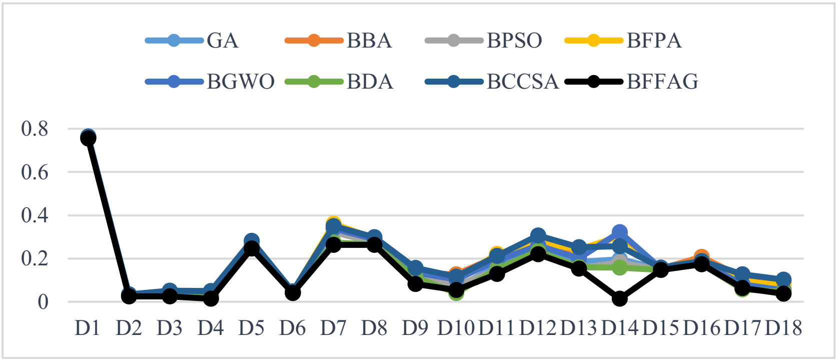


Fig. 23 Results of different implementation of the proposed BFFAG approach

Fig. 24 Comparison of the proposed BFFAG approach with other algorithms in terms of the mean of different performances



to the number of means after improving the method, the mean of the objective function (Mean) of the proposed method is negative, i.e. it decreases compared to most of the competing methods. Therefore, it can be concluded that the new algorithm reduces the optimal mean.

5.3 Experiments on emotion analysis dataset

In this sub-section, the performance of the proposed approaches is evaluated on a large dataset with more features. To this end, the emotion analysis dataset as one of the largest datasets is used. Here, we have used the popular emotion analysis data in IMDB Large Movie Review Dataset, where the opinions of 50,000 IMDB users are stored. This dataset contains two classes as negative and positive (i.e. positive and negative opinions). Because it takes a long time to analyze this dataset, researchers select a percentage of it and perform the necessary tests. In this paper, 2000 training data and 200 test data were used. The purpose of using this data is to construct a model that can correctly classify the given test texts into one of two classes of positive or negative. On the other hand, we need a dataset with features. Here, we use the common natural language processing libraries such as nltk, scikit-learn, and regular-expression (re).

In this experiment, the textual data is first read in Python programming language, and each of the opinions is classified into positive (1) and negative (0) groups. With the re and nltk libraries, punctuation marks are removed from the texts. At this point, the texts are converted to a list of separate words using the word_tokenize module. To extract the root of the words and remove the words' extras, LancasterStemmer is used as a word rooter module. This process is applied to the training and test data. We then use the TfidfVectorizer module in the nltk library to convert the resulting words into the input features for the classification algorithms. Each text in the algorithm is mapped to a vector according to the following points:

- The number of words used in the text.
- The repetition rate of each word in the text.
- The location of a particular word in the text relative to the other words and the adjacent words.
- Total number of unique words in all text collections.

Table 10 Paired Samples Test for Mean

		Paired Differences					t	df	Sig.(2-tailed)
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
					Lower	Upper			
Pair 1	BFFAG - FFA	−.0227944	.0224919	.0053014	−.0339794	−.0116095	−4.300	17	.000
Pair 2	BFFAS - FFA	−.0048111	.0130468	.0030752	−.0112991	.0016769	−1.565	17	.136
Pair 3	BFFAG - GA	.0834833	.0631142	.0148762	.0520974	.1148693	5.612	17	.000
Pair 4	BFFAG - BBA	−.5893278	2.3449170	.5527022	−1.7554276	.5767720	−1.066	17	.301
Pair 5	BFFAG - BPSO	.0498556	.0571544	.0134714	.0214333	.0782778	3.701	17	.002
Pair 6	BFFAG - BFPA	−.0185111	.0281459	.0066341	−.0325077	−.0045145	−2.790	17	.013
Pair 7	BFFAG - BGWO	.0642278	.0640673	.0151008	.0323679	.0960877	4.253	17	.001
Pair 8	BFFAG - BDA	.0735944	.0586976	.0138352	.0444048	.1027841	5.319	17	.000
Pair 9	BFFAG - BCCSA	.0596889	.0623222	.0146895	.0286968	.0906810	4.063	17	.001

Table 11 Paired Samples Test for average feature selection

		Paired Differences					t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
					Lower	Upper			
Pair 1	BFFAG - GA	−.4333333	2.3154214	.5457501	−1.5847653	.7180987	−.794	17	.438
Pair 2	BFFAG - BBA	.3222222	2.3979294	.5651974	−.8702400	1.5146845	.570	17	.576
Pair 3	BFFAG - BPSO	−1.2833333	2.3522830	.5544384	−2.4530961	−.1135705	−2.315	17	.033
Pair 4	BFFAG - BFPA	−4.9444444	4.4008318	1.0372860	−7.1329266	−2.7559623	−4.767	17	.000
Pair 5	BFFAG - BGWO	−3.9500000	4.3355236	1.0218927	−6.1060052	−1.7939948	−3.865	17	.001
Pair 6	BFFAG - BDA	−.0444444	2.1861118	.5152715	−1.1315723	1.0426834	−.086	17	.932
Pair 7	BFFAG - BCCSA	−3.4444444	4.7669477	1.1235804	−5.8149918	−1.0738971	−3.066	17	.007

After a complete preprocessing, 3675 features are built using the TfidfVectorizer algorithm, and also a feature is selected to determine the positive and negative classes. Here we use basic algorithms including GA, BBA, BPSO, BFPA, BGWO, BDA and BCCSA to compare with the proposed BFFAG approach. The Application architecture of the proposed approach is shown in Fig. 25.

Figure 26 shows convergence rate of the objective function of two proposed approaches and other comparative algorithms on the preprocessing with 2200 samples and 3675 features with different iterations.

From Figs. 26 and 27, it is seen that the BFFAG approach exhibits better convergence in 7 of 10 iterations, indicating that this approach performs better by increasing the number of iterations. This approach, however, performs poorly in two iterations of 20 and 80. Table 12 shows the results of the proposed and other algorithms in terms of the average number of selected features.

This table shows that the BFFAG approach performs very well. This approach has proven to work well in most iterations. The BFFAS approach also shows acceptable results compared to the FFA. Fig. 28 compares classification accuracy of two proposed approaches with that of other comparative algorithms in different number of iterations on the emotion analysis dataset.

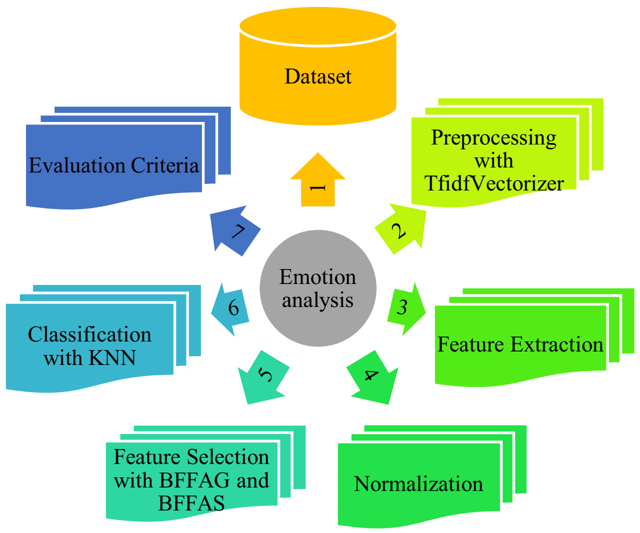


Fig. 25 Application Architecture

As can be seen from Fig. 28, the classification accuracy of the BFFAG approach is higher than that of the other methods, in addition to the better feature selection shown in Table 12. This is because that both feature selection and classification accuracy are considered in the objective function. According to Figs. 26, 27 and Table 12, it can be seen that the BFFAG approach outperforms other methods in terms of convergence, classification accuracy and the objective function on dataset emotional analysis.

6 Conclusions and future work

The FFA is a new meta-heuristic algorithm that is powerful in solving optimization problems due to its local and global memories and division of solutions into good and bad. In this paper, two new binary versions of FFA, namely BFFAS and BFFAG, was proposed for feature selection problem. In the proposed BFFAS and BFFAG approaches, respectively, the sigmoid transfer function and new genetic operators were introduced to convert the continuous FFA into binary versions. The proposed genetic operators, namely BGMU and BLMU, were used to update binary solutions based on global memory and total search space, and local and global memory, respectively. Also, the DM operator was utilized to maintain exploration in the FFA, rather than the parameter w. These three operators increase the performance of the original FFA. At the same time, the number of parameters of the FFA are reduced.

The experiments were implemented on 18 datasets from UCI datasets. Experimental results showed that our BFFAG approach outperforms the BFFAS and FFA in most criteria.

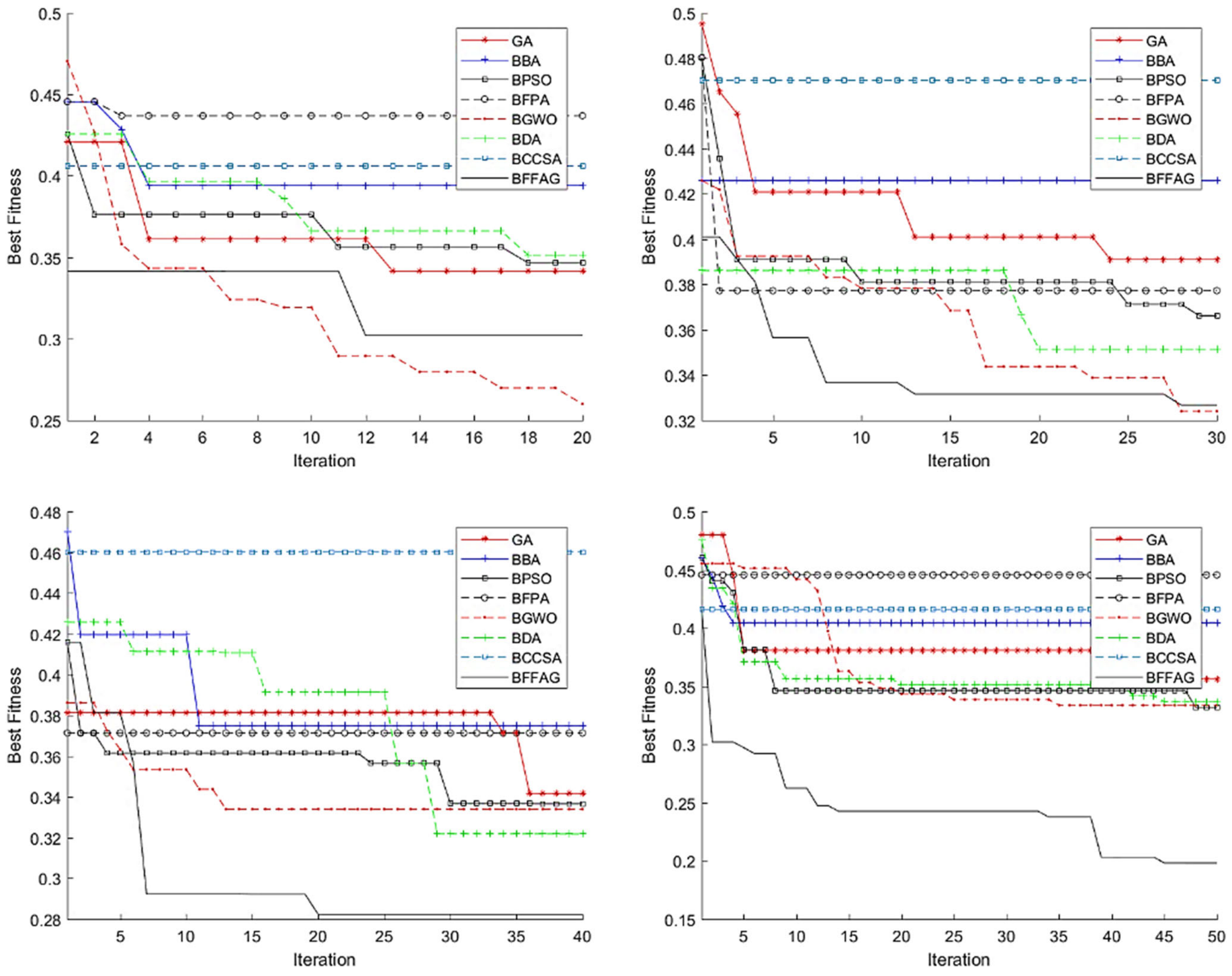


Fig 26 Convergence rate of the proposed approach and other algorithms on low iteration emotion analysis datasets

Furthermore, the BFFAG approach achieves better results in the objective function value, average number of features and classification accuracy compared to the state-of-the-art algorithms, namely GA, BBA, BPSO, BFPA, BGWO, BDA, and BCCS, on most of the datasets. We used t-test to better compare and analyze outputs of the methods. The t-test results showed that there is a significant difference in the mean value of the objective function as well as the average number of selected features of the proposed method in different datasets compared to the other methods.

Table 12 Average number of selected features of two proposed approaches and other algorithms on emotion analysis dataset

Iteration	GA	BBA	BPSO	BFPA	BGWO	BDA	BCCSA	BFFAG
10	1824	1712	1853	2350	2317	1673	1807	1641
20	1809	1742	1852	2371	2793	1781	1863	1842
30	1828	1802	1810	2359	2671	1762	1854	1841
40	1858	1892	1842	2354	2691	1836	1852	1780
50	1797	1897	1794	2343	2561	1862	1868	1797
60	1846	1820	1835	2337	2928	1897	1831	1820
70	1860	1890	1820	2375	2895	1746	1834	1834
80	1853	1856	1869	2352	2938	1708	1834	1813
90	1831	1814	1800	2352	2698	1831	1834	1802
100	1815	1789	1803	2359	2894	1813	1842	1726

Moreover, the proposed BFFAG approach was implemented on the popular IMDB Large Movie Review Dataset. In this case, 3675 attributes were extracted from this database to compare algorithms in high dimensions. The results of this

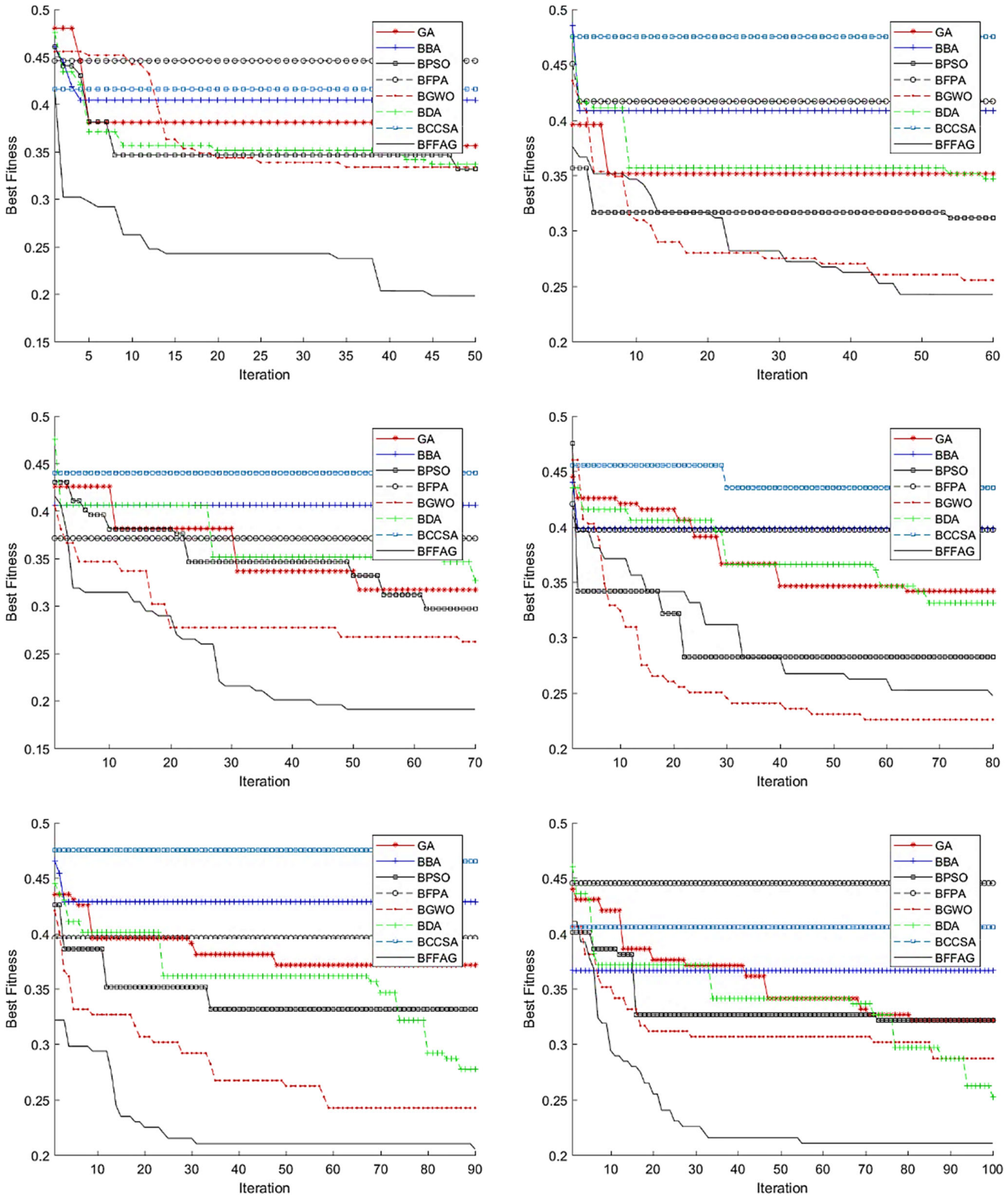


Fig. 27 The convergence rate of the proposed approach and other algorithms on the emotion analysis dataset with further iteration

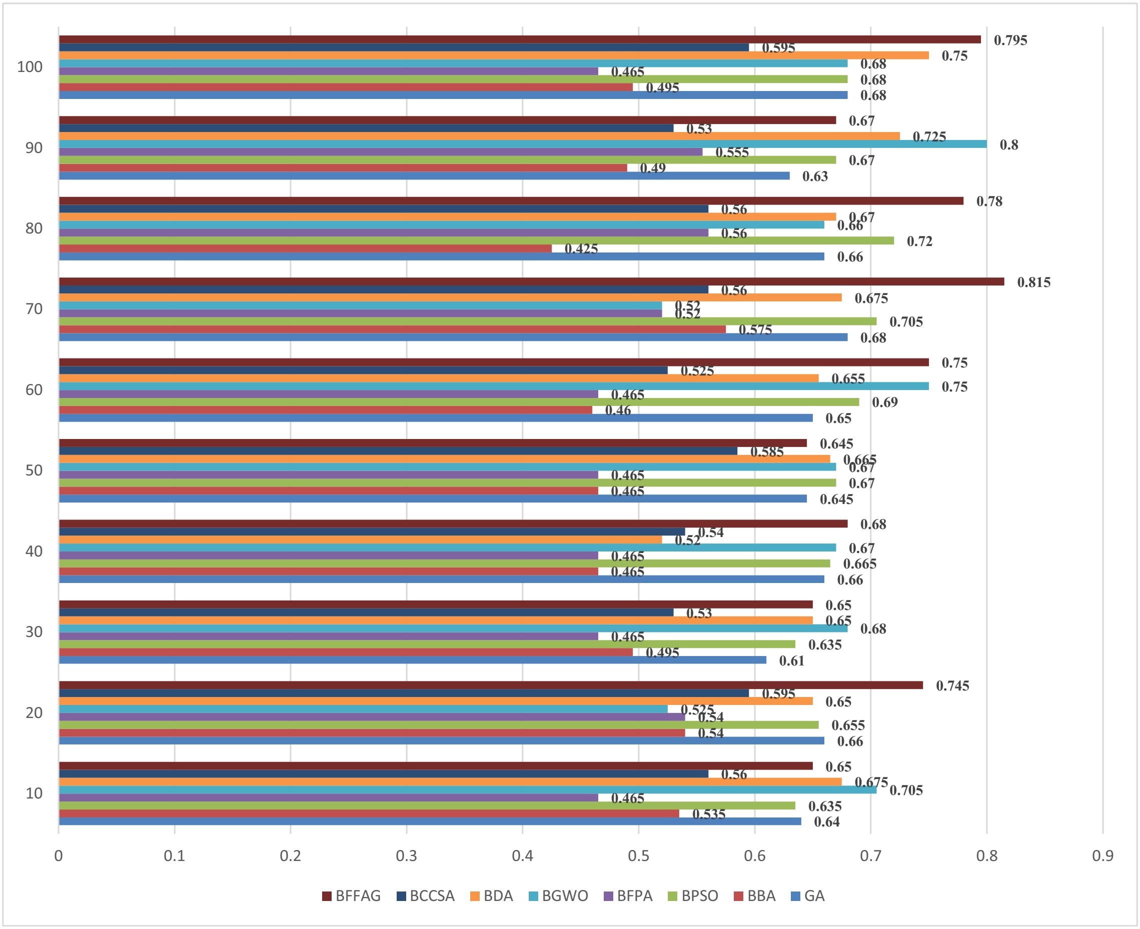


Fig. 28 classification accuracy of the two proposed approaches and other algorithms on the emotion analysis dataset with different iterations

experiment demonstrated the superiority of the BFFAG approach compared the other algorithms in terms of the convergence, average number of features and accuracy.

For future work, the proposed approaches can be applied to real world problems such as 0–1 knapsack problem, scheduling problem and etc. Also, the proposed approaches can be applied on datasets with dimensions larger than 500. In addition, other classifiers such as decision tree, support vector machine and naive Bayes can be used to evaluate the performance of the proposed approaches. Finally, investigating the impact of local search processes by proposed approaches on a large and small scales will be an interesting task for the future.

Acknowledgments The authors would like to thank the editor in chief, the associate editor, and reviewers for their constructive feedback during the review process. They genuinely appreciate the reviewers' comments for making this paper more complete.

References

1. Mafarja M, Mirjalili S (2018) Whale optimization approaches for wrapper feature selection. Appl Soft Comput 62:441–453
2. Sayed SA-F, Nabil E, Badr A (2016) A binary clonal flower pollination algorithm for feature selection. Pattern Recogn Lett 77:21–27
3. Zorarpacı E, Özel SA (2016) A hybrid approach of differential evolution and artificial bee colony for feature selection. Expert Syst Appl 62:91–103
4. Dong H et al (2018) A novel hybrid genetic algorithm with granular information for feature selection and optimization. Appl Soft Comput 65:33–46
5. Emary E, Zawbaa HM, Hassanien AE (2016) Binary ant lion approaches for feature selection. Neurocomputing 213:54–65
6. Mafarja M, Aljarah I, Faris H, Hammouri AI, al-Zoubi A'M, Mirjalili S (2019) Binary grasshopper optimization algorithm approaches for feature selection problems. Expert Syst Appl 117:267–286
7. Abualigah LMQ (2019) Feature selection, and enhanced krill herd algorithm for text document clustering: Springer
8. Rajamohana S, Umamaheswari K (2018) Hybrid approach of improved binary particle swarm optimization and shuffled frog leaping for feature selection. Comput Electr Eng 67:497–508
9. Faris H, Mafarja MM, Heidari AA, Aljarah I, al-Zoubi A'M, Mirjalili S, Fujita H (2018) An efficient binary salp swarm algorithm with crossover scheme for feature selection problems. Knowl-Based Syst 154:43–67
10. Rodrigues D, et al (2015) Binary flower pollination algorithm and its application to feature selection, In Recent advances in swarm intelligence and evolutionary computation, Springer. p. 85–100
11. Sayed GI, Hassanien AE, Azar AT (2017) Feature selection via a novel chaotic crow search algorithm. Neural Comput and Applic 31:171–188

12. Gharehchopogh FS, Gholizadeh H (2019) A comprehensive survey: whale optimization algorithm and its applications. Swarm Evol Comput 48:1–24
13. Gharehchopogh FS, Shayanfar H, Gholizadeh H (2019) A comprehensive survey on symbiotic organisms search algorithms. Artif Intell Rev: p. 1–48
14. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
15. Shayanfar H, Gharehchopogh FS (2018) Farmland fertility: a new metaheuristic algorithm for solving continuous optimization problems. Appl Soft Comput 71:728–746
16. Mafarja MM, Mirjalili S (2017) Hybrid whale optimization algorithm with simulated annealing for feature selection. Neurocomputing 260:302–312
17. Shehu HA, Tokat S (2020) A Hybrid Approach for the Sentiment Analysis of Turkish Twitter Data. In the international conference on artificial intelligence and applied mathematics in engineering. Part of the lecture notes on data engineering and communications technologies book series (LNDECT, volume 43) pp 182–190
18. Zhang Y, Wang S, Phillips P, Ji G (2014) Binary PSO with mutation operator for feature selection using decision tree applied to spam detection. Knowl-Based Syst 64:22–31
19. Xiang J, Han XH, Duan F, Qiang Y, Xiong XY, Lan Y, Chai H (2015) A novel hybrid system for feature selection based on an improved gravitational search algorithm and k-NN method. Appl Soft Comput 31:293–307
20. Hancer E, Xue B, Karaboga D, Zhang M (2015) A binary ABC algorithm based on advanced similarity scheme for feature selection. Appl Soft Comput 36:334–348
21. Emary E, Zawbaa HM, Hassanien AE (2016) Binary grey wolf optimization approaches for feature selection. Neurocomputing 172:371–381
22. Wan Y, Wang M, Ye Z, Lai X (2016) A feature selection method based on a modified binary-coded ant colony optimization algorithm. Appl Soft Comput 49:248–258
23. Abualigah LM, Khader AT, al-Betar MA, Alomari OA (2017) Text feature selection with a robust weight scheme and dynamic dimension reduction to text document clustering. Expert Syst Appl 84:24–36
24. Pashaei E, Aydin N (2017) Binary black hole algorithm for feature selection and classification on biological data. Appl Soft Comput 56:94–106
25. Mafarja MM, Eleyan D, Jaber I, Hammouri A, Mirjalili S (2017) Binary dragonfly algorithm for feature selection. In 2017 International Conference on New Trends in Computing Sciences (ICTCS) pp 12–17
26. Chen Y-P, Li Y, Wang G, Zheng YF, Xu Q, Fan JH, Cui XT (2017) A novel bacterial foraging optimization algorithm for feature selection. Expert Syst Appl 83:1–17
27. Abualigah LM, Khader AT, Hanandeh ES (2018) Hybrid clustering analysis using improved krill herd algorithm. Appl Intell 48(11): 4047–4071
28. Abualigah LM, Khader AT, Hanandeh ES (2018) A combination of objective functions and hybrid krill herd algorithm for text document clustering analysis. Eng Appl Artif Intell 73:111–125
29. Allam M, Nandhini M (2018) Optimal feature selection using binary teaching learning based optimization algorithm. J King Saud Univ Comput Inf Sci 1–13
30. Mafarja M, Aljarah I, Heidari AA, Faris H, Fournier-Viger P, Li X, Mirjalili S (2018) Binary dragonfly optimization for feature selection using time-varying transfer functions. Knowl-Based Syst 161:185–204
31. Papa JP, Rosa GH, de Souza AN, Afonso LCS (2018) Feature selection through binary brain storm optimization. Comput Electr Eng 72:468–481
32. Mafarja MM, Mirjalili S (2018) Hybrid binary ant lion optimizer with rough set and approximate entropy reducts for feature selection. Soft Comput:1–17
33. De Souza, RCT, Dos Santos Coelho L, De Macedo CA, Perezan, J (2018) A V-Shaped Binary Crow Search Algorithm for Feature Selection. In 2018 IEEE Congress on Evolutionary Computation (CEC) pp 1–8
34. Arora S, Anand P (2019) Binary butterfly optimization approaches for feature selection. Expert Syst Appl 116:147–160
35. Hussien AG, Hassanien AE, Houssein EH, Bhattacharyya S, Amin M (2019) S-shaped binary whale optimization algorithm for feature selection, In Recent trends in signal and image processing. Part of the Advances in Intelligent Systems and Computing book series (AISC, volume 727) pp 79–87
36. Santana CJ Jr, Macedo M, Siqueira H, Gokhale A, Bastos-Filho CJA (2019) A novel binary artificial bee colony algorithm. Futur Gener Comput Syst 98:180–196
37. Yan C, Ma J, Luo H, Patel A (2019) Hybrid binary coral reefs optimization algorithm with simulated annealing for feature selection in high-dimensional biomedical datasets. Chemom Intell Lab Syst 184:102–111
38. Zhang Y, Gong DW, Gao XZ, Tian T, Sun XY (2020) Binary differential evolution with self-learning for multi-objective feature selection. Inf Sci 507:67–85
39. Abdel-Basset M, el-Shahat D, el-henawy I, de Albuquerque VHC, Mirjalili S (2020) A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection. Expert Syst Appl 139:112824
40. Unler A, Murat A (2010) A discrete particle swarm optimization method for feature selection in binary classification problems. Eur J Oper Res 206(3):528–539
41. Kabir MM, Shahjahan M, Murase K (2012) A new hybrid ant colony optimization algorithm for feature selection. Expert Syst Appl 39(3):3747–3763
42. Barani F, Mirhosseini M, Nezamabadi-pour H (2017) Application of binary quantum-inspired gravitational search algorithm in feature subset selection. Appl Intell 49(180):304–318
43. Al-Tashi Q, Rais HM, Abdulkadir SJ, Mirjalili S, Alhussian H (2020) A Review of grey wolf optimizer-based feature selection methods for Classification. Evolutionary machine learning techniques. Part of the Algorithms for Intelligent Systems book series (AIS) pp 273–286
44. Liao TW, Kuo RJ (2017) Five discrete symbiotic organisms search algorithms for simultaneous optimization of feature subset and neighborhood size of KNN classification models. Appl Soft Comput 64:581–595
45. Mafarja M, Aljarah I, Heidari AA, Hammouri AI, Faris H, Ala'M AZ, Mirjalili S (2017) Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems. Knowl-Based Syst 145:25–45
46. Asuncion A, Newman DJ (2007) UCI machine learning repository. Irvine, CA: University of california, School of Information and Computer Science. http://www.ics.uci.edu/~mlearn/MLRepository.html. Accessed 2019.8.18
47. Sivanandam S, Deepa S (2008) Genetic algorithm optimization problems. In Introduction to genetic algorithms
48. Mirjalili S, Mirjalili SM, Yang X-S (2014) Binary bat algorithm. Neural Comput Applic 25(3–4):663–681
49. Mirjalili S, Lewis A (2013) S-shaped versus V-shaped transfer functions for binary particle swarm optimization. Swarm Evol Comput 9:1–14
50. Sayed GI, Hassanien AE, Azar AT (2019) Feature selection via a novel chaotic crow search algorithm. Neural Comput Applic 31(1): 171–188