Mathematical Biosciences and Engineering

MBE, 18(4): 3813–3854.
DOI: 10.3934/mbe.2021192
Received: 09 February 2021
Accepted: 21 April 2021
Published: 30 April 2021

http://www.aimspress.com/journal/MBE

Research article

An efficient binary Gradient-based optimizer for feature selection

Yugui Jiang[1,3], Qifang Luo[1,3,*], Yuanfei Wei[2], Laith Abualigah[4] and Yongquan Zhou[1,3]

[1] College of Artificial Intelligence, Guangxi University for Nationalities, Nanning 530006, China
[2] Xiangsihu College of Gunagxi University for Nationalities, Nanning, Guangxi 532100, China
[3] Guangxi Key Laboratories of Hybrid Computation and IC Design Analysis, Nanning 530006, China
[4] Faculty of Computer Sciences and Informatics, Amman Arab University, Amman 11953, Jordan

* **Correspondence:** Email: l.qf@163.com; Tel: +8618978886258; Fax: +867713265523.

Abstract: Feature selection (FS) is a classic and challenging optimization task in the field of machine learning and data mining. Gradient-based optimizer (GBO) is a recently developed metaheuristic with population-based characteristics inspired by gradient-based Newton's method that uses two main operators: the gradient search rule (GSR), the local escape operator (LEO) and a set of vectors to explore the search space for solving continuous problems. This article presents a binary GBO (BGBO) algorithm and for feature selecting problems. The eight independent GBO variants are proposed, and eight transfer functions divided into two families of S-shaped and V-shaped are evaluated to map the search space to a discrete space of research. To verify the performance of the proposed binary GBO algorithm, 18 well-known UCI datasets and 10 high-dimensional datasets are tested and compared with other advanced FS methods. The experimental results show that among the proposed binary GBO algorithms has the best comprehensive performance and has better performance than other well known metaheuristic algorithms in terms of the performance measures.

Keywords: Gradient-based optimizer (GBO); transfer function; binary gradient-based optimizer; feature selection (FS)

1. Introduction

With the rapid development of information technology, big data has been a persistently hot topic and the hotly anticipated artificial intelligence relies on the development of big data [1]. One of the reasons why big data has become a buzzing topic is that due to the increasing computing power of computers, the datasets to be processed have become larger and larger, where the datasets contain more and more attributes, making the task of machine learning in data mining complicated. If a dataset contains n features, then 2^n solutions need to be generated and evaluated [2,3]. If n is small, then the total number of feature subsets is small and the optimal feature subset can usually be obtained by exhaustive search. However, when n becomes large enough to reach a defined value, it is no longer possible to enumerate all the feature subsets, which is computationally expensive. How to handle these large datasets becomes paramount. Because these datasets often contain unimportant, redundant, and noisy features that reduce the efficiency of the classifier, choosing the right features is the key to solving this problem [4,5]. Pre-processing functions in the data mining process include Feature selection (FS), which aims to reduce the dimensionality of the data by eliminating irrelevant, redundant, or noisy features, thereby improving the efficiency of machine learning algorithms, such as classification accuracy [6]. FS approach, an important technique in machine learning and data mining, has been extensively researched over the past 20 years. Feature selection has been widely applied in areas including e.g., text classification [7,8], face recognition [9,10], cancer classification [11], genetic classification [12,13], financial [14], recommendation systems [15], customer relationship management [16], cancer diagnosis [17], image classification [18], medical technology [19], etc.

As usually, FS methods can be classified as Filter [20,21] or Wrapper [22,23], depending on whether they are independent of the subsequent learning algorithm. Filter is independent of the subsequent learning algorithm and generally uses the statistical performance of all training data to evaluate features directly, which has the advantage of being fast, but the evaluation deviates significantly from the performance of the subsequent learning algorithm. Wrapper uses the training accuracy of subsequent learning algorithms to evaluate a subset of features, which has the advantage of being less biased but is computationally intensive and relatively difficult for large data sets. It is based on the fact that the selected subset is ultimately used to construct the classification model so that if the features that achieve high classification performance are used directly in the construction of the classification model, a classification model with high classification performance is obtained. This method is slower than the Filter method, but the size of the optimized feature subset is much smaller, which is good for identifying key features; it is also more accurate, but less generalized and has higher time complexity.

There is another FS method which is the embedded FS method [24,25]. In the filtered and wrapped FS methods, the FS process is clearly separated from the learner training process. In contrast, embedded FS automatically performs FS during the learner training process, which is an integration of the FS process and the learner training process, and both are done in the same optimization process, i.e., FS is performed automatically during the learner training process. Embedded FS is most commonly used for L1 regularization and L2 regularization. Generally, the larger the regularization term, the simpler the model and the smaller the coefficients, when the regularization term increases to a certain level, all the feature coefficients will tend to 0. In this process, some of the feature coefficients will become 0 first, and the feature selection process is realized. Logistic regression, linear regression, and decision tree can be used as base learners for regularized feature selection, and only algorithms that can get feature coefficients or can get feature importance can be used as base learners for embedded selection.

The FS process can be divided into supervised feature selection [26] and unsupervised feature selection [27,28], depending on whether the original data sample contains information about the pattern category or not. Supervised feature selection is the process of selecting a feature set using the relationships between features and between features and categories, given a pattern category.

Unsupervised feature selection refers to the selection of features in the original dataset by the relationship between the features themselves in the dataset.

How to adopt the right method to solve the FS problem is crucial. Through the study of many researchers, the FS problems can be solved by using a variety of search methods, such as exhaustive search, greedy algorithms, and random search. However, most of the existing FS methods tend to fall into local optima or are computationally expensive. FS problem is an NP-hard problem where the optimal solution can only be guaranteed by exhaustive search. The use of metaheuristics makes it possible to obtain suboptimal solutions without examining the entire space of solutions. The superiority of metaheuristics stems from the ability to find an acceptable solution in a reasonable amount of time. [29] Recently, metaheuristic algorithms have shown superior performance in FS problems to find the best features. [30,31] Genetic Algorithms (GA) [32], Particle Swarm Optimization (PSO) [33], Ant Colony Optimization (ACO) [34], Whale Optimization Algorithm (WOA) [35], Grey Wolf Optimization (GWO) [36], Grasshopper Optimization Algorithm (GOA) [37], Gravitational Search Algorithms (GSA) [38], Slime Mould Algorithm (SMA) [39], Hunger Games Search (HGS) [40], Gradient-based optimizer (GBO) [41], and Dragonfly Algorithms (DA) [42] are some of the well-known metaheuristic algorithms. Compared to traditional algorithms, metaheuristics algorithms have better solutions.

The Gradient-based optimizer (GBO) is a novel gradient-based Newton's method [43] optimization algorithm proposed by Iman Ahmadianfar in 2020. The GBO is a new metaheuristic approach with population-based characteristics inspired by the gradient-based Newton's method, uses two main operators: gradient search rule (GSR) and local escaping operator (LEO), and a set of vectors to explore the search space. In general, the GBO has better optimization outcomes compared to other established metaheuristics. The GBO demonstrates its best performance among competitor algorithms in terms of exploration and exploitation. The GSR adopts the gradient-based approach, which enhances the exploration tendency and speeds up the convergence, hence obtaining a better position in the search space. The LEO enables the proposed GBO to get rid of local optima. The original version of GBO was designed to solve continuous optimization problems. However, many optimization problems (e.g., FS) have discrete decision variables and search spaces, and for discrete problems with discrete search spaces, the GBO does not provide the best solution. We are inspired to improve the binary version of the GBO by the superior performance of the GBO in continuous search space. In this paper, a binary version of GBO is proposed and designed to solve the FS problem. In this approach, two transfer functions [44] belonging to two families (i.e., S-shaped and V-shaped) are used to convert the continuous solution into the binary solution. This method is called BGBO_S and BGBO_V. The proposed method provides a new idea for solving the FS problem, which should not be limited to the improvement of the inherent algorithm, but can try to get good results as well when the newly proposed algorithm is applied to the FS problem. Meanwhile, many real-world problems are discrete, and GBO itself is a metaheuristic algorithm with a good mechanism. Since it has just been proposed, there is no binary version yet. The proposed method also provides a wider application of the GBO algorithm

The remaining paper is organized as follows. The related works on feature selection are described in Section 2. The original GBO is introduced in Section 3. The proposed binary GBO (BGBO) is proposed in Section 4. In Section 5, the BGBO are tested on 18 standard UCI datasets and 10 high-dimensional datasets, and the results are compared with the well known binary metaheuristic algorithms. We conclude in Section 6.

2. Related works

There has been quite a bit of work in the area of metaheuristics for the FS problem, and variables and FS problem have been the focus of research in many application areas [45], and FS plays an important role in classification [46]. A large number of metaheuristic algorithms for FS problem have been proposed in the literature. The search capability of binary metaheuristic algorithms is dominated by wrapper-based feature selection algorithms. Traditional optimization methods cannot solve complex optimization problems better and it is difficult to obtain satisfactory solutions. Thus, a more effective method, the metaheuristic algorithm, has been proposed and applied by an increasing number of scholars. Metaheuristic algorithms are the product of combining stochastic algorithms and local search algorithms. Metaheuristic algorithms can be divided into four categories: evolutionary algorithms, swarm intelligence, physics-based methods, and human-based methods.

The first category of metaheuristic algorithms is inspired by the Darwinian Theory of evolution, and evolutionary algorithms simulate the rules of evolution in nature. Genetic algorithms (GA) [32] are one of the representatives of evolutionary algorithms. The three steps of selection, crossover, and mutation are the main steps of genetic algorithms to update the population for optimization purposes. There are various improved versions of genetic algorithms as well as applications to feature selection problems. Siedlecki et al. [47] used genetic algorithms (GA) to select features and find near-optimal feature subsets from large feature sets. Leardi et al. [48] demonstrated that the subsets of variables selected by genetic algorithms are generally more efficient than those obtained by classical feature selection methods. Oh et al. [49] proposed a novel hybrid genetic algorithm for feature selection. The method embeds the local search operation into the hybrid GA to tune the search, which improves the algorithm performance and effectively controls the subset size. The convergence of the algorithm is enhanced. In addition, evolutionary algorithms include Differential Evolution (DE) algorithms [50], and Xue et al. [51] studied Differential Evolution (DE) for multi-objective feature selection in classification.

The second class of metaheuristic algorithms: swarm intelligence, inspired by the social behavior of animals in a herd, shares information about all individuals in the optimization process. Particle Swarm Optimization (PSO) [33], Gray Wolf Optimizer (GWO) [36], Ant Colony Optimization (ACO) [34], Artificial Bee Colony (ABC) [52], Whale Optimization Algorithm (WOA) [35], Harris hawks optimization (HHO) [53], and Marine Predators Algorithm (MPA) [54] are among the representative algorithms of this class of metaheuristics. In particular, QO Saber et al. [55] proposed a particle swarm optimization algorithm with a logistic regression model and proved that the proposed method has a competitive performance. K Chen et al. [56] proposed hybrid particle swarm optimization (HPSO-SSM) with a spiral mechanism, and the results showed that the HPSO-SSM method has high performance in solving classification problems. B Xue et al. [57] proposed a multi-objective particle swarm optimization (PSO) study for feature selection. Emery, E et al. [58] proposed a novel binary version of gray wolf optimization (GWO) and used it to select the best subset of features for classification purposes. P Hu et al. [59] analyzed the range of values of an under-binary condition and proposed a new parameter update equation to balance the ability of global search and local search. The proposed method introduces new transfer functions and proposes new parameter update equations. These methods show remarkable performance and have fast convergence, but they tend to fall into local optimal states. Q. Tu et al. [60] proposed a multi-strategy ensemble GWO (MEGWO). The proposed MEGWO contains three different search strategies to update the solution. Mafarja et al. [61]

proposed two binary variants of the WOA algorithm to search for the optimal feature subset for classification purposes. MM Mafarja et al. [62] used two hybridization models to design different feature selection techniques based on the whale optimization algorithm (WOA). R. K Agrawal et al. [63] proposed the quantum whale optimization algorithm (QWOA) for feature selection, which is a combination of quantum concept and whale optimization algorithm (WOA). The approach enhances the versatility and convergence of the classical WOA for feature selection and extends the prospects of nature-inspired feature selection methods based on high-performance but low-complexity wrappers.

The third category of metaheuristic algorithms: physics-based methods, inspired by the physical laws of nature, simulate the physical laws in the optimization process to find the best. Common algorithms include Simulated Annealing (SA) [64], Gravitational Search Algorithm (GSA) [38], Lightning Search Algorithm (LSA) [65], Multi-verse Optimizer (MVO) [66], Electromagnetic Field Optimization (EFO) [67], Chemical Reaction Optimization (CRO) [68], and Henry Gas Solubility Optimization (HGSO) [69]. Meiri et al. [70] used simulated annealing (SA) method for specifying large-scale linear regression models. Lin et al. [71] proposed a simulated annealing (SA) method for parameter determination and feature selection of SVM, called SA-SVM. Rashedi et al. [72] introduced a binary version of the algorithm (GSA). Sushama et al. [73] used a wrapper-based approach for disease prediction analysis of medical data using GSA and k-NN. Rao et al. [74] proposed a feature selection technique based on a hybrid of binary chemical reaction optimization (BCRO) and binary chemical reaction optimization-binary particle swarm optimization (HBCRO-BPSO) in this paper to optimize the number of selected features and improve the classification accuracy. This method optimizes the number of features and improves the classification accuracy and computational efficiency of the ML algorithm. However, it still does not attempt to handle ultra-high-dimensional datasets with a large number of samples in FS. Neggaz et al. [75] proposed a novel dimensionality reduction method by using Henry Gas Solubility Optimization (HGSO) algorithm to select important features to improve classification accuracy. The method generates 100% accuracy for classification problems with more than 11,000 features and is valid on both low and high-dimensional datasets. However, HGSO also maintains certain limitations. Since HGSO maintains multiple control parameters, this may compromise its applicability compared to other well-known methods such as GWO [36] and HHO [53].

The last type of metaheuristic algorithms: human-based approaches, inspired by human interactions or human behavior in society. For examples: Teaching–learning-based optimization (TLBO) [76], Imperialist Competitive Algorithm (ICA) [77], Volleyball Premier League Algorithm (VPL) [78] and Cultural Evolution Algorithm (CEA) [79]. In which, Mohan Allam et al. [80] proposed a new wrapper-based feature selection method called the binary teaching learning based optimization (FS-BTLBO) algorithm. Mousavirad et al. [81] proposed an improved imperialist competition algorithm to apply this proposed algorithm to the feature selection process. Keramati et al. [82] proposed a new FS method based on cultural evolution. The proposed methods provide new solutions to the feature selection problem, but there are still problems of insufficient accuracy and excessive number of features.

From the above work related to metaheuristics, we can know that a great number of metaheuristics are successful applied to the FS problem. The above work related to different types of metaheuristic algorithms on feature selection problem proves that metaheuristic algorithms have promising performance in solving FS problems, especially in terms of accuracy. In addition, they can produce better results by using a smaller number of features. However, there are still some problems, such as the number of tested datasets is small and not comprehensive enough to include low-dimensional, high-dimensional or different types of datasets, the problem of slow convergence that exists in most metaheuristics, and the lack of significant effect on the number of features selected for high-dimensional datasets. Based on the No Free Lunch (NFL) rule [83], no single metaheuristic algorithm can solve all problems, which indicates that a particular algorithm may provide very promising results for a set of problems, but the same method may be inefficient for a different set of problems. Gradient-based optimizer is a novel gradient-based metaheuristic algorithm proposed by Iman Ahmadianfar [41] in 2020. The GBO algorithm is a new metaheuristic algorithm that has been recently proposed and has not been systematically applied to feature selection problems. The gradient-based optimization-seeking mechanism (GSR and LEO) of the GBO algorithm makes it feasible to make an appropriate trade-off between exploration and exploitation. Therefore, in this paper, we propose binary GBO for solving FS problems, mapping continuous GBO into discrete forms using S-shaped and V-shaped transfer functions, and try to apply it in solving high-dimensional dataset problems. The following is a brief description of the GBO algorithm.

3. Gradient-based optimizer (GBO)

The metaheuristic algorithm was first proposed by Iman Ahmadianfar et al. in 2020 to solve optimization problems related to engineering applications. Exploration and exploitation are the two main phases in metaheuristic algorithms, which aim to improve the speed of convergence and/or local optimum avoidance of the algorithm when searching for a target/position. The GBO is managed to create a proper trade-off between exploration and exploitation to uses two main operators: gradient search rule (GSR) and local escaping operator (LEO). A simple introduction of this algorithm is described below:

3.1. Gradient search rule (GSR)

First, GBO proposes the first operator GSR, which helps the GBO to consider stochastic behavior in the optimization process to facilitate the exploration and avoidance of local optima. And the direction movement (DM) is added to GSR, which is used to perform a suitable local search trend to facilitate the convergence speed of the GBO algorithm. Based on the GSR and DM, the following equation is used to update the position of current vector (x_n^m).

$$X1_n^m = x_n^m - randn \times \rho_1 \times \frac{2\Delta x \times x_n^m}{x_{worst} - x_{best} + \varepsilon} + rand \times \rho_2 \times (x_{best} - x_n^m) \tag{1}$$

where

$$\rho_1 = x \times rand \times \alpha - \alpha \tag{2}$$

$$\alpha = \left| \beta \times \sin(\frac{3\pi}{2} + \sin(\beta \times \frac{3\pi}{2})) \right| \tag{3}$$

$$\beta = \beta_{\min} + (\beta_{\max} + \beta_{\min}) \times (1 - (\frac{m}{M})^3)^2 \tag{4}$$

where $\beta_{\min}$ and $\beta_{\max}$ are 0.2 and 1.2, respectively, m is the number of iterations, and M is the total number of iterations. $randn$ is a normally distributed random number, and ε is a small number within the range of [0, 0.1]. ρ_2 can be given by:

$$\rho_2 = 2 \times rand \times \alpha - \alpha \tag{5}$$

$$\Delta x = rand(1:N) \times |step| \tag{6}$$

$$step = \frac{(x_{best} - x_{r1}^m) + \delta}{2} \tag{7}$$

$$\delta = 2 \times rand \times (\left| \frac{x_{r1}^m + x_{r2}^m + x_{r3}^m + x_{r4}^m}{4} - x_n^m \right|) \tag{8}$$

where $rand(1:N)$ is a random number with N dimensions, $r1, r2, r3$ *and* $r4$ $r1 \neq r2 \neq r3 \neq r4 \neq n$ are different integers randomly chosen from [1, *N*], $step$ is a step size, which is determined by x_{best} and x_{r1}^m. By replacing the position of the best vector (x_{best}) with the current vector (x_n^m) in Eq (1), the new vector ($X2_n^m$) can be generated as follows:

$$X2_n^m = x_{best} - randn \times \rho_1 \times \frac{2\Delta x \times x_n^m}{yp_n^m - yq_n^m + \varepsilon} + rand + \rho_2 \times (x_{r1}^m - x_{r2}^m) \tag{9}$$

where

$$yp_n = rand \times (\frac{[z_{n+1} + x_n]}{2} + rand \times \Delta x) \tag{10}$$

$$yq_n = rand \times (\frac{[z_{n+1} + x_n]}{2} - rand \times \Delta x) \tag{11}$$

Based on the positions $X1_n^m$, $X2_n^m$, and the current position (X_n^m), the new solution at the next iteration (x_n^{m+1}) can be defined as:

$$x_n^{m+1} = r_a \times (r_b \times X1_n^m + (1 - r_a) \times X2_n^m) + (1 - r_a) \times X3_n^m \tag{12}$$

$$X3_n^m = X_n^m - \rho_1 \times (X2_n^m - X1_n^m) \tag{13}$$

3.2. Local escaping operator (LEO)

LEO is the second operator introduced by GBO. LEO is introduced to make GBO still effective in the face of complex high-dimensional problems. The LEO generates a solution with a superior performance (x_{LEO}^{m}) by using several solutions, which include the best position (x_{best}), the solutions $X1_{n}^{m}$ and $X2_{n}^{m}$, two random solutions x_{r1}^{m} and x_{r2}^{m}, and a new randomly generated solution (x_{k}^{m}). The solution X_{LEO}^{m} is generated by the following scheme:

$if \;\; rand < pr$

$if \;\; rand < 0.5$

$$X_{LEO}^{m} = X_{n}^{m+1} + f_1 \times (u_1 \times x_{best} - u_2 \times x_{k}^{m}) + f_2 \times \rho_1 \times (u_3 \times (X2_{n}^{m} - X1_{n}^{m}) + u_2 \times (x_{r1}^{m} - x_{r2}^{m}))/2$$

$$X_{n}^{m+1} = X_{LEO}^{m} \tag{14}$$

$else$

$$X_{LEO}^{m} = X_{best} + f_1 \times (u_1 \times x_{best} - u_2 \times x_{k}^{m}) + f_2 \times \rho_1 \times (u_3 \times (X2_{n}^{m} - X1_{n}^{m}) + u_2 \times (x_{r1}^{m} - x_{r2}^{m}))/2$$

$$X_{n}^{m+1} = X_{LEO}^{m} \tag{15}$$

End

End

where f_1 is a uniform random number in the range of [-1,1], f_2 is a random number from a normal distribution with mean of 0 and standard deviation of 1, pr is the probability, and $u_1, u_2, and\ u_3$ are three random numbers, which are defined as:

$$u_1 = \begin{cases} 2 \times rand & if\ \mu_1 < 0.5 \\ 1 & otherwise \end{cases} \tag{16}$$

$$u_2 = \begin{cases} rand & if\ \mu_1 < 0.5 \\ 1 & otherwise \end{cases} \tag{17}$$

$$u_3 = \begin{cases} rand & if\ \mu_1 < 0.5 \\ 1 & otherwise \end{cases} \tag{18}$$

where $rand$ is a random number in the range of [0, 1], and μ_1 is a number in the range of [0, 1]. The above equations can be simplified:

$$u_1 = L_1 \times 2 \times rand + (1 - L_1) \tag{19}$$

$$u_2 = L_1 \times rand + (1 - L_1) \tag{20}$$

$$u_3 = L_1 \times rand + (1 - L_1) \tag{21}$$

where L_1 is a binary parameter with a value of 0 or 1. If parameter μ_1 is less than 0.5, the value of L_1 is 1, otherwise, it is 0. To determine the solution x_{k}^{m} in Eq (12), the following scheme is suggested.

$$x_k^m = \begin{cases} x_{rand} & if\ \mu_2 < 0.5 \\ x_p^m & otherwise \end{cases} \tag{22}$$

$$x_{rand} = X_{\min} + rand(0,1) \times (X_{\max} - X_{\min}) \tag{23}$$

where x_{rand} is a new solution, x_p^m is a randomly selected solution of the population ($p \in [1, 2, \cdots, N]$), and μ_2 is a random number in the range of [0, 1]. Eq (22) can be simplified as:

$$x_k^m = L_2 \times x_p^m + (1 - L_2) \times x_{rand} \tag{24}$$

where L_2 is a binary parameter with a value of 0 or 1. If μ_2 is less than 0.5, the value of L_2 is 1, otherwise, it is 0. The pseudo code of the GBO algorithm is shown in Algorithm 1.

Algorithm 1. Pseudo code of the GBO algorithm.

Initialization
Assign values for parameters $pr, \varepsilon\ and\ M$
Generate an initial population $X_0 = [x_{0,1}, x_{0,2}, \cdots, x_{0,D}]$
Evaluate the objective function value $f(X_0), n = 1, 2, .., N$.
Specify the best and worst solutions X_{best}^m and X_{worst}^m
    While ($m < M$)
        for n = 1 : N
            for i = 1 : D
                Select randomly $r1 \neq r2 \neq r3 \neq r4 \neq n$ in the range of [1, N]
                Calculate the position $x_{n,i}^{m+1}$ using Eq (12)
            end for
            Local escaping operator
            if $rand < pr$
                Calculate the position X_{LEO}^m using Eq (14) or Eq (15)
                $X_n^{m+1} = x_{LEO}^m$
            end
            Update the positions X_{best}^m and X_{worst}^m
        end for
        $m = m + 1$
    end
Return X_{best}^m

4. Proposed binary gradient-based optimizer (BGBO)

4.1. Motivation

The GBO algorithm is a novel population-based metaheuristic search method to solving the continuous problem [84]. The GBO algorithm is derived from a gradient-based search method that uses Newton's method to explore the better regions in the search space. Two operators (i.e., gradient-based rule (GSR) and local escape operator (LEO)) were introduced in GBO and mathematically computed to facilitate the exploration and exploitation of the search. Newton's method is used as a search engine in GSR to enhance the exploration and exploitation process, while LEO is used to deal with complex problems in GBO. GBO shows superior performance in solving optimization problems compared to other optimization methods in the literature. Due to the above advantages of GBO and since it has not been used to solve FS problems, searching for the best subset of features in FS is a challenging problem, especially in wrapper-based methods. This is because the selected subset needs to be evaluated by a learning algorithm (e.g., classifier) in each optimization step. Therefore, a suitable optimization method is needed to reduce the number of evaluations in this paper; we propose a method for solving the FS problem. On this basis, we present the motivation for using this algorithm as a search method in a wrapper-based FS process. According to the nature of FS probability, the search space can be represented by binary values [0, 1], and binary arithmetic is much simpler than continuous arithmetic, so we propose a binary version of GBO to solve the FS problem.

4.2. Our proposed binary GBO (BGBO)

The proposed binary GBO has two key parts, the first one is how to map from continuous values to [0, 1] and the second one is how to update the position of the population. Transfer function [44] is the simplest method in binary GBO, which maps continuous values to [0, 1] and then decomposes them into 0 and 1 based on probability. It preserves the structure of GBO and other operations that move the position of the population in the binary space. The transfer functions are divided into two main categories according to their shapes: S-shaped and V-shaped. Figure 1 shows these two families of transfer functions.

In the S-shaped transfer function, the vectors values are converted into probability values within the [0, 1] range. As shown in Eq (25) [85].

$$T(X_i^d) = \frac{1}{1+e^{-X_i^d}} \tag{25}$$

where X_i^d represents the position in the *d*-th dimension of the *i*-th individual in the GBO algorithm. The location of each vector, based on the probability values obtained from $T(X_i^d)$, will be updated.

$$X_{t+1}^d = \begin{cases} 1 & if\ \ rand < T(X_t^d) \\ 0 & if\ \ rand \geq T(X_t^d) \end{cases} \tag{26}$$

where X_{t+1}^d represents the *t*-th vectors' position at next-iteration in the *d*-th dimension.

For next, the Hyperbolic tan (V-shaped) function [86] is another transfer function and mathematical formulation is given below:

$$T(X_i^d) = \left|\tanh(X_i^d)\right| \tag{27}$$

where X_i^d represents the position in the d-th dimension of the i-th individual in the GBO algorithm. Based on the probability values obtained from Eq (27), the current position of each vector is updated in the next iteration. Using Eq (28), the search space is transformed into a binary search space.

$$X_{t+1}^d = \begin{cases} \neg X_t^d & if \ rand < T(X_t^d) \\ X_t^d & if \ rand \geq T(X_t^d) \end{cases} \tag{28}$$

Table 1 shows the mathematical expressions of all transfer functions that S-shaped and V-shaped transfer functions are included. There are four S-shaped transfer functions: S1, S2, S3, and S4. There are also four V-shaped transfer functions, namely: V1, V2, V3, and V4.

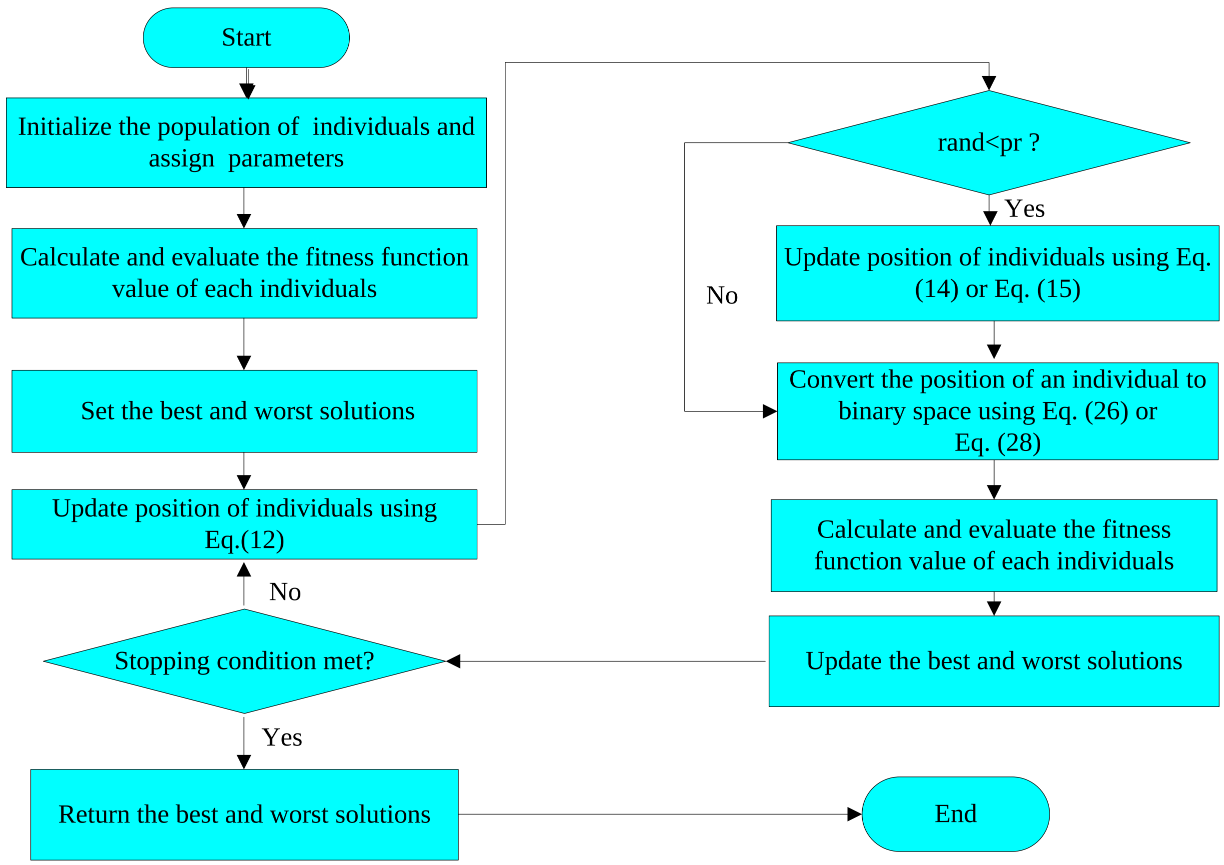

Figure 1. Flowchart of the BGBO.

S-shaped functions and V-shaped functions, with a suitable probability, the exploration and exploitation ability of individuals in the population is the best. And at the beginning of the iteration, the exploration ratio of these functions is high relative to the exploitation ratio. As the value is higher, the probability of individuals in the population changing their current position is higher. These transfer functions are designed to facilitate the exploration and utilization of the binary search space by the individuals of the population. Therefore, both families of transfer functions (S-shaped and V-shaped) are used to transfer continuous solutions of GBO into the binary search space “0” and “1”, and the methods are called BGBO_S and BGBO_V. Algorithm 2 is the pseudo-code for this method, as shown in the following. Meanwhile the flow chart of BGBO algorithm is shown in Figure 1.

Algorithm 2 Pseudo code of the BGBO algorithm

Initialization
Assign values for parameters pr, ε *and* M
Generate an initial population $X_0 = \left[x_{0,1}, x_{0,2}, \cdots, x_{0,D}\right]$
Evaluate the objective function value $f(X_0), n = 1,2,.., N$
Specify the best and worst solutions X_{best}^m and X_{worst}^m
    while ($m < M$)
        for $n = 1 : N$
            for $i = 1 : D$
            Select randomly $r1 \neq r2 \neq r3 \neq r4 \neq n$ in the range of [1, N]
              Calculate the position $x_{n,i}^{m+1}$ using Eq (12)
            end for
          Local escaping operator
          if *r and* < *pr*
            Calculate the position X_{LEO}^m using Eq (14) or Eq (15)
            $X_n^{m+1} = x_{LEO}^m$
          end
  The probability of converting a population to a binary space 0 or 1. Using Eq (26) or Eq (28)
          Compute the fitness of all populations
          Update the positions X_{best}^m and X_{worst}^m
        end for
      $m = m + 1$
    end
Return X_{best}^m

4.3. Binary GBO for FS problems

There are usually a large number of noisy and irrelevant features in data mining, and for these irrelevant features, partial processing is usually needed, otherwise, it will cause the waste of data processing resources and make the error probability of processing results more difficult, thus increasing the difficulty of the learning task. For example, in the classification task, if there are a large number of irrelevant features, the learning time of the classifier may be longer and the classification accuracy may be lower. As the dimensionality of the data increases and the dataset contains more and more messages, it is very difficult to handle large data, and the cost of computation time increases.

Therefore, it needs to reduce the features of the dataset effectively and keep the key features. Datasets are usually represented by a matrix whose rows represent instances (or samples) and columns represent attributes (or features). Feature selection is a common technique used in data mining and machine learning and is an effective and well-proven method. Most researchers focus on high precision and low feature methods, and feature selection is one such method. In this section, the wrapper method is used to implement feature selection. The binary version of the algorithm corresponding to the above eight transfer functions (BGBO_S1, BGBO_S2, BGBO_S3, BGBO_S4, BGBO_V1, BGBO_V2, BGBO_V3, and BGBO_V4) is applied to the feature selection problem.

Table 1. V-Shaped and S-Shaped Family.

S-shaped Transfer function		V-shaped Transfer function	
Name	Mathematical formulation	Name	Mathematical formulation
S1	$\mathrm{T}(x)=\frac{1}{1+e^{-2x}}$	V1	$\mathrm{T}(x)=\left\lvert \mathrm{erf}\,(\frac{\sqrt{\pi}}{2}x)\right\rvert$
S2	$\mathrm{T}(x)=\frac{1}{1+e^{-x}}$	V2	$\mathrm{T}(x)=\lvert\tanh(x)\rvert$
S3	$\mathrm{T}(x)=\frac{1}{1+e^{(-x/2)}}$	V3	$\mathrm{T}(x)=\left\lvert(x)/\sqrt{1+x^2}\right\rvert$
S4	$\mathrm{T}(x)=\frac{1}{1+e^{(-x/3)}}$	V4	$\mathrm{T}(x)=\left\lvert\frac{2}{\pi}\arctan(\frac{\pi}{2}x)\right\rvert$

4.3.1. Fitness function

As already known in the previous subsections, it is significant to choose an effective search strategy for FS methods. Since the proposed method is a wrapper-based approach, then a learning algorithm (e.g., a classifier) should be involved in the evaluation process. In this work, a well-known classifier, namely, the k-NN classifier [87] is used as an evaluator, and the k-NN classifier classifies unlabeled instances by measuring the distance between a given unlabeled instance and its k nearest instances [88]. It is based on the principle that if a majority of the k most similar samples of a sample in a given feature space belongs to a certain class, then that sample also belongs to that class. The classification accuracy of the selected features is incorporated into the proposed fitness function. The classification accuracy obtained is better when the features in the subset are correlated. The classification accuracy obtained is better when the number of features in the subset is smaller. Having a higher classification accuracy is one of the goals of the FS method, and another important goal is to recalculate the number of selected features; the fewer the number of features in the solution, the better the solution will be. The mathematical formulation of the fitness function is shown below:

$$fitness=\alpha\gamma_R(D)+\beta\frac{|M|}{|N|} \tag{29}$$

where $\gamma_R(D)$ represents the classification error rate corresponding to the currently selected feature subset of the classifier, $|M|$ represents the number of features currently selected, $|N|$ is the total number of features, α and β are two weight coefficients to reflect the classification rate and length of the subset, α is the random number between [0, 1], and $\beta = 1 - \alpha$.

4.3.2. Computational complexity

To have a better understanding of the implementation process of the BGBO feature selection algorithm proposed in this paper, the following subsections analyze the computational complexity of BGBO. The time complexity is first analyzed as follows.

1) Population initialization *O* (*n*d),* where n is the population size and d is the dimension size (i.e., the number of features size).

2) The k-NN classifier training takes *O* (*n*m*), where n is still the population size and m is the instance size.

3) The time required to evaluate the fitness value of each population individual is *O* (*n*).

4) The time required for BGBO execution (i.e., the process of updating the position of population individuals) is *O* (*n*d*).

5) The time required to map the population of individuals to the binary space *O* (*n*d*).

6) Repeat steps 2-5 above until the maximum number of iterations T is satisfied.

The time complexity required to perform steps 2-6 above is *O* (*n*d*m*T*), and the final time complexity is *O* (*n*d*m*T*K*) when run independently *K* times.

The space complexity is then analyzed as follows.

1) The space required for population initialization is *O* (*n*d*).

2) The space required for the k-NN classifier is *O* (*m*s*), where s is the number of features that have been selected.

From the above analysis, n and s can be ignored, so the space complexity is *O* (*m*d*).

The above is the analysis of the time complexity and space complexity of the BGBO algorithm proposed in this paper when applied to feature selection, for which the time complexity and space complexity are mutually affected. When pursuing a better time complexity, the performance of the space complexity may be worse, i.e., it may lead to occupying more storage space; on the contrary, when pursuing a better space complexity, the performance of the time complexity may be worse, i.e., it may lead to occupying a longer running time. Time complexity and space complexity are incompatible, and we need to strike a balance between them.

5. Experiment results and discuss

The detailed descriptions of the 18 benchmark datasets are shown in Table 2. These datasets have distinct characteristics in nature. These datasets were extracted from the UCI repository [89]. These datasets were used to test the proposed method. All experiments were implemented in MatlabR2017a and executed on an Intel Core i3 machine with a CPU frequency of 3.70 GHz and 8 GB of RAM. The maximum number of iterations was 100.

As a preliminary study, we conducted experiments on the effect of different population sizes on the basic method (i.e., on the classification accuracy of BGBO), and thus evaluated BGBO at different population sizes (i.e., 10, 20, 30, and 50). The PenglungEW dataset was used for the experiments, and the average accuracy and time spent by the classifier were used as evaluation criteria. Because of the large dimensionality of this dataset, the sensitivity is high. The higher sensitivity allows the algorithm to respond significantly too small changes in parameters. Table 3 shows the experimental results for the PenglungEW dataset with different population sizes and the maximum number of iterations.

Table 2. List of used datasets.

No.	Datasets	Instances	Number of features (d)	Number of classes (k)
1	Breast_cancer_wisconsin	569	32	2
2	IonosphereEW	351	34	2
3	SonarEW	208	60	2
4	zoo	101	17	6
5	Parliment1984	435	17	2
6	Iris	150	4	3
7	Wdbc	569	30	2
8	PenglungEW	73	325	2
9	Lymphography	148	18	4
10	KrvskpEW	3196	36	2
11	SPECT	267	22	2
12	Clean1	476	166	2
13	Semeion	1593	256	10
14	Glass	214	9	6
15	Coil	1440	1024	20
16	Wine	178	13	3
17	Segmentation	178	13	3
18	Vote	300	16	2

Table 3. Average accuracy results and Time according to different combinations of population size (n) and the number of max iteration on the PenglungEW dataset.

Population Size(n)	Max Iterations	Accuracy	Time
10	100	1.0000	202.23 s
20	100	0.9978	401.84 s
30	100	1.0000	590.49 s
50	100	0.9956	985.71 s
10	150	1.0000	298.56 s
20	150	1.0000	605.30 s
30	150	1.0000	886.67 s
50	150	1.0000	1481.35 s

The effect of varying the population size (10, 20, 30, and 50) and varying the number of iterations (100 and 150) on the classification accuracy of the dataset PenglungEW can be seen in Table 3, where it can be seen that the range of variation in classification accuracy is small and that increasing the population size does not always improve the results, and similarly, increasing the number of iterations has little effect on the results. Therefore, we set the population size to 10 and set the number of iterations to 100 in all the next experiments as a trade-off between classification accuracy and the overhead of the algorithm running time.

As mentioned in the previous subsection, there are two weighting coefficients α and β in the fitness function. They reflect the importance of the classification error rate and the number of selected features on the performance of the algorithm, respectively. To further investigate the effect of different weight coefficients α and β on the experimental results, different combinations are set up for experiments, and Table 4 shows the experimental results for different combinations.

Table 4. Average accuracy results, Time, Average fitness and Average number of feature according to different combinations of α and β on the PenglungEW dataset.

α	β	Accuracy	Fitness	Number of feature	Time
0.5	0.5	0.9807	0.0284	12.2000	199.64
0.6	0.4	0.9863	0.0218	11.1000	199.33
0.7	0.3	1.0000	0.0132	14.4000	201.50
0.8	0.2	1.0000	0.0070	11.5000	201.70
0.9	0.1	1.0000	0.0051	16.8000	202.94
0.99	0.01	1.0000	0.0002	6.9667	209.04

The results are shown in Table 4. In general, decreasing the value of β and increasing the value of α, the accuracy has improved, but when α is 0.7 and β is 0.3, the accuracy has reached 1. Therefore, increasing the reference standard, the fitness value, and the number of selected features, according to the results, when the value of α is determined to be 0.99 and the value of β is 0.01, the fitness value and the number of selected features are the best, so to be able to make a reasonable comparison with other methods. In this paper, α and β are set to 0.99 and 0.01 in the subsequent experiments, respectively. Additionally, k in the Euclidean distance formula used to evaluate the feature subset of the k-NN classifier was set to 5 [90].

Table 5. Parameters setup for the variations of BGBO.

Parameter	Values
Population size	10
Number of iterations	100
Dimension	Number of features
Number of runs for each method	30
α	0.99
β	0.01
k	5
Other parameters	Pr = 0.5

The parameters used in BGBO are shown in Table 5. Each variation of BGBO has tested 30 independent runs. The measurement criteria used in the comparison of BGBO are fitness values, classification accuracy and selection feature size.

1) Fitness values are obtained from fitness function by using the selected features on the benchmark datasets. (The mean, standard deviation of the fitness function is calculated).
2) Classification accuracy is obtained from the classifier by using the selected features on the benchmark datasets.
3) Selection feature size is the mean number of the selected features.

Table 6. Comparison between different variations of BGBO on the mean of fitness values (as Mean) and the standard deviation of fitness values (as S.D).

Dataset	BGBO_S1		BGBO_S2		BGBO_S3		BGBO_S4		BGBO_V1		BGBO_V2		BGBO_V3		BGBO_V4	
	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D
Breast_cancer _wisconsin	0.0054	0.0006	0.0044	0.0006	0.0040	0.0007	0.0038	0.0005	0.0020	0.0012	0.0018	0.0010	**0.0017**	0.0008	0.0020	0.0013
IonosphereEW	0.1635	0.0064	0.1572	0.0074	0.1516	0.0082	0.1526	0.0079	0.0679	0.0180	0.0676	0.0188	**0.0326**	0.0114	0.0697	0.0205
SonarEW	0.0661	0.0114	0.0583	0.0094	0.0654	0.0123	0.0556	0.0096	0.0763	0.0135	**0.0390**	0.0218	0.0543	0.0194	0.0704	0.0217
zoo	0.0038	0.0000	0.0031	0.0001	0.0029	0.0006	0.0021	0.0004	0.0019	0.0001	0.0028	0.0004	**0.0012**	0.0002	0.0020	0.0003
Parliment1984	0.0099	0.0045	0.0277	0.0070	0.0162	0.0023	0.0099	0.0049	0.0120	0.0000	0.0088	0.0058	**0.0020**	0.0003	0.0229	0.0020
Iris	0.0710	0.0000	0.0355	0.0000	0.0355	0.0000	0.0355	0.0000	**0.0025**	0.0000	0.0075	0.0000	**0.0025**	0.0000	0.0050	0.0000
Wdbc	0.0046	0.0007	0.0312	0.0006	0.0076	0.0035	0.0281	0.0035	0.0018	0.0004	0.0052	0.0040	**0.0013**	0.0002	0.0086	0.0036
PenglungEW	0.1418	0.0121	0.0750	0.0010	0.0052	0.0002	0.0046	0.0001	0.0511	0.0331	**0.0002**	0.0001	0.0004	0.0001	0.0423	0.0323
Lymphography	0.0867	0.0165	0.0796	0.0202	0.0660	0.0154	0.0960	0.0221	0.0844	0.0287	0.0537	0.0170	**0.0279**	0.0128	0.0348	0.0003
KrvskpEW	0.0491	0.0039	0.0416	0.0051	0.0474	0.0077	0.0426	0.0047	**0.0374**	0.0092	0.0433	0.0092	0.0423	0.0089	0.0428	0.0110
SPECT	0.2343	0.0140	0.1519	0.0138	0.2111	0.0152	0.1623	0.0133	0.1903	0.0180	0.1664	0.0160	0.1606	0.0132	**0.1509**	0.0161
Clean1	0.0687	0.0051	0.0760	0.0066	0.0723	0.0070	0.0772	0.0082	0.0803	0.0146	0.0801	0.0083	0.0544	0.0135	**0.0536**	0.0089
Semeion	0.0229	0.0010	0.0194	0.0017	0.0308	0.0021	0.0316	0.0017	0.0204	0.0041	0.0181	0.0035	**0.0094**	0.0023	0.0271	0.0060
Glass	0.0297	0.0185	0.0230	0.0084	0.0101	0.0129	0.0246	0.0007	0.0701	0.0000	0.0470	0.0000	**0.0010**	0.0000	**0.0010**	0.0000
Coil	0.0250	0.0023	0.0271	0.0017	0.0261	0.0028	0.0247	0.0022	0.0204	0.0053	0.0154	0.0042	**0.0059**	0.0027	0.0123	0.0037
Wine	0.0053	0.0007	0.0033	0.0006	0.0028	0.0004	0.0030	0.0004	0.0024	0.0002	0.0025	0.0003	**0.0015**	0.0000	0.0024	0.0002
Segmentation	0.0045	0.0006	0.0060	0.0069	0.0030	0.0003	0.0026	0.0004	0.0023	0.0000	0.0017	0.0003	**0.0016**	0.0004	0.0024	0.0002
Vote	0.0113	0.0083	0.0546	0.0041	0.0040	0.0006	0.0194	0.0009	0.0178	0.0000	0.0126	0.0074	**0.0013**	0.0000	0.0483	0.0047
Rand first	0		0		0		0		2		2		13		3	
Sum rank	113		106		94		94		76		63		25		70	
Average rank	6.27		5.88		5.22		5.22		4.22		3.50		1.38		3.88	
Final rank	8		7		5		6		4		2		1		3	

Table 7. Comparison between different variations of BGBO on the mean of accuracy values (M.Acc) and the standard deviation of accuracy values (as S.D).

Dataset	BGBO_S1		BGBO_S2		BGBO_S3		BGBO_S4		BGBO_V1		BGBO_V2		BGBO_V3		BGBO_V4	
	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D
Breast_cancer _wisconsin	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
IonosphereEW	0.8467	0.0130	0.8533	0.0099	0.8595	0.0093	0.8576	0.0109	0.9457	0.0121	0.9481	0.0109	**0.9652**	0.0153	0.9457	0.0165
SonarEW	0.9524	0.0084	0.9482	0.0126	0.9482	0.0094	0.9496	0.0079	0.9412	0.0171	0.9412	0.0240	**0.9612**	0.0200	0.9405	0.0174
zoo	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
Parliment1984	0.9958	0.0056	0.9770	0.0074	0.9877	0.0029	0.9939	0.0058	0.9885	0.0000	0.9943	0.0066	**1.0000**	0.0000	0.9778	0.0030
Iris	0.9333	0.0000	0.9667	0.0000	0.9667	0.0000	0.9667	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
Wdbc	**1.0000**	0.0000	0.9737	0.0000	0.9974	0.0041	0.9766	0.0042	**1.0000**	0.0000	0.9971	0.0043	**1.0000**	0.0000	0.9938	0.0041
PenglungEW	0.8629	0.0124	0.9291	0.0015	**1.0000**	0.0000	**1.0000**	0.0000	0.9489	0.0336	**1.0000**	0.0000	0.9933	0.0203	0.9578	0.0327
Lymphography	0.9189	0.0168	0.9256	0.0209	0.9389	0.0154	0.9067	0.0221	0.9176	0.0294	0.9485	0.0174	**0.9689**	0.0153	0.9666	0.0002
KrvskpEW	0.9580	0.0038	0.9643	0.0047	0.9576	0.0075	0.9625	0.0046	**0.9644**	0.0099	0.9593	0.0096	0.9604	0.0088	0.9598	0.0115
SPECT	0.7704	0.0141	**0.8522**	0.0141	0.7912	0.0156	0.8415	0.0137	0.8094	0.0188	0.8342	0.0165	0.8377	0.0103	0.8501	0.0165
Clean1	0.9379	0.0051	0.9295	0.0069	0.9323	0.0071	0.9274	0.0085	0.9207	0.0148	0.9207	0.0086	0.9463	0.0136	**0.9477**	0.0094
Semeion	0.9842	0.0010	0.9863	0.0019	0.9743	0.0022	0.9733	0.0018	0.9814	0.0039	0.9840	0.0034	**0.9883**	0.0038	0.9746	0.0058
Glass	0.9736	0.0181	0.9798	0.0080	0.9923	0.0127	0.9767	0.0002	0.9302	0.0000	0.9535	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
Coil	0.9815	0.0025	0.9787	0.0018	0.9793	0.9793	0.9804	0.0023	0.9798	0.0054	0.9853	0.0040	**0.9946**	0.0028	0.9881	0.0037
Wine	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
Segmentation	**1.0000**	0.0000	0.9982	0.0070	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
Vote	0.9928	0.0084	0.9500	0.0044	**1.0000**	0.0000	0.9833	0.0000	0.9833	0.0000	0.9889	0.0080	**1.0000**	0.0000	0.9522	0.0058
Rand first	5		4		6		5		7		6		14		7	
Sum rank	72		85		70		73		71		61		27		59	
Average rank	4		4.72		3.89		4.05		3.94		2.39		1.5		3.33	
Final rank	6		8		4		7		5		2		1		3	

Table 8. Comparison between different variations of BGBO on the mean number of the selected features (M.NF) and the standard deviation of the mean number of the selected features (as S.D).

Dataset	BGBO_S1		BGBO_S2		BGBO_S3		BGBO_S4		BGBO_V1		BGBO_V2		BGBO_V3		BGBO_V4	
	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D
Breast_cancer _wisconsin	15.0667	1.6386	12.1667	0.9499	10.8333	0.9855	10.2000	1.0635	10.8333	0.9855	**3.7667**	0.9353	4.1667	1.3153	3.9333	1.4126
IonosphereEW	19.3667	3.7645	16.4667	3.0141	14.1667	2.7803	13.0000	2.5997	14.1667	2.7803	3.4000	0.8137	3.3667	0.6150	**3.3333**	0.8841
SonarEW	44.0588	4.1754	35.7647	3.7170	32.0000	**2.7157**	30.3529	4.6360	32.0000	2.7157	**10.1765**	2.8990	10.2222	4.8780	11.2500	6.5676
zoo	5.6500	0.5871	4.8500	0.6708	3.9000	0.6407	4.1000	0.5525	3.9000	0.6407	3.1053	0.3153	3.1053	0.3153	**3.0526**	0.2294
Parliment1984	9.2000	1.4563	7.8667	1.0080	6.5667	1.2229	6.2000	1.6060	**1.0000**	0.0000	5.0000	1.1547	3.1538	0.5547	1.4138	1.5473
Iris	2.0000	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	3.0000	0.0000	**1.0000**	0.0000	2.0000	0.0000
Wdbc	13.9333	1.9640	15.5667	1.9241	14.9667	2.6061	14.8667	2.8374	5.4000	1.2421	6.9167	1.7299	**3.9333**	0.6397	7.5294	2.5029
PenglungEW	194.7667	35.5699	155.8333	14.2492	170.0000	7.3250	149.5333	3.6647	17.0000	8.1325	**5.6000**	2.6987	12.5667	3.6359	15.5000	7.6508
Lymphography	11.5667	1.6750	10.6667	1.5830	9.9000	2.6826	6.5333	1.3060	5.1111	1.4530	4.9091	0.7007	3.2000	0.7746	**3.1000**	0.3051
KrvskpEW	27.0667	3.0618	22.6333	2.9182	19.8333	2.8416	19.7333	3.2582	**7.6333**	3.6340	10.8095	2.4417	11.1200	3.0458	10.6333	3.0792
SPECT	15.5333	1.5253	12.2667	1.9815	9.7333	1.9815	11.8667	1.8889	**3.6333**	1.5643	5.0714	2.2596	5.0000	0.7071	5.4444	1.2935
Clean1	120.2000	14.3657	103.2667	8.7491	87.0667	6.7156	87.9000	7.0042	30.2333	12.8404	27.1000	12.5240	**20.9000**	10.4958	31.5667	15.9410
Semeion	192.6667	18.0370	155.6000	10.8329	142.7000	7.1252	135.2000	6.8249	52.5000	12.5114	59.6000	21.3519	56.1667	19.2158	**51.5000**	21.0316
Glass	3.4667	1.0417	3.0000	1.1447	2.4000	1.0034	1.5000	0.5724	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
Coil	680.8000	115.9005	617.0333	46.2366	567.7000	25.2752	543.5667	17.8281	**34.7333**	14.5435	86.6000	88.7467	53.4667	23.3722	54.4000	21.1702
Wine	6.8333	0.9499	4.2667	0.7397	3.6000	0.4983	3.8667	0.5074	3.0667	0.2537	3.2333	0.4302	**2.0667**	0.2537	3.0667	0.2537
Segmentation	5.8333	0.7466	5.4667	1.0080	3.8667	0.4342	3.4333	0.5040	3.0000	0.0000	2.1667	0.3791	**2.1000**	0.5477	3.0667	0.2537
Vote	6.6333	1.6078	8.1000	1.6049	6.4000	1.0372	4.6667	1.4700	2.0000	0.0000	2.5333	0.8996	2.0000	0.0000	**1.6333**	1.6501
Rand first	5		4		6		5		7		6		14		7	
Sum rank	72		85		70		73		71		61		27		59	
Average rank	4		4.72		3.89		4.05		3.94		2.39		1.5		3.33	
Final rank	6		8		4		7		5		2		1		3	

5.1. Comparison between the BGBO based approaches

The results of the average classification accuracy are shown in Table 7. Among them, BGBO_V3 achieved the best results, it managed to reach the highest precision in nine data sets, and in addition, it had the highest precision value among all methods in five data sets, so it was placed in the first place in the overall ranking. BGBO_V2 ranked second, it achieved the highest precision in six data sets, but it ranked better overall, so it ranked second. BGBO_V4 is in third place, and although it has six datasets that achieve the highest precision and additionally has the highest precision value of all methods in one dataset. It ranks relatively low in the overall ranking compared to BGBO_V2. Similarly, BGBO_S3 ranks fourth and BGBO_V1 ranks fifth. BGBO_S1, BGBO_S4, and BGBO_S2 rank sixth seventh, and eighth, respectively. In addition, their standard deviation, we can see that the standard deviation of BGBO_V3 in all nine data sets is 0, which is sufficient to prove that it is BGBO_V3 a robust method.

In Table 8, the proposed BGBO method is compared in terms of the average number of selected features.BGBO_V3 has the smallest average number of features in 14 datasets, while BGBO_V1 and BGBO_V4 have the smallest values in 7 datasets. BGBO_V3 and BGBO-S3 had the smallest values in 6 data sets, while BGBO_S1, BGBO_S2, and BGBO-S4 had the smallest number of features in 5, 4, and 5 data sets, respectively. As for the standard deviation values, BGBO_V3 proved to be a robust method that obtained small deviation values in the data set of 7. The superiority of BGBO_V3 was also confirmed by the average adaptation results shown in Table 6.

5.2. Comparison with other metaheuristic-based approaches

Table 9. Parameters of BGBO and comparison algorithms.

10	Parameters	Values
BPSO	c_1, c_2	2, 2
	ω	0.1
BWOA	au	[2, 0]
	b	1
BDA	α	0.99
	β	0.01
BBA	$Q_{\min}, Q_{\max}$	0, 2
	A *Loudess*, r *Pulse rate*	0.5, 0.5
BGOA	interval	[0, 2.079]
	l, f	1.5, 0.5
BGWO	au	[2, 0]
BHHO	au	[2, 0]
BGBO	pr	0.5

In this section, BGBO_V3, which has the best performance among the eight methods introduced above, is considered as the best method among the methods proposed in this work. The proposed BGBO_V3 is compared with other existing state-of-the-art binary metaheuristics such as BPSO, BWOA, BDA, BBA, BGOA, BGWO, and BHHO. Table 9 shows the used parameters of all algorithms in the next experiments. Again, the performance is compared and analyzed by the average fitness value, the average classification accuracy, and the average number of selected features.

Table 10 shows the experimental results of the average fitness values of BGBO_V3 and other metaheuristic-based methods. From Table 10, it can be seen that BGBO_V3 has the best results in 89% of the datasets (16 out of 18), which indicates that BGBO_V3 has the best performance. BDA and BBA achieved first place in two datasets, KrvskpEW and SonarEW, respectively, and BPSO, BWOA, BGOA, BGWO, and BHHO are not ranked first in any of the 18 datasets. This indicates that the performance of the proposed method is significant. In addition, it can be seen from Table 10 that BGBO_V3 performs significantly better than other methods in some datasets, such as Lymphography and IonosphereEW. In addition, comparing the standard deviations of the various methods, BGBO_V3 is also more stable. In terms of the final average ranking, BGBO_V3 ranked first, followed by BHHO, BDA, BWOA, BGOA, BBA, BGWO, and BPSO.

In Table 11, BGBO_V3 is compared with other methods based on the average classification accuracy. Among the 18 datasets, BGBO_V3 ranked first with 13 datasets compared to 1, 1, 4, 6, 2, 3, and 7 for BPSO, BWOA, BDA, BBA, BGOA, BGWO, and BHHO, respectively. The comparison with k-NN classifier also shows that BGBO_V3 has higher classification accuracy. In terms of the number of top-ranked methods, BGBO_V3 performed much better than the other methods. In addition, BGBO_V3 achieves an average classification accuracy of 100% on the Breast_cancer_wisconsin, Zoo, Parliment1984, Iris, Glass, Wine, Segmentation, and Vote datasets. Specifically, the classification accuracy of BGBO_V3 reaches 99% for both the high-dimensional datasets PenglungEW and Coil, indicating that the proposed method in this paper can better solve high-dimensional data. This indicates that the proposed method is important in practical applications. In addition, the standard deviation results show that BGBO_V3 is more stable than other methods on some data sets. The final ranking BGBO_V3 is also in the first place. It is followed by BHHO, BBA, BDA, BGOA, BGWO, BPSO, and BWOA.

Inspecting the results of the number of selected features from Table 12, we observe that BGBO_V3 outperforms the other algorithms, followed by BPSO, BBA, BDA, BWOA, BHHO, BGWO, and BGOA, with very competitive results. For most of the datasets, the differences between the average numbers of features selected between the algorithms are very small. However, it is worth noting that for the PenglungEW dataset, the highest dimensional dataset with 325 features, BGBO_V3 achieves the best classification accuracy of 99.78% with the smallest number of features (only 12.5667 features on average). As well as for Coil, the highest dimensional dataset with 1024 features, BGBO_V3 achieves the best classification accuracy of 99.46% with the smallest number of features (only 53.4667 features on average). This shows the advantage of this method when dealing with high-dimensional datasets.

Table 10. Comparison between BGBO_V3 and state-of-art methods based on the mean of fitness values (Mean) and the standard deviation of fitness values (as S.D).

Dataset	BGBO_V3		BPSO		BWOA		BDA		BBA		BGOA		BGWO		BHHO	
	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D
Breast_cancer _wisconsin	**0.0017**	**0.0008**	0.0324	0.0038	0.0225	0.0033	0.0106	0.0022	0.0131	0.0074	0.0025	0.0026	0.0290	0.0021	0.0047	**0.0008**
IonosphereEW	**0.0326**	0.0114	0.0642	0.0080	0.0890	0.0086	0.0729	0.0183	0.0683	0.0114	0.0910	0.0083	0.1172	0.0087	0.0719	**0.0061**
SonarEW	0.0543	0.0194	0.1547	0.0128	0.1206	0.0182	0.0582	0.0198	**0.0523**	0.0261	0.1182	0.0160	0.0571	**0.0122**	0.0801	0.0132
zoo	**0.0020**	**0.0002**	0.1251	0.0073	0.0035	0.0010	0.0032	**0.0002**	0.0022	0.0003	0.0089	0.0122	0.0029	0.0004	0.0034	0.0005
Parliment1984	**0.0020**	**0.0003**	0.0634	0.0103	0.0182	0.0053	0.0167	0.0027	0.0229	0.0038	0.0269	0.0028	0.0493	0.0005	0.0426	0.0047
Iris	**0.0025**	**0.0000**	0.0750	**0.0000**	0.0355	**0.0000**	0.0050	**0.0000**	0.0355	**0.0000**	0.0358	0.0013	0.0355	**0.0000**	0.0380	**0.0000**
Wdbc	**0.0013**	**0.0002**	0.0724	0.0096	0.0048	0.0027	0.0128	0.0031	0.0051	0.0043	0.0213	0.0006	0.0309	0.0068	0.0038	0.0008
PenglungEW	**0.0004**	**0.0001**	0.1710	0.0081	0.1242	0.0383	0.0700	0.0011	0.0675	0.0177	0.1528	0.0256	0.2539	0.0271	0.0034	0.0004
Lymphography	**0.0279**	0.0128	0.1143	0.0079	0.1292	0.0371	0.1044	0.0197	0.1361	0.0170	0.0730	0.0146	0.0430	0.0112	0.1371	**0.0011**
KrvskpEW	0.0423	0.0089	0.1322	0.0085	0.0570	0.0102	**0.0407**	0.0074	0.0787	0.0053	0.0519	0.0048	0.0650	0.0080	0.0455	**0.0043**
SPECT	**0.1606**	0.0132	0.1836	0.0274	0.2251	0.0243	0.1943	0.0171	0.2172	0.0258	0.1811	0.0142	0.1859	0.0132	0.1724	**0.0114**
Clean1	**0.0544**	0.0135	0.1954	0.0071	0.0784	0.0081	0.0633	0.0157	0.1479	0.0113	0.0711	0.0057	0.0653	0.0077	0.0647	**0.0067**
Semeion	**0.0094**	**0.0023**	0.1384	0.0035	0.0235	0.0053	0.0169	0.0033	0.0957	0.0052	0.0152	0.0018	0.0160	0.0024	0.0169	0.0014
Glass	**0.0010**	**0.0000**	0.0788	**0.0000**	0.0176	0.0104	0.0246	0.0010	0.0626	**0.0000**	0.0061	0.0099	0.0659	0.0106	0.0022	0.0004
Coil	**0.0059**	0.0027	0.1548	0.0024	0.0163	0.0033	0.0231	0.0024	0.1358	0.0043	0.0206	**0.0019**	0.0485	0.0019	0.0183	0.0027
Wine	**0.0015**	**0.0000**	0.0687	0.0035	0.0026	0.0008	0.0056	0.0061	0.0034	0.0006	0.0039	0.0005	0.0043	0.0005	0.0032	0.0006
Segmentation	**0.0016**	**0.0004**	0.0864	0.0020	0.0027	0.0005	0.0812	0.0115	0.0030	0.0005	0.0031	0.0005	0.0042	0.0005	0.0032	0.0007
Vote	**0.0013**	**0.0000**	0.0590	0.0124	0.0109	0.0079	0.0022	0.0009	0.0576	0.0119	0.0220	0.0040	0.0034	0.0030	0.0176	0.0048
Rand first	16		0		0		1		1		0		0		0	
Sum rank	20		127		83		71		86		85		94		70	
Average rank	1.11		7.06		4.61		3.94		4.78		4.72		5.22		3.89	
Final rank	1		8		4		3		6		5		7		2	

Table 11. Comparison between BGBO_V3 and state-of-art methods based on the mean of accuracy values (M.Acc) and the standard deviation of fitness values (as S.D).

Dataset	BGBO_V3		BPSO		BWOA		BDA		BBA		BGOA		BGWO		BHHO		k-NN
	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	Acc
Breast_cancer _wisconsin	**1.0000**	0.0000	0.9737	0.0124	0.9820	0.0035	0.9918	0.0022	**1.0000**	0.0000	0.9901	0.0030	0.9743	0.0022	**1.0000**	0.0000	0.9912
IonosphereEW	**0.9681**	0.0117	0.9391	0.0083	0.9136	0.0086	0.9291	0.0182	0.9314	0.0060	0.9119	0.0085	0.8862	0.0088	0.9314	0.0060	0.8571
SonarEW	0.9476	0.0202	0.9333	0.0202	0.8833	0.0188	0.9452	0.0199	**0.9508**	0.0265	0.8865	0.0162	0.9460	0.0124	0.9235	0.0138	0.8571
zoo	**1.0000**	0.0000	0.9400	0.0423	0.9481	0.0761	**1.0000**	0.0000	**1.0000**	0.0000	0.9967	0.0127	**1.0000**	0.0000	**1.0000**	0.0000	0.9500
Parliment1984	**1.0000**	0.0000	0.9540	0.0000	0.9773	0.0607	0.9877	0.0029	0.9793	0.0047	0.9778	0.0029	0.9540	0.0000	0.9621	0.0056	0.9770
Iris	**1.0000**	0.0000	**1.0000**	0.0000	0.8389	0.0619	**1.0000**	0.0000	0.9667	0.0000	0.9667	0.0000	0.9667	0.0000	0.9667	0.0000	**1.0000**
Wdbc	**1.0000**	0.0000	0.9819	0.0128	0.9619	0.0121	0.9899	0.0032	0.9977	0.0046	0.9825	0.0000	0.9740	0.0071	**1.0000**	0.0000	0.9825
PenglungEW	0.9978	0.0122	0.9356	0.0122	0.9942	0.0088	0.9331	0.0011	0.9349	0.0178	0.8505	0.0005	0.8305	0.0025	**1.0000**	0.0000	0.8667
Lymphography	**0.9736**	0.0137	0.9367	0.0237	0.7851	0.0411	0.8983	0.0202	0.8667	0.0175	0.9310	0.0150	0.9622	0.0115	0.8667	0.0000	0.9000
KrvskpEW	0.9604	0.0088	0.9425	0.0104	0.7567	0.1432	**0.9639**	0.0070	0.9566	0.0118	0.9541	0.0046	0.9393	0.0082	0.9601	0.0044	0.9296
SPECT	0.8393	0.0135	0.8377	0.0320	**0.9145**	0.0787	0.8076	0.0294	0.7843	0.0261	0.8226	0.0145	0.8170	0.0132	0.8302	0.0119	0.6604
Clean1	0.9463	0.0136	0.9042	0.0150	0.9382	0.1012	0.9405	0.0158	**0.9512**	0.0098	0.9330	0.0059	0.9404	0.0080	0.9400	0.0071	0.8842
Semeion	**0.9925**	0.0023	0.9732	0.0079	0.9737	0.0913	0.9875	0.0033	0.9832	0.0052	0.9896	0.0019	0.9903	0.0025	0.9885	0.0016	0.9781
Glass	**1.0000**	0.0000	0.9302	0.0000	0.9716	0.0062	0.9767	0.0000	0.9535	0.0000	0.9969	0.0081	0.9357	0.0100	**1.0000**	0.0000	0.9250
Coil	**0.9946**	0.0028	0.9740	0.0034	0.9736	0.0203	0.9813	0.0023	0.9858	0.0041	0.9841	0.0020	0.9574	0.0020	0.9861	0.0030	0.9757
Wine	**1.0000**	0.0000	0.9898	0.0154	0.9764	0.0717	0.9986	0.0062	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**
Segmentation	**1.0000**	0.0000	0.9546	0.0154	0.9444	0.1126	0.9681	0.0316	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**
Vote	**1.0000**	0.0000	0.9667	0.0000	0.9925	0.0085	**1.0000**	0.0000	0.9456	0.0123	0.9822	0.0042	0.9994	0.0030	0.9850	0.0051	0.9333
Rand first	13		1		1		4		6		2		3		7		3
Sum rank	23		112		123		73		67		87		92		58		113
Average rank	1.27		6.22		6.83		4.05		3.72		4.83		5.11		3.22		6.27
Final rank	1		7		9		4		3		5		6		2		8

Table 12. Comparison between BGBO_V3 and state-of-art methods based on the mean number of the selected features (M.NF) and the standard deviation of the mean number of the selected features (asS.D).

Dataset	BGBO_V3		BPSO		BWOA		BDA		BBA		BGOA		BGWO		BHHO	
	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D	M.NF	S.D
Breast_cancer _wisconsin	**4.1667**	1.3153	6.0000	0.0000	14.1500	2.8149	7.4000	1.5669	8.7667	2.0457	16.9000	2.6307	10.6667	1.8998	14.0000	2.5495
IonosphereEW	**3.3667**	1.0554	12.9667	3.0454	11.5500	2.8186	9.0333	2.6061	7.8000	2.0240	12.9000	2.2796	15.4333	2.0957	13.5000	3.2059
SonarEW	**10.2222**	5.8765	21.6000	2.7990	30.7500	4.7780	24.0667	3.5129	21.3667	4.5975	34.8333	4.9416	22.0000	3.8952	26.0000	5.0839
zoo	**3.1053**	0.3153	4.4333	1.4065	25.3000	7.2410	5.1000	0.3078	3.5000	0.5086	9.0000	1.2594	4.6667	0.6609	5.4000	0.8433
Parliment1984	3.1538	0.5547	**1.6667**	0.5467	6.8182	1.8878	7.3333	1.1244	3.8500	1.6311	7.8000	1.7301	6.0333	0.8087	8.0000	2.1602
Iris	**1.0000**	0.0000	**1.0000**	0.0000	8.7000	2.7549	2.0000	0.0000	**1.0000**	0.0000	1.1333	0.5074	**1.0000**	0.0000	2.0000	0.0000
Wdbc	3.9333	0.6397	5.9667	1.3257	**1.3571**	0.7450	8.5000	1.7622	8.3000	1.8597	11.7667	1.6955	15.3000	1.9502	11.3333	2.3452
PenglungEW	**12.5667**	5.5207	136.3333	5.4414	15.6667	5.6347	123.0000	8.8318	99.4667	7.3237	156.9667	10.3140	157.9333	19.0985	110.4286	14.1287
Lymphography	**3.2000**	1.5102	4.2000	0.8469	112.0500	56.4880	6.7500	1.5517	7.4000	1.6103	8.5000	2.3007	10.0000	1.6189	9.2000	1.9322
KrvskpEW	11.1200	3.0458	11.0333	1.5196	6.8000	3.7947	18.0500	3.9799	**5.8000**	0.6644	23.1667	4.4728	17.4333	2.7753	21.7000	3.6833
SPECT	**5.0000**	0.6633	5.1333	1.4559	23.6842	7.3866	8.2500	2.4895	7.9000	2.2644	12.1667	1.7237	10.3667	2.0759	9.5000	2.0736
Clean1	20.9000	10.4958	71.4333	5.5936	**8.0588**	4.6834	73.4000	6.8626	63.3333	5.2413	80.4667	5.2636	104.9000	5.9094	88.7000	26.9528
Semeion	**56.1667**	15.9300	105.6333	4.4216	86.3333	39.1504	118.3333	9.5171	74.1333	5.2176	128.9667	7.1943	168.0667	6.0168	147.3333	26.8229
Glass	**1.0000**	0.0000	**1.0000**	0.0000	136.6500	38.8875	1.5667	0.9714	**1.0000**	0.0000	2.9667	2.5527	2.1667	1.1167	2.2000	0.4216
Coil	**53.4667**	23.3722	466.1667	8.4897	438.1000	194.4766	467.9000	21.7895	429.6667	14.6836	502.3667	17.5528	650.0333	17.0162	469.6667	70.6417
Wine	**2.0667**	0.0000	2.6667	0.4795	5.1000	1.6190	5.4500	1.0501	4.4000	0.8208	5.4000	0.6992	5.6500	0.5871	4.1250	0.8345
Segmentation	**2.1000**	0.5477	2.3667	0.5561	5.4000	2.0633	2.5500	0.6048	3.8500	0.5871	4.4000	0.6992	5.5000	0.6883	4.1000	0.8756
Vote	2.0000	0.0000	**1.9000**	0.6618	5.5500	1.4318	3.5500	1.5035	5.9667	2.0083	7.1000	1.2959	4.6333	0.8503	4.3500	0.6708
Rand first	13		4		2		0		3		0		1		0	
Sum rank	25		48		94		83		52		120		112		104	
Average rank	1.39		2.67		5.22		4.61		2.88		6.67		6.22		5.78	
Final rank	1		2		5		4		3		8		7		6	

In terms of the fitness function of feature selection, the fitness function is optimal only when the accuracy and the number of selected features are simultaneously good. From the analysis of the experimental results in Tables 10–12, the fitness value of the BGBO_V3 algorithm proposed in this paper is superior to other algorithms in most data sets. Besides, the BGBO_V3 algorithm obtains a better subset of features while maintaining high accuracy. Although the other algorithms have higher accuracy, the optimal feature subsets obtained by them are not satisfactory. Figures 2–10 shows the average convergence curves of various algorithms on different data sets. It is clear from the convergence plots that BGBO_V3 outperforms the other methods in terms of the average fitness value from the beginning of the iteration. It is worth mentioning that BGBO_V3 has the fastest convergence on most of the datasets, obtaining better results in balancing the exploration and exploitation capabilities.

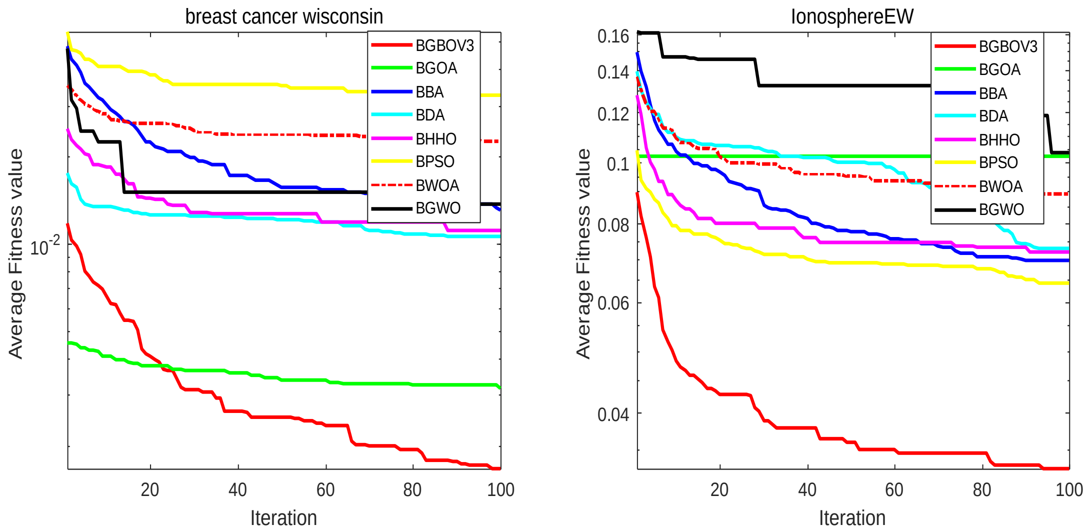

Figure 2. Convergence curves for the Breast_cancer_wisconsin and IonosphereEW dataset.

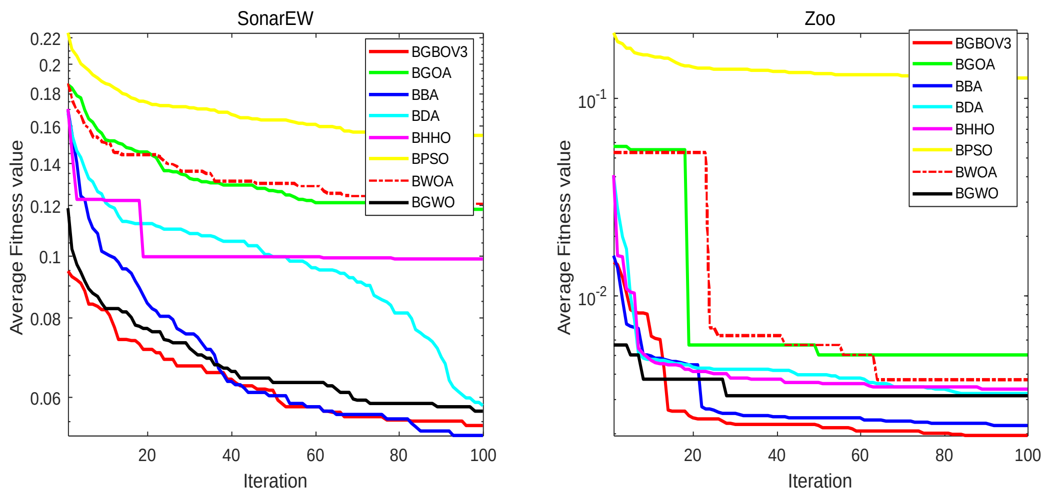

Figure 3. Convergence curves for the SonarEW and Zoo dataset.

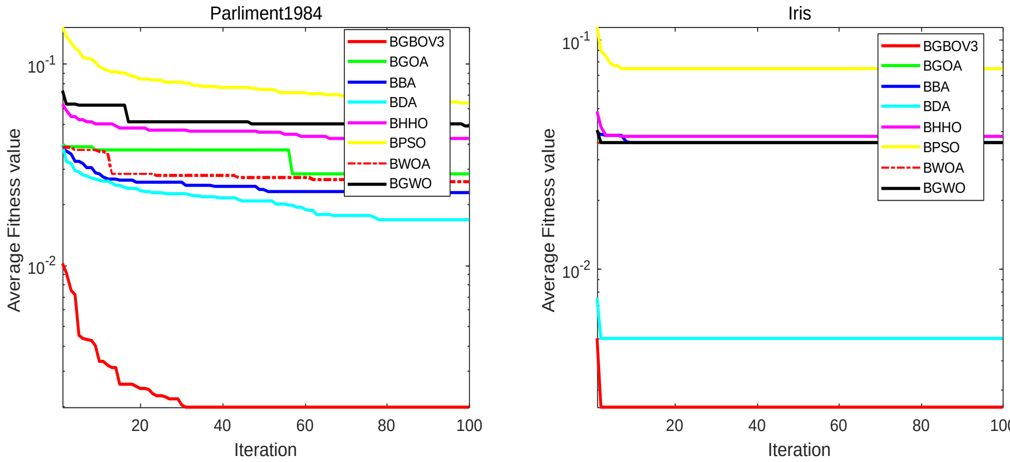

Figure 4. Convergence curves for the Parliment1984 and Iris dataset.

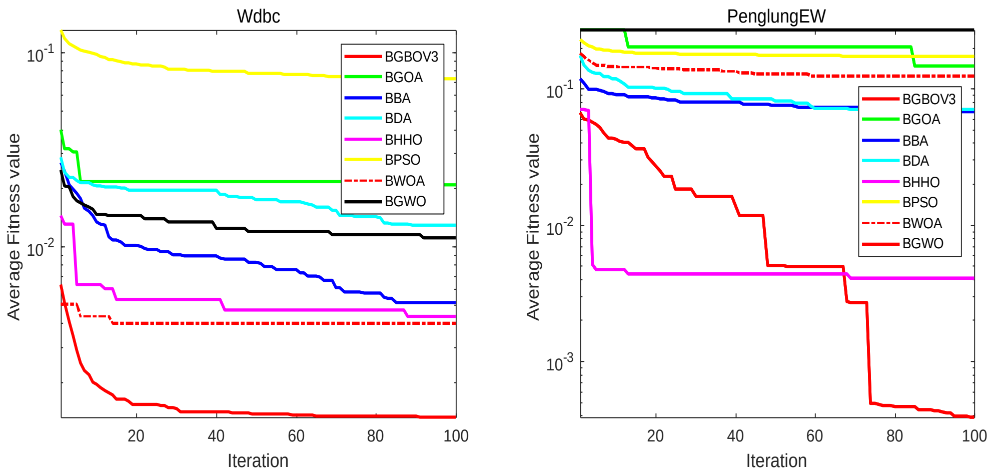

Figure 5. Convergence curves for the Wdbc and PenglungEW dataset.

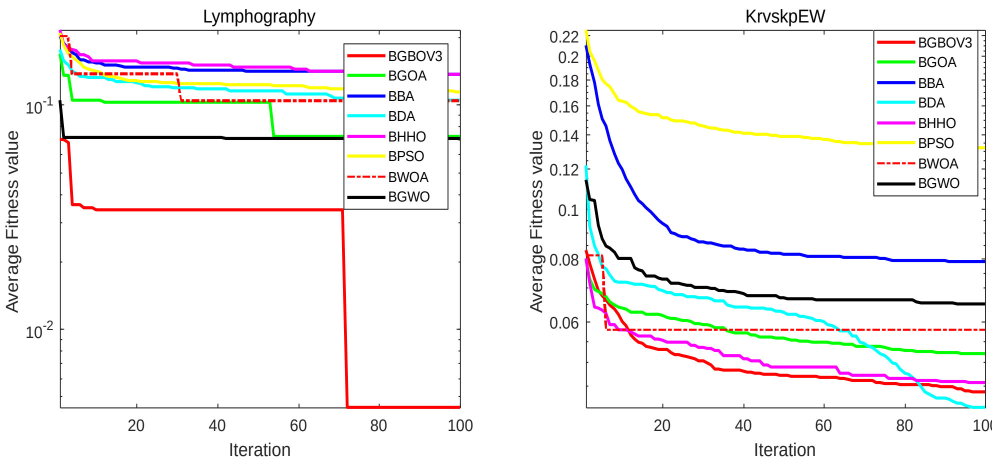

Figure 6. Convergence curves for the Lymphography and KrvskpEW dataset.

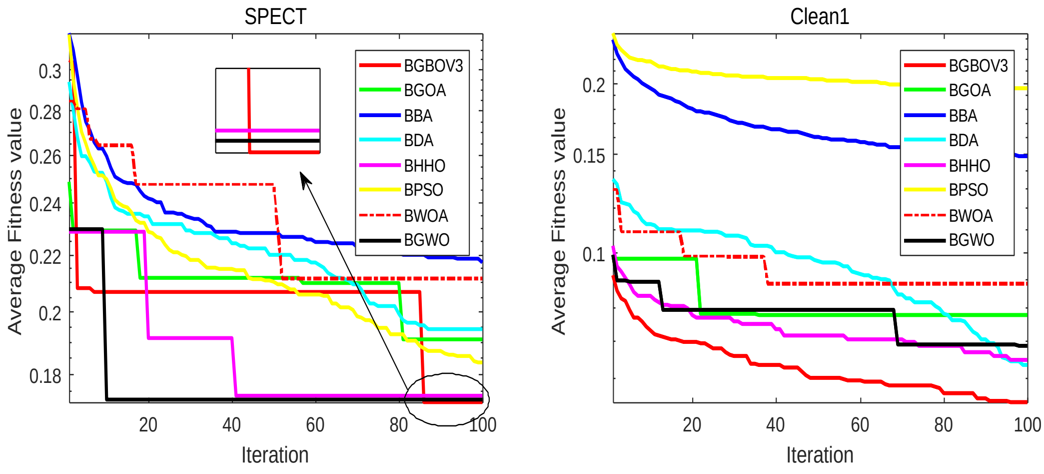

Figure 7. Convergence curves for the SPECT and Clean1 dataset.

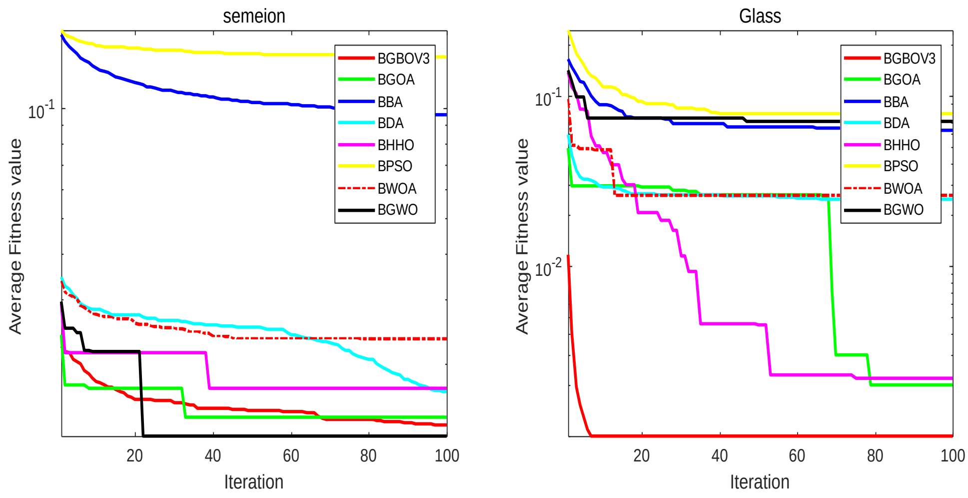

Figure 8. Convergence curves for the semeion and Glass dataset.

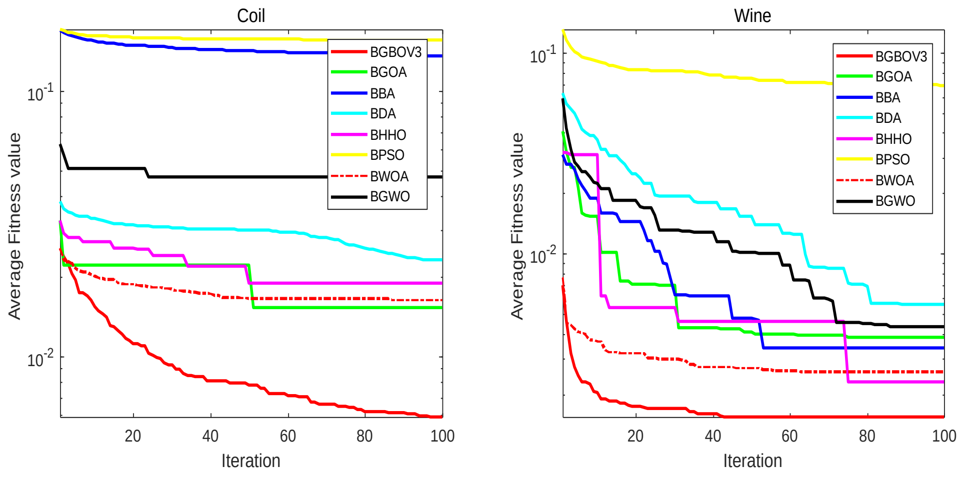

Figure 9. Convergence curves for the Coil and Wine dataset.

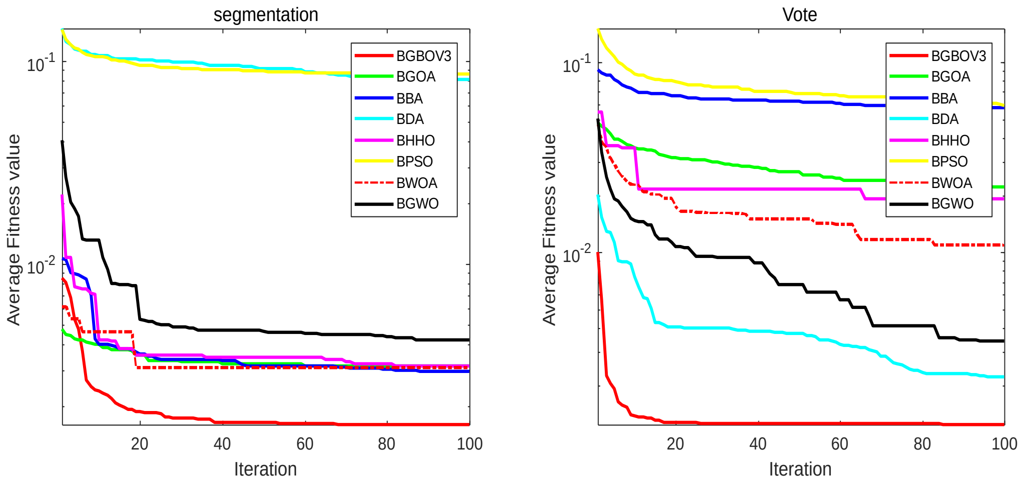

Figure 10. Convergence curves for the segmentation and Vote dataset.

Table 13 The *p*-values of Wilcoxon signed-rank test for the fitness value result of the BGBO_V3 vs. other algorithms ($p \geq 0.05$ are underlined).

Dataset	BPSO	BWOA	BDA	BBA	BGOA	BGWO	BHHO
Breast_cancer _wisconsin	2.46E-11	2.42E-09	3.82E-11	5.60E-11	2.30E-11	2.40E-11	4.20E-09
IonosphereEW	2.75E-11	2.70E-09	1.16E-10	2.68E-11	2.74E-11	2.73E-11	2.67E-10
SonarEW	2.94E-11	4.77E-09	0.023159	**<u>0.899934</u>**	2.96E-11	0.023094	4.71E-10
zoo	2.83E-12	1.13E-11	4.71E-11	0.002544	3.32E-12	6.81E-11	5.83E-11
Parliment1984	4.86E-08	1.57E-05	1.46E-07	3.40E-07	1.58E-07	6.99E-08	1.57E-09
Iris	1.38E-09	5.58E-06	1.61E-05	1.38E-09	2.60E-08	1.38E-09	4.57E-08
Wdbc	1.50E-11	3.20E-10	1.10E-09	1.40E-11	1.25E-11	1.14E-09	2.12E-09
PenglungEW	2.88E-11	3.72E-09	2.92E-09	3.58E-11	2.95E-11	2.94E-11	1.52E-07
Lymphography	1.77E-11	1.49E-11	1.99E-09	2.55E-11	2.70E-11	2.53E-11	1.48E-11
KrvskpEW	2.81E-11	5.18E-10	0.013447	2.23E-12	1.24E-06	3.12E-11	**<u>0.582786</u>**
SPECT	1.73E-04	3.81E-08	7.27E-08	1.99E-10	7.24E-09	5.76E-08	3.36E-06
Clean	3.00E-11	2.47E-05	0.025874	3.00E-11	4.66E-08	5.86E-04	1.09E-05
Semeion	3.01E-11	1.05E-07	0.01122	3.00E-11	**<u>0.08234</u>**	0.006661	0.023141
Glass	1.69E-14	3.79E-11	2.49E-13	1.69E-14	5.31E-06	5.79E-13	6.29E-10
Coil	3.00E-11	4.86E-09	3.02E-11	3.02E-11	3.00E-11	3.02E-11	7.34E-06
Wine	7.29E-13	1.31E-07	5.63E-11	3.10E-11	7.80E-09	4.39E-11	5.34E-08
Segmentation	5.15E-13	2.86E-09	3.06E-11	3.76E-10	2.85E-08	5.96E-11	1.81E-08
Vote	5.59E-13	1.86E-11	5.59E-07	1.14E-12	1.00E-12	7.43E-13	1.37E-11

A nonparametric Wilcoxon rank-sum test was performed at the 5% significance level to verify whether there was a statistical difference between the results of the fitness values of BGBO_V3 and the respective comparison methods. Table 13 shows the p-values of BGBO_V3 concerning each comparison method. In terms of fitness values, we observed that BGBO_V3 exhibited statistically significant classification accuracy in all datasets compared to BPSO, BWOA, BDA, and BGWO. Significant differences (p-values less than 0.05) exist for all datasets. Comparing BGBO_V3 with BBA, BGOA, and BHHO, we can note that BGBO_V3 has p-values greater than 0.05 in the SonarEW, Semeion, and KrvskpEW datasets, respectively.

5.3. Comparison on high-dimensional datasets

From the previous subsection, we know that BGBO_V3 achieves excellent results on high-dimensional datasets (PenglungEW and Coil). Therefore, to further test the performance of the algorithm proposed in this paper, this section conducts experiments with 10 high-dimensional datasets. These datasets were downloaded from the UCI database and the database of Arizona State University (ASU) [91]. The information (number of instances, features, and classes) of the high-dimensional datasets is shown in Table 14. Tables 15–17 show the experimental results of the average fitness value, average classification accuracy and, the average number of selected features obtained by various methods on the high-dimensional datasets.

Table 14. List of used high-dimensional datasets.

No.	Datasets	Instances	Number of features (d)	Number of classes (k)
1.	arcene	200	10,000	2
2.	ORL	400	1024	40
3.	orlraws10P	100	10,304	10
4.	PCMAC	1943	3289	2
5.	pixraw10P	100	10,000	10
6.	warpPIE10P	210	2420	10
7.	RELATHE	1427	4322	2
8.	Yale	165	1024	15
9.	warpAR10P	130	2400	10
10.	Leukemia	72	7129	2

We first inspect the fitness function averages of all the compared algorithms on the high-dimensional datasets. The results are shown in Table 15. From Table 15, we can notice that BGBO_V3 proposed in this paper achieves the best results in 80% of the datasets (8 out of 10), among the other methods, only BWOA and BBA achieve the best results in ORL and Yale, respectively. In addition, looking at the standard deviation again, we can see that BGBO_V3 achieves 0 for all four datasets (i.e., arcene, PCMAC, RELATHE, and Leukemia), indicating that the method has excellent stability on high-dimensional datasets. In the final comparative ranking, BGBO_V3 is first, followed by BDA, BGWO, BBA, BPSO, BHHO, BWOA, and BGOA in order.

Table 16 shows the experimental results of the average classification accuracy of all methods on high-dimensional datasets. From the comparison of the average classification accuracy, BGBO_V3 ranks first in the average classification accuracy on 8 out of 10 high-dimensional datasets, accounting for 80% of all datasets. Other methods ranked first in the number of datasets far less than BGBO_V3. In particular, the average classification accuracy of BGBO_V3 reached 100% on the three datasets of arcene, RELATHE, and Leukemia. In addition, the standard deviation, it can be seen that BGBO_V3 achieves 0 on all four datasets (i.e., arcene, PCMAC, RELATHE, and Leukemia), which indicates from both the average fitness value and the average accuracy that BGBO_V3 has excellent performance as well as good stability on high-dimensional datasets. In the final ranking of the average classification accuracy comparison, BGBO_V3 is still the first, while the KNN machine learning method is ranked last among all methods. The remaining rankings are BGWO, BDA, BHHO, BBA, BPSO, BWOA, and BGOA in that order.

Finally, Table 17 shows the experimental results of all methods for the average number of selected features. As can be seen from the table, BGBO_V3 can obtain the smallest number of features on all data sets. Especially from the numerical point of view, the feature values obtained by BGBO_V3 are far smaller than those obtained by the other methods, even with a difference of hundreds of times. This indicates that BGBO_V3 can find a smaller subset of features compared to other methods. Similarly, the standard deviation of BGBO_V3 is within the acceptable range. In terms of the final ranking, BGBO_V3 ranks first, followed by BHHO, BDA, BBA, BPSO, BWOA, BGOA, and BGWO. Figures 11–15 show the average convergence curves of the various algorithms on the high-dimensional dataset. From the convergence plots, it is clear that BGBO_V3 achieves a good convergence rate from the beginning of the iteration to about the 50th generation, and the average fitness value is better than the other methods. The convergence curve of BGBO_V3 can be clearly distinguished from 50 to 100 generations, which shows the advantage of the method relative to other methods. This is due to the excellent mechanism of the GBO algorithm itself, which enables us to balance the process of exploration and exploitation in the algorithm.

5.4. Comparisons with metaheuristics in the literature

In this section, the average classification accuracy of BGBO_V3 is compared with other methods in the literature. These experimental results are also obtained in the same experimental setting with a good reference effect. Since the selected datasets are only partially the same, 10 datasets are selected for comparison and analysis in this work, which contains some important low-dimensional, high-dimensional datasets, respectively. Table 18 shows the experimental data of the average classification accuracy of BGBO_V3 with various methods in the literature.

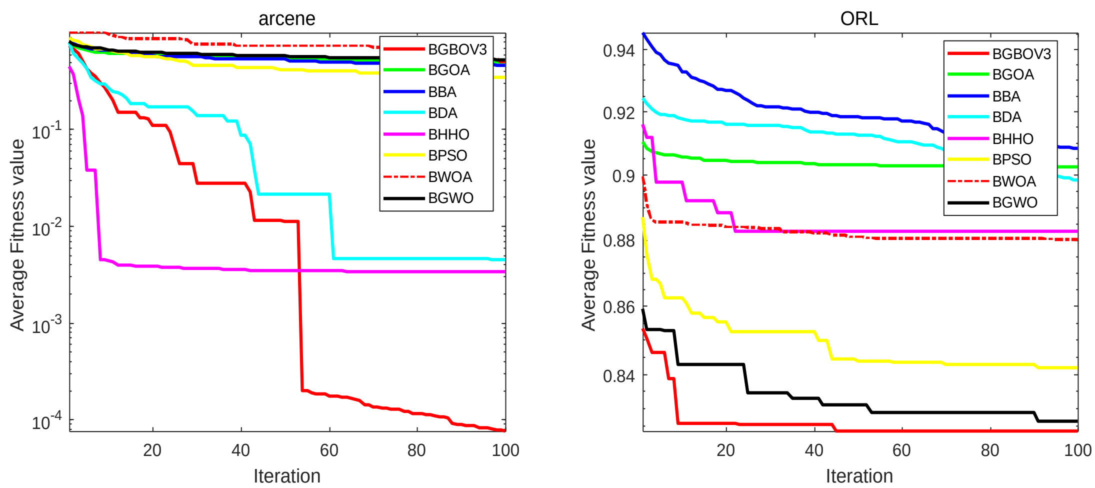

Figure 11. Convergence curves for the arcene and ORL dataset.

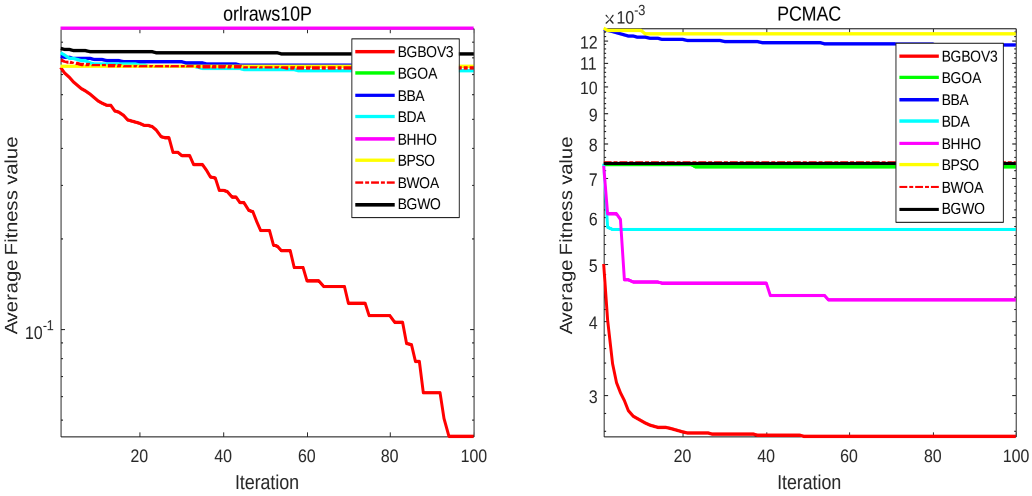

Figure 12. Convergence curves for the orlraws10P and PCMAC dataset.

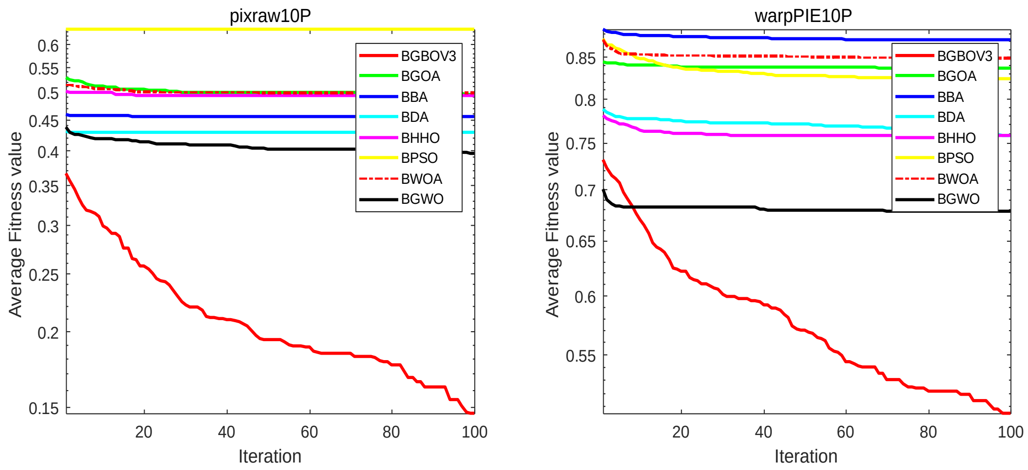

Figure 13. Convergence curves for the pixraw10P and warpPIE10P dataset.

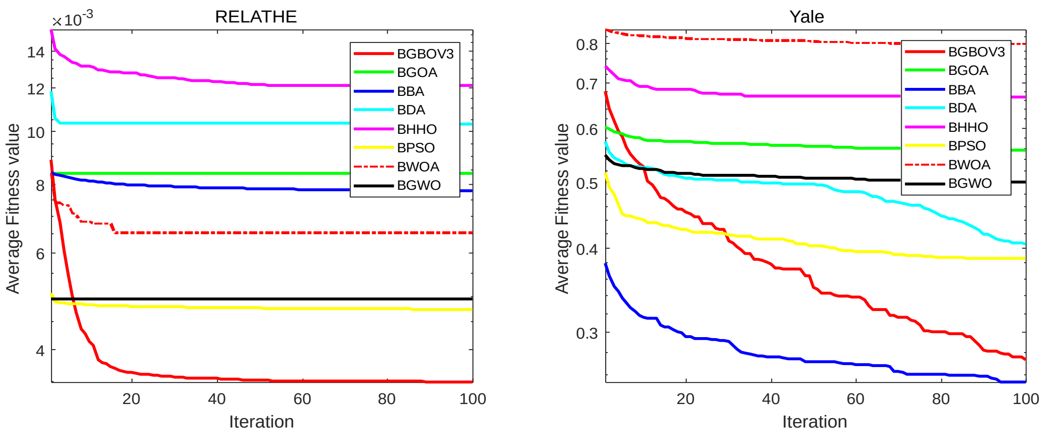

Figure 14. Convergence curves for the RELATHE and Yale dataset.

Table 15. Comparison between different variations of BGBO on the mean of fitness values (as Mean) and the standard deviation of fitness values (as S.D).

Dataset	BGBO_V3		BPSO		BWOA		BDA		BBA		BGOA		BGWO		BHHO	
	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D
arcene	**0.0001**	0.0000	0.3405	0.2431	0.4996	0.4949	0.0045	0.0003	0.4557	0.1617	0.4999	0.0000	0.5265	0.0728	0.0034	0.0003
ORL	**0.8134**	0.0087	0.8418	0.0072	0.8799	0.0019	0.8983	0.0093	0.9081	0.0207	0.9023	0.0021	0.8155	0.0073	0.8838	0.0028
orlraws10P	**0.0441**	0.1141	0.7473	0.0000	0.9949	0.0000	0.7194	0.0298	0.7449	0.0472	0.9949	0.0000	0.6379	0.0622	0.9916	0.0001
PCMAC	**0.0025**	0.0000	0.0123	0.0000	0.0074	0.0001	0.0057	0.0003	0.0118	0.0000	0.0073	0.0000	0.0074	0.0000	0.0042	0.0002
pixraw10P	**0.1461**	0.0383	0.6348	0.0000	0.4973	0.0091	0.4278	0.0004	0.4545	0.0000	0.4947	0.0143	0.3956	0.0185	0.4937	0.0158
warpPIE10P	**0.5047**	0.0531	0.8230	0.0059	0.8481	0.0013	0.7580	0.0126	0.8710	0.0027	0.8366	0.0042	0.6779	0.0142	0.7579	0.0079
RELATHE	**0.0035**	0.0000	0.0047	0.0000	0.0049	0.0001	0.0103	0.0003	0.0078	0.0000	0.0084	0.0000	0.0049	0.0001	0.0121	0.0001
Yale	0.2731	0.0877	0.3858	0.0239	0.7983	0.0131	0.4054	0.0446	**0.2507**	0.0325	0.5573	0.0132	0.4999	0.0069	0.6664	0.0173
warpAR10P	**0.1210**	0.0557	0.2815	0.0231	0.4967	0.0159	0.4063	0.0361	0.2031	0.0605	0.4486	0.0170	0.5068	0.0168	0.3502	0.0224
Leukemia	**0.0000**	0.0000	0.1463	0.0000	0.0756	0.0000	0.0036	0.0004	0.0044	0.0000	0.1915	0.0341	0.0049	0.0001	0.1419	0.0129
Rand first	9		0		1		0		1		0		0		0	
Sum rank	11		48		57		41		47		62		42		51	
Average rank	1.1		4.8		5.7		4.1		4.7		6.2		4.2		5.1	
Final rank	1		5		7		2		4		8		3		6	

Table 16. Comparison between BGBO_V3 and state-of-art methods based on the mean of accuracy values (M.Acc) and the standard deviation of fitness values (as S.D).

Dataset	BGBO_V3		BPSO		BWOA		BDA		BBA		BGOA		BGWO		BHHO		k-NN
	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	M.Acc	S.D	S.D
arcene	**1.0000**	0.0000	0.6611	0.2456	0.5000	0.5000	**1.0000**	0.0000	0.5444	0.1634	0.5000	0.0000	0.4732	0.0735	**1.0000**	0.0000	0
ORL	**0.1799**	0.0085	0.1546	0.0072	0.1176	0.0016	0.0974	0.0094	0.0873	0.0209	0.0949	0.0022	0.1828	0.0074	0.1126	0.0024	0.0968
orlraws10P	**0.9556**	0.1153	0.2500	0.0000	0.2000	0.1010	0.2778	0.0302	0.2522	0.0477	0.2100	0.0100	0.3611	0.0632	0.3310	0.0130	0.2000
PCMAC	**0.9974**	0.0000	0.9923	0.0000	**0.9974**	0.0000	**0.9974**	0.0000	0.9923	0.0000	**0.9974**	0.0000	**0.9974**	0.0000	**0.9974**	0.0000	**0.9974**
pixraw10P	**0.8524**	0.0387	0.3636	0.0000	0.5035	0.0091	0.5714	0.0000	0.5455	0.0000	0.5064	0.0146	0.6066	0.0188	0.5042	0.0161	0.5455
warpPIE10P	**0.4904**	0.0537	0.1736	0.0060	0.1498	0.0013	0.2388	0.0129	0.1245	0.0028	0.1611	0.0043	0.3211	0.0146	0.2381	0.0079	0.2353
RELATHE	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	0.9930	0.0000	0.9965	0.0000	0.9965	0.0000	**1.0000**	0.0000	0.9895	0.0000	0.9930
Yale	0.7244	0.0886	0.6152	0.0242	0.2000	0.0133	0.5951	0.0450	**0.7508**	0.0329	0.4434	0.0133	0.5013	0.0070	0.3305	0.0176	0.2143
warpAR10P	**0.8779**	0.0562	0.7205	0.0235	0.5042	0.0162	0.5941	0.0365	0.7991	0.0611	0.5529	0.0172	0.4944	0.0170	0.6499	0.0230	0.3636
Leukemia	**1.0000**	0.0000	0.8571	0.0000	0.9286	0.0000	**1.0000**	0.0000	**1.0000**	0.0000	0.8119	0.0350	**1.0000**	0.0000	0.8595	0.0130	0.9286
Rand first	9		1		3		2		1		1		3		2		1
Sum rank	11		52		54		35		49		61		33		48		63
Average rank	1.1		5.2		5.4		3.5		4.9		6.1		3.3		4.8		6.3
Final rank	1		6		7		3		5		8		2		4		9

Table 17. Comparison between BGBO_V3 and state-of-art methods based on the mean number of the selected features (M.NF) and the standard deviation of the mean number of the selected features (as S.D).

Dataset	BGBO_V3		BPSO		BWOA		BDA		BBA		BGOA		BGWO		BHHO	
	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D	Mean	S.D
arcene	**75.5000**	40.7361	4965.8333	47.3964	4583.0667	70.8706	4516.3333	275.6651	4700.6333	67.5862	4924.8000	46.1195	4968.5333	50.3969	3393.0667	348.4594
ORL	**163.6667**	49.8879	506.2667	16.1222	661.6000	19.7809	486.8667	21.2225	469.2667	17.5655	636.3667	45.4870	656.7333	17.7627	537.9231	106.9778
orlraws10P	**72.4333**	68.2403	4959.2333	19.0692	5073.4000	30.8965	4529.3000	505.7059	4747.5333	53.6642	5074.5667	29.0833	5585.3333	600.5217	1691.8667	107.3013
PCMAC	**1.2000**	0.6325	1528.9375	8.5827	1603.9000	18.3216	1041.1000	94.7599	1362.4000	13.6154	1552.5185	8.0640	1603.9333	14.4149	556.5000	55.2816
pixraw10P	**61.2000**	72.6135	4793.8000	18.8833	5817.2333	725.0792	3471.0000	398.1363	4535.0667	25.3540	6013.4333	420.9199	6148.7000	535.0142	2825.2000	230.9463
warpPIE10P	**31.6667**	21.9440	1183.5333	22.6529	1536.4333	92.6490	1059.2000	93.3722	1027.9000	22.1474	1472.0000	83.6120	1404.7000	150.2690	880.1333	108.0991
RELATHE	**2.5000**	1.7171	2040.4333	17.4488	2129.6667	25.7177	1449.7667	125.4453	1852.4000	16.9779	2112.2000	17.4877	2128.1000	31.9378	721.9333	55.4520
Yale	**24.1000**	20.0213	502.1667	12.3095	637.1333	45.6876	465.9000	22.4136	410.7667	11.9241	646.6667	16.8448	632.0667	33.1859	375.8000	104.6683
warpAR10P	**31.0667**	29.0896	1152.1333	29.6377	1417.1667	156.5142	1064.7000	87.4505	1016.9000	26.3705	1423.4333	111.6959	1501.8333	94.6464	857.8660	149.6586
Leukemia	**18.0333**	22.2842	3452.6333	24.9171	3498.6333	26.9885	2560.7667	277.5908	3170.1000	23.1701	3787.0667	418.3822	3524.7667	40.7784	1997.3000	136.6206
Rand first	10		0		0		0		0		0		0		0	
Sum rank	10		50		66		32		36		67		72		22	
Average rank	1		5		6.6		3.2		3.6		6.7		7.2		2.2	
Final rank	1		5		6		3		4		7		8		2	

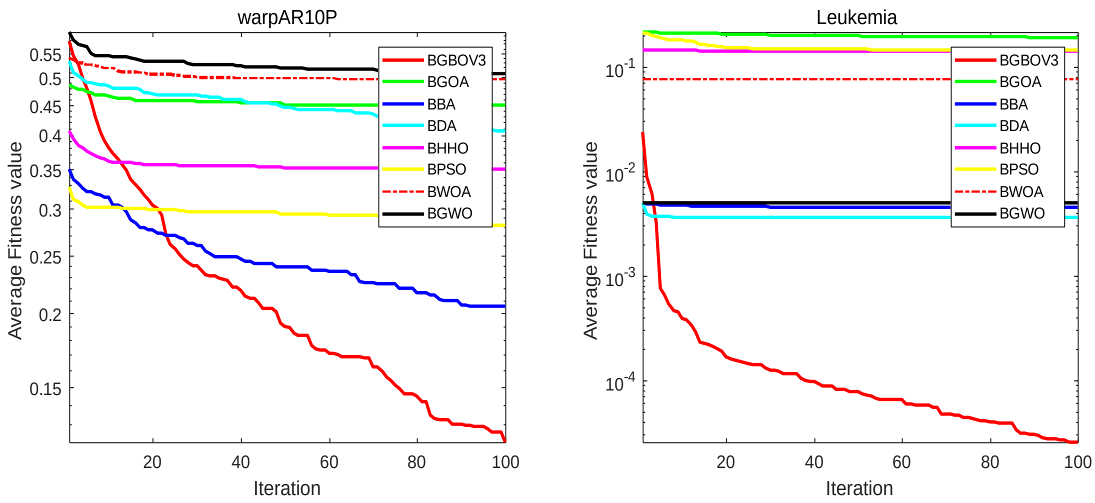

Figure 15. Convergence curves for the warpAR10P and Leukemia dataset.

Table 18. Classification accuracies of the BGBO_V3 versus other metaheuristics from the specialized literature.

Dataset	BGBO_V3	BSSA_S3_CP [93]	WOA-CM [94]	BGOA-M [95]	GA [93]	RBDA [92]	BGSA [92]	bGWO1 [96]
Wine	**1.000**	0.993	0.959	0.989	0.937	0.991	0.951	0.930
IonosphereEW	0.968	0.918	0.919	0.946	0.876	**0.970**	0.881	0.807
Lymphography	**0.973**	0.890	0.807	0.912	0.758	0.930	0.781	0.744
Zoo	**1.000**	**1.000**	0.980	0.958	0.946	**1.000**	0.939	0.879
SonarEW	0.948	0.937	0.852	0.915	0.754	**0.964**	0.888	0.731
Vote	**1.000**	0.951	0.939	0.963	0.808	0.996	0.931	0.912
KrvskpEW	0.960	0.964	0.866	0.974	0.940	**0.975**	0.908	0.944
PenglungEW	**0.998**	0.877	0.972	0.934	0.672	0.959	0.919	0.600
Leukemia	**1.000**	0.989	0.982	-	0.705	-	-	-
Clean1	**0.946**	0.879	-	-	0.862	-	-	
Rank first	7	1	0	0	0	4	0	0
Sum rank	15	33	42	28	61	14	48	60
Average rank	1.50	3.30	4.67	3.50	6.10	1.75	6.00	7.5
Final rank	1	3	5	4	7	2	6	8

From Table 18, we can see that the proposed method in this paper ranks first in seven datasets, including low-dimensional datasets and high-dimensional datasets, respectively, and we can see that BGBO_V3 can achieve excellent results not only in low-dimensional but also in high-dimensional with good performance. Secondly, RBDA has four datasets ranked first, and BSSA_S3_CP has one dataset ranked first. The final ranking of BGBO_V3 is first, and RBDA, BSSA_S3_CP, BGOA-M, WOA-CM, BGSA, GA, and bGWO1 are ranked in order. Thus, the BGBO_V3 proposed in this paper achieves better performance than other methods in solving the FS problem.

In this paper, eight variants of BGBO are proposed using transfer functions. This corresponds to s-shaped and v-shaped eight transfer functions, respectively, for mapping continuous search spatial to discrete search spatial, and in applying to the feature selection problem. The experimental results show that very promising results are achieved compared to the currently popular wrapper-based FS methods. The primary reason is that the algorithm is very reliable and efficient due to the excellent performance of GBO itself, making the algorithm a better choice for the wrapper-based FS method. Moreover, on challenging high-dimensional datasets, the results show clearly that the proposed BGBO has better results in solving high-dimensional problems compared to other methods. Thus, the algorithm has excellent performance and outstanding stability in dealing with challenging problems such as high-dimensional datasets, which shows the potential of the algorithm in solving high-dimensional problems. Among the eight variants proposed in this paper, BGBO_V3 has the best performance. The main reason for the analysis is that different transfer functions have different slopes of curves and different probabilities of changing the position of population individuals. A reasonable probability can balance the exploration and exploitation ability of population individuals, and it is not idealized that the larger the probability of changing the location of the population individuals is better. In addition, an excellent exploration mechanism of the algorithm itself is essential. In the BGBO algorithm based on Newton's method, the GBO algorithm has good two operators (GSR) and (LEO), both of which have their own merit-seeking mechanism, and its perfect mechanism and simple variable parameters balance the exploration and exploitation of the algorithm well, making the individuals better able to make the algorithm proceed in the optimal direction. The transfer function also deserves credit for its own simple and easy-to-understand mechanism. So the combination of the algorithm itself and the transfer function has a clear advantage overall.

6. Conclusions and future works

In this paper, a binary version of the GBO algorithm is proposed to solve the FS problem, and eight variants of the binary gradient optimizer are proposed using transfer functions (i.e., s-shaped and v-shaped). The transfer functions are used to map the continuous search space to the discrete search space and are used in the proposed algorithm. To benchmark the proposed algorithm, 18 standard UCI benchmark datasets are used. The experimental results show that BGBO_V3 has the best performance among the proposed algorithms. The experimental results compare BGBO_V3 with the most popular and better-performing methods in the literature. By analyzing the experimental results, it is demonstrated that the BGBO_V3 algorithm has a high performance among the existing methods for solving FS problems, especially in high-dimensional data sets. Besides, the proposed algorithm has a high convergence speed, which enables the algorithm to find the exact minimum feature set faster. Combining the above experimental results, process discussion and conclusion analysis, it can be concluded that the proposed binary version of the GBO algorithm has advantages in solving FS, and it is worth considering when facing high-dimensional data sets.

In future work, more practical discrete applications combining the method, such as 0-1 backpack problem, TSP problem, scheduling tasks, etc.. The use of other classifiers as evaluators is also a very good research direction, such as extreme learning machines, neural networks and support vector machines, etc.. The performance of the algorithm can be further investigated by comparing different classifiers.

Acknowledgments

This work is supported by National Science Foundation of China under Grants No. 62066005, and Project of Guangxi Natural Science Foundation under Grant No. 2018GXNSFAA138146.

Conflict of interests

The authors declare no conflict of interest.

References

1. M. Chen, S. Mao, Y. Liu, Big data: A Survey, *Mobile Netw. Appl.*, **19** (2014), 171–209.
2. I. Guyon, A. Elisseeff, An introduction of variable and feature selection, *J. Mach. Learn Res.*, **3** (2003), 1157–1182.
3. Y. Wan, M. Wang, Z. Ye, X. Lai, A feature selection method based on modified binary coded ant colony optimization algorithm, *Appl. Soft Comput.*, **49** (2016), 248–258.
4. H. Liu, H. Motoda, *Feature selection for knowledge discovery and data mining*, Kluwer Academic, 2012.
5. Z. Sun, G. Bebis, R. Miller, Object detection using feature subset selection, *Pattern Recogn.,* **37** (2004), 2165–2176.
6. H. Liu, H. Motoda, *Feature Extraction, Construction and Selection: A Data Mining Perspective* Springer Science & Business Media, Boston, MA, 1998.
7. Z. Zheng, X. Wu, R. K. Srihari, Feature selection for text categorization on imbalanced data, *ACM Sigkdd Explor. Newsl.*, **6** (2004), 80–89.
8. H. Uguz, A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm, *Knowl.-Based Syst.*, **24** (2011), 1024–1032.
9. H. K. Ekenel, B. Sankur, Feature selection in the independent component subspace for face recognition, *Pattern Recogn. Lett.*, **25** (2004), 1377–1388.
10. H. R. Kanan, K. Faez, An improved feature selection method based on ant colony optimization (ACO) evaluated on face recognition system, *Appl. Math. Comput.*, **205** (2008), 716–725.
11. F. Model, P. Adorjan, A. Olek, C. Piepenbrock, Feature selection for DNA methylation based cancer classification, *Bioinformatics*, **17** (2001), S157–S164.
12. N. Chuzhanova, A. J. Jones, S. Margetts, Feature selection for genetic sequence classification, *Bioinformatics*, **14** (1998), 139–143.
13. S. Tabakhi, A. Najafi, R. Ranjbar, P. Moradi, Gene selection for microarray data classification using a novel ant colony optimization, *Neurocomputing*, **168** (2015), 1024–1036.
14. D. Liang, C. F. Tsai, H. T. Wu, The effect of feature selection on financial distress prediction, *Knowl.-Based Syst.*, **73** (2015), 289–297.
15. M. Ramezani, P. Moradi, F. A. Tab, Improve performance of collaborative filtering systems using backward feature selection, in *The 5th Conference on Information and Knowledge Technology*, (2013), 225–230.
16. B. Tseng, Tzu Liang Bill, C. C. Huang, Rough set-based approach to feature selection in customer relationship management, *Omega*, **35** (2007), 365–383.

17. R. Sawhney, P. Mathur, R. Shankar, A firefly algorithm based wrapper-penalty feature selection method for cancer diagnosis, in *International Conference on Computational Science and Its Applications*, Springer, Cham, (2018), 438–449.
18. B. Guo, R. I. Damper, S. R. Gunn, J. D. B. Nelson, A fast separability-based feature-selection method for high-dimensional remotely sensed image classification, *Pattern Recogn.*, **41** (2008), 1653–1662.
19. R. Abraham, J. B. Simha, S. S. Iyengar, Medical datamining with a new algorithm for feature selection and naive bayesian classifier, in *10th International Conference on Information Technology (ICIT 2007)*, IEEE, 2007, 44–49.
20. L. Yu, H. Liu, Feature selection for high-dimensional data: A fast correlation-based filter solution, in *Proceedings of the 20th international conference on machine learning (ICML-03)*, (2003), 856–863.
21. L. Cosmin, J. Taminau, S. Meganck, D. Steenhoff, A survey on filter techniques for feature selection in gene expression microarray analysis, *IEEE/ACM Trans. Comput. Biol. Bioinf.*, **9** (2012), 1106–1119.
22. S. Maldonado, R. Weber, A wrapper method for feature selection using support vector machines, *Inform. Sci.*, **179** (2009), 2208–2217.
23. J. Huang, Y. Cai, X. Xu, A hybrid genetic algorithm for feature selection wrapper based on mutual information, *Pattern Recogn. Lett.*, **28** (2007), 1825–1844.
24. C. Tang, X. Liu, X. Zhu, J. Xiong, M. Li, J. Xia, et al., Feature selective projection with low-rank embedding and dual laplacian regularization, *IEEE Trans. Knowl. Data. Eng.*, **32** (2019), 1747–1760.
25. C. Tang, M. Bian, X. Liu, M. Li, H. Zhou, P. Wang, et al., Unsupervised feature selection via latent representation learning and manifold regularization, *Neural Networks*, **117** (2019), 163–178.
26. S. Sharifzadeh, L. Clemmensen, C. Borggaard, S. Støier, B. K. Ersbøll, Supervised feature selection for linear and non-linear regression of L*a*b color from multispectral images of meat, *Eng. Appl. Artif. Intel.*, **27** (2013), 211–227.
27. C. Tang, X. Zheng, X. Liu, L. Wang, Cross-view locality preserved diversity and consensus learning for multi-view unsupervised feature selection, *IEEE Trans. Knowl. Data. Eng.*, 2021.
28. C. Tang, X. Zhu, X. Liu, L. Wang, Cross-View local structure preserved diversity and consensus learning for multi-view unsupervised feature selection, in Proceedings of the AAAI Conference on Artificial Intelligence, **33** (2019), 5101–5108.
29. M. Dorigo, *Optimization, Learning and Natural Algorithms*, Phd Thesis Politecnico Di Milano, 1992.
30. C. Lai, M. J. T. Reinders, L. Wessels, Random subspace method for multivariate feature selection, *Pattern Recogn. Lett.*, **27** (2006), 1067–1076.
31. B. Xue, M. Zhang, W. N. Browne, X. Yao, A survey on evolutionary computation approaches to feature selection, *IEEE Trans. Evol. Comput.*, **20** (2016), 606–626.
32. J. J. Grefenstette, Optimization of control parameters for genetic algorithms, *IEEE Trans. Syst. Man Cybern.*, **16** (1986), 122–128.
33. B. G. Obaiahnahatti, J. Kennedy, A new optimizer using particle swarm theory, in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, IEEE, (1995), 39–43.

34. K. Chen, F. Y. Zhou, X. F. Yuan, Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection, *Expert Syst. Appl.*, **128** (2019), 140–156.
35. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Softw.*, **95** (2016), 51–67.
36. S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Softw*., **69** (2014), 46–61.
37. S. Shahrzad, S. Mirjalili, A. Lewis, Grasshopper optimisation algorithm: Theory and application, *Adv. Eng. Softw*., **105** (2017), 30–47.
38. E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, GSA: a gravitational search algorithm, *Inform. Sci.*, **179** (2009), 2232–2248.
39. S. Li, H. Chen, M. Wang, A. A. Heidari, S. Mirjalili, Slime mould algorithm: A new method for stochastic optimization, *Future Gener. Comput. Syst.*, **111** (2020), 300–323.
40. Y. Yang, H. Chen, A. A. Heidari, A. H. Gandomi, Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts, *Expert Syst. Appl.*, **171** (2021), 114864.
41. I. Ahmadianfar, O. Bozorg-Haddad, X. Chu, Gradient-based optimizer: A new metaheuristic optimization algorithm, *Inform. Sci.*, **540** (2020), 131–159.
42. S. Mirjalili, Dragonfly algorithm: a new meta-heuristic optimization technique for solving single–objective, discrete, and multi-objective problems, *Neural Comput. Appl.*, **27** (2016), 1053–1073.
43. T. J. Ypma, Historical development of the Newton-Raphson method, *SIAM Rev.*, **37** (1995), 531–551.
44. S. Mirjalili, A. Lewis, S-shaped versus V-shaped transfer functions for binary particle swarm optimization, *Swarm Evol. Comput.*, **9** (2013), 1–14.
45. H. Liu, J. Li, L. Wong, A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns, *Genome Inform.*, **13** (2002), 51–60.
46. M. Dash, H. Liu, Feature selection for classification, *Intell. Data Anal.*, **1** (1997), 131–156.
47. W. Siedlecki, J. Sklansky, A note on genetic algorithms for large-scale feature selection, *Pattern Recogn. Lett.*, **10** (1989), 335–347.
48. R. Leardi, R. Boggia, M. Terrile, Genetic algorithms as a strategy for feature selection, *J. Chemom.*, **6** (1992), 267–281.
49. I. S. Oh, J. S. Lee, B. R. Moon, Hybrid genetic algorithms for feature selection, *IEEE Trans. Pattern Anal.*, **26** (2004), 1424–1437.
50. R. Storn, K. Price, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optim.*, **11** (1997), 341–359.
51. B. Xue, W. Fu, M. Zhang, Differential evolution (DE) for multi-objective feature selection in classification, in *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*, (2014), 83–84.
52. D. Karaboga, B. Akay, A comparative study of artificial bee colony algorithm, *Appl. Math. Comput.*, **214** (2009), 108–132.
53. A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, *Future Gener. Comput. Syst.*, **97** (2019), 849–872.
54. A. Faramarzi, M. Heidarinejad, S. Mirjalili, A. H. Gandomi, Marine predators algorithm: A nature-inspired metaheuristic, *Expert Syst. Appl.*, **152** (2020), 113377.
55. O. S. Qasim, Z. Algamal, Feature selection using particle swarm optimization-based logistic regression model, *Chemom. Intell. Lab. Syst.*, **182** (2018), 41–46.

56. K. Chen, F. Y. Zhou, X. F. Yuan, Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection, *Expert Syst. Appl.*, **128** (2019), 140–156.
57. B. Xue, M. Zhang, W. N. Browne, Particle swarm optimization for feature selection in classification: a multi-objective approach, *IEEE Trans. Cybern.*, **43** (2013), 1656–1671.
58. E. Emary, H. M. Zawbaa, A. E. Hassanien, Binary grey wolf optimization approaches for feature selection, *Neurocomputing*, **172** (2016), 371–381.
59. P. Hu, J. S. Pan, S. C. Chu, Improved binary grey wolf optimizer and its application for feature selection, *Knowl.-Based Syst.*, **195** (2020), 105746.
60. T. Qiang, X. Chen, X. Liu, Multi-strategy ensemble grey wolf optimizer and its application to feature selection, *Appl. Soft Comput.*, **76** (2019), 16–30.
61. M. M. Mafarja, S. Mirjalili, Whale optimization approaches for wrapper feature selection, *Appl. Soft Comput.*, **62** (2018), 441–453.
62. M. M. Mafarja, S. Mirjalili, Hybrid whale optimization algorithm with simulated annealing for feature selection, *Neurocomputing*, **260** (2017), 302–312.
63. R. K. Agrawal, B. Kaur, S. Sharma, Quantum based whale optimization algorithm for wrapper feature selection, *Appl. Soft Comput.*, **89** (2020), 106092.
64. C. R. Hwang, Simulated annealing: Theory and applications, *Acta. Appl. Math.*, **12** (1988), 108–111.
65. H. Shareef, A. A. Ibrahim, A. H. Mutlag, Lightning search algorithm, *Appl. Soft Comput.*, **36** (2015), 315–333.
66. S. Mirjalili, S. M. Mirjalili, A. Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization, *Neural Comput. Appl.*, **27** (2016), 495–513.
67. H. Abedinpourshotorban, S. M. Shamsuddin, Z. Beheshti, D. N. A. Jawawib, Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm, *Swarm Evol. Comput.*, **26** (2016), 8–22.
68. A. Y. S. Lam, V. O. K. Li, Chemical-reaction-inspired metaheuristic for optimization, *IEEE Trans. Evol. Comput.*, **14** (2009), 381–399.
69. F. A. Hashim, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, S. Mirjalili, Henry gas solubility optimization: A novel physics-based algorithm, *Future Gener. Comp. Syst.*, **101** (2019), 646–667.
70. R. Meiri, J. Zahavi, Using simulated annealing to optimize the feature selection problem in marketing applications, *Eur. J. Oper. Res.*, **171** (2006), 842–858.
71. S. W. Lin, Z. J. Lee, S. C. Chen, T. Y. Tseng, Parameter determination of support vector machine and feature selection using simulated annealing approach, *Appl. Soft Comput.*, **8** (2008), 1505–1512.
72. E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, BGSA: Binary gravitational search algorithm, *Nat. Comput.*, **9** (2010), 727–745.
73. S. Nagpal, S. Arora, S. Dey, Feature selection using gravitational search algorithm for biomedical data, *Procedia Comput. Sci.*, **115** (2017), 258–265.
74. P. C. S. Rao, A. J. S. Kumar, Q. Niyaz, P. Sidike, V. K. Devabhaktuni, Binary chemical reaction optimization based feature selection techniques for machine learning classification problems, *Expert Syst. Appl.*, **167** (2021), 114169.
75. N. Neggaz, E. H. Houssein, K. Hussain, An efficient henry gas solubility optimization for feature selection, *Expert Syst. Appl.*, **152** (2020), 113364.
76. R. V. Rao, V. J. Savsani, D. P. Vakharia, Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. *Comput. Aided Design*, **43** (2011), 303–315.

77. S. Hosseini, A. A. Khaled, A survey on the imperialist competitive algorithm metaheuristic: implementation in engineering domain and directions for future research, *Appl. Soft Comput.*, **24** (2014), 1078–1094.
78. R. Moghdani, K. Salimifard, Volleyball premier league algorithm, *Appl. Soft Comput.*, **64** (2017), 161–185.
79. H. C. Kuo, C. H. Lin, Cultural evolution algorithm for global optimizations and its applications, *J. Appl. Res. Technol.*, **11** (2013), 510–522.
80. M. Allam, M. Nandhini, Optimal feature selection using binary teaching learning based optimization algorithm, *J. King Saud Univ.-Comput. Inform. Sci.*, **10** (2018).
81. S. J. Mousavirad, H. Ebrahimpour-Komleh, Feature selection using modified imperialist competitive algorithm, in *ICCKE 2013*, IEEE, (2013), 400–405.
82. A. Keramati, M. Hosseini, M. Darzi, A. A. Liaei, Cultural algorithm for feature selection, in *The 3rd International Conference on Data Mining and Intelligent Information Technology Applications*, IEEE, (2011), 71–76.
83. D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, *IEEE Trans. Evol. Comput.*, **1** (1997), 67–82.
84. A. Fink, S. Vo, Solving the continuous flow-shop scheduling problem by metaheuristics, *Eur. J. Oper. Res.*, **151** (2003), 400–414.
85. J. Kennedy, R. C. Eberhart, A discrete binary version of the particle swarm algorithm, in *1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation*, IEEE, **5** (1997), 4104–4108.
86. E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, BGSA: binary gravitational search algorithm, *Nat. Comput.*, **9** (2010), 727–745.
87. N. S. Altman, An introduction to kernel and nearest-neighbor nonparametric regression, *Am. Stat.*, **46** (1992), 175–185.
88. F. Pernkopf, Bayesian network classifiers versus selective k-NN classifier, *Pattern Recogn.*, **38** (2005), 1–10.
89. A. Asuncion, D. Newman, *UCI Machine Learning Repository*, University of California, 2007.
90. E. Emary, H. M. Zawbaa, A. E. Hassanien, Binary ant lion approaches for feature selection, *Neurocomputing*, **213** (2016), 54–65.
91. Arizona State University's (ASU) repository, Available from: http://featureselection.asu.edu/datasets.php.
92. A. I. Hammouri, M. Mafarja, M. A. Al-Betar, M. A. Awadallah, I. Abu-Doush, An improved Dragonfly Algorithm for feature selection, *Knowl.-Based Syst.*, **203** (2020), 106131.
93. H. Faris, M. M. Mafarja, A. A. Heidari, I. Aljarah, A. M. Al-Zoubi, S. Mirjalili, et al., An Efficient Binary Salp Swarm Algorithm with Crossover Scheme for Feature Selection Problems, *Knowl.-Based Syst.*, **154** (2018), 43–67.
94. M. Mafarja, S.Mirjalili, Whale optimization approaches for wrapper feature selection, *Appl. Soft Comput.*, **62** (2018), 441–453.
95. M. Mafarja, I. Aljarah, H. Faris, A. I. Hammouri, A. M. Al-Zoubi, S. Mirjalili, Binary grasshopper optimisation algorithm approaches for feature selection problems, *Expert Syst. Appl.*, **117** (2019), 267–286.

96. E. Emary, H. M. Zawbaa, A. E. Hassanien, Binary grey wolf optimization approaches for feature selection, *Neurocomputing*, **172** (2016), 371–381.