

# A Self-adaptive Binary Cat Swarm Optimization Using New Time-Varying Transfer Function for Gene Selection in DNA Microarray Expression Cancer Data

yousef sharafi

Islamic Azad University

Mohammad Teshnehlab (✉ [Teshnehlab@eetd.kntu.ac.ir](mailto:Teshnehlab@eetd.kntu.ac.ir))

KN Toosi: KN Toosi University of Technology

---

## Research Article

**Keywords:** Self-adaptive parameters, Microarray expression cancer data, Gene selection, Binary cat swarm optimization algorithm, Multi-objective optimization.

**Posted Date:** March 7th, 2022

**DOI:** <https://doi.org/10.21203/rs.3.rs-1010398/v1>

**License:**   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

---

# A Self-adaptive Binary Cat Swarm Optimization Using New Time-Varying Transfer Function for Gene Selection in DNA Microarray Expression Cancer Data

**Yousef Sharafi**

Department of Computer,  
Science and Research Branch,  
Islamic Azad university,  
Tehran, Iran  
Email: y.sharafi@srbiau.ac.ir

**Mohammad Teshnehlab\*<sup>1</sup>**

Industrial Control Center of  
Excellence, Electrical Engineering  
Department, K.N.Toosi University,  
Tehran, Iran  
Email: teshnehlab@eetd.kntu.ac.ir

## Abstract

Microarray technology is beneficial in terms of diagnosing various diseases, including cancer. Despite all DNA microarray benefits, the high number of genes versus the low number of samples has always been a crucial challenge for this technology. Accordingly, we need new optimization algorithms to select optimal genes for faster disease diagnosis. In this article, a new version of the binary cat optimization algorithm, named SBCSO, for gene selection in DNA microarray expression cancer data is presented. The main contributions in this paper are listed as follows: First, the opposition-based learning (OBL) mechanism is employed to improve the proposed algorithm's population members' diversity. Second, a time-varying V-shape transfer function is employed to balance the two phases of exploration and extraction in the proposed algorithm. Third, the MR and  $\lambda$  parameters in the proposed algorithm are adapted over time, and finally, single-objective and multi-objective approaches are proposed to solve the gene selection problems. The fifteen datasets pertinent to microarray data of various cancer types are employed to compare the proposed method with other well-known binary optimization algorithms. The experiments' results indicate that the proposed algorithm has a better capability to select the optimal genes for a faster disease diagnosis.

**Keywords:** Self-adaptive parameters, Microarray expression cancer data, Gene selection, Binary cat swarm optimization algorithm, Multi-objective optimization.

## 1. Introduction

Microarray data are extremely useful in terms of diagnosis and prediction of various diseases such as cancer. These data are obtained based on individual's DNA and their genetic information. DNA microarray technology simultaneously investigates the expression of thousands of genes, so it is a significant breakthrough in diagnosing and treating various diseases, particularly cancer. One of the ever-present, significant challenges in microarray data analysis is the selection of optimal genes. The high number of genes, in addition to the low number of samples in microarray data, will bring about some problems in terms of data analysis. Also, in some cases, the high proportion of genes to sample numbers increases the possibility of selecting an unsuitable gene. Nowadays, data analysis could be complicated without statistical analysis and intelligent algorithms [3,11].

---

\* Corresponding Author.

The traditional methods take the vast majority of genes into account as effective genes in diseases [4]. Investigations indicate that most DNA microarray gene expression cancer data are not of informational value based on classification criteria [5]. Thus, feature (gene) selection and diagnosis and elimination process of irrelevant genes play pivotal roles in the analysis of microarray data obtained from DNA. Hence, researchers are in the pursuit of introducing novel algorithms of feature selection [1,2].

During the last decades, various algorithms are introduced to solve the feature (gene) selection problem, including filter-based, wrapper-based, hybrid, and embedded algorithms, among which the filter-based and wrapper-based methods are more prevalent [6,16]. Wrapper-based methods take advantage of machine learning algorithms as a cost function to evaluate a subset of features. Some of the wrapper-based algorithms introduced in recent decades for selecting a subset of features in various applications include ISSA [7], BWSSO [8], QWOA [9], and TLBOSA [10]. Besides, the filter-based algorithms make use of statistical methods to select a subset of features [12]. The hybrid method employs the two wrapper and filter methods to improve the feature selection algorithm's effectiveness. In this regard, we can name WFACOFs [13], IG-GA [14], and BDE-xRank [15] algorithms.

When comparing the wrapper-based and filter-based methods, it is indicated that wrapper-based algorithms have better results. However, the major drawback of these methods is numerous calculations. Accordingly, researchers have always pursued novel methods of evolutionary optimization algorithms to accelerate the selection process of a subset of optimal features by reducing the computational load [6].

In recent decades, different evolutionary optimization methods inspired by nature have been introduced, among which gray wolf optimizer (GWO) [17], particle swarm optimization (PSO) [18], and cat swarm optimization (CSO) [19] algorithms are included. The structure of all evolutionary algorithms comprises an initial population that is evolved in an evolutionary approach. Suitable diversity in the members of an initial population can play a pivotal part in population convergence towards optimal global solutions. Suppose the diversity of population members is not appropriate. In that case, the two exploitation and exploration phases will not be balanced well, and the evolutionary algorithms can get trapped in a local minimum. Various investigations have been conducted to improve the diversity of population members and the effectiveness of evolutionary algorithms. The opposition-based learning (OBL) method is one of the most that has recently captured a variety of researchers' interests. OBL is a new concept in machine learning, which is inspired by the contrasting relationships between entities [20]. This investigation employs the OBL mechanism to improve the diversity of initial population members and the middle generation [50-52].

The cat swarm optimization (CSO) algorithm is presented based on cats' group behavior in two phases of tracing and seeking [19]. In the CSO algorithm, the percentage of cats' involvement in both phases is determined based on the mixture rate (MR) parameter's extent. In essence, the CSO is a continuous algorithm, and the feature selection is an optimization problem, the variables' spaces of which are binary. In evolutionary algorithms, some transfer functions, including linear, S-shape, and V-shape functions, are usually employed to convert the continuous space to binary space. Optimization algorithms employing S-shape functions include BPSO [21], BGWO [22] and BCSO [23]. Also, the BGSA [24], BBA [25], BDA [26] algorithms take advantage of V-shape functions for various applications. Recently, a new type of transfer function is introduced, the form of which varies by time. In the article [23], the PSO binary optimization algorithm is introduced based on time-varying transfer functions to strike a balance between exploitation and exploration.

Also, in dragonfly optimization algorithm, the time-varying transfer functions are employed in feature selection [24]. In the article [25], a time-varying mirrored S-shaped transfer function is used for BPSO algorithms to solve feature selection problems. The binary version of the binary whale optimization algorithm (BWOA) algorithm is also provided based on a time-varying transfer function to solve feature selection problems [26]. In this paper, a new time-varying V-shape transfer function is employed to convert the continuous space to the binary space in the CSO algorithm. The proposed transfer function has an adaptive parameter striking a balance between two exploration and exploitation processes in the proposed algorithm.

In the feature selection problem, the F1 and F2, which are inversely related functions, naturally convert this problem into a multi-objective one. The F1 and F2 functions refer to the number of features selected and classification error rate, respectively. In this research, single-objective and multi-objective approaches are proposed to solve the feature selection problem. The proposed algorithm is applied to 15 benchmark datasets to optimize the feature selection. The experimental results indicate that the proposed binary algorithm has reported better performance in comparison to other well-known binary optimization algorithms.

The contributions of this investigation are summarized as follows:

- The improved CSO binary algorithm is presented.
- An initial combined population with a suitable diversity is presented to solve binary problems based on OBL and uniform distribution. The OBL mechanism is utilized in the evolution of population members of the middle generation.
- A new time-varying V-shape transfer functions are employed to balance two exploration and exploitation phases.
- Single-objective and multi-objective approaches are provided to solve the gene selection problem.
- The MR and  $\lambda$  parameters in the proposed algorithm are adapted over time.
- The proposed algorithm results are investigated on fifteen datasets pertinent to microarray data of various cancers, and the outcomes are reported.

The structure of this article is as follows: A summary of the CSO algorithm and a review of OBL and TF are presented in section 2. In section 3, the proposed SB-CSO algorithm is presented. In section 4, the proposed SB-CSO algorithm's test results compared to other well-known binary optimization algorithms are reported. Finally, a conclusion is presented.

## 2. Background

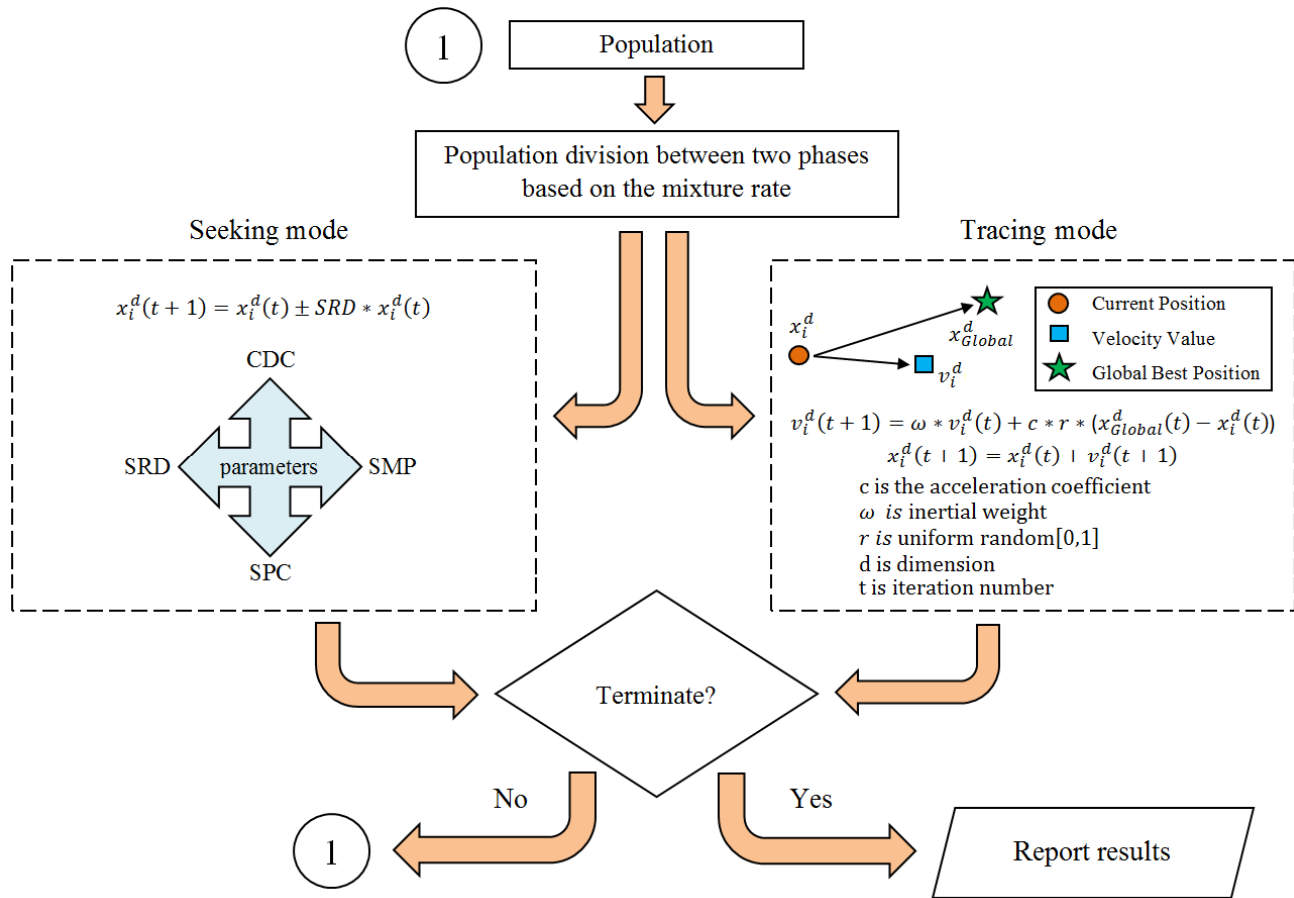
In this section, the fundamental concepts of cat swarm optimization algorithm, OBL, and a review of the transfer functions are presented.

### 2.1 The standard cat swarm optimization algorithm

The cat swarm optimization algorithm is an algorithm inspired by the group behavior of cats in nature. Cats usually conduct two behaviors in nature. First, they rest most of the time, watching their surroundings intelligently. Second, they are aware of everything happening around them, and as soon as they see a target, they move quickly towards it. These two behaviors are considered to be the principle of the design and implementation of the CSO algorithm [27].

In the CSO algorithm, these two behaviors are stimulated in two phases of seeking and tracing. In the seeking phase, cats observe their surroundings, and in the tracing phase, they move

towards a suitable target. In the CSO algorithm, the number of cats falling under the tracing phase category is determined based on the mixture rate (MR) parameter, and the rest of the cats fall into the seeking phase category. Figure 1 demonstrates the general process of the CSO algorithm.



**Fig. 1** The overall procedure of the CSO algorithm.

The four parameters of the seeking phase in the CSO algorithm are defined as follows:

- Counts of Dimension to Change (**CDC**)
- Self-Position Consideration (**SPC**)
- Seeking Range of the selected Dimension (**SRD**)
- Seeking Memory Pool (**SMP**)

Several copies of every cat in the seeking phase will be created (the copy number equal to SMP parameter), and each copy will change separately based on SRD and CDC parameters (similar to the mutation operator in a genetic algorithm). The value of the SPC parameter is either TRUE or FALSE. In case it is TRUE, the present cat competes with other modified copies, as well. Finally, the best solution in terms of fitness is transferred to the next algorithm generation. Any cat in the tracing phase will move to the position of the cat in the best position [27].

Researchers have introduced several improved versions of the CSO algorithm, so far. Particularly in the article [29], an improved version of this algorithm is presented based on the normal mutation to achieve a faster convergence. The article [28] has also improved the global and local search of this algorithm by modifying some of the equations.

## 2.2 The binary cat swarm optimization algorithm

In the research literature, some binary versions of the cat optimization algorithm are presented. For the first time in 2013, a binary version of the cat algorithm addressing the zero-one knapsack problem was introduced [37]. In this algorithm, a simple sigmoid function was introduced to convert the continuous space into binary space. In the article [38], one binary version of the cat algorithm based on the V-shape transfer function is presented to solve the one-zero knapsack problem. In the article [39], the updating equations of a cat position in the tracing phase changed, and the zero-one knapsack problem is solved based on it. The pseudo-code of the original binary cat swarm algorithm (BCSO) is indicated in Algorithm 1.

---

**Algorithm 1:** Binary Cat Swarm Optimization Algorithm (BCSO)

---

1. **Initialization:** popSize, Max<sub>NFE</sub>, MR, nfe=0, etc.
  2. **Initial population.**
  3. **While** (nfe < Max<sub>NFE</sub>) **do**
  4.     Division of population members between two phases based on the MR.
  5.     **For** i=1 to popSize
  6.         **If** (cat[i].flag == 1) **then**
  7.             //Seeking phase (update  $x(i)$ ).
  8.         **Else**
  9.             //Tracing phase (update  $x(i)$ ).
  10.         **End if**
  11.     **End for**
  12.     Convert continuous to binary space.
  13.     Evaluate population (cost function value).
  14.     **If** (termination condition) **then**
  15.         Go to step 21.
  16.     **Else**
  17.         Go to step 3.
  18.     **End if**
  19.     Update nfe.
  20. **End while**
  21. **Output:** The report of the result.
- 

## 2.3 Opposition-based learning (OBL)

In the evolutionary algorithms, the population members' diversity plays a pivotal part in preventing the early premature convergence and also getting stuck in a local optimum. Accordingly, various methods, including OBL and chaotic maps, are presented to increase the population members' diversity [20, 30, 31]. The concept of OBL is demonstrated in the Figure below (also see equation 1).



**Fig. 2** Opposition-based learning

$$\bar{x} = upper + lower - x \quad (1)$$

The *upper* and *lower* values are the ceiling and the floor of the search space, respectively. Given that the search space is binary in the feature selection problems, the opposite of a solution is calculated employing the following equation.

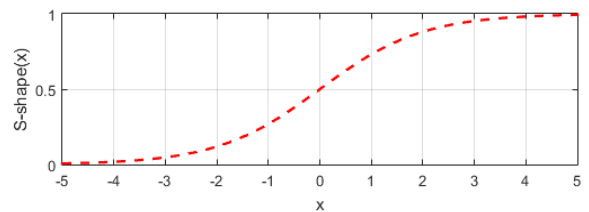
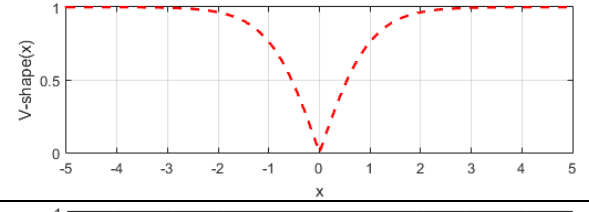
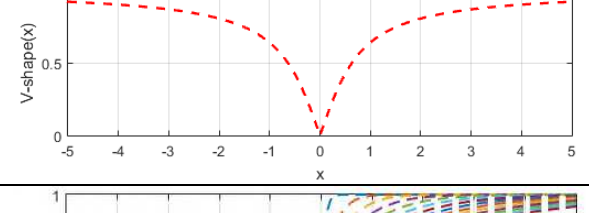
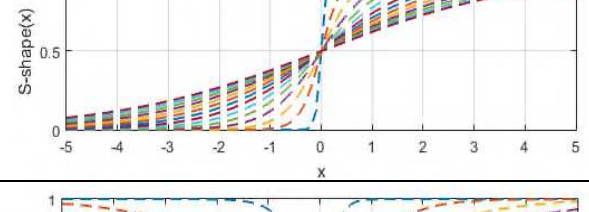
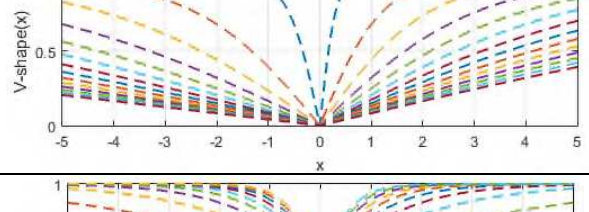
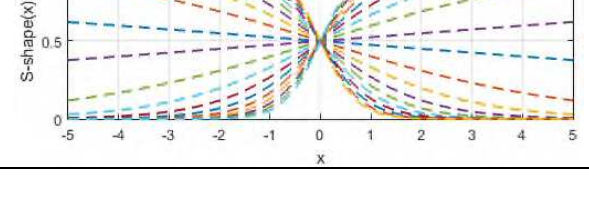
$$\bar{x} = 1 - x \quad (2)$$

In recent decades, researchers have taken advantage of various OBL mechanisms in some evolutionary optimization algorithms. In this regard, we can name Quasi-Opposition, Super-Opposition, Quasi-Reflection OBL, and Generalized OBL methods [32,33]. In this paper, the OBL mechanism is utilized to improve the proposed algorithm's population members' diversity.

## 2.4 A brief overview of existing transfer functions

Various transfer functions have been employed to convert the continuous space to the binary space in evolutionary optimization algorithms to solve feature selection problems. Among the well-known transfer functions capturing researchers' interests in recent years are V-shape and S-shape transfer functions. It is worth mentioning that the shape of these transfer functions can vary by the time during the optimization process. Accordingly, the shape of these transfer functions can be either static or time-varying. In Table 1, a review of some transfer functions is presented.

**Table 1** Some common transfer functions.

Name	Shape Type	Time-varying	Transfer function	Shape
S1[34]	S-shape	NO	$\frac{1}{1+e^{-x}}$	
V1[35]	V-shape	NO	$ \tanh(x) $	
V2[36]	V-shape	NO	$\left  \frac{2}{\pi} \arctan\left(\frac{\pi}{2}x\right) \right $	
S2[24]	S-shape	YES	$\frac{1}{1+\exp(-\frac{2}{\alpha}x)}$ , $0.01 \leq \alpha \leq 3$	
V3[24]	V-shape	YES	$\begin{cases} \frac{2}{1+\exp(-\frac{x}{3\alpha})} - 1 & x > 0 \\ 1 - \frac{2}{1+\exp(-\frac{x}{3\alpha})} & x \leq 0 \end{cases}$ , $0.01 \leq \alpha \leq 4$	
S3[25]	S-shape	YES	$\frac{1}{1+\exp(\sigma x)}$ / $\frac{1}{1+\exp(-\sigma x)}$ , $\sigma = (\sigma_{\max} - \sigma_{\min})\left(\frac{\text{iter}}{\text{maxiter}}\right) + \sigma_{\min}$	

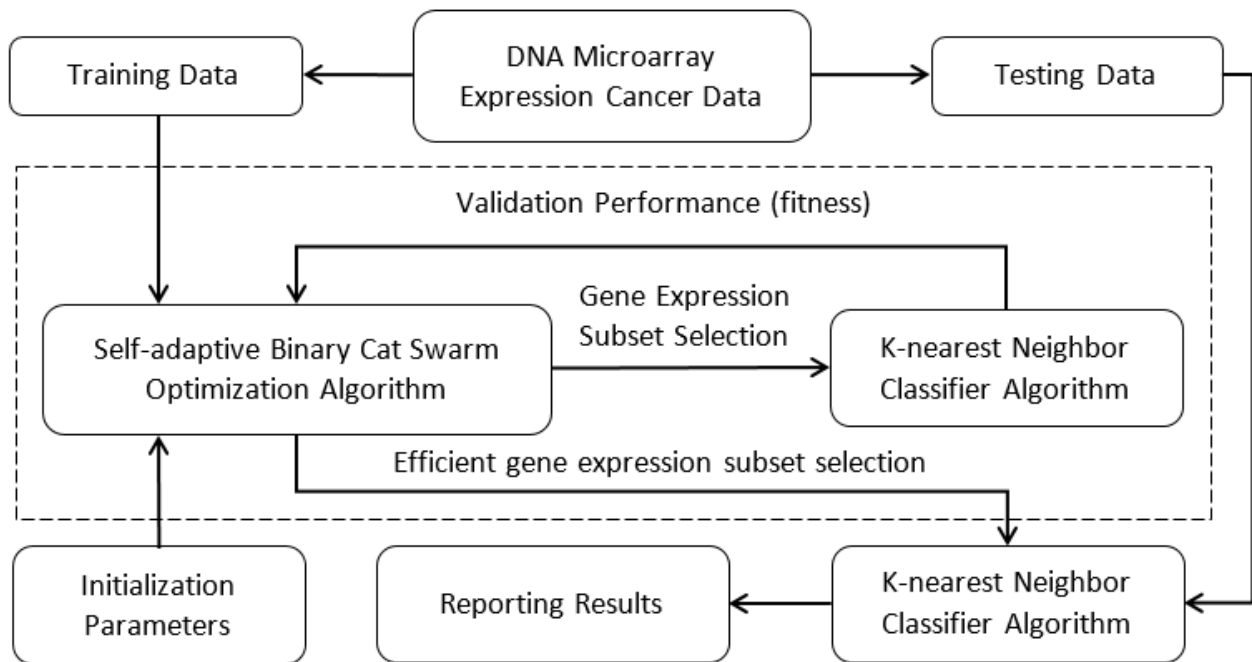
### 3. The proposed self-adaptive binary cat swarm optimization algorithm

This section presents the binary version of the self-adaptive cat swarm optimization algorithm for the gene selection in DNA Microarray Expression Cancer data. In the proposed binary algorithm named SBCSO, the OBL mechanism is utilized to increase the population members' diversity. Enhancing the population members' diversity contributes highly to preventing the evolutionary optimization algorithms from getting stuck in a local minimum. In subsection 3.1, the method of using the OBL mechanism in the proposed algorithm is presented. The feature (gene) selection is a binary optimization problem. Therefore, the transfer functions are required to convert an optimization algorithm in continuous space to binary space. In subsection 3.2, a new time-varying



V-shape transfer function containing a time-varying parameter is presented. The role of this parameter is to balance two phases of exploration and exploitation. The cat swarm optimization algorithm comprises two phases of seeking and tracing, in which the number of cats in each phase is determined based on the mixture rate (MR) parameter. In subsection 3.3, an automatic mechanism for adapting the proposed algorithm's parameters, including the MR and  $\lambda$  parameters, is presented ( $\lambda$  is a controlling parameter balancing the exploration and the exploitation phases). In subsection 3.4, the cost function value for solving the feature selection problem is defined in the form of single and multi-objective functions. In subsection 3.5, the final conditions of the proposed algorithm are presented.

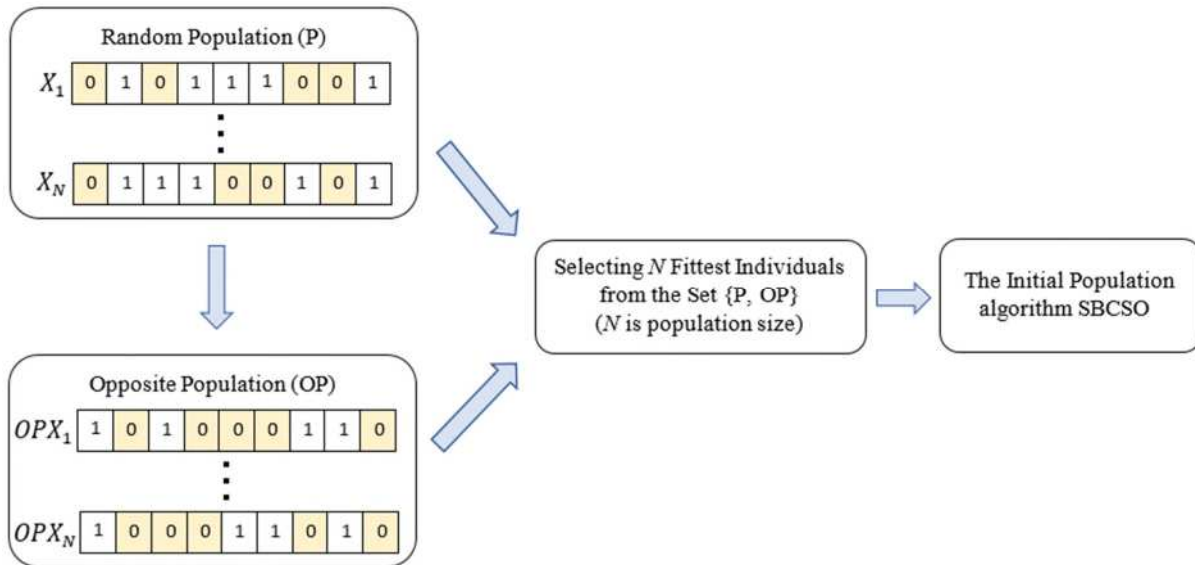
In order to obtain the best subset of features, the K-nearest neighbor (KNN) algorithm with  $K = 5$  is employed to determine the performance of each feature's subset for classification. The block diagram of the SBCSO algorithm for selecting the optimal feature subset of a dataset is demonstrated in Fig. 3. The details of the proposed algorithms are presented below.



**Fig. 3** Block diagram of self-adaptive binary cat swarm optimization algorithm.

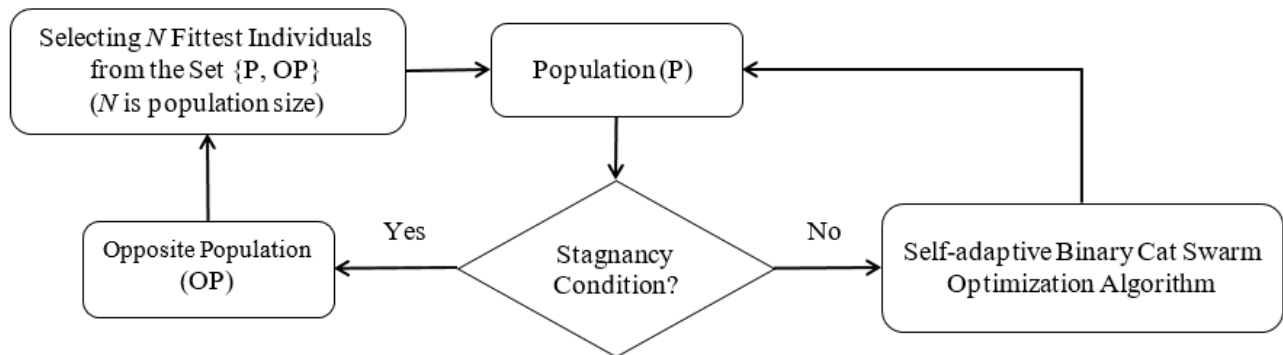
### 3.1 Initial population strategy

The OBL mechanism is employed in two parts of the proposed algorithm to improve the population members' diversity. First, a population titled the subset  $\{p\}$  with a uniform distribution is initially created in the first part, as indicated in Fig. 4. In the next step, a population titled  $\{OP\}$  is created using the OBL mechanism.  $N$  better solutions for the initial population are selected from  $\{P, OP\}$  set to calculate their cost functions.



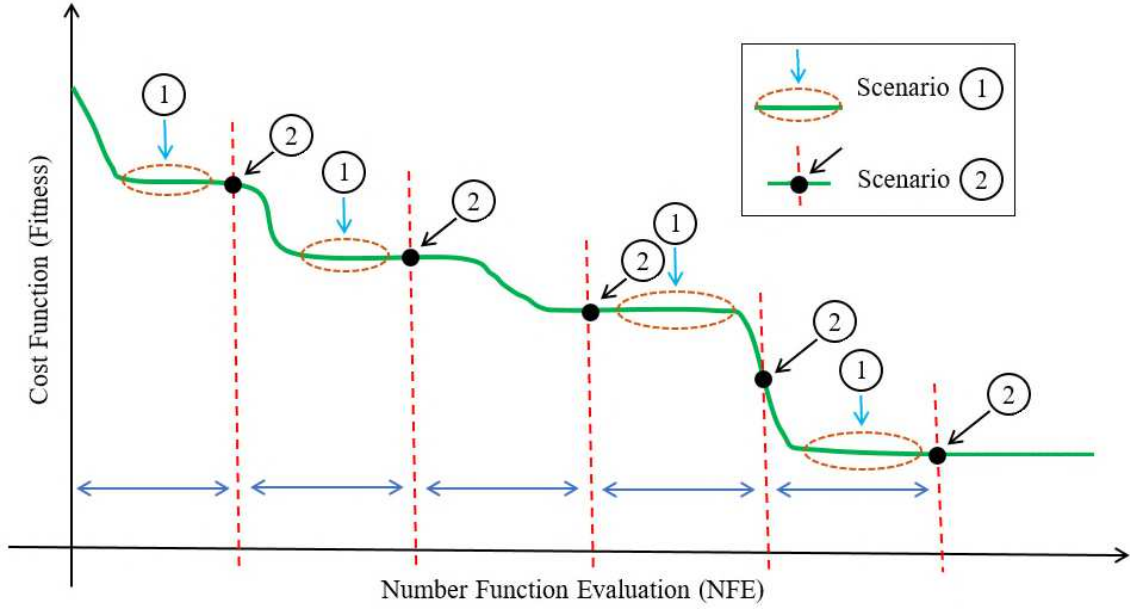
**Fig. 4** The proposed initial population based on OBL.

Most evolutionary algorithms come across stagnancy over time as population diversity decreases. In this research, as demonstrated in Fig. 5, if the proposed algorithm comes across the stagnancy condition, the OBL mechanism will become active, improving the population members' diversity.



**Fig. 5** The use of the OBL mechanism for avoiding the stagnancy conditions.

In the SBCSO algorithm, the OBL mechanism can become active under particular circumstances. As indicated in Fig. 6, two scenarios are introduced. In the first scenario, if the algorithm comes across a stagnancy in terms of a decline in the cost function (fitness) value, the OBL mechanism will become active. In the second scenario, the OBL mechanism is active after several defined steps. In the proposed algorithm, the first scenario is employed. In fact, if the value of the cost function remains unchanged after “*step*” successive steps, the OBL mechanism will be employed (the value of *step* in this article is considered 10).



**Fig. 6** The scenarios of employing the OBL mechanism.

### 3.2 Position updating based on new time-varying V-shape transfer function

The evolutionary algorithms usually employ two types of transfer functions, S-shape and V-shape, to convert continuous space to binary space. In the S-shape transfer functions, the solution is converted to the binary space on the basis of equation 3.

$$x_i^d(t+1) = \begin{cases} 1 & \text{if rand} < \text{Sfunction}(x_i^d(t+1)) \\ 0 & \text{if rand} \geq \text{Sfunction}(x_i^d(t+1)) \end{cases} \quad (3)$$

The expression  $x_i^d(t+1)$  is the value of d-th dimension of the i-th solution at the t-th generation. The phrase rand is a random number with a uniform distribution in [0, 1] range.

In the V-shape transfer functions, the solution is converted to the binary space on the basis of equation 4.

$$x_i^d(t+1) = \begin{cases} 1 - x_i^d(t) & \text{if rand} < \text{Vfunction}(x_i^d(t+1)) \\ x_i^d(t) & \text{if rand} \geq \text{Vfunction}(x_i^d(t+1)) \end{cases} \quad (4)$$

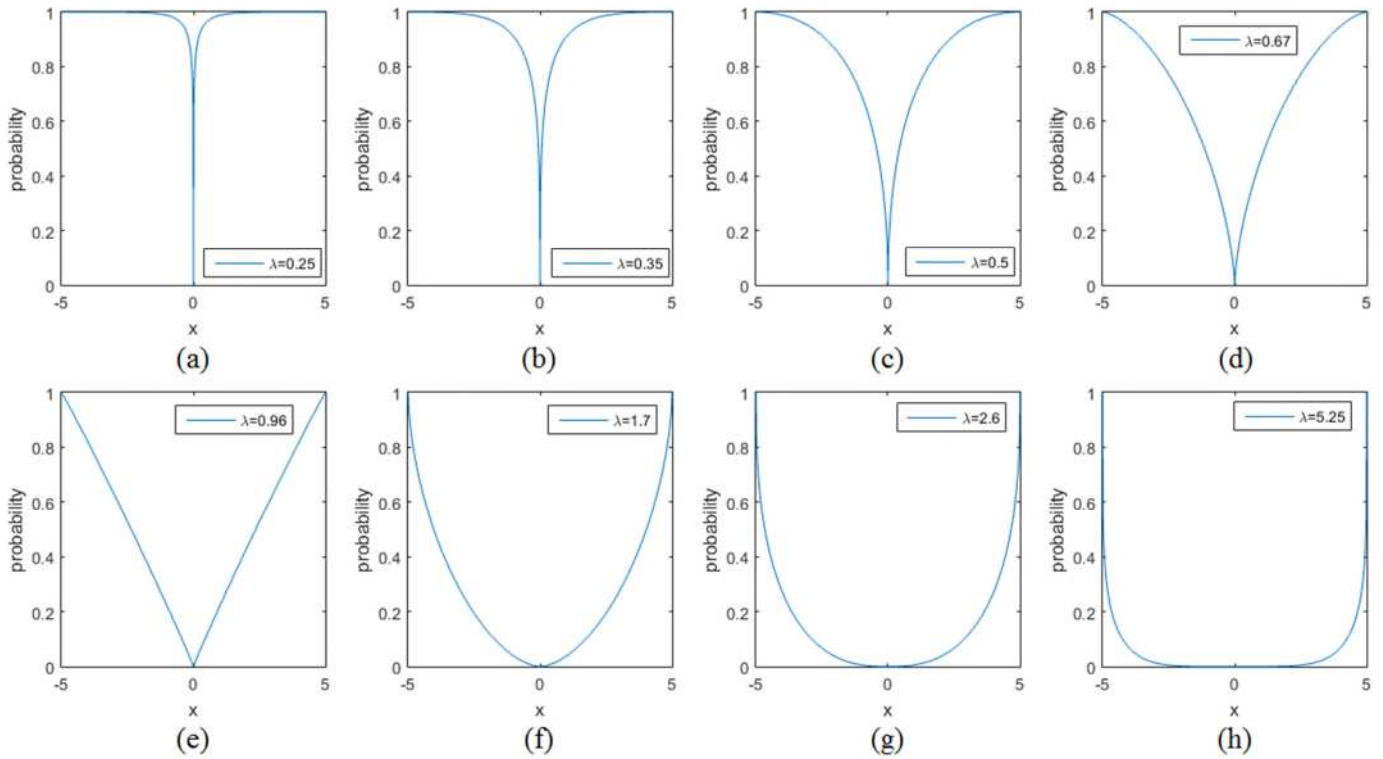
In this paper, a new time-varying V-shape transfer function is presented as indicated below.

$$\text{TVTF}(x, \lambda) = - \left( 1 - \left| \frac{x}{\beta} \right|^\lambda \right)^{\frac{1}{\lambda}} + 1, \beta = \max(x) \quad (5)$$

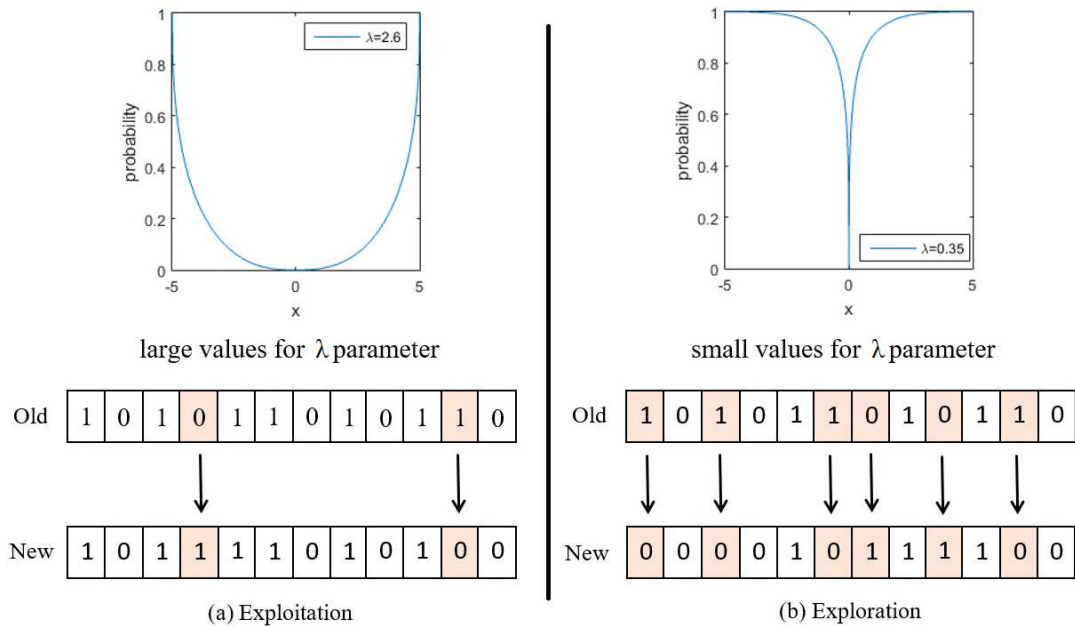
In this equation,  $\lambda$  is a controlling parameter balancing the exploration and the exploitation phases in the optimization process. In Fig. 7, the influence of the parameter on the proposed transfer function is demonstrated.

As can be seen in Fig. 7, for the small values of  $\lambda$ , the transfer function has values approximate to one. Also, according to Fig. 8, the variations on the solution are high, and the algorithm is in the exploration phase.

For the larger values of  $\lambda$ , on the other hand, the transfer function has values approximate to zero. According to Fig. 8, the variations on the solution are low, and the algorithm is in the exploitation phase.



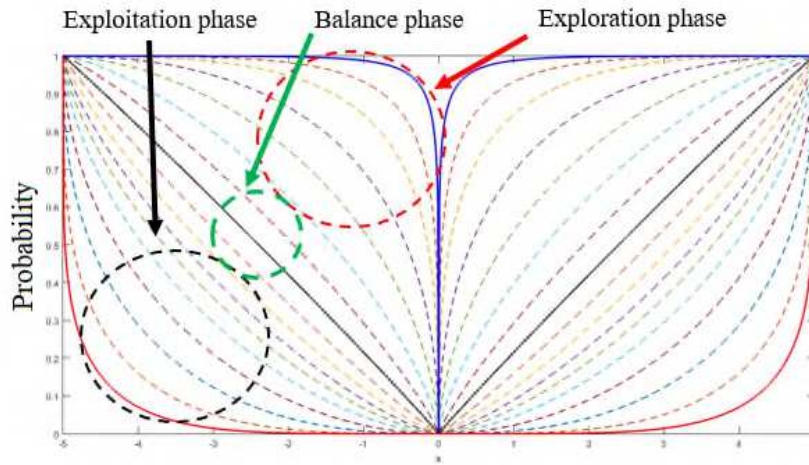
**Fig. 7** The effect of the  $\lambda$  parameter values on the shape of the proposed transfer function.



**Fig. 8** The effect of  $\lambda$  value on the intensity of the changes on a solution.

Figure 9 demonstrates an overview of the different shapes of the proposed transfer function. The evolutionary algorithms usually start to operate with the exploration phase and enter the exploitation phase over time. If such a scenario is supposed to occur in the proposed algorithm, it ought to start with small value of the  $\lambda$  parameter, and  $\lambda$  must be increase over time. This matter is easily performed as the  $\lambda$  parameter linearly increases according to equation 6. This method's major drawback is that it compels the algorithm to move in a predefined path from the exploration phase to the exploitation phase. And sometimes, it makes the algorithm get stuck in the local minimum and be unable to get out it. In subsection 3.3, a new method for determining the  $\lambda$  parameter's value in each step of the proposed algorithm is presented.

$$\lambda_{iter} = \left(\frac{iter}{maxiter}\right)(\lambda_{max} - \lambda_{min}) + \lambda_{min}, \quad \lambda_{min} = 0.25, \quad \lambda_{max} = 5.25 \quad (6)$$

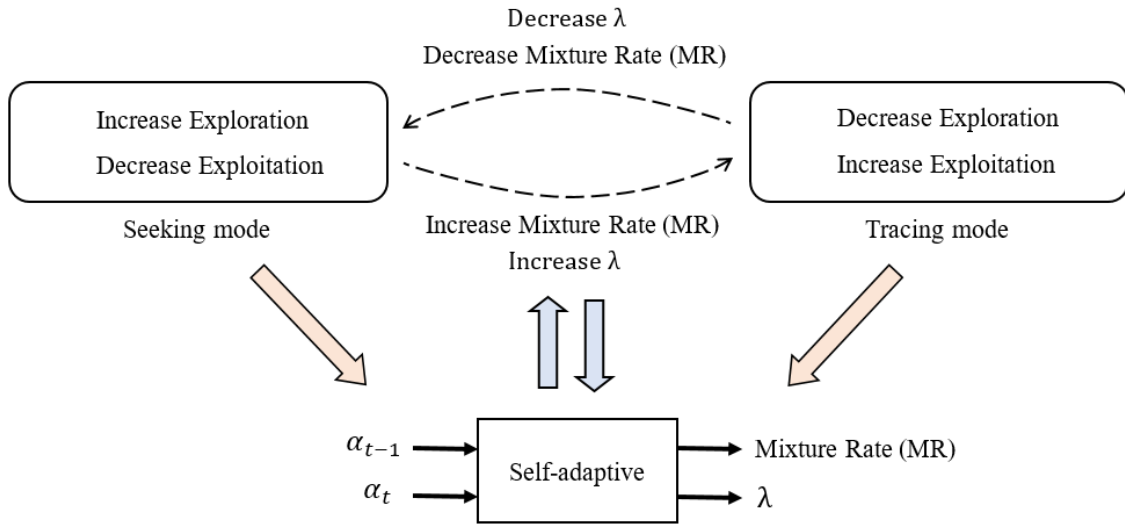


**Fig. 9** The proposed time-varying V-shape transfer function.

### 3.3 The self-adaptive parameters in the proposed method

The  $\lambda$  and MR parameters in the proposed SBCSO algorithm play a pivotal role in placing the population members in two exploration and exploitation phases. In each step of the proposed algorithm, the MR parameter value indicates how many members of the population are in the exploitation phase. Also, the value of the  $\lambda$  parameter in the transfer function indicates the intensity of changes on one solution.

Figure 10 demonstrates the self-adaptive approach of the  $\lambda$  and MR parameters in the proposed SBCSO algorithm. As can be seen in this Figure, the binary cat optimization algorithm operates in two phases of seeking and tracing. In the seeking phase, the concentration is on global search (increase in exploration), and the tracing phase concentrates on local search (increase in exploitation). The evolutionary algorithms cannot have the best performance until they can divide their population members well between these two phases and determine the phase in which they should operate more during the optimization process. This investigation employs  $\lambda$  and MR parameters to control the balance between these two phases at any moment. The update procedure of these parameters will be discussed later.



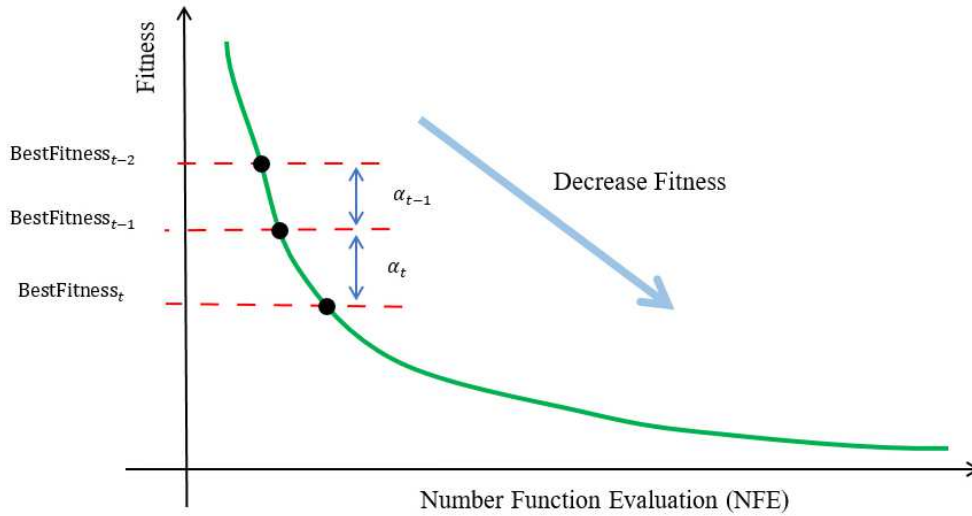
**Fig. 10** The self-adaptive approach of  $\lambda$  and MR parameters in the proposed algorithm.

As can be seen in Fig. 11, the cost function curve can be considered as feedback of the algorithm optimization process's performance. The variation of fitness value is calculated in two successive steps according to the following equations.

$$\alpha_{t-1} = |BestFitness_{t-2} - BestFitness_{t-1}| \quad (7)$$

$$\alpha_t = |BestFitness_{t-1} - BestFitness_t| \quad (8)$$

$BestFitness_t$  is the best cost function value in the t-th step.  $\alpha_t$  is the difference in the value of the cost function changes in two consecutive steps of t and t-1.  $\alpha_{t-1}$  is the difference in the value of the cost function in two successive steps of t-1 and t-2.



**Fig. 11** Cost function value curve.

In the proposed algorithm is adapted to  $\lambda$  and MR parameters based on  $\alpha_t$  and  $\alpha_{t-1}$  according to the following pseudo-code. When the algorithm gets stuck in the local optimum, the cost function

curve remains unchanged ( $\alpha_t = \alpha_{t-1} = 0$ ). In this case, to increase the diversity of the population members, the search mechanism should be in the exploration phase (by reducing the values of  $\lambda$  and MR). When the slope of the cost function curve's variations is low ( $\alpha_t < \alpha_{t-1}$ ), it indicates that either the population members' diversity decreases or we are close to a local or global optimum. Accordingly, in order to increase the population members' diversity, the search mechanism ought to be in the exploration phase (by reducing the  $\lambda$  and MR values). When the slope of the cost function value's variations is high ( $\alpha_{t-1} < \alpha_t$ ), it means that the search route is suitable, and we can increase the convergence speed and place the search mechanism in the exploitation phase (by increasing the values of  $\lambda$  and MR).

```
[MR, λ] = Self-adaptive(αt, αt-1)
  if (αt-1 < αt) //Exploitation phase
    {
      MR = min(MR + 0.02, MRmax)
      λ = min(λ + 0.15, λmax)
    }
  elseif //Exploration phase
    {
      MR = max(MR - 0.02, MRmin)
      λ = max(λ - 0.15, λmin)
    }
λmin = 0.25, λmax = 5.25, MRmin = 0.02, MRmax = 0.90
```

### 3.4 Fitness function

The definition of a cost function for an optimization problem is of paramount importance. A gene selection problem as a minimization problem is expressed as follows:

$$\text{Minimum } F(x) = (f_1(x), f_2(x)) \quad (9)$$

The  $f_1(x)$  is classification accuracy and  $f_2(x)$  is the number of selected genes. The objective functions of  $f_1(x)$  and  $f_2(x)$  for a specific dataset are defined as follows:

$$f_1(x) = \frac{\text{Number of wrongly predicted}}{\text{Total number of instances}} \quad (10)$$

$$f_2(x) = \frac{\text{Number of selected features}}{\text{Total number of features}} \quad (11)$$

### 3.4.1 Single objective approach

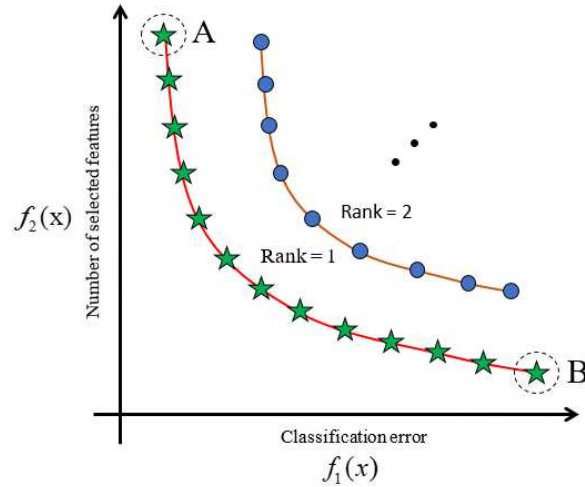
In the single-objective approach, the fitness function is defined as below:

$$fitness = w_1 \cdot f_1(x) + w_2 \cdot f_2(x), \quad w_1 + w_2 = 1 \quad (12)$$

The values of  $w_1$  and  $w_2$  are the weights to control the effect of objective functions on the final fitness function's final value. In the test results, the values 0.95 and 0.6 are considered for parameter  $w_1$ .

### 3.4.2 Multi-objective approach

In the single-objective approach to the gene selection problem, the objective functions of  $f_1(x)$  and  $f_2(x)$  are minimized simultaneously, and ultimately, an optimal solution is obtained. In the multi-objective approach, according to Fig. 12, a set of answers titled Pareto front (PF). Based on the non-dominated sorting algorithm, the best answers are ranked first. As demonstrated in Fig. 12, solution 'A' comprises the maximum selected genes and the minimum classification errors. On the contrary, solution 'B' contains the minimum selected genes and the maximum classification errors.



**Fig. 12** The Pareto front for solving the gene selection problem.

Note that the proposed SBCSO algorithm is a single-objective version. In a multi-objective approach, the cost function is defined as follows:

$$fitness_i = (m \cdot Rank_i) - CD_i \quad (13)$$

A definition of the concept of the crowding distance (CD) is:

$$CD(x_c) = \sum_{j=1}^m \frac{f_j(x_a) - f_j(x_b)}{f_j^{max} - f_j^{min}} \quad (14)$$

where  $f_i^{max}$  and  $f_i^{min}$  are maximum and minimum of the value of the  $i$ -th objective function,  $m$  is the number of the objective functions,  $f_i(x_a)$  and  $f_i(x_b)$  are the  $i$ -th objective function value related to the two members close to the solution  $x_c$  in the space of the Pareto front.

The multi-objective approach of the proposed SBCSO algorithm runs exactly as its single-objective version does. The only difference is that in the single-objective approach, the SBCSO



algorithm aims to minimize the cost function of equation 12. Nevertheless, in the multi-objective approach, the SBCSO algorithm aims to minimize the cost function of equation 13.

### 3.5 Termination criteria

One of the evolutionary algorithms' challenges is that it is not clear when the algorithm will reach the optimal solution. Also, it might not reach the optimal solution at all and get stuck in a local optimum. Therefore, in evolutionary algorithms, usually, the end of the search process is considered based on either the number of evaluations of the objective function (*NFE*) or the number of steps. In this paper, the *NFE* is employed. The pseudo-code of the proposed SBCSO algorithm is indicated in Algorithm 2.

---

**Algorithm 2:** Self-adaptive Binary Cat Swarm Optimization Algorithm (SBCSO)

---

1. **Initialization:** popSize, Max<sub>NFE</sub>,  $\lambda$ , MR, nfe=0, etc.
  2. **Initial population** (subsection 3.1).
  3. **While** (nfe < Max<sub>NFE</sub>) **do**
  4.     Division of population members between two phases based on the MR.
  5.     **For** i=1 to popSize
  6.         **If** (cat[i].flag == 1) **then**
  7.             //Seeking phase (update  $x(i)$ ).
  8.         **Else**
  9.             //Tracing phase (update  $x(i)$ ).
  10.         **End if**
  11.     **End for**
  12.     Convert continuous to binary space (subsection 3.2).
  13.     Evaluate population (cost function value).
  14.     Adaptation of the  $\lambda$  and MR (subsection 3.3).
  15.     **If** (stagnancy conditions happen) **then**
  16.         Go to step 20.
  17.     **Else**
  18.         Go to step 22.
  19.     **End if**
  20.     Improvement of the population members based on the OBL (subsection 3.1).
  21.     Evaluate population (cost function value).
  22.     **If** (termination condition) **then**
  23.         Go to step 29.
  24.     **Else**
  25.         Go to step 3.
  26.     **End if**
  27.     Update nfe.
  28. **End while**
  29. **Output:** The report of the result.
-

## 4. Experimental results and discussions

In this section, in order to investigate the proposed algorithm performance, the results of the fifteen datasets simulation related to the microarray data of different types of cancers, compared to the other common methods, are reported in several subsections as follows. The assessment results of the SBCSO algorithm, compared to the binary single-objective optimization algorithms, such as the BACO [13], BBA [25], BCSO [23], BDA [26], BGSA [35], BGA [42], BPSO [21], and BGWO [22], are reported in subsection 4.3. Finally, the assessment results of the SBCSO algorithm, compared to the binary multi-objective optimization algorithms, such as BMOCSO [46], BNSGA-II [45], BMODE [43], BMOPSO [44], and BMOBBA [47], are reported in subsection 4.5. All of the simulations were executed in MATLAB on a computer equipped with a Core i7 CPU and 16G RAM. In this study, the results of the proposed SBCSO algorithm are compared based on the statistical tests together with different algorithms. Friedman, Quade, and Wilcoxon are among the statistical tests used in the assessment of the results [48,49].

### 4.1 Dataset description

Some details of 15 datasets of different types of cancer microarray data, including the numbers of samples, genes, and data class of each dataset, are indicated in Table 1. Each dataset is categorized into two training and testing parts based on the k-fold cross-validation in the results' assessment. In this research, to ensure the experiments' results on different datasets, the k=10 value is used for the k-fold cross validation mechanism. The optimization process of all feature selection algorithms is executed on train data, and after selecting the optimal features, the assessment results of classification on test data are reported.

In the suggested wrapper-based method, a K-nearest neighbor (KNN) algorithm is used to select an optimal subset of features. The KNN algorithm obtains the optimal features based on the train data with the lowest classification error. According to the gained features, the classification accuracy of the test data is achieved. In the KNN algorithm, the K value and also the distance method is obtained based on the trial and error method. In this study, the KNN algorithm, K=5 and Euclidean distance method are taken into consideration.

**Table 1** Microarray datasets.

ID	Dataset	#Genes	#Instances	#Classes	Keywords
DS01	Colon Cancer	2000	60	2	continuous, binary
DS02	Central Nervous System Cancer	7129	60	2	discrete, binary
DS03	Blood1 Cancer	7129	72	2	discrete, binary
DS04	Ovarian Cancer	15154	253	2	continuous, binary
DS05	Blood2 Cancer	12582	72	3	discrete, multi-class
DS06	SRBCT Cancer	2308	83	4	continuous, multi-class
DS07	Breast Cancer	2905	168	2	continuous, binary
DS08	Prostate Cancer	12600	102	2	continuous, binary
DS09	Myeloma Cancer	12625	173	2	continuous, binary
DS10	Lung Cancer	12533	181	2	continuous, binary
DS11	Lymphoma Cancer	7129	77	2	continuous, binary
DS12	Crohn's Cancer	22283	127	3	continuous, multi-class
DS13	Huntington's Cancer	22283	31	2	continuous, binary
DS14	Miscellaneous1 Cancer	1413	217	3	continuous, multi-class
DS15	Miscellaneous2 Cancer	10100	50	2	continuous, binary

### 4.2 Parameter settings

All of the values related to the parameters of the different algorithms are provided in Table 2.

**Table 2** Parameters values of compared algorithms.

Parameter setting	Value	
Initial population size	{ 100, 200}	
Stopping criteria	The number of fitness evaluation (NFE)	20000
	Number of generations	500
Problem search space	Domain	[0 1]
	Dimension	Number of features
Objective function parameter	W1	{0.95, 0.60}
Binary gravitational search algorithm (BGSA) [35]	$\alpha$	20
	Gravitational constant (G0)	100
Binary particle swarm optimization (BPSO) [21], binary multi-objective particle swarm optimization (BMOPSO) [44]	c1	2.05
	c2	2.05
	Inertia weight (w)	0.9 to 0.1
Binary cat swarm optimization (BCSO) [23], binary multi-objective cat swarm optimization (BMOCSSO) [46], Proposed SBCSO	c1	2.05
	Inertia weight (w)	0.9 to 0.1
	Mixture rate (MR)	0.5
	Seeking memory pool (SMP)	5
	counts of dimension to change (CDC)	0.2
	self-position considering (SPC)	TRUE
Binary grey wolf optimization (BGWO) [22]	No parameter	
Binary multi-objective differential evolution (BMODE) [43]	Crossover rate	0.9
	Mutation factor	0.6
Binary genetic algorithm (BGA) [42], binary multi-objective genetic algorithm (BNSGA-II) [45]	Crossover rate	0.9
	Mutation rate	1/dimension
Binary dragonfly algorithm (BDA) [26]	beta	1.5
	Inertia weight (w)	0.9 to 0.2
Binary bat algorithm (BBA) [25], binary multi-objective bat optimization algorithm (BMOBBA) [47]	$\alpha$	0.9
	$\gamma$	0.9
Binary ant colony optimization (BACO) [13]	Evaporation rate (p)	0.1
	Pheromone (alpha)	1
	Visibility (beta)	0.02

### 4.3 Comparing the proposed algorithm (SBCSO) with other binary evolutionary algorithms

This section compares the proposed algorithm with other single-objective binary optimization algorithms, such as BACO [13], BBA [25], BCSO [23], BDA [26], BGSA [35], BGA [42], BPSO [21], and BGWO [22]. All of the different algorithms' parameters are indicated in Table 2. The results obtained from 20-times independent execution of the proposed algorithm, compared to the other binary algorithms, are reported in Tables 3 to 10. In order to assess the SBCSO algorithm performance, the two values of 0.6 and 0.95 have been used for the w1 parameter in formula fitness (see equation 12). The results of the Tables 3 and 6 are based on w1=0.95. Accordingly, the results of Tables 4 and 7 are on the basis of w1=0.60. The fitness values of all benchmark datasets are indicated in Tables 3 and 4. As demonstrated in Table 3, the SBCSO algorithm has the lowest fitness value in 12 out of 15 benchmark datasets. The BGA algorithm has obtained better results in datasets DS02, DS05, and DS09, compared to the SBCSO algorithm. Furthermore, as indicated in Table 4, the SBCSO algorithm has the lowest fitness value among all of the datasets, all of which are on the basis of w1=0.60.

**Table 3** Comparison between the binary optimization algorithms based on the fitness average ( $w1=0.95$ ).

#DS	BACO [13]	BBA [25]	BCSO [23]	BDA [26]	BGSA [35]	BGA [42]	BPSO [21]	BGWO [22]	SBCSO
DS01	1.78E-01 ± 7.52E-02	1.32E-01 ± 8.25E-02	1.32E-01 ± 2.02E-02	1.31E-01 ± 8.31E-02	1.47E-01 ± 6.06E-02	1.16E-01 ± 1.78E-02	1.32E-01 ± 3.23E-02	1.41E-01 ± 8.66E-02	<b>8.19E-02 ± 8.72E-03</b>
DS02	2.62E-01 ± 6.25E-02	1.98E-01 ± 1.29E-02	2.31E-01 ± 7.79E-02	2.29E-01 ± 8.47E-02	1.99E-01 ± 6.43E-02	<b>1.65E-01 ± 7.06E-02</b>	2.15E-01 ± 6.95E-02	2.23E-01 ± 6.24E-02	1.66E-01 ± 2.37E-02
DS03	1.57E-01 ± 4.51E-02	1.16E-01 ± 4.05E-02	1.17E-01 ± 7.12E-02	1.29E-01 ± 7.36E-02	1.17E-01 ± 2.49E-02	8.82E-02 ± 4.92E-02	1.16E-01 ± 4.56E-02	1.26E-01 ± 6.67E-02	<b>3.57E-02 ± 2.04E-02</b>
DS04	8.43E-02 ± 7.01E-02	5.40E-02 ± 2.12E-02	6.94E-02 ± 5.05E-02	7.01E-02 ± 6.59E-02	6.85E-02 ± 3.13E-02	4.21E-02 ± 4.68E-02	6.76E-02 ± 5.38E-02	7.28E-02 ± 2.19E-02	<b>1.88E-02 ± 1.13E-02</b>
DS05	1.44E-01 ± 3.81E-02	7.69E-02 ± 7.65E-02	7.76E-02 ± 5.68E-02	9.06E-02 ± 5.40E-02	7.71E-02 ± 4.34E-02	<b>3.61E-02 ± 3.29E-02</b>	6.38E-02 ± 5.06E-02	8.67E-02 ± 4.04E-02	3.65E-02 ± 1.54E-02
DS06	1.17E-01 ± 2.30E-02	6.95E-02 ± 5.35E-02	8.12E-02 ± 3.49E-02	7.06E-02 ± 5.23E-02	7.00E-02 ± 2.33E-02	2.20E-02 ± 2.86E-03	6.99E-02 ± 3.10E-02	6.80E-02 ± 6.51E-02	<b>2.01E-02 ± 2.60E-02</b>
DS07	3.13E-01 ± 1.85E-02	2.88E-01 ± 8.70E-02	2.96E-01 ± 1.04E-02	3.01E-01 ± 7.20E-02	2.96E-01 ± 7.54E-02	2.58E-01 ± 7.95E-02	2.96E-01 ± 1.68E-02	2.95E-01 ± 3.08E-02	<b>2.46E-01 ± 7.40E-02</b>
DS08	1.37E-01 ± 7.82E-02	9.90E-02 ± 5.98E-02	1.08E-01 ± 3.81E-02	1.08E-01 ± 5.11E-02	8.95E-02 ± 4.21E-02	7.89E-02 ± 1.61E-02	9.97E-02 ± 2.92E-02	1.17E-01 ± 2.47E-02	<b>6.67E-02 ± 2.92E-02</b>
DS09	2.06E-01 ± 7.24E-02	1.77E-01 ± 4.12E-02	1.83E-01 ± 2.93E-02	1.87E-01 ± 4.23E-02	1.84E-01 ± 1.77E-02	<b>1.55E-01 ± 2.06E-02</b>	1.83E-01 ± 8.54E-02	1.95E-01 ± 5.60E-02	<b>1.64E-01 ± 1.48E-02</b>
DS10	8.83E-02 ± 4.42E-02	6.13E-02 ± 2.44E-02	7.14E-02 ± 6.96E-02	6.99E-02 ± 1.59E-02	7.16E-02 ± 6.49E-02	2.10E-02 ± 1.55E-02	7.21E-02 ± 3.01E-02	7.50E-02 ± 7.24E-02	<b>3.09E-03 ± 3.41E-03</b>
DS11	1.36E-01 ± 6.15E-02	8.57E-02 ± 4.03E-02	8.62E-02 ± 7.49E-02	5.98E-02 ± 5.26E-02	7.45E-02 ± 3.81E-02	6.07E-02 ± 4.51E-02	6.27E-02 ± 6.01E-02	8.23E-02 ± 5.98E-02	<b>3.49E-02 ± 2.70E-02</b>
DS12	2.26E-01 ± 3.49E-02	1.67E-01 ± 8.39E-02	1.67E-01 ± 4.44E-02	1.52E-01 ± 2.48E-02	1.66E-01 ± 8.24E-02	1.36E-01 ± 8.84E-02	1.53E-01 ± 4.51E-02	1.65E-01 ± 3.06E-02	<b>1.07E-01 ± 4.27E-02</b>
DS13	5.53E-02 ± 1.68E-02	2.28E-02 ± 2.17E-02	2.39E-02 ± 1.41E-02	2.38E-02 ± 1.23E-02	2.48E-02 ± 8.43E-03	1.75E-02 ± 3.87E-03	5.46E-02 ± 4.91E-02	5.93E-02 ± 2.90E-02	<b>2.28E-03 ± 3.72E-03</b>
DS14	3.21E-02 ± 3.98E-03	2.16E-02 ± 1.09E-02	1.65E-02 ± 7.31E-03	2.15E-02 ± 1.39E-02	2.61E-02 ± 8.80E-03	1.06E-02 ± 2.17E-03	2.60E-02 ± 2.75E-02	2.83E-02 ± 1.18E-02	<b>5.56E-04 ± 3.80E-04</b>
DS15	2.53E-01 ± 1.24E-02	1.76E-01 ± 6.95E-02	1.94E-01 ± 5.00E-02	1.94E-01 ± 4.84E-02	1.76E-01 ± 8.24E-02	1.37E-01 ± 5.88E-02	1.76E-01 ± 5.94E-02	1.87E-01 ± 7.44E-02	<b>1.25E-01 ± 5.61E-02</b>

**Table 4** Comparison between the binary optimization algorithms based on the fitness average ( $w1=0.60$ ).

#DS	BACO [13]	BBA [25]	BCSO [23]	BDA [26]	BGSA [35]	BGA [42]	BPSO [21]	BGWO [22]	SBCSO
DS01	3.18E-01 ± 1.48E-02	2.50E-01 ± 6.46E-02	2.71E-01 ± 1.34E-02	2.41E-01 ± 1.57E-02	2.64E-01 ± 5.17E-02	2.35E-01 ± 1.77E-02	2.50E-01 ± 7.55E-02	2.80E-01 ± 6.78E-02	<b>7.39E-02 ± 2.20E-02</b>
DS02	3.62E-01 ± 2.39E-02	3.16E-01 ± 4.13E-02	3.11E-01 ± 7.65E-02	2.88E-01 ± 7.43E-02	2.91E-01 ± 1.48E-02	2.77E-01 ± 4.19E-02	3.05E-01 ± 5.22E-02	3.46E-01 ± 6.25E-02	<b>1.35E-01 ± 6.02E-02</b>
DS03	2.81E-01 ± 8.61E-02	2.27E-01 ± 8.36E-02	2.39E-01 ± 1.42E-02	2.32E-01 ± 6.90E-02	2.44E-01 ± 3.15E-02	1.51E-01 ± 4.38E-02	2.55E-01 ± 5.38E-02	2.72E-01 ± 4.34E-02	<b>2.21E-02 ± 1.87E-02</b>
DS04	2.22E-01 ± 1.26E-02	1.79E-01 ± 5.49E-02	1.87E-01 ± 8.05E-02	1.54E-01 ± 6.35E-02	1.91E-01 ± 2.52E-02	5.35E-02 ± 3.01E-02	2.01E-01 ± 4.69E-02	2.29E-01 ± 2.25E-02	<b>3.69E-03 ± 2.69E-03</b>
DS05	2.64E-01 ± 3.01E-02	2.30E-01 ± 3.32E-02	2.21E-01 ± 5.94E-02	1.89E-01 ± 3.12E-02	2.19E-01 ± 7.60E-02	1.53E-01 ± 8.86E-02	2.12E-01 ± 6.84E-02	2.57E-01 ± 5.67E-02	<b>2.61E-02 ± 9.51E-03</b>
DS06	2.66E-01 ± 4.38E-02	2.23E-01 ± 1.75E-02	2.37E-01 ± 5.79E-02	2.00E-01 ± 4.77E-02	2.28E-01 ± 6.57E-02	1.03E-01 ± 6.60E-02	2.18E-01 ± 6.11E-02	2.58E-01 ± 1.55E-02	<b>2.68E-02 ± 2.62E-02</b>
DS07	3.75E-01 ± 7.23E-02	3.42E-01 ± 4.39E-02	3.37E-01 ± 1.73E-02	3.20E-01 ± 3.13E-02	3.54E-01 ± 2.23E-02	2.56E-01 ± 3.25E-02	3.58E-01 ± 4.52E-02	3.75E-01 ± 4.66E-02	<b>1.58E-01 ± 8.00E-02</b>
DS08	2.67E-01 ± 3.04E-02	2.45E-01 ± 2.79E-02	2.39E-01 ± 6.34E-02	2.28E-01 ± 7.76E-02	2.47E-01 ± 3.76E-02	1.96E-01 ± 7.24E-02	2.51E-01 ± 6.40E-02	2.72E-01 ± 5.82E-02	<b>6.04E-02 ± 3.16E-02</b>
DS09	3.12E-01 ± 2.41E-02	2.89E-01 ± 6.77E-02	2.92E-01 ± 4.79E-02	2.66E-01 ± 2.22E-02	2.94E-01 ± 3.73E-02	2.29E-01 ± 5.86E-02	2.91E-01 ± 2.53E-02	3.06E-01 ± 2.94E-02	<b>1.11E-01 ± 8.34E-02</b>
DS10	2.42E-01 ± 6.18E-02	2.08E-01 ± 6.43E-02	2.23E-01 ± 6.09E-02	1.95E-01 ± 8.56E-02	2.21E-01 ± 2.67E-02	1.24E-01 ± 6.67E-02	2.24E-01 ± 2.89E-02	2.41E-01 ± 5.86E-02	<b>1.31E-02 ± 1.27E-02</b>
DS11	2.39E-01 ± 5.66E-02	2.15E-01 ± 5.33E-02	2.15E-01 ± 7.96E-02	1.90E-01 ± 3.12E-02	2.20E-01 ± 3.54E-02	1.95E-01 ± 1.95E-02	2.16E-01 ± 8.52E-02	2.35E-01 ± 4.84E-02	<b>3.88E-02 ± 1.11E-02</b>
DS12	3.27E-01 ± 4.24E-02	2.95E-01 ± 4.59E-02	2.58E-01 ± 3.93E-02	2.77E-01 ± 7.11E-02	2.72E-01 ± 6.02E-02	2.35E-01 ± 7.18E-02	2.87E-01 ± 8.46E-02	3.06E-01 ± 2.54E-02	<b>1.08E-01 ± 2.11E-02</b>
DS13	2.51E-01 ± 3.78E-02	1.83E-01 ± 2.20E-02	2.11E-01 ± 5.69E-02	1.82E-01 ± 3.10E-02	2.08E-01 ± 1.36E-02	1.41E-01 ± 7.04E-02	2.10E-01 ± 2.94E-02	2.35E-01 ± 6.50E-02	<b>1.35E-02 ± 2.29E-02</b>
DS14	1.87E-01 ± 7.57E-02	1.38E-01 ± 4.44E-02	1.17E-01 ± 8.10E-02	1.47E-01 ± 4.13E-02	1.70E-01 ± 7.15E-02	5.58E-02 ± 3.23E-02	1.72E-01 ± 7.47E-02	1.87E-01 ± 4.02E-02	<b>5.34E-03 ± 2.88E-04</b>
DS15	3.43E-01 ± 5.12E-02	2.98E-01 ± 8.07E-02	2.97E-01 ± 5.70E-02	2.74E-01 ± 2.24E-02	2.91E-01 ± 2.60E-02	2.59E-01 ± 4.26E-02	3.00E-01 ± 6.99E-02	3.27E-01 ± 7.32E-02	<b>1.10E-01 ± 3.55E-02</b>

In order to have a quick examination of the algorithms' results, the comparison between all of the algorithms for all of the datasets is conducted in Fig. 13, Fig. 14, and Table 5. In the ranking of the algorithms, the white box indicates the first rank, and the black box indicates the last rank among all.

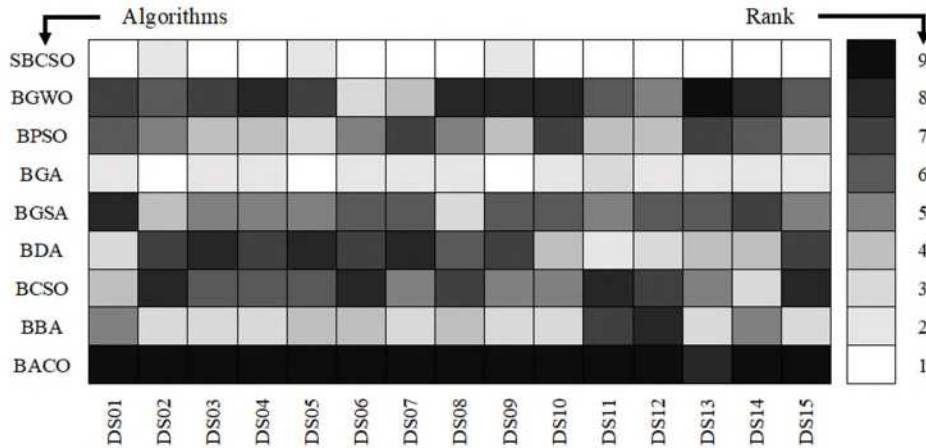


Fig. 13 Ranking the algorithms based on the average value of the fitness ( $w_1=0.95$ ).

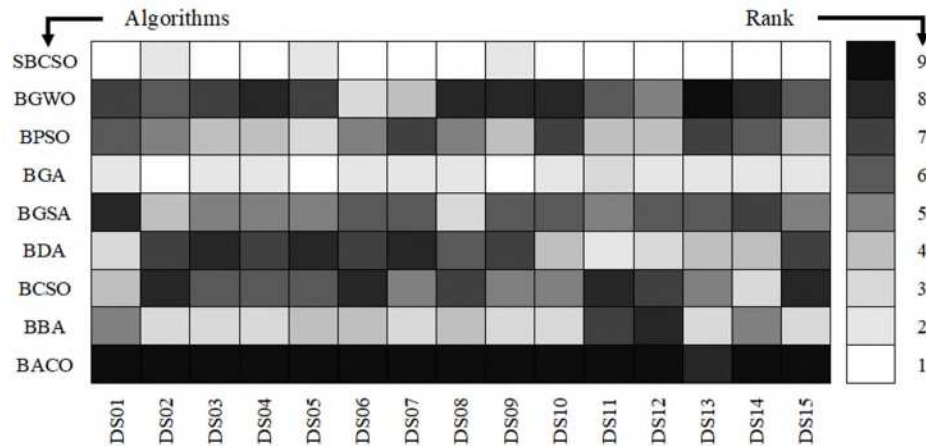


Fig. 14 Ranking the algorithms based on the average value of the fitness ( $w_1=0.60$ ).

Table 5 Ranking the algorithms based on the average value of the fitness and accuracy.

Metric	W1	BACO	BBA	BCSO	BDA	BGSA	BGA	BPSO	BGWO	SBCSO
Fitness	0.60	8.73	5.00	5.20	3.27	5.53	2.07	5.93	8.27	<b>1.00</b>
	0.95	8.93	4.07	6.07	5.67	5.53	1.87	5.00	6.67	<b>1.20</b>
Accuracy	0.60	8.67	4.33	4.20	7.20	4.93	2.07	4.73	7.00	<b>1.87</b>
	0.95	8.80	5.20	6.00	6.20	5.47	1.80	5.07	4.73	<b>1.73</b>
Overall Ranking		9.00	3.00	5.00	7.00	5.00	2.00	4.00	8.00	<b>1.00</b>
(Average ranking number)		(8.78)	(4.65)	(5.37)	(5.58)	(5.37)	(1.95)	(5.18)	(6.67)	<b>(1.45)</b>

Tables 6 and 7 demonstrates the results of the best classification error along with several features selected from each dataset for all of the algorithms (table 6 for the values of  $w_1=0.95$  and table 7 for the values

of  $w_1=0.60$ ). As demonstrated in table 6, the SBCSO algorithm has the least value of classification error in 10 datasets among 15 benchmark datasets. Compared to the SBCSO algorithm, the BGA algorithm in three data sets of DS05, DS07, DS08, and DS09 has gained better results. The BGWO algorithm in the DS02 dataset has a lesser classification error than the SBCSO algorithm. On the other hand, according to tables 6 and 7, the SBCSO algorithm has selected a lesser number of genes compared to the other algorithms.

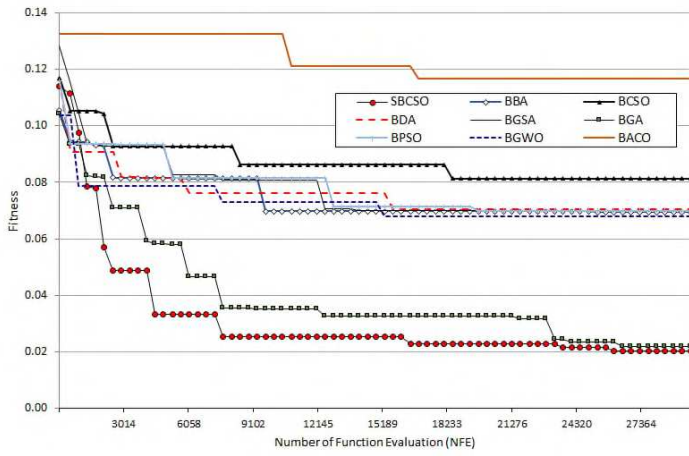
**Table 6** The best classification error and the number of genes selected for each dataset ( $w_1=0.95$ ).

#DS	Metric	BACO [25]	BBA [24]	BCSO [25]	BDA [48]	BGSA [72]	BGA [68]	BPSO [39]	BGWO [28]	SBCSO
DS01	Error (%)	1.62E-01	1.13E-01	1.13E-01	1.13E-01	1.29E-01	9.68E-02	1.13E-01	1.13E-01	<b>8.06E-02</b>
	#Features	776	793	791	754	775	771	808	1094	<b>170</b>
DS02	Error (%)	1.90E-01	1.83E-01	1.56E-01	1.59E-01	1.22E-01	1.50E-01	2.00E-01	<b>1.18E-01</b>	1.67E-01
	#Features	1400	401	1416	1328	1425	388	436	<b>1904</b>	125
DS03	Error (%)	1.11E-01	7.22E-02	7.13E-02	8.67E-02	7.17E-02	4.60E-02	7.19E-02	6.21E-02	<b>1.80E-02</b>
	#Features	1471	1356	1405	1320	1384	1270	1372	1904	<b>532</b>
DS04	Error (%)	6.26E-02	3.16E-02	4.74E-02	4.74E-02	4.74E-02	2.37E-02	4.34E-02	3.95E-02	<b>1.58E-02</b>
	#Features	377	363	370	380	356	297	399	535	<b>58</b>
DS05	Error (%)	1.26E-01	5.55E-02	5.55E-02	6.94E-02	5.55E-02	<b>1.39E-02</b>	4.17E-02	5.55E-02	2.78E-02
	#Features	611	608	625	620	612	<b>578</b>	609	855	255
DS06	Error (%)	8.24E-02	3.57E-02	4.76E-02	3.51E-02	3.54E-02	1.16E-02	3.55E-02	1.84E-02	<b>7.46E-03</b>
	#Features	707	657	665	688	672	609	668	934	<b>241</b>
DS07	Error (%)	3.03E-01	2.80E-01	2.86E-01	2.92E-01	2.86E-01	<b>2.50E-01</b>	2.86E-01	2.74E-01	2.56E-01
	#Features	585	523	562	557	569	<b>4.79E+02</b>	575	812	63
DS08	Error (%)	1.18E-01	7.84E-02	8.82E-02	8.71E-02	6.86E-02	<b>5.88E-02</b>	7.84E-02	8.82E-02	5.88E-02
	#Features	1234	1236	1220	1261	1223	<b>1.16E+03</b>	1269	1692	545
DS09	Error (%)	1.91E-01	1.62E-01	1.68E-01	1.73E-01	1.68E-01	<b>1.39E-01</b>	1.68E-01	1.73E-01	1.62E-01
	#Features	1222	1191	1222	1127	1246	<b>1149</b>	1204	1554	519
DS10	Error (%)	6.69E-02	3.87E-02	4.97E-02	4.97E-02	4.97E-02	1.23E-02	4.97E-02	4.42E-02	<b>1.10E-03</b>
	#Features	1243	1233	1212	1138	1224	1052	1245	1656	<b>155</b>
DS11	Error (%)	1.15E-01	6.49E-02	6.49E-02	3.89E-02	5.19E-02	3.89E-02	3.89E-02	5.19E-02	<b>2.60E-02</b>
	#Features	1113	1026	1049	977	1075	1013	1099	1411	<b>437</b>
DS12	Error (%)	2.20E-01	1.57E-01	1.57E-01	1.41E-01	1.57E-01	1.25E-01	1.41E-01	1.51E-01	<b>1.05E-01</b>
	#Features	1013	1041	1019	1025	1008	995	1061	1243	<b>415</b>
DS13	Error (%)	3.92E-02	6.76E-03	7.09E-03	7.04E-03	7.36E-03	5.37E-03	3.94E-02	4.08E-02	<b>6.77E-04</b>
	#Features	1048	949	995	989	1034	719	997	1193	<b>95</b>
DS14	Error (%)	1.53E-02	8.62E-03	7.45E-03	1.22E-02	9.67E-03	6.07E-03	9.66E-03	1.02E-02	<b>1.30E-04</b>
	#Features	447	341	241	253	430	124	429	474	<b>11</b>
DS15	Error (%)	2.46E-01	1.66E-01	1.86E-01	1.86E-01	1.66E-01	1.26E-01	1.66E-01	1.69E-01	<b>1.23E-01</b>
	#Features	1145	1103	1092	1058	1123	1064	1112	1630	<b>510</b>

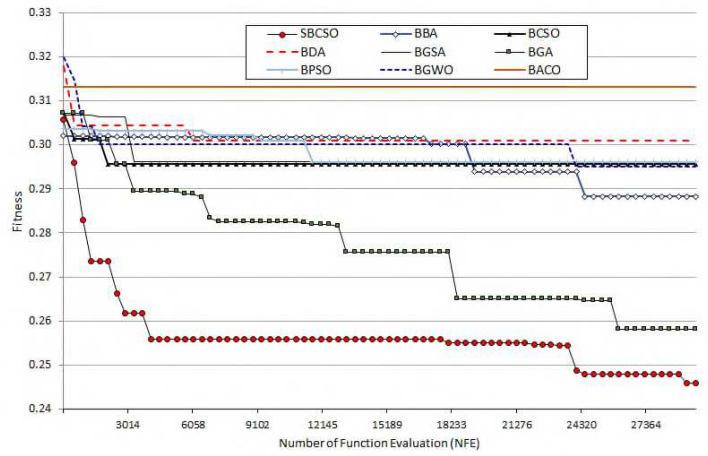
**Table 7** The best classification error and the number of genes selected for each dataset ( $w_1=0.60$ ).

#DS	Metric	BACO [25]	BBA [24]	BCSO [25]	BDA [48]	BGSA [72]	BGA [68]	BPSO [39]	BGWO [28]	SBCSO
DS01	Error (%)	2.04E-01	9.66E-02	1.29E-01	1.61E-01	1.13E-01	9.66E-02	9.66E-02	1.29E-01	<b>8.06E-02</b>
	#Features	782	770	774	576	784	710	767	812	<b>102</b>
DS02	Error (%)	1.36E-01	6.06E-02	6.40E-02	1.04E-01	<b>3.06E-02</b>	4.10E-02	4.45E-02	4.50E-02	1.42E-01
	#Features	701	699	682	565	<b>682</b>	630	696	876	126
DS03	Error (%)	1.51E-01	6.92E-02	8.31E-02	1.11E-01	1.11E-01	2.76E-02	1.11E-01	1.39E-01	<b>1.39E-02</b>
	#Features	678	661	676	591	631	481	671	674	<b>49</b>
DS04	Error (%)	8.85E-02	5.66E-02	4.97E-02	8.82E-02	5.01E-02	8.32E-03	6.97E-02	8.33E-02	<b>3.15E-04</b>
	#Features	337	291	314	203	322	97	318	358	<b>7</b>
DS05	Error (%)	1.26E-01	6.59E-02	5.20E-02	1.04E-01	6.53E-02	8.05E-03	5.15E-02	4.04E-02	<b>1.37E-03</b>
	#Features	611	618	618	412	585	482	588	758	<b>82</b>
DS06	Error (%)	2.07E-01	1.32E-01	1.37E-01	1.32E-01	1.43E-01	3.95E-02	1.21E-01	1.73E-01	<b>1.03E-02</b>
	#Features	424	431	463	361	427	238	437	462	<b>62</b>
DS07	Error (%)	3.21E-01	2.80E-01	2.80E-01	3.09E-01	2.97E-01	2.56E-01	2.91E-01	3.21E-01	<b>2.38E-01</b>
	#Features	531	505	492	390	510	299	531	531	<b>43</b>
DS08	Error (%)	1.19E-01	6.85E-02	7.83E-02	1.08E-01	8.81E-02	<b>5.87E-02</b>	9.79E-02	1.18E-01	6.86E-02
	#Features	1234	1282	1212	1029	1226	<b>1012</b>	1209	1269	121
DS09	Error (%)	1.96E-01	1.73E-01	1.73E-01	1.96E-01	1.79E-01	1.50E-01	1.73E-01	1.85E-01	<b>1.44E-01</b>
	#Features	1226	1171	1185	938	1175	874	1180	1229	<b>155</b>
DS10	Error (%)	1.38E-01	9.85E-02	1.16E-01	1.23E-01	1.02E-01	3.38E-02	1.12E-01	1.26E-01	<b>3.57E-03</b>
	#Features	1192	1118	1151	911	1197	776	1176	1238	<b>82</b>
DS11	Error (%)	7.41E-02	3.88E-02	<b>2.58E-02</b>	3.88E-02	3.88E-02	2.58E-02	3.88E-02	5.17E-02	2.60E-02
	#Features	1038	1027	<b>1068</b>	890	1050	959	1029	1091	124
DS12	Error (%)	2.99E-01	2.52E-01	2.00E-01	2.60E-01	2.12E-01	1.78E-01	2.42E-01	2.34E-01	<b>1.41E-01</b>
	#Features	1070	1043	996	877	1044	927	1024	1198	<b>168</b>
DS13	Error (%)	1.89E-01	8.58E-02	1.22E-01	1.55E-01	1.21E-01	6.63E-02	1.22E-01	1.34E-01	<b>6.32E-03</b>
	#Features	996	951	998	647	982	735	990	1124	<b>70</b>
DS14	Error (%)	2.42E-02	1.38E-02	8.57E-03	1.41E-02	7.11E-02	1.10E-02	1.50E-02	8.15E-02	<b>1.85E-04</b>
	#Features	561	421	364	450	414	160	531	450	<b>17</b>
DS15	Error (%)	2.55E-01	1.80E-01	1.60E-01	1.80E-01	1.60E-01	<b>1.20E-01</b>	1.80E-01	2.20E-01	<b>1.20E-01</b>
	#Features	1440	1438	1520	1257	1479	<b>1414</b>	1456	1478	<b>288</b>

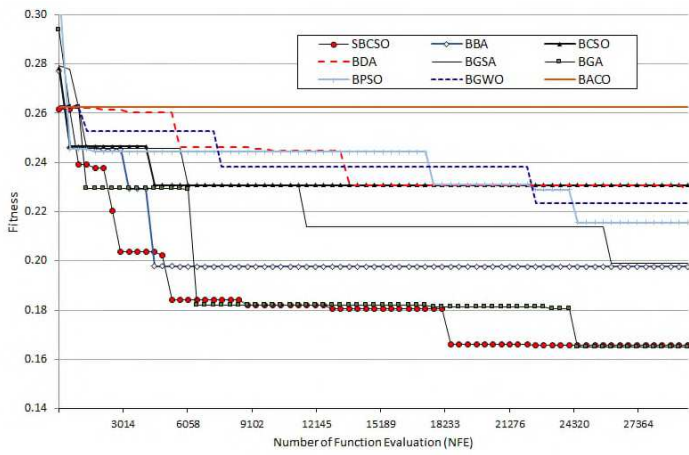
Figures 15-24 indicate the fitness decline curve for different algorithms during the optimization process. As can be seen, the convergence velocity and the slope of fitness decline in the SBCSO are more than the other algorithms.



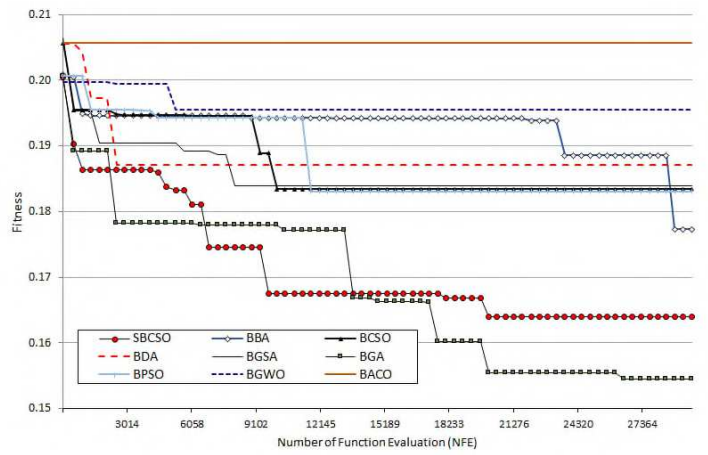
**Fig. 15** The curve of average fitness decline for the DS03 dataset for different algorithms.



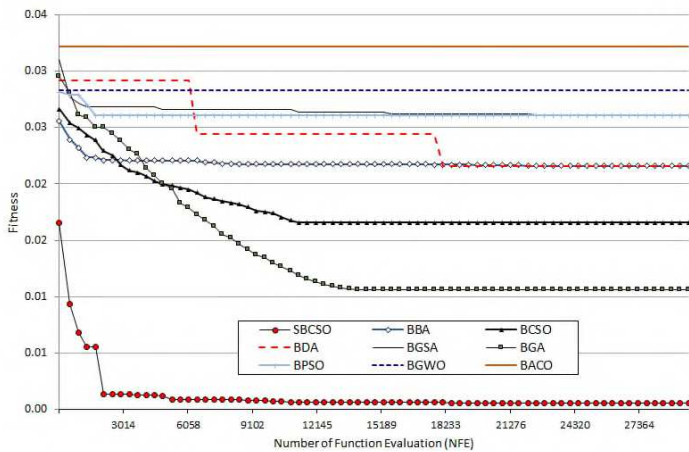
**Fig. 16** The curve of average fitness decline for the DS07 dataset for different algorithms.



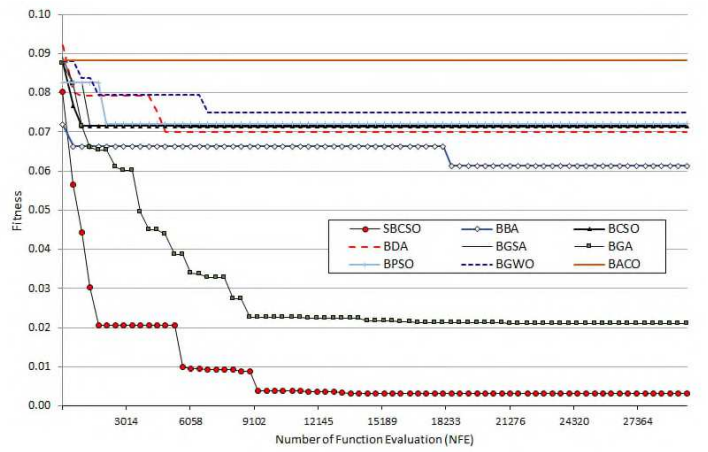
**Fig. 17** The curve of average fitness decline for the DS02 dataset for different algorithms.



**Fig. 18** The curve of average fitness decline for the DS09 dataset for different algorithms.

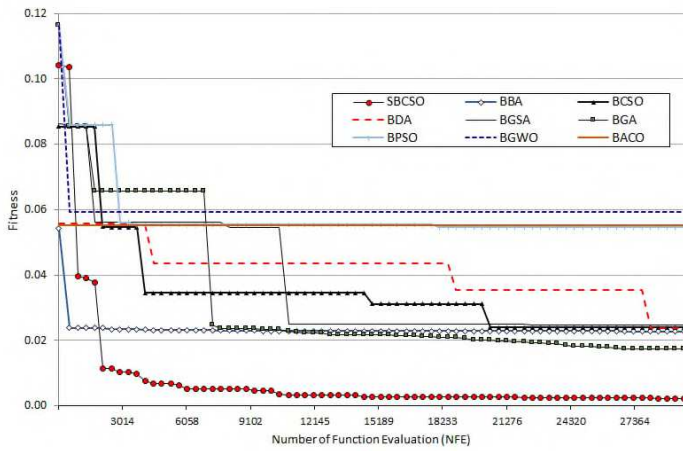


**Fig. 19** The curve of average fitness decline for the DS014 dataset for different algorithms.

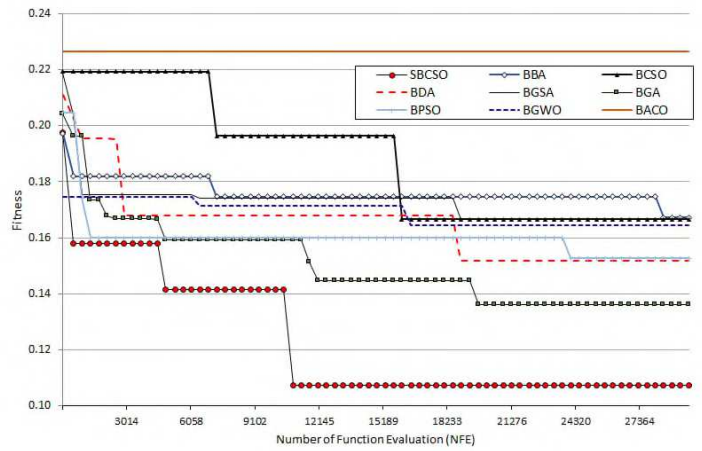


**Fig. 20** The curve of average fitness decline for the DS10 dataset for different algorithms.

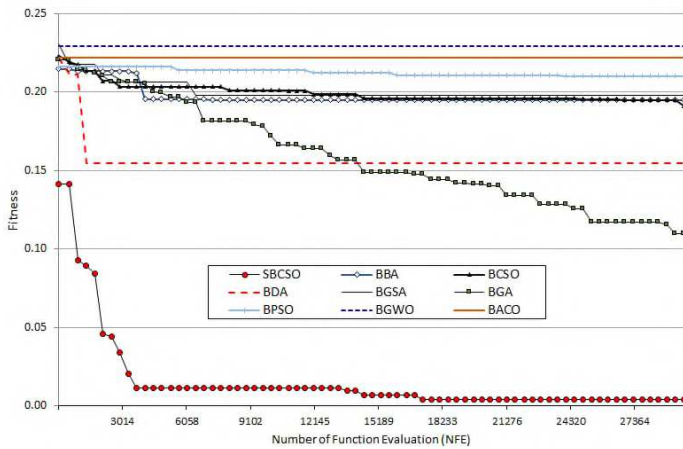




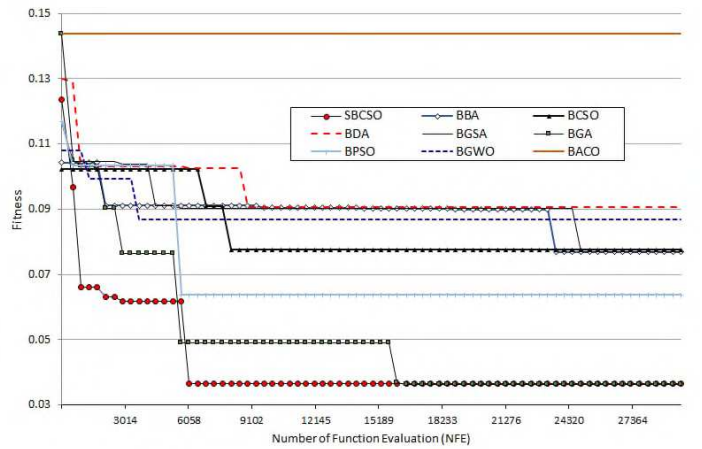
**Fig. 21** The curve of average fitness decline for the DS13 dataset for different algorithms.



**Fig. 22** The curve of average fitness decline for the DS12 dataset for different algorithms.



**Fig. 23** The curve of average fitness decline for the DS04 dataset for different algorithms.



**Fig. 24** The curve of average fitness decline for the DS05 dataset for different algorithms.

The Friedman, Sign, and Quade statistical tests are employed to analyze more of the proposed algorithm's results than other algorithms. The two procedures of pairwise comparisons and multiple comparisons are employed in the results of the reports. The Sign and Wilcoxon tests fall into the pairwise comparisons classification, and the Friedman and Quade tests fall into the multiple comparison's classification.

As indicated in Table 8, the SBCSO algorithm has the best performance in comparison to other algorithms in Quade and Friedman tests and attained the 1.45 and 1.31 ranks for the Friedman and Quade tests, respectively. The BGA and BBA algorithms attained the second and third ranks in the Friedman and Quade tests, respectively. The BACO algorithm is ranked the last among all of the algorithms.

**Table 8** Friedman test and Quade test results.

algorithm	Friedman	Quade
SBCSO	<b>1.45</b>	<b>1.31</b>
BACO	8.78	8.82
BBA	4.65	4.84
BCSO	5.37	5.50
BDA	5.58	5.05
BCSO	5.37	5.31
BGA	1.95	2.01
BPSO	5.18	5.19
BGWO	6.67	6.98
statistic	318.09	64.58
p-value	3.75e-08	3.41e-04

Table 9 reports the results of the Sign test. Table 9 indicates the number of times each algorithm wins another algorithm at both 0.05 and 0.1 levels. Table 9 clearly demonstrates that the SBCSO algorithm with the certainty level of  $\alpha=0.05$  has won all of the other algorithms. The BGA and BBA algorithms are ranked second and third, respectively, and the BACO algorithm is ranked the latest among all other algorithms.

**Table 9** Sign test results .

	SBCSO	BACO	BBA	BCSO	BDA	BGSA	BGA	BPSO	BGWO
SBCSO	-	59	57	57	58	58	47	59	58
$\alpha=$		0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
BACO	1	-	0	0	1	1	0	2	8
$\alpha=$									
BBA	3	60	-	35	38	40	1	38	46
$\alpha=$		0.05			0.05	0.05		0.05	0.05
BCSO	3	60	25	-	30	31	2	28	39
$\alpha=$		0.05							0.05
BDA	2	59	22	30	-	28	2	26	36
$\alpha=$		0.05							0.1
BGSA	2	59	20	29	32	-	2	29	45
$\alpha=$		0.05							0.05
BGA	13	60	59	58	58	58	-	58	59
$\alpha=$		0.05	0.05	0.05	0.05	0.05		0.05	0.05
BPSO	1	58	22	32	34	31	2	-	49
$\alpha=$		0.05							0.05
BGWO	2	52	14	21	24	15	1	11	-
$\alpha=$		0.05							

Table 10 reports the results of the Wilcoxon test. In table 10, the values of R, -R+, and p-value are calculated for all binary comparisons pertinent to SBCSO. Table 10 clearly demonstrates that the proposed algorithm wins over all of the other algorithms with the certainty level of  $\alpha=0.01$ .

**Table 10** Wilcoxon test results .

Comparison	R+	R-	p-value	level of significance
SBCSO versus BACO	1796	34	1.71E-11	$\alpha = 0.01$
SBCSO versus BBA	1732	98	1.51E-10	$\alpha = 0.01$
SBCSO versus BCSO	1740	90	1.83E-10	$\alpha = 0.01$
SBCSO versus BDA	1792	38	4.44E-11	$\alpha = 0.01$
SBCSO versus BGSA	1792	38	4.1E-10	$\alpha = 0.01$
SBCSO versus BGA	1527	303	1.94E-07	$\alpha = 0.01$
SBCSO versus BPSO	1796	34	1.24E-10	$\alpha = 0.01$
SBCSO versus BGWO	1792	38	3.56E-10	$\alpha = 0.01$

#### 4.4 Classification performance

The influence of employing the SBCSO algorithm in the feature selection process for each dataset is calculated to examine the performance and evaluate the SBCSO algorithm. In Table 11, the FSL and FSA values refer to the number of main features and the number of the selected feature of each dataset. The Accuracy and Time values for each dataset are also calculated before and after the feature selection process. The KNN algorithm with the value of K=5 and the k-fold validation mechanisms with the value of k=10 are employed to calculate the accuracy. As indicated in Table 11, the SBCSO positively influences the Accuracy and Time, and in all of the datasets, better performance is reported (with a much lesser number of features).

**Table 11** The impact of using the SBCSO algorithm in the feature selection process.

DS#	Before applying SBCSO			After applying SBCSO		
	FSL	Accuracy (%)	Time (sec)	FSA	Accuracy (%)	Time (sec)
DS01	2000	75.81	0.9559	170	<b>91.9</b>	0.8605
DS02	7129	53.33	0.316	125	<b>83.3</b>	0.1688
DS03	7129	83.33	0.3425	532	<b>98.2</b>	0.1609
DS04	15154	91.30	2.1674	58	<b>98.4</b>	0.1756
DS05	12582	77.78	0.4485	255	<b>97.2</b>	0.162
DS06	2308	83.13	0.2128	241	<b>99.3</b>	0.1535
DS07	2905	68.45	0.3425	63	<b>74.4</b>	0.1624
DS08	12600	84.31	0.6592	545	<b>94.1</b>	0.1795
DS09	12625	78.03	1.147	519	<b>83.8</b>	0.2146
DS10	12533	92.82	1.2662	155	<b>99.9</b>	0.1857
DS11	7129	79.22	0.3476	437	<b>97.4</b>	0.1597
DS12	22283	77.17	1.0112	415	<b>89.5</b>	0.18
DS13	22283	93.55	0.3074	95	<b>99.9</b>	0.1649
DS14	1413	99.08	0.2985	11	<b>100</b>	0.1683
DS15	10100	64.00	0.324	510	<b>87.7</b>	0.1576

## 4.5 The comparison between the SBCSO algorithm and the binary multi-objective optimization algorithms

In this section, the results of the multi-objective SBCSO algorithm procedure are examined. In subsection 3.4, a discussion has taken place regarding the objective functions and the multi-objective SBCSO algorithm procedure. The proposed algorithm's results have been compared with BNSGA-II, BMODE, BMOCSSO, BMOPSO, and BMOBBA algorithms. Given that the Pareto-optimal front related to the datasets is not available, the two  $\Delta$  and HV criteria were used for the results' assessment. These criteria are briefly explained as follows.

The hypervolume (HV) criterion is used to assess the convergence velocity of the proposed algorithm towards the optimal Pareto front. The HV indicates the dominated space value. A large amount of the HV demonstrates that the Pareto front level is closer to the optimal Pareto front.

$$HV = \text{volume}(\bigcup_{i=1}^{|PF|} \beta_i) \quad (15)$$

The spread metric ( $\Delta$ ) criterion is used for diversity assessment in the Pareto front level. A small amount of the  $\Delta$  indicates that the Pareto front level is more ordered.

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - \bar{d}|}{d_f + d_l + (N - 1) * \bar{d}} \quad (16)$$

The comparison results between two  $\Delta$  and HV criteria for 20-times independent execution of each algorithm are reported in Table 12-15. As indicated in Table 12-15, the SBCSO algorithm has obtained better results based on the HV criteria. Accordingly, it demonstrates that the SBCSO algorithm can obtain more non-dominated solutions rather than the other algorithms, not to mention that it has a higher convergence velocity. Regarding the  $\Delta$  criterion, the SBCSO algorithm has obtained better answers in DS05, DS06, DS08, DS09, DS10, DS11, DS12, DS13, DS14, and DS15. The BNSGA-II algorithm in the DS03 and DS04 datasets, the algorithm BMODE in the DS01 and DS07 datasets, and also the BMOPSO algorithm in the DS02 dataset have gained better answers based on the  $\Delta$  criterion.

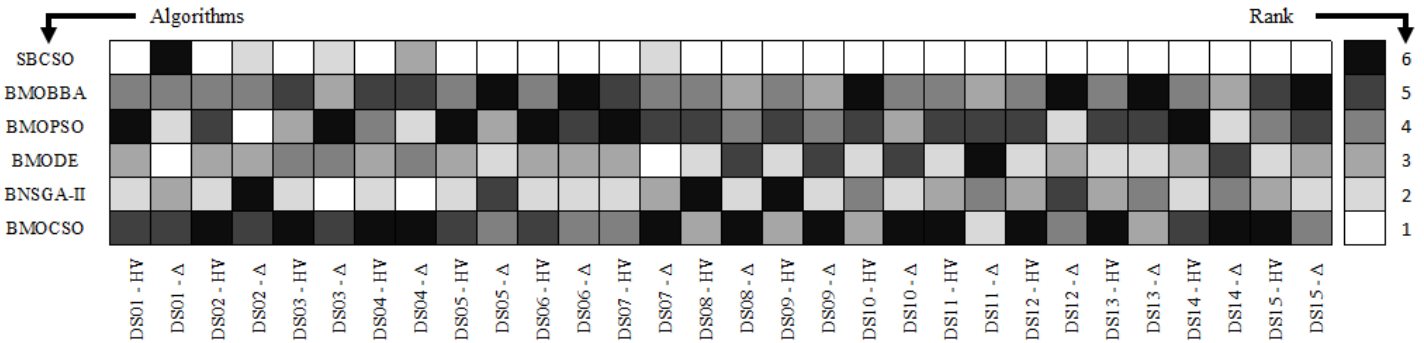
**Table 12** The result of comparing two  $\Delta$  and HV criteria for different algorithms.

DS#	Metric	BMOCSSO	BNSGA-II	BMODE	BMOPSO	BMOBBA	SBCSO
DS01	HV	1.81E-01 ± 2.92E-02	8.37E-01 ± 1.99E-02	4.49E-01 ± 8.17E-02	1.40E-01 ± 2.92E-02	2.86E-01 ± 4.34E-02	<b>8.86E-01 ± 1.40E-02</b>
	$\Delta$	9.96E-01 ± 1.55E-01	9.73E-01 ± 1.66E-01	<b>8.49E-01 ± 2.18E-01</b>	9.52E-01 ± 2.59E-01	9.92E-01 ± 5.19E-02	1.05E+00 ± 1.39E-01
DS02	HV	8.67E-02 ± 7.32E-02	6.07E-01 ± 3.78E-01	1.46E-01 ± 4.93E-02	8.78E-02 ± 4.42E-02	1.30E-01 ± 3.70E-02	<b>7.24E-01 ± 8.20E-02</b>
	$\Delta$	9.98E-01 ± 4.02E-01	1.05E+00 ± 2.43E-02	9.73E-01 ± 4.65E-01	<b>9.57E-01 ± 3.68E-01</b>	9.87E-01 ± 2.49E-01	9.64E-01 ± 2.93E-01
DS03	HV	1.03E-01 ± 3.95E-02	7.58E-01 ± 1.89E-02	1.99E-01 ± 7.24E-02	2.27E-01 ± 4.12E-02	1.51E-01 ± 2.93E-02	<b>8.56E-01 ± 4.23E-02</b>
	$\Delta$	9.96E-01 ± 1.26E-01	<b>9.36E-01 ± 2.35E-01</b>	9.93E-01 ± 4.82E-01	1.03E+00 ± 2.78E-01	9.87E-01 ± 2.65E-01	9.46E-01 ± 1.23E-01
DS04	HV	1.60E-01 ± 1.77E-02	6.37E-01 ± 6.15E-02	5.74E-01 ± 3.77E-01	3.43E-01 ± 1.83E-01	1.82E-01 ± 5.60E-02	<b>9.07E-01 ± 1.48E-02</b>
	$\Delta$	9.92E-01 ± 2.50E-01	<b>9.11E-01 ± 3.16E-01</b>	9.62E-01 ± 3.43E-01	9.13E-01 ± 2.04E-01	9.67E-01 ± 1.90E-01	9.27E-01 ± 4.94E-01
DS05	HV	1.71E-01 ± 2.88E-02	6.89E-01 ± 1.48E-01	2.26E-01 ± 7.57E-02	1.35E-01 ± 1.12E-02	1.86E-01 ± 1.34E-02	<b>8.19E-01 ± 2.35E-02</b>
	$\Delta$	1.00E+00 ± 2.85E-02	1.00E+00 ± 4.44E-01	9.62E-01 ± 4.58E-01	9.85E-01 ± 4.00E-01	1.07E+00 ± 5.84E-02	<b>9.11E-01 ± 1.38E-01</b>
DS06	HV	3.60E-01 ± 2.63E-01	8.13E-01 ± 6.85E-02	4.70E-01 ± 2.63E-01	3.40E-01 ± 1.86E-01	3.70E-01 ± 2.23E-01	<b>8.62E-01 ± 3.37E-02</b>
	$\Delta$	1.00E+00 ± 1.74E-01	9.45E-01 ± 3.43E-01	9.69E-01 ± 7.69E-02	1.03E+00 ± 3.63E-01	1.07E+00 ± 6.23E-02	<b>9.28E-01 ± 3.30E-01</b>
DS07	HV	3.32E-01 ± 3.00E-01	7.16E-01 ± 2.51E-02	4.32E-01 ± 2.78E-01	1.79E-01 ± 2.47E-02	1.94E-01 ± 3.95E-02	<b>8.32E-01 ± 6.00E-02</b>
	$\Delta$	1.06E+00 ± 2.52E-01	9.69E-01 ± 3.92E-01	<b>9.62E-01 ± 3.60E-01</b>	9.90E-01 ± 4.53E-01	9.84E-01 ± 4.47E-01	9.67E-01 ± 1.74E-01
DS08	HV	5.95E-01 ± 3.14E-01	4.62E-01 ± 4.16E-02	6.63E-01 ± 3.72E-01	4.83E-01 ± 3.13E-01	5.03E-01 ± 2.00E-01	<b>7.97E-01 ± 4.49E-02</b>

	$\Delta$	$1.40E+00 \pm 3.52E-01$	$9.97E-01 \pm 1.07E-01$	$1.33E+00 \pm 2.50E-02$	$1.26E+00 \pm 3.75E-01$	$1.19E+00 \pm 2.55E-01$	<b><math>8.93E-01 \pm 2.45E-01</math></b>
DS09	HV	$5.98E-01 \pm 1.84E-01$	$4.83E-01 \pm 1.29E-01$	$6.48E-01 \pm 2.08E-01$	$4.83E-01 \pm 2.09E-01$	$4.99E-01 \pm 3.29E-01$	<b><math>8.52E-01 \pm 7.36E-02</math></b>
	$\Delta$	$1.38E+00 \pm 4.53E-01$	$1.06E+00 \pm 3.09E-01$	$1.23E+00 \pm 3.13E-01$	$1.23E+00 \pm 4.31E-01$	$1.16E+00 \pm 4.05E-01$	<b><math>8.39E-01 \pm 2.93E-01</math></b>
DS10	HV	$6.61E-01 \pm 2.61E-01$	$5.39E-01 \pm 1.58E-01$	$7.02E-01 \pm 7.49E-02$	$5.33E-01 \pm 2.18E-01$	$5.17E-01 \pm 1.47E-01$	<b><math>8.53E-01 \pm 8.51E-02</math></b>
	$\Delta$	$1.39E+00 \pm 9.96E-02$	$9.91E-01 \pm 1.28E-01$	$1.31E+00 \pm 4.44E-01$	$1.06E+00 \pm 2.41E-02$	$1.13E+00 \pm 2.50E-01$	<b><math>9.26E-01 \pm 9.23E-02</math></b>
DS11	HV	$3.15E-01 \pm 1.52E-01$	$6.45E-01 \pm 2.25E-01$	$6.92E-01 \pm 2.53E-01$	$5.10E-01 \pm 2.39E-01$	$5.30E-01 \pm 9.10E-02$	<b><math>8.50E-01 \pm 3.41E-02</math></b>
	$\Delta$	$9.98E-01 \pm 4.90E-01$	$1.03E+00 \pm 3.59E-01$	$1.11E+00 \pm 2.55E-01$	$1.04E+00 \pm 2.41E-01$	$1.02E+00 \pm 3.92E-02$	<b><math>9.88E-01 \pm 3.44E-01</math></b>
DS12	HV	$1.91E-01 \pm 4.77E-02$	$4.89E-01 \pm 9.99E-02$	$6.37E-01 \pm 3.39E-01$	$4.49E-01 \pm 8.60E-02$	$4.72E-01 \pm 9.81E-02$	<b><math>7.69E-01 \pm 2.37E-02</math></b>
	$\Delta$	$9.99E-01 \pm 3.08E-02$	$1.03E+00 \pm 4.50E-02$	$9.80E-01 \pm 2.66E-01$	$9.41E-01 \pm 5.74E-02$	$1.10E+00 \pm 4.11E-01$	<b><math>9.11E-01 \pm 4.11E-01</math></b>
DS13	HV	$2.65E-01 \pm 2.82E-02$	$5.82E-01 \pm 1.80E-01$	$6.59E-01 \pm 1.31E-01$	$5.03E-01 \pm 3.70E-01$	$5.10E-01 \pm 1.78E-01$	<b><math>8.72E-01 \pm 2.48E-02</math></b>
	$\Delta$	$1.02E+00 \pm 3.64E-01$	$1.03E+00 \pm 8.34E-02$	$1.00E+00 \pm 3.33E-01$	$1.07E+00 \pm 2.64E-01$	$1.10E+00 \pm 4.87E-01$	<b><math>8.42E-01 \pm 3.28E-01</math></b>
DS14	HV	$5.99E-01 \pm 3.63E-01$	$7.88E-01 \pm 8.84E-02$	$7.53E-01 \pm 4.51E-02$	$5.54E-01 \pm 5.33E-02$	$6.18E-01 \pm 1.11E-01$	<b><math>9.70E-01 \pm 4.27E-02</math></b>
	$\Delta$	$1.30E+00 \pm 4.02E-01$	$1.13E+00 \pm 2.32E-01$	$1.16E+00 \pm 2.22E-01$	$1.03E+00 \pm 4.14E-01$	$1.08E+00 \pm 5.09E-02$	<b><math>1.01E+00 \pm 7.53E-02</math></b>
DS15	HV	$2.03E-01 \pm 5.76E-02$	$5.03E-01 \pm 1.12E-01$	$6.34E-01 \pm 2.45E-01$	$4.51E-01 \pm 2.87E-01$	$4.20E-01 \pm 9.65E-02$	<b><math>7.76E-01 \pm 1.94E-02</math></b>
	$\Delta$	$1.11E+00 \pm 9.50E-02$	$1.01E+00 \pm 2.02E-01$	$1.04E+00 \pm 4.17E-01$	$1.24E+00 \pm 4.04E-01$	$1.30E+00 \pm 3.96E-02$	<b><math>9.67E-01 \pm 2.06E-01</math></b>

**Table 13** Ranking the algorithms based on the average value of the HV and  $\Delta$ .

Metric	BMOCSO	BNSGA-II	BMODE	BMOPSO	BMOBBA	SBCSO
HV	5.00	2.93	2.60	5.07	4.40	<b>1.00</b>
$\Delta$	4.80	3.07	3.47	3.60	4.40	<b>1.67</b>
Overall Ranking	6	2	3	4	5	<b>1</b>
(Average ranking number)	(4.90)	(3.00)	(3.03)	(4.33)	(4.40)	<b>(1.33)</b>



**Fig. 25** Ranking the algorithms based on the average value of all metric values.

For further examination of Table 14, the results of statistical tests show the comparison of different algorithms. Table 14 shows that SBCSO is the best performing algorithm of the comparison with a rank of 1.33 and 1.12 for the Friedman and Quade tests, respectively.

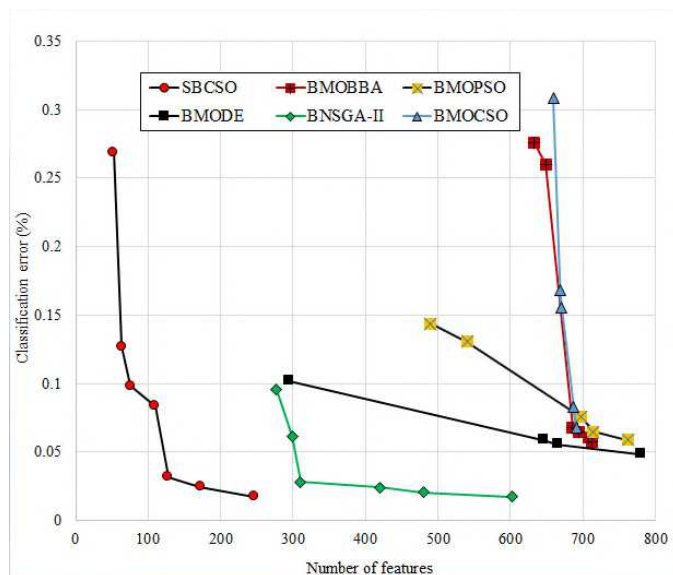
**Table 14** Friedman and Quade tests of all metric values.

algorithm	Friedman	Quade
SBCSO	<b>1.33</b>	<b>1.12</b>
BMOCSO	4.90	5.15
BNSGA-II	3.00	2.72
BMODE	3.03	3.02
BMOPSO	4.33	4.63
BMOBBA	4.40	4.36
statistic	73.94	27.67
p-value	9.18e-15	4.36e-05

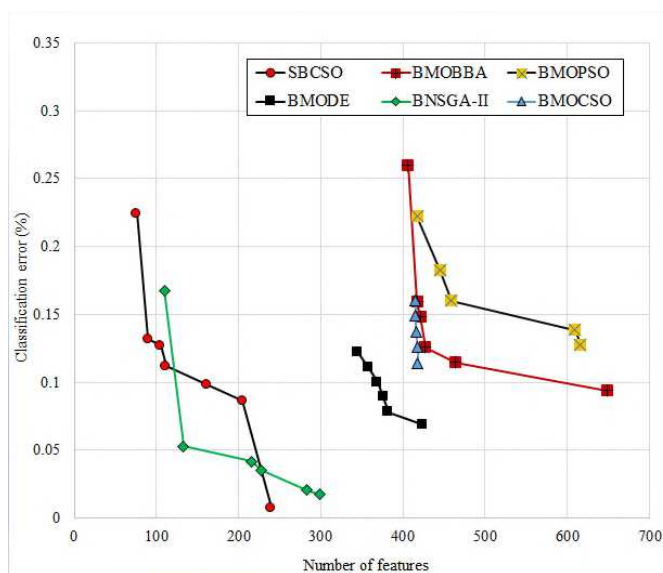
**Table 15** Wilcoxon test results.

Comparison	R+	R-	p-value	level of significance
SBCSO versus BMOCSO	449	16	2.6E-06	$\alpha = 0.01$
SBCSO versus BNSGA-II	412	53	7.69E-06	$\alpha = 0.01$
SBCSO versus BMODE	427	38	1.02E-05	$\alpha = 0.01$
SBCSO versus BMOPSO	413	52	5.75E-06	$\alpha = 0.01$
SBCSO versus BMOBBA	449	16	3.18E-06	$\alpha = 0.01$

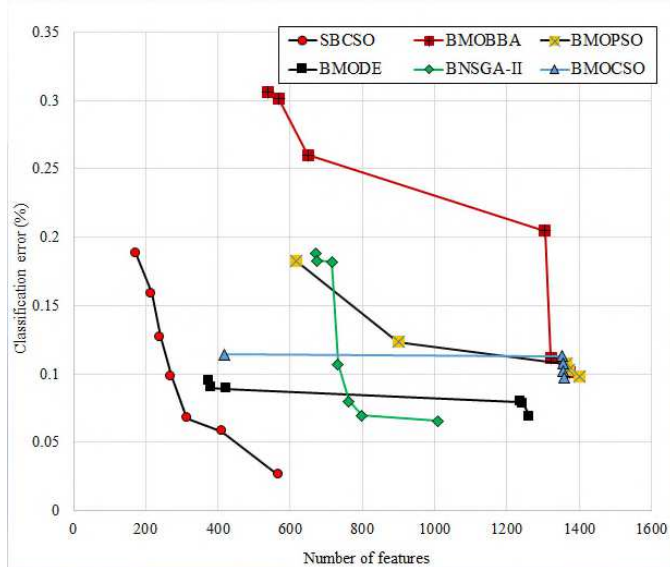
Figures 26-31 demonstrate the Pareto front diagram obtained by different algorithms. In these figures, the vertical axis represents the classification error, and the horizontal axis represents the number of the selected features for each dataset. As indicated in the figures, the SBCSO algorithm has a faster convergence to the POF non-dominated solutions (at the same time compared to other algorithms). The multi-objective SBCSO algorithm simultaneously minimizes the number of selected features and the classification error.



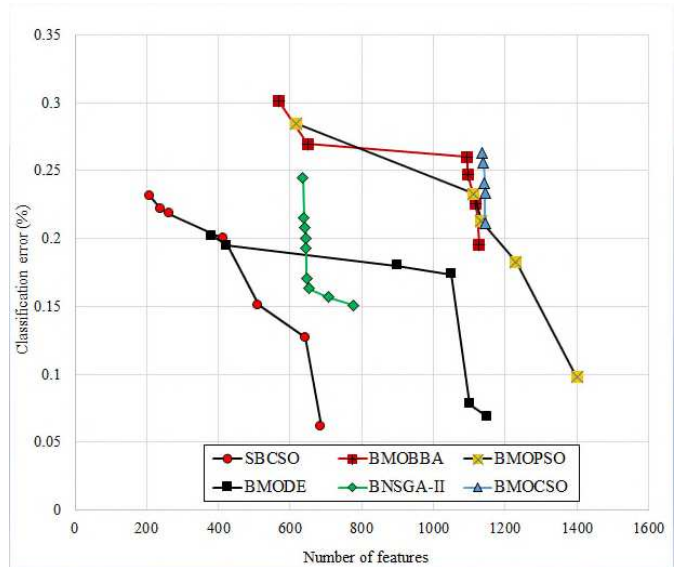
**Fig. 26** Pareto front of non-dominated solutions for the DS04 dataset.



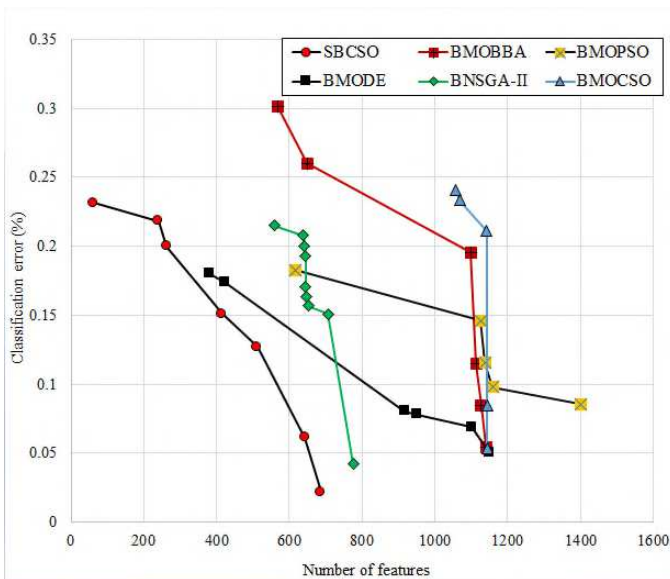
**Fig. 27** Pareto front of non-dominated solutions for the DS06 dataset.



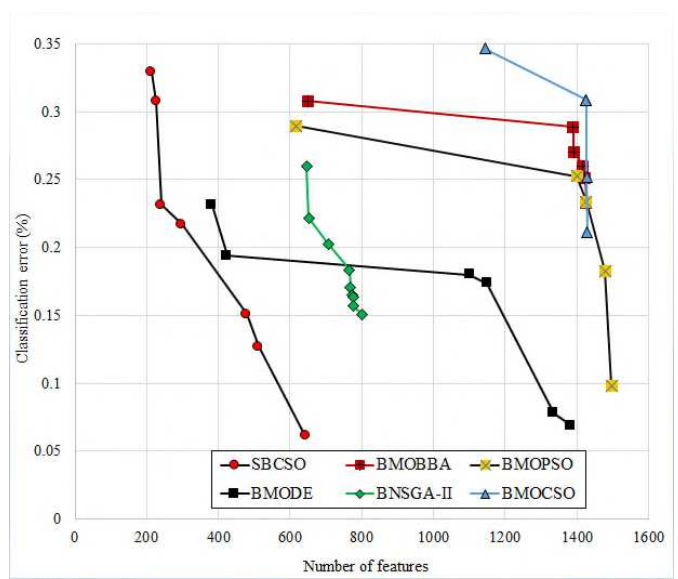
**Fig. 28** Pareto front of non-dominated solutions for the DS10 dataset.



**Fig. 29** Pareto front of non-dominated solutions for the DS12 dataset.



**Fig. 30** Pareto front of non-dominated solutions for the DS13 dataset.



**Fig. 31** Pareto front of non-dominated solutions for the DS15 dataset.

## 5. Conclusion

This article proposes a new version of cat binary optimization algorithms, titled SBCSO, in order to select genes at DNA microarray expression cancer data. The high number of genes against the low number of samples has always been a challenge to microarray technology. The proposed algorithm consists of 4 main sections. In the first section, an opposition-based learning (OBL) mechanism is employed for population members' diversity. In the second section, a new time-varying V-shape transfer function that varies by time is employed. In the third section, the  $MR$  and  $\lambda$  of the proposed algorithm are adapted over time. And in the fourth section, a single-objective and multi-objective approach is proposed in order to solve the gene selection problem. The fifteen datasets pertinent to the microarray data of different types of cancer have been employed in order to compare the proposed method with other ubiquitous methods. The results of the experiments indicate that the proposed algorithm is highly capable in selecting optimal gene sets required for the faster diagnosis of a vast majority of diseases.

## **Compliance with Ethical Standards**

### **A. AUTHORSHIP RESPONSIBILITY**

I certify that ALL of the following statements are correct

- ❖ The manuscript represents valid work; neither this manuscript nor one with substantially similar content under my authorship has been published or is being considered for publication elsewhere (except as described in the manuscript submission); and copies of any closely related manuscripts are enclosed in the manuscript submission; **AND**
- ❖ For manuscripts with more than one author, I agree to allow the corresponding author to serve as the primary correspondent with the editorial office and to review and sign off on the final proofs prior to publication; or, if I am the only author, I will be the corresponding author and agree to serve in the roles described above.
- ❖ For manuscripts that are a report of a study, I confirm that this work is an accurate representation of the trial results.

### **B. AUTHORSHIP CRITERIA**

To fulfill all of the criteria for authorship, every author of the manuscript must have made substantial contributions to ALL of the following aspects of the work:

- ❖ Conception and planning of the work that led to the manuscript or acquisition, analysis and interpretation of the data, or both; **AND**
- ❖ Drafting and/or critical revision of the manuscript for important intellectual content; **AND**
- ❖ Approval of the final submitted version of the manuscript.

I certify that I fulfill ALL of the above criteria for authorship.

### **C. AUTHORSHIP CONTRIBUTION**

I certify that I have participated sufficiently in the work to take public responsibility for the entire content of the manuscript.

### **D. FUNDING DISCLOSURES**

I certify that no funding has been received for the conduct of this study and/or preparation of this manuscript.

### **E. CONTRIBUTOR DISCLOSURES**

I certify that all persons who have made substantial contributions to this manuscript but who do not fulfill the authorship criteria are listed with their specific contributions in the Acknowledgments section in the manuscript, and that all persons named in the Acknowledgments section have given me written permission to be named in the manuscript.

### **F. CONFLICT OF INTEREST DISCLOSURES**

I have no conflicts of interest to declare



## References

- [1] Schena, M., Shalon, D., Heller, R., Chai, A., Brown, P.O. and Davis, R.W., 1996. Parallel human genome analysis: microarray-based expression monitoring of 1000 genes. *Proceedings of the National Academy of Sciences*, 93(20), pp.10614-10619.
- [2] Venkataramana, L., Jacob, S.G., Ramadoss, R., Saisuma, D., Haritha, D. and Manoja, K., 2019. Improving classification accuracy of cancer types using parallel hybrid feature selection on microarray gene expression data. *Genes & genomics*, 41(11), pp.1301-1313.
- [3] Tabares-Soto, R., Orozco-Arias, S., Romero-Cano, V., Bucheli, V.S., Rodríguez-Sotelo, J.L. and Jiménez-Varón, C.F., 2020. A comparative study of machine learning and deep learning algorithms to classify cancer types based on microarray gene expression data. *PeerJ Computer Science*, 6, p.e270.
- [4] Glazier AM. Finding Genes That Underlie Complex Traits. *Science*. 2002; 298(5602): 2345- 9.
- [5] Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A. and Bloomfield, C.D., 1999. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439), pp.531-537.
- [6] Chandrashekar G, Sahin F (2014) A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1), pp.16-28
- [7] Tubishat, M., Idris, N., Shuib, L., Abushariah, M.A. and Mirjalili, S., 2020. Improved Salp Swarm Algorithm based on opposition based learning and novel local search algorithm for feature selection. *Expert Systems with Applications*, 145, p.113122.
- [8] Kalaimani, V. and Umagandhi, R., 2020. A novel wrapper FS based on binary swallow swarm optimization with score-based criteria fusion for gene expression microarray data. *Materials Today: Proceedings*.
- [9] Agrawal, R.K., Kaur, B. and Sharma, S., 2020. Quantum based Whale Optimization Algorithm for wrapper feature selection. *Applied Soft Computing*, p.106092.
- [10] Shukla, A.K., Singh, P. and Vardhan, M., 2019. A new hybrid wrapper TLBO and SA with SVM approach for gene expression data. *Information Sciences*, 503, pp.238-254.
- [11] Hambali, M.A., Oladele, T.O. and Adewole, K.S., 2020. Microarray Cancer Feature Selection: Review, Challenges and Research Directions. *International Journal of Cognitive Computing in Engineering*
- [12] Lazar, C., Taminau, J., Meganck, S., Steenhoff, D., Coletta, A., Molter, C., de Schaezen, V., Duque, R., Bersini, H. and Nowe, A., 2012. A survey on filter techniques for feature selection in gene expression microarray analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9(4), pp.1106-1119.
- [13] Ghosh, M., Guha, R., Sarkar, R. and Abraham, A., 2019. A wrapper-filter feature selection technique based on ant colony optimization. *Neural Computing and Applications*, pp.1-19.
- [14] Yang, C.H., Chuang, L.Y. and Yang, C.H., 2010. IG-GA: a hybrid filter/wrapper method for feature selection of microarray data. *Journal of Medical and Biological Engineering*, 30(1), pp.23-28.
- [15] Apolloni, J., Leguizamón, G. and Alba, E., 2016. Two hybrid wrapper-filter feature selection algorithms applied to high-dimensional microarray experiments. *Applied Soft Computing*, 38, pp.922-932.
- [16] Almgren, N. and Alshamlan, H., 2019. A survey on hybrid feature selection methods in microarray gene expression data for cancer classification. *IEEE Access*, 7, pp.78533-78548.
- [17] Faris, H., Aljarah, I., Al-Betar, M.A. and Mirjalili, S., 2018. Grey wolf optimizer: a review of recent variants and applications. *Neural computing and applications*
- [18] Wang, D., Tan, D. and Liu, L., 2018. Particle swarm optimization algorithm: an overview. *Soft Computing*, 22(2), pp.387-408.
- [19] Kumar, Y. and Singh, P.K., 2018. Improved cat swarm optimization algorithm for solving global optimization problems and its application to clustering. *Applied Intelligence*, 48(9), pp.2681-2697.
- [20] Mahdavi, S., Rahnamayan, S. and Deb, K., 2018. Opposition based learning: A literature review. *Swarm and evolutionary computation*, 39, pp.1-23.

- [21] Chuang, L.Y., Chang, H.W., Tu, C.J. and Yang, C.H., 2008. Improved binary PSO for feature selection using gene expression data. *Computational Biology and Chemistry*, 32(1), pp.29-38.
- [22] Hu, P., Pan, J.S. and Chu, S.C., 2020. Improved Binary Grey Wolf Optimizer and Its application for feature selection. *Knowledge-Based Systems*, p.105746.
- [23] Islam, M.J., Li, X. and Mei, Y., 2017. A time-varying transfer function for balancing the exploration and exploitation ability of a binary PSO. *Applied Soft Computing*, 59, pp.182-196.
- [24] Mafarja M, Aljarah I, Heidari AA, Faris H, Fournier-Viger P, Li X, Mirjalili S (2018) Binary dragonfly optimization for feature selection using time-varying transfer functions. *Knowledge-Based Systems*, 161, pp.185-204
- [25] Beheshti Z (2020) A time-varying mirrored S-shaped transfer function for binary particle swarm optimization. *Information Sciences*, 512, pp.1503-1542
- [26] Kahya, M.A., Altamir, S.A. and Algamal, Z.Y., 2020. Improving whale optimization algorithm for feature selection with a time-varying transfer function. *Numerical Algebra, Control & Optimization*, 11(1), p.87.
- [27] Chu, S.C. and Tsai, P.W., 2007. Computational intelligence based on the behavior of cats. *International Journal of Innovative Computing, Information and Control*, 3(1), pp.163-173.
- [28] Kumar, Y. and Singh, P.K., 2018. Improved cat swarm optimization algorithm for solving global optimization problems and its application to clustering. *Applied Intelligence*, 48(9), pp.2681-2697.
- [29] L. Pappula and D. Ghosh, "Cat swarm optimization with normal mutation for fast convergence of multimodal functions," *Applied Soft Computing*, vol. 66, pp. 473–491, 2018.
- [30] Abd Elaziz, M. and Mirjalili, S., 2019. A hyper-heuristic for improving the initial population of whale optimization algorithm. *Knowledge-Based Systems*, 172, pp.42-63.
- [31] Zhao, X., Yang, F., Han, Y. and Cui, Y., 2020. An Opposition-Based Chaotic Salp Swarm Algorithm for Global Optimization. *IEEE Access*, 8, pp.36485-36501.
- [32] Rojas-Morales, N., Rojas, M.C.R. and Ureta, E.M., 2017. A survey and classification of opposition-based metaheuristics. *Computers & Industrial Engineering*, 110, pp.424-435.
- [33] Dhargupta, S., Ghosh, M., Mirjalili, S. and Sarkar, R., 2020. Selective opposition based grey wolf optimization. *Expert Systems with Applications*, p.113389.
- [34] Kennedy J, Eberhart RC (1997) A discrete binary version of the particle swarm algorithm, in: *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation.*, 1997 IEEE International Conference on, IEEE, 1997, pp. 4104-4108
- [35] Rashedi E, Nezamabadi-Pour H, Saryazdi S (2010) BGSA: binary gravitational search algorithm. *Natural Computing*, 9(3), pp.727-745
- [36] Mirjalili S, Lewis A (2013) S-shaped versus V-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation*, 9, pp.1-14
- [37] Sharafi, Y., Khanesar, M.A. and Teshnehlab, M., 2013, September. Discrete binary cat swarm optimization algorithm. In *2013 3rd IEEE International Conference on Computer, Control and Communication (IC4)* (pp. 1-6). IEEE.
- [38] Siqueira, H., Figueiredo, E., Macedo, M., Santana, C.J., Bastos-Filho, C.J. and Gokhale, A.A., 2018, November. Boolean binary cat swarm optimization algorithm. In *2018 IEEE Latin American Conference on Computational Intelligence (LA-CCI)* (pp. 1-6). IEEE.
- [39] Siqueira, H., Santana, C., Macedo, M., Figueiredo, E., Gokhale, A. and Bastos-Filho, C., 2020. Simplified binary cat swarm optimization. *Integrated Computer-Aided Engineering*, (Preprint), pp.1-15.
- [40] Haznedar, Bulent; Arslan, Mustafa Turan; KALINLI, Adem (2017), "Microarray Gene Expression Cancer Data", *Mendeley Data*, V2, doi: 10.17632/ymp2tst2hh.2
- [41] Zexuan Zhu, Y. S. Ong and M. Dash, "Markov Blanket-Embedded Genetic Algorithm for Gene Selection", *Pattern Recognition*, Vol. 49, No. 11, 3236-3248, 2007." if you use the datasets.
- [42] Lu, J., Zhao, T. and Zhang, Y., 2008. Feature selection based-on genetic algorithm for image annotation. *Knowledge-Based Systems*, 21(8), pp.887-891.

- [43] Zhang, Y., Gong, D.W., Gao, X.Z., Tian, T. and Sun, X.Y., 2020. Binary differential evolution with self-learning for multi-objective feature selection. *Information Sciences*, 507, pp.67-85.
- [44] Annavarapu, C.S.R., Dara, S. and Banka, H., 2016. Cancer microarray data feature selection using multi-objective binary particle swarm optimization algorithm. *EXCLI journal*, 15, p.460.
- [45] Soyel, H., Tekguc, U. and Demirel, H., 2011. Application of NSGA-II to feature selection for facial expression recognition. *Computers & Electrical Engineering*, 37(6), pp.1232-1240.
- [46] Gao, X.Z., Nalluri, M.S.R., Kannan, K. and Sinharoy, D., 2021. Multi-objective optimization of feature selection using hybrid cat swarm optimization. *Science China Technological Sciences*, 64(3), pp.508-520.
- [47] Rodrigues, D., Pereira, L.A., Nakamura, R.Y., Costa, K.A., Yang, X.S., Souza, A.N. and Papa, J.P., 2014. A wrapper approach for feature selection based on bat algorithm and optimum-path forest. *Expert Systems with Applications*, 41(5), pp.2250-2258.
- [48] Carrasco, J., García, S., Rueda, M.M., Das, S. and Herrera, F., 2020. Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review. *Swarm and Evolutionary Computation*, 54, p.100665.
- [49] Derrac, J., García, S., Molina, D. and Herrera, F., 2011. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1), pp.3-18.
- [50] Tubishat, M., Idris, N., Shuib, L., Abushariah, M.A. and Mirjalili, S., 2020. Improved Salp Swarm Algorithm based on opposition based learning and novel local search algorithm for feature selection. *Expert Systems with Applications*, 145, p.113122.
- [51] Ewees, A.A., Abd Elaziz, M. and Oliva, D., 2021. A new multi-objective optimization algorithm combined with opposition-based learning. *Expert Systems with Applications*, 165, p.113844.
- [52] Yu, X., Xu, W. and Li, C., 2021. Opposition-based learning grey wolf optimizer for global optimization. *Knowledge-Based Systems*, p.107139.