Applied Intelligence
https://doi.org/10.1007/s10489-022-04201-z

Opposition-based sine cosine optimizer utilizing refraction learning and variable neighborhood search for feature selection

Bilal H. Abed-alguni[1] · Noor Aldeen Alawad[1] · Mohammed Azmi Al-Betar[2] · David Paul[3]

Accepted: 21 September 2022

Abstract
This paper proposes new improved binary versions of the Sine Cosine Algorithm (SCA) for the Feature Selection (FS) problem. FS is an essential machine learning and data mining task of choosing a subset of highly discriminating features from noisy, irrelevant, high-dimensional, and redundant features to best represent a dataset. SCA is a recent metaheuristic algorithm established to emulate a model based on sine and cosine trigonometric functions. It was initially proposed to tackle problems in the continuous domain. The SCA has been modified to Binary SCA (BSCA) to deal with the binary domain of the FS problem. To improve the performance of BSCA, three accumulative improved variations are proposed (i.e., IBSCA1, IBSCA2, and IBSCA3) where the last version has the best performance. IBSCA1 employs Opposition Based Learning (OBL) to help ensure a diverse population of candidate solutions. IBSCA2 improves IBSCA1 by adding Variable Neighborhood Search (VNS) and Laplace distribution to support several mutation methods. IBSCA3 improves IBSCA2 by optimizing the best candidate solution using Refraction Learning (RL), a novel OBL approach based on light refraction. For performance evaluation, 19 real-wold datasets, including a COVID-19 dataset, were selected with different numbers of features, classes, and instances. Three performance measurements have been used to test the IBSCA versions: classification accuracy, number of features, and fitness values. Furthermore, the performance of the last variation of IBSCA3 is compared against 28 existing popular algorithms. Interestingly, IBCSA3 outperformed almost all comparative methods in terms of classification accuracy and fitness values. At the same time, it was ranked 15 out of 19 in terms of number of features. The overall simulation and statistical results indicate that IBSCA3 performs better than the other algorithms.

Keywords Sine cosine algorithm · Refraction learning · Opposition-based learning · Mutation methods · Laplace distribution · Feature selection

1 Introduction

Back in 2003, the amount of generated data was around five exabytes. Nowadays, the same amount of data, and even more, is produced within two days [1]. This rapid increase in the volume, velocity and variety of data raises challenges and, at the same time, opportunities. Dealing with such data is a challenge, but there are opportunities to utilize the data for beneficial applications [2].

In order to perform data mining, data are first pre-processed [3], which involves cleaning and preparing the data to best meet the requirements of input for later stages. One possible pre-processing step is Feature Selection (FS) [3], which is a method of choosing a subset of features of a dataset that can best represent the data accurately without redundancy, noise, or repetition. FS is used in a wide number of applications, including data classification [4–6], data clustering [7–9], image processing [10–13], and text categorization [14, 15].

Generally speaking, FS techniques are either based on an evaluation criterion or on a search strategy. Evaluation criterion-based methods can be further classified as either filters or wrappers. The main difference between these two is the absence or existence (respectively) of a learning algorithm in the process to evaluate feature subsets. Chi-Square [16], Gain Ratio [17], Information Gain [18], support vector machines [19], ReliefF [20, 21], and hybrid ReliefF [22, 23] are filter methods. They depend upon correlations between features and classes in the dataset.

✉ Bilal H. Abed-alguni
Bilal.h@yu.edu.jo

Extended author information available on the last page of the article.

Wrapper FS methods [24], on the other hand, utilize learning algorithms. A disadvantage of wrapper FS methods is the high computational cost, however they often give precise results.

Due to the huge search space, the FS problem has been shown to be NP-Hard [25, 26]. Thus, it is costly and time-consuming to employ exact methods to find a solution. However, when searching for approximate solutions, randomization searching strategies, such as sequential forward, sequential backward, random, and heuristic [27], often enhance results. Further, metaheuristic algorithms often lead to efficient implementations of various FS methods.

Metaheuristic algorithms use heuristic strategies or guidelines in optimization algorithms to solve complex optimization problems (e.g., FS problem) in real time. Unlike single-purpose algorithms, metaheuristic algorithms can be used for many different optimization problems [27–33]. One major category of metaheuristic algorithms is Swarm Intelligence (SI), where creature swarms are the main inspiration (e.g., ants, flocks, bees) [34]. SI algorithms have been tested with various optimization problems, including FS. For instance, the authors of [35] utilize the powerful SI algorithm Grey Wolf Optimizer (GWO) with an FS problem, and the results reported a respectable performance. Similarly, the Antlion Optimizer (ALO) [36] has been successfully used as a wrapper for a FS strategy, and the Whale Optimization Algorithm (WOA) has been utilised in several different implementations of FS algorithms [37–40], as has Particle Swarm Optimization (PSO) [41], Artificial Bee Colony (ABC) [42], Ant Colony Optimization (ACO) [43, 44], Gravitational Search Algorithm [45], and the Salp Swarm Algorithm (SSA) [46–48].

Indeed, the hardness of tackling the FS problem is considerably increased with an increase of the original problem's dimensions. For instance, when the FS data has n features, its search space has 2^n different solutions. Thus, any metaheuristic algorithm used to tackle such an FS problem often requires modification to work well given the complex nature of the FS search space. This is also mentioned in the No Free Lunch (NFL) theorem [49], which states that no superior algorithm can achieve the best performance for all optimization problems or even for the same optimization problem with different instances. Therefore, research opportunities are still available to introduce new/modified metaheuristic algorithms for FS problems.

Besides the previously mentioned SI algorithms, metaheuristics algorithms can imitate a physical rule, evolutionary phenomena, or human-based technique [50]. To this end, Seyedali Mirjalili proposed a metaheuristic algorithm called the Sine Cosine Algorithm (SCA) [50] in 2016. SCA is a population-based algorithm inspired by the sine and cosine trigonometric functions. The simplicity, robustness and efficiency of the algorithm are SCA's main advantages. Those characteristics have motivated others to implement SCA for different optimization problems. For example, truss structure optimization is an architecture-based optimization problem [51] where SCA has been applied. SCA has also been adapted to support the travelling salesperson problem [52], text categorization [53], image segmentation [54], object tracking [55], unit commitment [56], optimal design of a shell and tube evaporator [57], abrupt motion tracking [58], and parameter optimization for support vector regression [59].

Because real-world problems are complex and have constraints, researchers have attempted to enhance SCA in a number of different ways. Firstly, SCA operators have been modified to deal with particular problems [60–63]. Alternatively, SCA has been hybridized with i) local-based algorithms [52, 64, 65], ii) population-based algorithms [66, 67], iii) operators from other optimization algorithms [65, 68]. For instance, in [62] the SCA exploration and exploitation phases were managed by a nonlinear conversion parameter. In addition, to help avoid local optima, the position update equation was modified. Another example of SCA hybridization is improving exploitation utilizing the Nelder-Mead simplex concept and the Opposition-based learning (OBL) searching strategy [64]. Further, the diversification of SCA has been enhanced by integrating SCA with a random mutation and gaussian local search technique [65]. Quite recently, Al-betar et al. [69] introduced a memetic version of SCA to solve the economic load dispatch problem. In this approach, adaptive β-hill climbing [70] was hybridized with the optimization framework of SCA to better balance exploration and exploitation.

SCA was initially proposed for continuous decision variables. However, with a mapping function (transforming a continuous search space to binary), a binary SCA (BSCA) version was introduced in [71], where it was implemented for an FS optimization problem, and verified to be an efficient technique. The performance, accuracy, capability, and variety of decision variables' types are the factors that motivated us to conduct the research described in this paper. We propose three versions of the Improved Binary Sine Cosine Algorithm (i.e., IBSCA1, IBSCA2, and IBSCA3) for the FS problem, in which different approaches of exploration and exploitation are conducted. Consequently, this leads to the following contributions:

- We apply Opposition Based Learning (OBL) in IBSCA1 to ensure a diverse population of solutions. The use of OBL is expected to expand the search region and improve the solution's approximation.

- IBSCA2 builds on IBSCA1 and includes Variable Neighborhood Search (VNS) and Laplace distribution to explore the search space using several mutation methods (swap, insert, inverse, or random mutation). One of the advantages of VNS is that the mutated solution may break out of a local optimum.
- IBSCA3 builds on IBSCA2 and enhances the best candidate solution using Refraction Learning (RL). RL is a novel opposition learning approach that is based on the principle of light refraction. It is expected to improve the ability of IBSCA3 to jump out of local optima.
- The three exploration techniques are applied in an incremental manner, where IBSCA3 implements all of the three exploration techniques. Our purpose here is to show that the incremental integration of each exploration method gradually improves the performance of IBSCA and eventually leads to a strong optimization algorithm (IBSCA3).
- The candidate solutions produced by the optimization process of SCA and RL are continuous. Therefore, we used the V3 transfer function to convert the values of continuous decision variables into binary ones. V3 was selected based on extensive simulations on eight binary transfer functions (4 S-shaped and 4 V-shaped transfer functions). The experimental results indicated that V3 is the most viable transfer function.
- We evaluate the variations of IBSCA utilizing 19 well-known datasets (18 FS datasets from UCI repository and a COVID-19 dataset). IBSCA3 is found to be the most efficient version of IBSCA (Section 5.2).
- The performance of IBSCA3 was evaluated and compared to 10 popular binary algorithms (Section 5.3). The overall simulation results indicate that IBSCA3 outperformed all the compared algorithms in terms of accuracy and number of features selected over most of the datasets.
- We compared IBSCA3 to 10 state-of-the-art algorithms that adopt OBL-enhanced methods, VNS and Laplace distribution (Section 5.4). We found that IBSCA3 produces the best results among the results of the compared algorithms.
- We compared IBSCA3 to seven popular variations of SCA (Section 5.5). The experimental results indicate that IBSCA3 is the most accurate algorithm.

The accumulative advantages proposed for IBSCA are included in IBSCA3 where the method has the ability to diversify the search through Opposition Based Learning (OBL) and intensify the search through Variable Neighborhood Search (VNS) while also having the ability to escape local optima through Refraction Learning (RL). By means of these improvements, a superior method (i.e., IBSCA3) is introduced for the FS problem.

In general, the overall simulation results indicate that IBSCA3 outperforms the compared algorithms, based on accuracy and number of features selected, over almost all tested datasets. Note that there are two main differences between IBSCA3 and the other hybrid optimization algorithms that attempt to solve the FS problem. First, IBSCA3 is the only hybrid algorithm that combines OBL, RL, VNS and Laplace distribution in a single algorithm. Second, IBSCA3 is the first such algorithm to include Laplace distribution inside VNS.

The rest of the paper is organized as follows: SCA optimization problem implementations and versions are highlighted in Section 2. Section 3 then reviews the binary Sine Cosine algorithm and the objective function used. The newly proposed Improved Binary SCA with multiple exploration and exploitation approaches (IBSCA) for solving the FS problem is presented in Section 4. For the purpose of evaluation, the algorithms' performances over different experiments are compared and discussed in Section 5. Lastly, Section 6 summarises the work and presents potential future research avenues.

2 Related work

Several discrete variations of SCA have been developed to solve the FS problem [48, 61, 72–78]. This section examines recently proposed variations of the SCA for global optimization and solving the FS problem.

El-kenawy and Ibrahim [72] introduced a binary hybrid optimization algorithm (Binary SC-MWOA) that includes the SCA algorithm and a modified Whale Optimization algorithm. Binary SC-MWOA converts the continuous candidate solutions generated by the optimization operators of the SC and whale optimization algorithms into binary discrete solutions that can be used for the FS problem using the sigmoid function. Binary SC-MWOA was evaluated over 10 UCI repository datasets and compared to a number of popular optimization algorithms including the Grey Wolf Optimizer (GWO) [79], Whale Optimization Algorithms (WOA) [80] and memetic firefly algorithm. The Binary SC-MWOA was able to find an optimum subset of features with the best category error.

Neggaz et al. [48] presented a new hybrid optimization algorithm for FS called ISSAFD that combines the optimization operators of the SC algorithm and the Disrupt Operator of the Salp Swarm Optimizer (SSA). ISSAFD optimizes followers' positions in the SSA algorithm using sinusoidal mathematical functions similar to those in SCA operators. The disrupt operator diversifies the population of candidate solutions in the algorithm. The performance of ISSAFD was compared to many optimization algorithms including SSA, SCA, binary GWO (bGWO), PSO, ALO, and Genetic Algorithm (GA) over four well-known datasets. The simulation results suggested that ISSAFD was more accurate, had higher sensitivity, and chose fewer features than the other tested FS algorithms.

Hussain et al. [73] suggested an algorithm to solve continuous optimization problems and the FS problem called SCHHO that integrates the SCA algorithm in the Harris Hawks Optimization (HHO) algorithm. The goal of SCHHO is to use SCA as an exploration method in HHO. In addition, the exploitation ability of HHO is improved in SCHHO by having candidate solutions adjust dynamically to help avoid staying in local optima. As reported in [73], SCHHO performs much better than popular optimization algorithms, including Dragonfly algorithm (DA), grasshopper optimization algorithm (GOA), GWO, WOA, and SSA.

The wrapper-based Improved SCA (ISCA) [61] adds an Elitism strategy to SCA as well as a mechanism to update the best solution. The experimental results in [61] suggest that ISCA provides more accurate results and fewer features than GA, PSO and the original SCA algorithm.

Abd Elaziz et al., [74] proposed SCADE, an algorithm that combines the differential evolution (DE) algorithm with the SCA algorithm. DE's optimization operators are used at each iteration of SCA to improve its population of solutions. This helps the SCA algorithm avoid local optima. SCADE's performance was assessed over eight UCI datasets with comparison to three popular algorithms (social spider optimization (SSO), ABC and ACO [74]), with SCADE obtaining the best results.

Abualigah and Dulaimi [75] introduced the hybrid SCA and GA algorithm (SCAGA) for solving the FS problem. In SCAGA, the genetic optimization operators (crossover and mutation) are used to improve the optimization process of SCA and balance between its exploration and exploitation of candidate solutions. SCAGA was compared to SCA, PSO, and ALO using 16 UCI datasets. SCAGA was found to be a better feature-selection method than the other tested algorithms in terms of the maximum obtained accuracy and minimal obtained features.

Sindhu et al., [77] proposed an algorithm named Improved Biogeography Based Optimization (IBBO) for solving the FS problem. IBBO attempts to improve the optimization process of Biogeography Based Optimization (BBO) by employing the optimization operators of SCA after the migration operator of BBO. The performance of IBBO was compared to the performance of popular optimization algorithms such as BBO, SCA, GA, PSO, and ABC using four popular datasets. The simulation results suggest that IBBO is more accurate and selects fewer features compared to the other FS algorithms.

SCA may get stuck in sub-optimal regions during its optimization process. This is because its exploration operators (i.e., the two trigonometric functions of SCA) are unable to efficiently explore the search space. Abd Elaziz et al., [76] proposed Opposition-based SCA (OBSCA), which is a variation of SCA that uses the OBL technique to improve the performance of SCA. In OBSCA, OBL selects the best candidate solutions and generates their opposite solutions in an attempt to lead to more accurate solutions. OBSCA was compared in [76] to several optimization algorithms including SCA, Harmony Search (HS), GA, and PSO using standard optimization test functions and real-world engineering problems. OBSCA performed competitively compared to the other algorithms.

Kumar and Bharti [78] proposed the Hybrid Binary PSO and SCA algorithm (HBPSOSCA). In this algorithm, a V-shaped transfer function converts continuous candidate solutions into binary solutions. The effectiveness of HBPSOSCA was compared in [78] to binary PSO, modified BPSO with chaotic inertia weight, binary moth flame optimization algorithm, binary DA, binary WOA, binary SCA, and binary ABC using 10 standard benchmark functions and seven real-world datasets. The conducted experiments showed that HBPSOSCA exhibited better performance in most of the tested cases.

ASOSCA [81] is a hybrid optimization algorithm based on the Atom Search Optimization (ASO) algorithm and the SCA algorithm. It is basically used for automatic clustering. In ASOSCA, SCA is used to improve the quality of candidate solutions (i.e., reduce the number of features and improve accuracy of the solutions) in ASO. The performance of ASOSCA was compared in [81] to other optimization methods (e.g., SCA, ASO, PSO) using 16 clustering datasets and different cluster validity indexes. ASOSCA performed better than the other tested algorithms.

The Artificial Algae Algorithm (AAA) is a metaheuristic for solving continuous optimization problems [82]. It was originally inspired by the living behaviors of microalgae, photosynthetic specie. Turkoglu et al. [83] proposed eight binary versions of the AAA algorithm for solving the FS problem. Each binary version of AAA uses a different transfer function (four V-shaped and four S-shaped transfer functions). The performance of the binary versions of AAA was compared to the performance of seven well-known optimization algorithms (BBA, binary CS, binary Firefly algorithm, binary GWO, binary Moth flame algorithm, binary PSO, binary WOA [83]) using the UCI datasets. The experimental results indicate that the binary versions of AAA outperform the other tested algorithms.

The Horse herd Optimization Algorithm (HOA) is a metaheuristic that simulates the survival behaviour of a pack of horses in solving NP-hard optimization problems [84]. Awadallah et al. [85] proposed fifteen binary versions of HOA (BHOA) for solving the FS problems. The fifteen variations of BHOA were created by combining three popular crossover operators (one-point, two-point and uniform operators) with three transfer-functions categories (S-shaped, V-shaped and U-shaped transfer functions). The versions of BHOA were tested and evaluated against each other using 24 real-world datasets and the experimental findings suggest that the best version of BHOA is the one with S-shape and one-point crossover.

The Black Widow Optimization (BWO) algorithm is a new population-based optimization algorithm that mimics the mating process of black-widow spiders to solve the continuous optimization problems [86]. However, the BWO algorithm converges slowly to solutions when attempting to solve hard optimization problems. Therefore, the enhanced version of BWO (SDABWO) was proposed in [87] to improve the convergence behaviour of BWO and solve the FS problem. Three techniques were integrated in SDABWO. First, the spouses of male spiders are chosen based on a computational procedure that takes into consideration the weight of female spiders and the distance between spiders. Second, the mutation operators of differential evolution are used in SDABWO at its mutation phase in order to escape from local optima. Lastly, the three key parameters of SDABWO (procreating rate, cannibalism rate, and mutation rate) are adjusted dynamically over the course of the simulation process of SDABWO. SDABWO was compared to the performance of five well-established optimization algorithms (GWO, PSO, DE, BOA, HHO) using 12 datasets from the UCI repository. The experimental results indicate that SDABWO outperforms the other compared algorithms.

The chimp optimization algorithm (ChOA) is an optimization algorithm that is inspired by the behaviour of individual chimps in their group hunting for prey [88]. This algorithm was originally proposed for solving continuous optimization problems. The binary chimp optimization algorithm (BChOA) for solving the FS problem was introduced in [89]. BChOA has two variations, which are a result of combining the chOA with the one-point crossover operator and two transfer-functions categories (S-shaped and V-shaped transfer functions). The two versions of BChOA were compared to six popular metaheuristics (GA, PSO, BA, ACO, firefly algorithm, and flower pollination) and the results revealed that the two versions of BChOA perform better than the other tested algorithms.

The Hunger Games Search Optimization (HGSO) algorithm is an optimization algorithm for continuous mathematical problems. It was inspired by the prey anxiety from being eaten by their predators [90]. Devi et al. [91] presented two binary versions of the HGSO algorithm for the FS problem. It uses V-shaped and S-shaped transfer functions to transfer continuous solutions to binary ones. Binary HGSO was compared to well-known optimization algorithms (e.g., binary GWO and BSCA) using 16 datasets from the UCI repository. The simulation results demonstrated that the binary HGSO are more accurate with less selected features than the other tested algorithms.

In summary, many of the hybrid SCA variations in this section, including Binary SC-MWOA, ISSAFD, SCHHO, SCADE, HBPSOSCA and SCAGA, have internal parameters that require fine tuning and use iterative-based optimization operators inside their optimization loops (e.g., the crossover and mutation operators in SCAGA). In general, when compared to traditional optimization algorithms, hybrid methods use more computations (e.g., ASOSCA, HBPSOSCA, SCHHO). We are encouraged to use SCA in this new work because the candidate solutions in SCA can easily be converted to binary solutions using the transfer function described in Section 4.3.

3 Binary version of sine cosine algorithm for FS

The Sine Cosine Algorithm (SCA) [50], summarized in code in Algorithm 3 and pictorially in Fig. 1, iteratively optimizes a population of candidate solutions using basic trigonometric functions. A candidate solution is usually made of m decision variables $X =< x_1, x_2, ..., x_m >$, each initially generated randomly between the lower (LB) and upper (UB) bound for the variable. Once an initial population of candidate solutions has been randomly generated, SCA uses the problem's fitness function to calculate a fitness value of each candidate solution. The iterative optimization process of SCA then begins, and the decision variables of each candidate solution X_i^t are updated as follows:

$$x_i^{t+1} = \begin{cases} x_i^t + r_1 \times sin(r_2) \times |r_3 P_i^t - x_i^t|, & r_4 < 0.5 \\ x_i^t + r_1 \times cos(r_2) \times |r_3 P_i^t - x_i^t|, & r_4 \geq 0.5 \end{cases} \quad (1)$$

where r_1, r_2, r_3 and r_4 are random numbers and P_i^t is the position of the destination point in x_i^t at iteration t. In detail, r_1 is used to balance between exploration and exploitation of the range of the trigonometric functions in (1). The value of r_1 is selected at each iteration of SCA as follows:

$$r_1 = a - t\frac{a}{T} \quad (2)$$

where a is a constant, t is the iteration number and T is the maximum number of iterations. $r_2 \in [0, 2\pi]$ specifies the distance and direction of the movement related to the destination. $r_3 \in [0, 2]$ determines the weight of the

```
1: {———- Step 1: Initialize SCA parameters ———-}
2: Initialize the parameters ($n, m, LB, UB$ and $T$).
3: {———- Step 2: Generate the initial search agents ———-}
4: $x_i^j = LB + (UB - LB) \times U(0,1), \quad \forall j = 1, 2, \ldots, n$, and $\forall i = 1, 2, \ldots, m$.
5: Calculate the fitness of each solution $x_i$
6: Find the best search agent $P$
7: {———- Step 3: Update the position of each search agent ———-}
8: while ($t \le T$) do
9:     $r_1 = a - t\frac{a}{T}$
10:    for $j$ = 1 to $n$ do
11:        for $i$ = 1 to $m$ do
12:            Calculate random values $r_2, r_3, r_4$
13:            if ($r_4$ <0.5) then
14:                $x_i^{t+1} = x_i^t + r_1 \times \sin(r_2) \times |r_3 P_i^t - x_i^t|$
15:            else
16:                $x_i^{t+1} = x_i^t + r_1 \times \cos(r_2) \times |r_3 P_i^t - x_i^t|$
17:            end if
18:        end for
19:        if ($f(x_i^{t+1}) < f(x_i^t)$) then
20:            $x_i^{t+1} = x_i^{t+1}$
21:            $f(x_i^{t+1}) = f(x_i^{t+1})$
22:        else
23:            $x_i^{t+1} = x_i^t$
24:            $f(x_i^{t+1}) = f(x_i^t)$
25:        end if
26:        if ($f(x_i^{t+1}) < P_i^t$) then
27:            $P_i^t = x_i^{t+1}$
28:            $f(P_i^t) = f(x_i^{t+1})$
29:        end if
30:    end for
31:    $t = t + 1$
32: end while
33: Return the best search agent $P$
```

Algorithm 1 SCA pseudo-code.

destination point P_i^t. The fourth parameter $r_4 \in [0, 1]$ is a number used to randomly choose one of the two options in (1).

The FS problem is a binary optimization problem. A hypercube represents its search space, and a bit flip in the candidate vector changes the candidate position in the search space ($X = \{x_1, x_2, ..., x_m\}$). However, given that SCA is originally for continuous optimization problems, there is a need for a mapping function. The transfer function (TF) proposed by [92] is utilized to map a candidate continuous value to its corresponding binary value. In this paper, the use of the TF is based on literature work described in [93].

In more detail, the use of the TF is conducted as follows. First, the probability of flipping a bit is calculated using (3). Where $v_i^d(t)$ refers to the velocity of the d^{th} dimension in the i^{th} step vector (velocity) for the current iteration (t). Next, the decision value is updated based on (4), in which a random number $r \in [0, 1]$ is generated and, if the probability of flipping $T(v_i^d(t))$ is greater than r, then a bit flip takes place on the i-th element of the position vector ($X_i(t+1)$). This TF is called V-shaped and is visualized in Fig. 2.

$$T(v_d^i(t)) = |(v_d^i(t))/\sqrt{1 + (v_d^i(t))^2}| \tag{3}$$

$$X(t+1) = \begin{cases} \neg X_t & r < T(v_k^i(t)) \\ X_t & r \ge T(v_k^i(t)) \end{cases} \tag{4}$$

3.1 Objective function

In every optimization problem, there must be an objective function, which is an evaluation function that is used to measure a solution's effectiveness. In the case of the FS optimization problem, a wrapper (optimizer) aims to i) minimize the number of the selected feature, and ii) increase the algorithm accuracy. Therefore, the developed objective function is as illustrated in (5). The focus is to minimize the classification error rate and the selection ratio, where the classification error rate is denoted as $ERR(D)$ and the selection ratio is calculated by dividing the selected number of features ($|R|$) over the total number of features ($|N|$). $\alpha \in [0, 1]$ is the weight assigned to the classification error rate, and $\beta = 1 - \alpha$ is the weight assigned to the selection ratio [94].

$$Fitness = \alpha \times ERR(D) + \beta \times \frac{|R|}{|N|} \tag{5}$$

4 Proposed algorithm: an improved binary sine cosine algorithm with multiple exploration and exploitation approaches for feature selection

We present three versions of our binary optimization algorithm called Improved Binary SCA with multiple exploration and exploitation approaches (IBSCA) which can be used to solve FS problems. Algorithm 2 and the flowchart in Fig. 3 present the details of this approach. Three exploration techniques are applied in an accumulative manner to the three versions of IBSCA (IBSCA1, IBSCA2, IBSCA3), where IBSCA3 uses all of the three exploration techniques. The three versions of IBSCA are as follows:

– IBSCA1: OBL is used as the exploration method.
– IBSCA2: Builds on IBSCA1 by additionally using the VNS method combined with the Laplace distribution to explore the search space using several mutation methods.
– IBSCA3: Builds on IBSCA2 by additionally using Refraction Learning to improve the current best candidate solution at each iteration of the optimization loop of SCA.

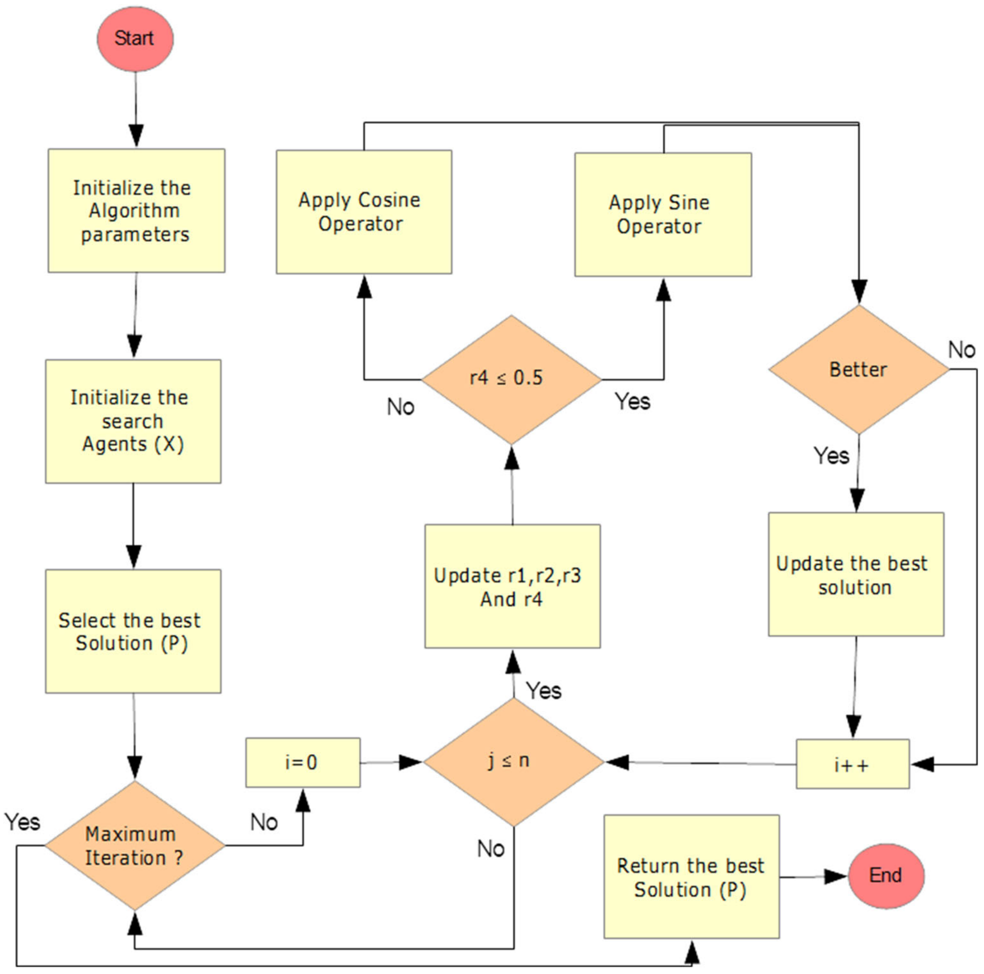

Fig. 1 The flowchart of SCA algorithm

4.1 Representation of candidate solutions

A candidate solution for a FS problem with m features is a vector of m binary decision variables. Given a candidate solution X, $x_i = 1$ means that the ith feature is included in X, whereas $x_i = 0$ means that it is not. Table 1 shows an example candidate solution consisting of 10 decision variables $X =< x_1 = 0, x_2 = 1, x_3 = 1, ..., x_{10} = 1 >$.

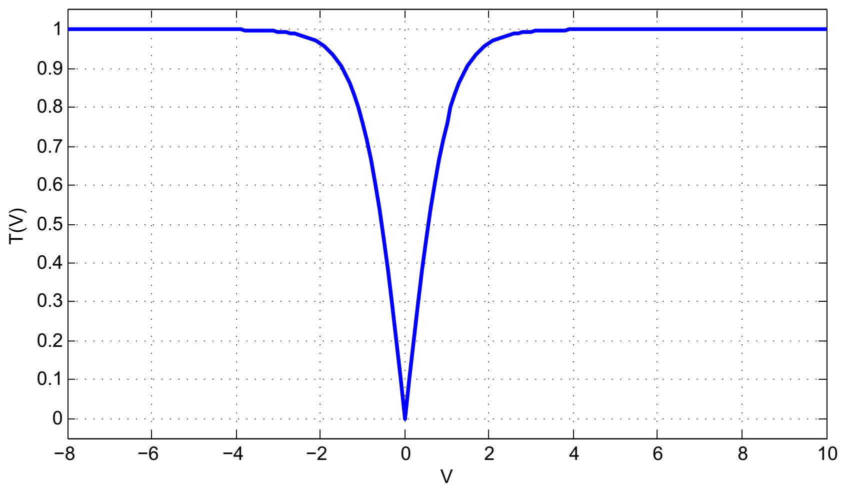

Fig. 2 V-shaped Transfer function

```
1:  Initialize the population of candidate solutions $\mathbf{X}_i$ $(i = 1, 2, ..., n)$ as follows:
    (a) Randomly generate $n/2$ binary candidate solutions
    (b) Apply OBL (7) to each decision variable in those generated candidate solutions to obtain a corresponding $n/2$ candidate solutions
2:  $t = 0$
3:  while $t < MaxItr$ do
4:      Evaluate each candidate solution using the fitness function
5:      Update $(P = X^*)$, the best solution obtained so far
6:      Generate random values $r_1, r_2, r_3$ and $r_4$
7:      Update all candidate solutions using (1)
8:      Apply the two-step transfer function (Section 4.3) to the updated candidate solutions.
9:      Apply RL to the best solution $X^*$
10:     Apply the two-step transfer function (Section 4.3) to the updated solution using RL.
11:     Randomly select, from the current population of solutions, a candidate solution (say $X_i^t$).
12:     Generate a random number $r \in [0, 1]$ based on the Laplace distribution.
13:     Select one of four moves based on the value of $r$:
```

$$x_i^{t+1} = \begin{cases} 0 \le r < 0.25 & \text{Apply swap operator to } X_i^t \\ 0.25 \le r < 0.5 & \text{Apply insert operator to } X_i^t \\ 0.5 \le r < 0.75 & \text{Apply inverse operator to } X_i^t \\ 0.75 \le r < 1.0 & \text{Apply random operator to } X_i^t \end{cases} \tag{6}$$

```
14:     $t = t + 1$
15: end while
16: return the best solution obtained so far
```

Algorithm 2 Improved Binary SCA with multiple exploration and exploitation approaches (IBSCA).

4.2 Population initialization

The performance of optimization algorithms can be improved by a diversified initial population of solutions [95–97]. One possible way to create a diverse initial population is by using the opposition-based learning (OBL) approach. OBL is an intelligent method developed from the observation that considering opposite candidate solutions can lead to improved search times [98]. It can be be applied to the decision variables in machine learning, optimization and search algorithms. For example, if $X = \langle x_1, x_2..., x_m \rangle$ is a candidate solution with m decision variables, the opposite candidate solution X^o is as follows:

$$X^o = \langle x_1^o, x_2^o..., x_m^o \rangle, \text{ where } x_i^o = LB_i + UB_i - x_i \tag{7}$$

where LB_i is the lower bound for variable i and UB_i is its upper bound.

The initialization stage is similar in all versions of IBSCA. In this stage, the first half of the population is generated randomly. The remainder of the population is generated by applying OBL to the first half (Line 1 in Algorithm 2). The use of OBL is expected to expand the search region and improve the solution's approximation.

While OBL can also be applied in the initialization stage of other optimization algorithms (e.g., Cuckoo Search [96, 99], Grey Wolf Optimizer [100], Whale Optimization [101]), as can be seen in Section 5.2, the performance of IBSCA using only OBL is slightly better than the performance of BSCA. This leads to it being a good base to later combine VNS, Laplace distribution, and RL to strongly improve IBSCA's performance.

4.3 Discretization strategy

Candidate solutions produced by the optimization process of SCA and RL are continuous. Therefore, we use two-step transfer functions to convert the continuous decision variables into binary ones (lines 8 and 10).

Table 2 shows eight binary transfer functions (4 S-shaped and 4 V-shaped transfer functions). We conducted extensive simulations to verify the efficiency of these transfer functions and found that V3 was the most viable transfer function. The experimental results in [93, 102] confirm our conclusion about V3. Thus, V3 is adopted in our experiments.

In V3, each decision variable x_i^j in candidate solution $X_i =< x_i^1, x_i^2, ..., x_i^m >$ at iteration t is used to calculate the probability of altering x_i^j to 0 or 1. The probability is

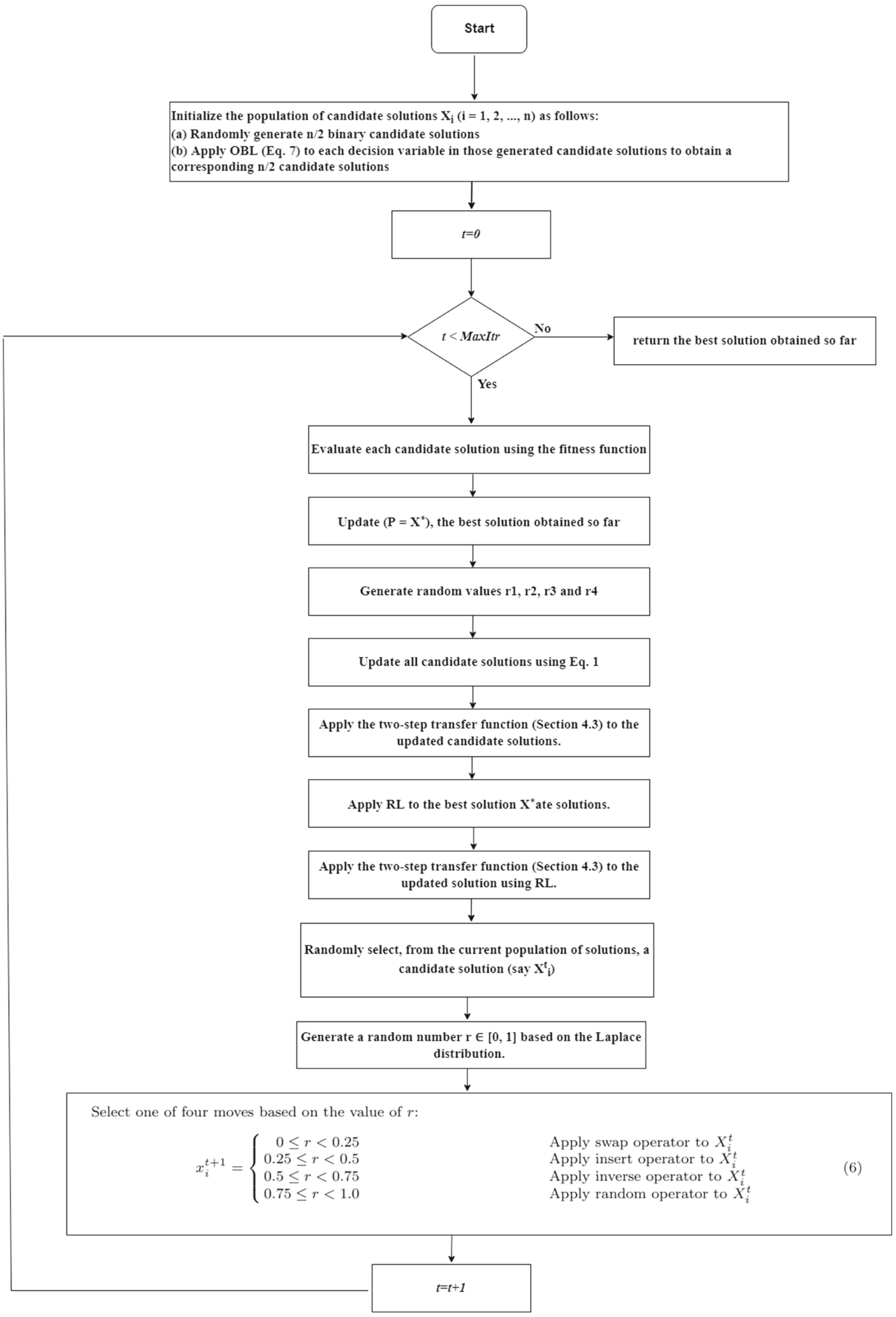
Start
Initialize the population of candidate solutions X_i (i = 1, 2, ..., n) as follows:
(a) Randomly generate n/2 binary candidate solutions
(b) Apply OBL (Eq. 7) to each decision variable in those generated candidate solutions to obtain a corresponding n/2 candidate solutions
t=0
t < MaxItr
No
return the best solution obtained so far
Yes
Evaluate each candidate solution using the fitness function
Update (P = X^*), the best solution obtained so far
Generate random values r1, r2, r3 and r4
Update all candidate solutions using Eq. 1
Apply the two-step transfer function (Section 4.3) to the updated candidate solutions.
Apply RL to the best solution X^*ate solutions.
Apply the two-step transfer function (Section 4.3) to the updated solution using RL.
Randomly select, from the current population of solutions, a candidate solution (say X^t_i)
Generate a random number r ∈ [0, 1] based on the Laplace distribution.
Select one of four moves based on the value of r:
$x_i^{t+1} = \begin{cases} 0 \le r < 0.25 & \text{Apply swap operator to } X_i^t \\ 0.25 \le r < 0.5 & \text{Apply insert operator to } X_i^t \\ 0.5 \le r < 0.75 & \text{Apply inverse operator to } X_i^t \\ 0.75 \le r < 1.0 & \text{Apply random operator to } X_i^t \end{cases}$ (6)
t=t+1

Fig. 3 The flowchart of IBSCA

Table 1 A sample binary candidate solution

Dimension	1	2	3	4	5	6	7	8	9	10
x_i	0	1	1	0	1	0	1	1	0	1

calculated as follows:

$$T(x_i^j(t)) = |x_i^j(t)/\sqrt{1+(x_i^j(t))^2}| \tag{8}$$

Then, $x_i^j(t)$ is set to 0 or 1 as follows:

$$x_i^j(t+1) = \begin{cases} 1 - x_i^j(t), & r < T(x_i^j(t)) \\ x_i^j(t), & r \geq T(x_i^j(t)) \end{cases} \tag{9}$$

where $r \in [0, 1]$ is generated randomly. The chance of flipping the new value $x_i^j(t+1)$ increases as the value $T(x_i^j(t))$ increases.

4.4 Fitness function

In wrapper FS methods, we seek to minimize the number of selected features while maximizing classification accuracy. These two conflicting goals should be taken into account in the fitness function. We adopted the following fitness function to be used in our proposed algorithm:

$$F(X) = \alpha \times ERR(D) + \beta \times \frac{|R|}{|N|} \tag{10}$$

where $F(X)$ is the fitness function of candidate solution X, ERR is the error rate obtained by a k-Nearest Neighbor classifier using X, $|R|$ is the number of features in X, $|N|$ is the total number of features in the dataset, α is the weight for ERR and $\beta = 1 - \alpha$ is the weight for the selection ratio ($|R|/|N|$).

4.5 Optimization loop

The optimization loop of IBSCA starts at Line 3 in Algorithm 2, and ends at line 15. The first step is to evaluate each candidate solution using the fitness function (Section 4.4). Then, the random parameters of the algorithm are initialized (r_1, r_2, r_3 and r_4) and the best solution is determined ($P = X^*$). Afterwards, all the candidate solutions are updated using (1) and the two-step transfer function (Section 4.3) is applied to the updated solutions to generate binary equivalences. In line 9, RL is applied to the best solution X^* as described in Section 4.5.1 and then the result is converted to a binary solution using the two-step transfer function. Finally, a combination of the variables neighborhood search with Laplace distribution (lines 11-14) is applied to a randomly selected solution from the current population, as described in Section 4.5.2.

4.5.1 Refraction learning

IBSCA3 applies RL to the current best solution to improve it. In this section, we describe RL and then show how it can be used in IBSCA3.

The refraction of light is caused by a light ray hitting an interface between two different mediums (e.g., air and water). The ray bends as its velocity changes when it moves toward the boundary between the two mediums. RL is an OBL method based on the principle of light refraction. The one-dimensional spatial refraction-learning process for the global optima X^* at iteration t is illustrated in Fig. 4 [95, 103].

The inverse of X^* can be calculated using refraction learning as follows:

$$X'^* = (\text{LB} + \text{UB})/2 + (\text{LB} + \text{UB})/(2k\eta) - X^*/(k\eta), \tag{11}$$

where η is the refraction index, given by:

$$\eta = \frac{\sin\,\theta_1}{\sin\,\theta_2}, \tag{12}$$

where $\sin\,\theta_1 = ((\text{LB} + \text{UB})/2 - X^*)/h$ and $\sin\,\theta_2 = (X'^* - (\text{LB} + \text{UB})/2)/h'$

In the above equations, X represents the incidence point (original candidate solution) while X' is the refraction point (opposite candidate solution). O denotes the center point of the search interval [LB, UB], h denotes the distance between $\boldsymbol{x}$ and O and h' denotes the distance between X' and O.

Table 2 S-shaped and V-shaped transfer functions

S-Shaped		V-Shaped	
Name	Function	Name	Function
S1	$\frac{1}{1+e^{-2x}}$	V1	$\left\|erf\left(\frac{\sqrt{\pi}}{2}x\right)\right\|$
S2	$\frac{1}{1+e^{-x}}$	V2	$\|\tan(x)\|$
S3	$\frac{1}{1+e^{\frac{-x}{2}}}$	V3	$\left\|\frac{x}{\sqrt{1+x^2}}\right\|$
S4	$\frac{1}{1+e^{\frac{-x}{3}}}$	V4	$\left\|\frac{2}{\pi}\arctan\left(\frac{\pi}{2}x\right)\right\|$

In general, (11) can handle n decision variables as follows:

$$x'^{*}_{j} = (\mathrm{LB}_j + \mathrm{UB}_j)/2 + (\mathrm{LB}_j + \mathrm{UB}_j)/(2k\eta) - x^{*}_{j}/(k\eta), \tag{13}$$

where x^{*}_{j} and x'^{*}_{j} are the jth decision variable of X^{*} and X'^{*}, respectively, and LB_j and UB_j are the lower and upper bounds of the jth decision variable, respectively.

In IBSCA3, (11) is applied to the best solution yet discovered (Line 9 in Algorithm 2).

4.5.2 Variables neighborhood search with laplace distribution

Two versions of IBSCA (IBSCA2 and IBSCA3) employ a combination of the Laplace distribution and VNS method. In this section, we first explain the Laplace distribution and VNS method and then show how they are applied in these algorithms.

Variable Neighborhood Search (VNS) is a powerful metaheuristic for solving combinatorial optimization problems. The primary goal when using VNS is to enhance a candidate solution by performing a series of operations (e.g., mutation) on a solution. This nearby solution may break out of a local optimum. The optimization process of VNS is iterative and moves between adjacent solutions in an attempt to identify a better candidate [97, 104].

The Laplace distribution is suitable for stochastic modeling because it is stable under geometric, rather than ordinary, summation [105, 106]. The Laplace distribution's density function is given by:

$$f(x) = \frac{1}{2b} e^{-\frac{|x-a|}{b}}, \tag{14}$$

where $-\infty < x < \infty$. The Laplace distribution is then defined as follows:

$$x_i^{t+1} = \begin{cases} \frac{1}{2} e^{\frac{|x-a|}{b}}, & x \le a \\ 1 - \frac{1}{2} e^{-\frac{|x-a|}{b}}, & x > a \end{cases} \tag{15}$$

where $a \in R$ is the location parameter and $b > 0$ is the scale parameter.

IBSCA2 and IBSCA3 employ a combination of the Laplace distribution and VNS method (lines 11 to 14 in Algorithm 2). In detail, these algorithms randomly pick a candidate solution x_i^t at iteration t from the current population of solutions. They then generate a random number $r \in [0, 1]$ using the Laplace distribution. r is then used as a probability to select one of four operations on the selected candidate solution (swap, insert, inverse, or random mutation), as follows:

$$x_i^{t+1} = \begin{cases} 0 \le r < 0.25 & \text{Apply swap operator to } x_i^t \\ 0.25 \le r < 0.5 & \text{Apply insert operator to } x_i^t \\ 0.5 \le r < 0.75 & \text{Apply inverse operator to } x_i^t \\ 0.75 \le r < 1.0 & \text{Apply random operator to } x_i^t \end{cases} \tag{16}$$

The swap operator randomly selects two decision variables in the candidate solution (say x_i and x_j) and then exchanges the values of x_i and x_j, as illustrated in Fig. 5.

The insert operator randomly selects two decision variables (say x_i and x_j) in the candidate solution and then shifts the values between x_{i+1} and x_{j-1} down one position, inserting x_i into x_{j-1}, as illustrated in Fig. 6.

The inverse operator, shown in Fig. 7, randomly selects two decision variables (x_i and x_j) in the candidate solution and then inverses the order of values from x_i to x_j.

The random operator, shown in Fig. 8, randomly selects a number of decision variables (say p) in the candidate solution and then flips the binary value of each selected decision variable.

4.6 Computational complexity of IBSCA

The purpose of this section is to show the detailed computational complexity of IBSCA. We assume that the cost of any basic vector operation is O(1) and we denoted MaxItr as M.

The computational complexity of IBSCA (Algorithm 2) can be calculated as follows:

- In Line 1(a), the generation of $n/2$ candidate solutions using a random generation function requires O($n/2$) operations.
- In Line 1(b), the generation of $n/2$ opposite candidate solutions using OBL (7) requires O($n/2$) operations.
- Line 2 requires O(1) operations.
- The internal operations inside the while loop (lines 3 to 15) are as follows:
    - The number of operations required to evaluate the fitness of the candidate solutions is O(n) operations (Line 4).
    - Updating the best candidate solution so far ($P = X^{*}$) requires O(n) operations (Line 5).
    - Generating four random numbers requires O(1) operations (Line 6).
    - Updating the candidate solutions using (1) requires O(n) operations (Line 7).
    - Applying the two-step transfer function (Section 4.3) to the updated candidate solutions requires O(n) operations (Line 8).

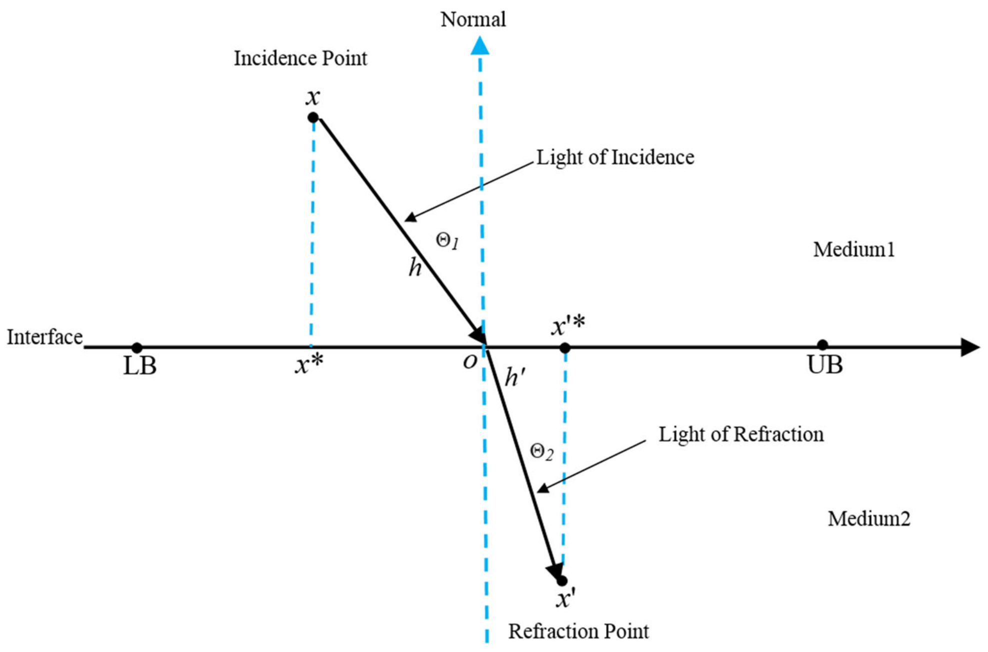

Fig. 4 Refraction Learning for the Global Optimal x^*

Fig. 5 Swap operator between x_3 and x_6

Original Solution X

Dimension	1	2	3	4	5	6	7	8	9	10
x_i	0	1	1	0	1	0	1	1	0	1

Solution X after swap

Dimension	1	2	3	4	5	6	7	8	9	10
x_i	0	1	0	0	1	1	1	1	0	1

Fig. 6 Insert operator between x_2 and x_9

Original Solution X

Dimension	1	2	3	4	5	6	7	8	9	10
x_i	0	1	0	0	1	1	1	1	0	1

Solution X after insert

Dimension	1	2	3	4	5	6	7	8	9	10
x_i	0	0	0	1	1	1	1	1	0	1

Fig. 7 Inverse operator between x_3 and x_6

Original Solution X

Dimension	1	2	3	4	5	6	7	8	9	10
x_i	0	1	1	0	1	0	1	1	0	1

Solution X after inverse

Dimension	1	2	3	4	5	6	7	8	9	10
x_i	0	1	0	1	0	1	1	1	0	1

Fig. 8 Random operator for x_3, x_6 and x_9

Original Solution X

Dimension	1	2	3	4	5	6	7	8	9	10
x_i	0	1	1	0	1	0	1	1	0	1

Solution X after inverse

Dimension	1	2	3	4	5	6	7	8	9	10
x_i	0	1	0	0	1	1	1	1	1	1

  - Applying RL to the best solution X^* requires O(1) operations (Line 9).
  - Applying the two-step transfer function (Section 4.3) to the updated solution using RL requires O(1) operations (Line 10).
  - Selecting a random solution from the current population of solutions (say X_i^t) requires O(1) operations (Line 11).
  - Generating a random number $r \in [0, 1]$ based on the Laplace distribution requires O(1) operations (Line 12).
  - Selecting one of four moves based on the value of r requires O(1) operations (Line 13).
  - Line 14 requires O(1) operations.

- Overall, the cost of the operations in the while loop (lines 3 to 15) is $O(M(n + n + 1 + n + n + 1 + 1 + 1 + 1 + 1 + 1))$, where M is the maximum number of iterations. This can be reduced to $O(M.n)$.
- The total number of operations in IBSCA (lines 1 to 16) is $O(n/2 + n/2 + 1 + M.n + 1)$. This can be reduced to $O(M.n)$ because $M.n$ is greater than $n + 2$.

In summary, the computational complexity of IBSCA is $O(M.n)$.

Table 3 Parameters Settings

Parameter	Value
Population size (Search agents)	10
Number of iterations	100
Dimension	Number of features
Number of runs	30
α in fitness function	0.99
a	2
r1	decreases linearly from a to 0
r2	a random number in the range [0 , 2π]
r3	a random number in the range [0 , 2]
r4	a random number in the range [0 , 1]
rLaplace	a random number in the range [0 , 1]

5 Experiments

In this section, we first demonstrate the performance of the three variations of IBSCA when solving the FS problem. The detailed characteristics of the used datasets are presented in Section 5.1. Section 5.2 provides a comparison of the convergence behavior of the original Binary Sine Cosine Algorithm (BSCA) [107] to the convergence behaviors of the three variations of IBSCA over the UCI datasets. Section 5.3 shows the performance of IBSCA3 in comparison to other well known FS algorithms.

Table 3 illustrates the parameter settings of our proposed approach. The values of the parameters of all of the algorithms have been finely tuned based on several experiments. Thus, the algorithms in this section were compared to each other based on their best parameter settings. Since the general feature of the optimization

Table 4 Datasets description

Dataset	No. of Attributes	No. of Objects	No. of Classes
Breastcancer	9	699	2
BreastEW	30	569	2
Exactly	13	1000	2
Exactly2	13	1000	2
HeartEW	13	270	2
Lymphography	18	148	4
M-of-n	13	1000	2
PenglungEW	325	73	7
SonarEW	60	208	2
SpectEW	22	267	2
CongressEW	16	435	2
IonosphereEW	34	351	2
KrvskpEW	36	3196	2
Tic_tac-toe	9	958	2
Vote	16	300	2
WaveformEW	40	5000	3
WineEW	13	178	3
Zoo	16	101	7
COVID-19 dataset	15	1085	2

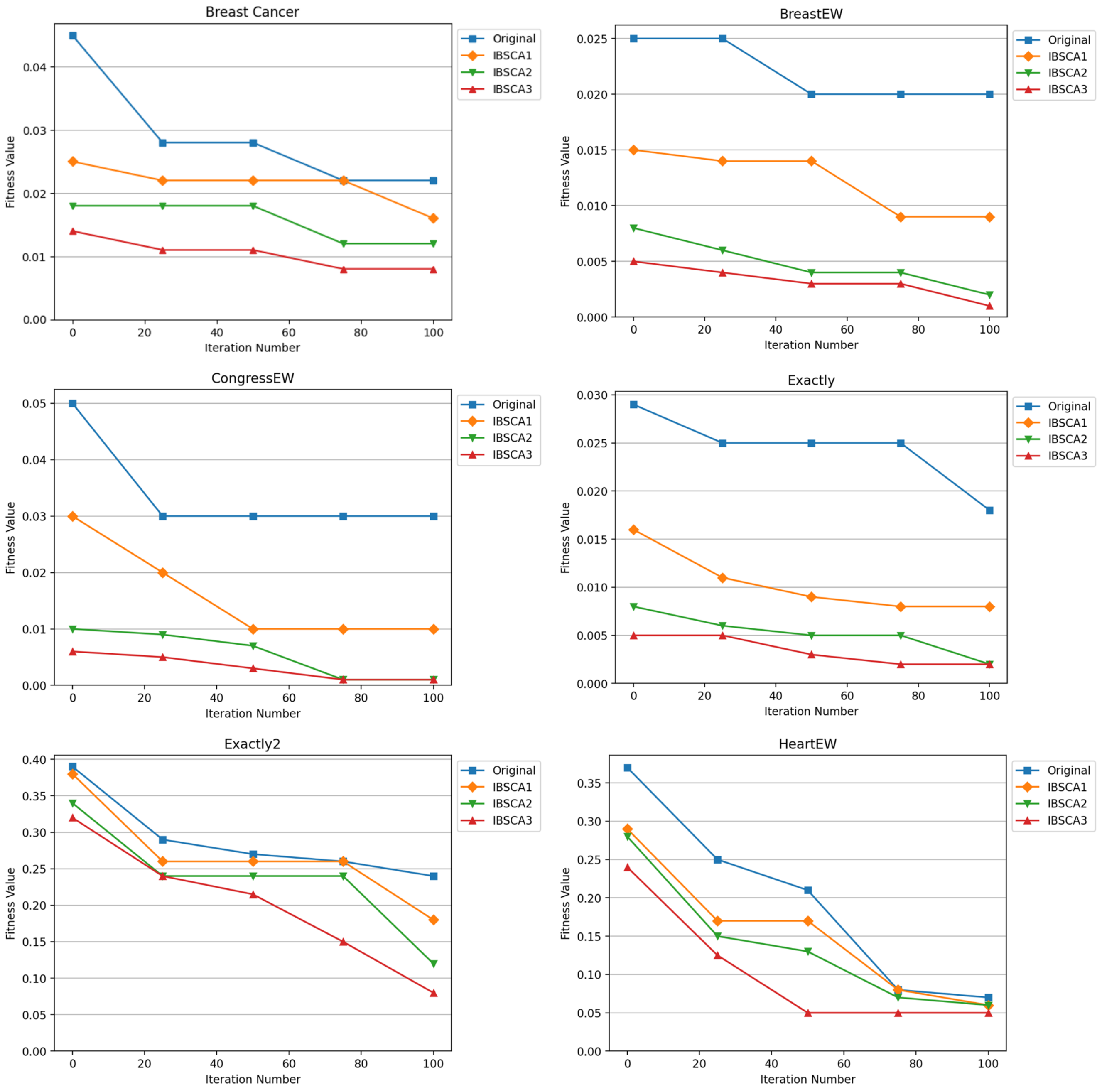

Fig. 9 Convergence behavior of BSCA, IBSCA1, IBSCA2 and IBSCA3 over the datasets: Breastcancer, BreastEW, CongressEW, Exactly, Exactly2 and HeartEW

algorithms is random in nature, we executed the algorithms for 30 independent runs. We executed our experiments on a Windows 7 computer with an Intel Core i7-3517U CPU @ 1.90GHz 2.40GHz and 8.0 GB memory.

5.1 Datasets properties

The performance of IBSCA was evaluated using nineteen datasets (18 from UCI repository [108] and a real-world COVID-19 dataset[1]. Table 4 provides a description of these datasets in terms of their dimensions, number of instances, and number of classes. All datasets were split randomly into 80 training instances and 20 testing instances [38] where the *k*-nearest neighbors classifier (KNN) is used. The

[1]https://github.com/Atharva-Peshkar/Covid-19-Patient-Health-Analytics

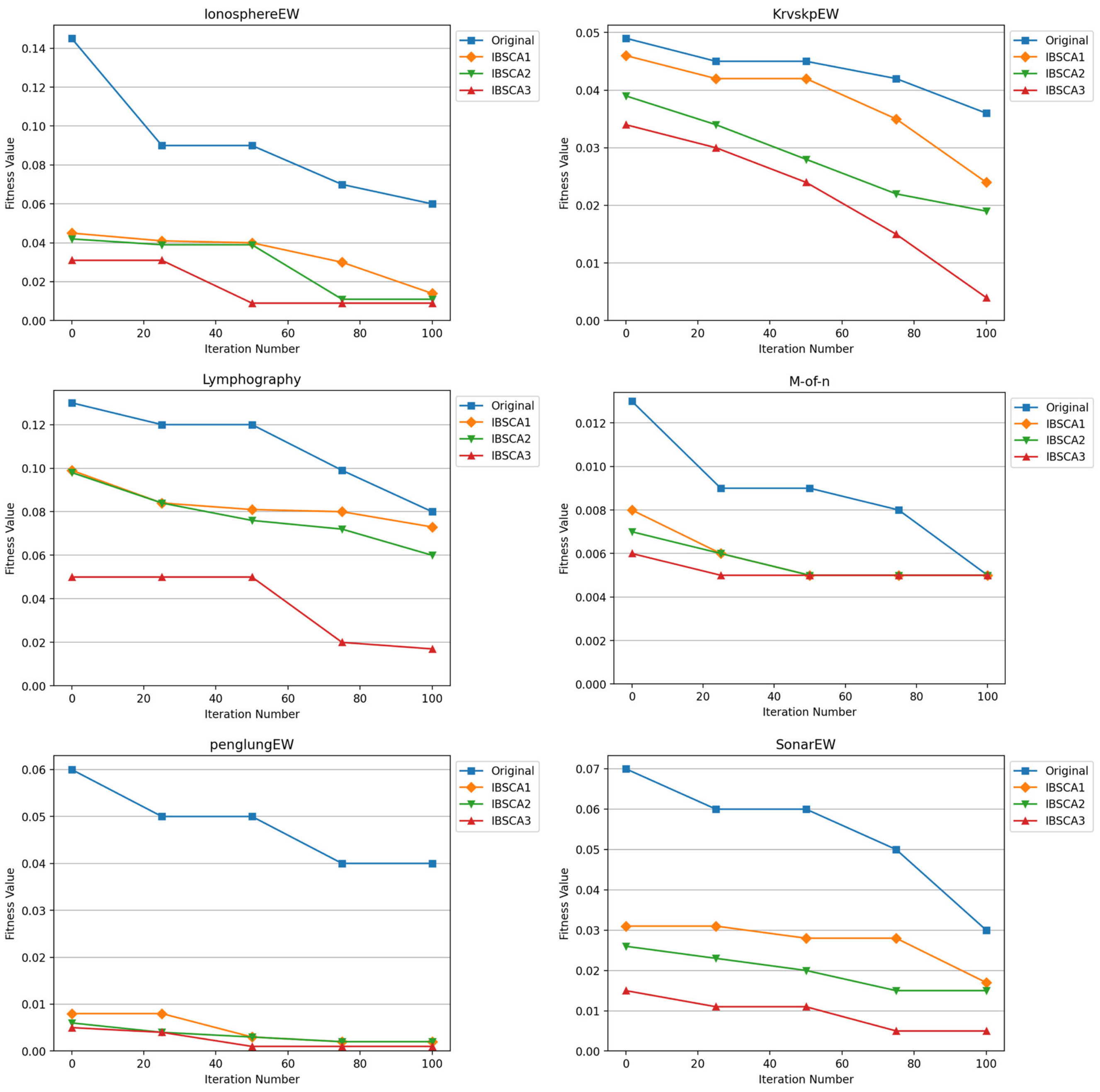

Fig. 10 Convergence behavior of BSCA, IBSCA1, IBSCA2 and IBSCA3 over the datasets: IonosphereEW, KrvskpEW, Lymphography, M-of-n, penglungEW and SonarEW

KNN technique is a supervised machine learning method for solving classification and regression problems [102].

5.2 Convergence behavior of BSCA vs three variations of IBSCA

Figures 9, 10 and 11 show the convergence behavior of BSCA, IBSCA1, IBSCA2 and IBSCA3 over the UCI datasets. In each chart of these figures, the *x-axis* represents the iteration number, and the *y-axis* represents the fitness value. The convergence charts show that IBSCA3 converges faster to good solutions than all of the other algorithms for all of the datasets. The superiority of IBSCA3 is mainly because it uses three exploration techniques. First, it uses OBL when initializing the population to improve quality and diversity. Second, it integrates the VNS and Laplace distribution to explore the search space using multiple mutation methods. Third, it uses RL to search

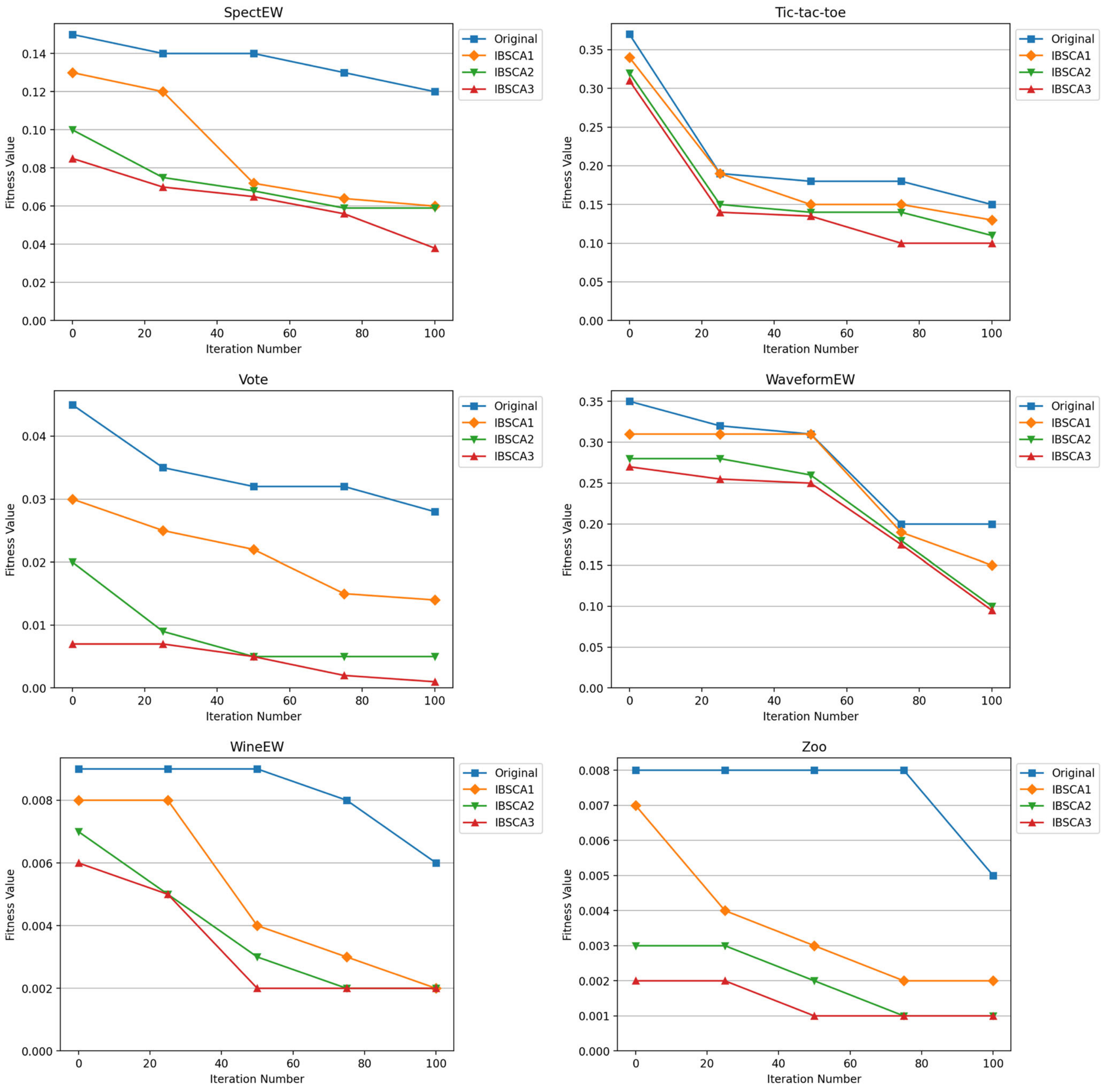

Fig. 11 Convergence behavior of BSCA, IBSCA1, IBSCA2 and IBSCA3 over the datasets: SpectEW, Tic-tac-toe, Vote, WaveformEW, WineEW and Zoo

the neighborhood of best candidate solutions for better solutions.

The second best performing algorithm was IBSCA2. It uses two exploration techniques compared to IBSCA3 that uses three techniques. IBSCA1 was the third best performing algorithm. It uses only one exploration technique. BSCA exhibits the worst convergence behavior compared to the other algorithms. This may be because it does not use any additional exploration techniques compared to the other algorithms.

5.3 Performance analysis of IBSCA3 compared to baseline algorithms

In this section, we present a comparison between IBSCA3 and other binary versions of the baseline algorithms: BSCA,

Table 5 Parameter settings of the baseline algorithms

Algorithm	Parameter settings
RBDA	Population size = 10, Number of iterations = 100, α = 0.99, and β = 0.01
LBDA	Population size = 10, Number of iterations = 100, α = 0.99, and β = 0.01
QBDA	Population size = 10, Number of iterations = 100, α = 0.99, and β = 0.01
SBDA	Population size = 10, Number of iterations = 100, α = 0.99, and β = 0.01
BGWO	Population size = 10, Number of iterations = 100, and a=[2, 0]
BGSA	Population size = 10, Number of iterations = 100, Gø= 100, and α = 20
BBA	Population size = 10, Number of iterations = 100, Frequency minimum Qmin = 0, Frequency maximum Qmax = 2, Loudness A = 0.5, and Pulse rate r = 0.5
CHIO	HIS = 30, Max_Age = 100, BRr = 0.01, Max_Itr = 100 , LB = 0 , UB = 1
CHIO-GC	HIS = 30, Max_Age = 100, BRr = 0.01, Max_Itr = 100 , LB = 0 , UB = 1

Random based Binary Dragonfly Algorithm (RBDA) [102], Linear based Binary Dragonfly Algorithm (LBDA) [102], Quadratic based Binary Dragonfly Algorithm (QBDA) [102], Sinusoidal based Binary Dragonfly Algorithm (SBDA) [102], Binary Gray Wolf Optimizer (BGWO) [109], Binary Gravitational Search Algorithm (BGSA) [109], and Binary Bat Algorithm (BBA) [109]. These algorithms were compared according to their classification accuracy, number of selected features, and their best fitness values. We also compared IBSCA to Coronavirus Herd Immunity Optimizer (CHIO) [110] and Coronavirus Herd Immunity Optimizer-Greedy Crossover (CHIO-GC) [110]. Table 5 shows the parameter settings of these algorithms, as in [102, 110].

Table 6 shows the average value and standard deviation of the results obtained by the proposed IBSCA3 algorithm, and the other compared algorithms, in terms of average classification accuracy. IBSCA3 outperforms the other algorithms and obtains the best classification accuracy on all the UCI and COVID-19 datasets.

Table 7 presents the average number of selected features for the tested algorithms. IBSCA3 outperforms the other tested algorithms on 14 out of 18 datasets. This is better than the second best algorithm (SBDA algorithm) which outperforms the remaining compared algorithms on 11 out of 18 datasets.

Table 8 illustrates the best fitness values obtained by the tested algorithms. We can observe that IBSCA3 shows superior performance over the other algorithms. It obtains the best fitness values on all datasets.

In summary, the enhanced version of the Binary Sine Cosine algorithm outperformed the other algorithms for all of the tested datasets, with IBSCA3 providing the highest classification accuracy and the lowest fitness function for all datasets with different dimensions, and the lowest average number of selected features in most cases. The overall results indicate that IBSCA3 converges faster than the other algorithms to the most accurate solutions with the least number of features.

The original SCA employs a random update method to update the solutions in the algorithm. This negatively affects the ability of SCA to balance between the exploration and exploitation of the search space. In contrast, IBSCA3 improves exploration and exploitation in the original SCA by employing several techniques. First, it employs an OBL approach to improve the diversity of initial population. Second, it integrates the VNS and Laplace distribution to explore the search space using multiple mutation methods. Third, it uses RL to search the neighborhood of the best candidate solutions for better ones. The overall results indicate that IBSCA3 improves the performance and convergence behavior of the original SCA in solving the FS problem.

5.4 Performance analysis of IBSCA3 compared to state-of-the-art algorithms that adopt OBL-enhanced methods, VNS and laplace distribution

In this section, we demonstrate a comparison between IBSCA3 and other new algorithms that incorporate OBL into their basic structure. These algorithms are: Improved Salp Swarm Algorithm based on opposition based learning and novel local search algorithm for feature selection (ISSA) [111], Improved Harris Hawks Optimization using elite opposition-based learning and novel search mechanism for feature selection (IHHO) [112], and New feature selection methods based on opposition-based learning and self-adaptive cohort intelligence for predicting patient no-shows (OSACI) [113]. We also compare IBSCA3 with other new algorithms that employ similar methods (VNS and Laplace distribution): A variable neighborhood search algorithm for human resource selection and optimization problem in the home appliance manufacturing industry (VNS-HRS) [114], Improving feature selection

Table 6 Average and standard deviation of classification accuracy for the proposed IBSCA3 algorithm in comparison to existing algorithms

Dataset	Metric	IBSCA3	BSCA	RBDA	LBDA	QBDA	SBDA	BGWO	BGSA	BBA	CHIO	CHIO-GC
Breastcancer	Avg	**0.997**	0.965	0.983	0.978	0.993	0.993	0.978	0.948	0.932	N/A	N/A
	StDev	0.000	0.002	0.004	0.002	0.001	0.000	0.01	0.02	0.051	N/A	N/A
BreastEW	Avg	**1.000**	0.979	**1.000**	0.987	0.980	0.975	0.923	0.928	0.913	0.899	0.94
	StDev	0.000	0.005	0.008	0.008	0.006	0.006	0.015	0.014	0.035	0.021	0.019
Exactly	Avg	**1.000**	0.985	**1.000**	**1.000**	0.994	**1.000**	0.835	0.732	0.602	N/A	N/A
	StDev	0.000	0.037	0.003	0.000	0.020	0.000	0.077	0.124	0.055	N/A	N/A
Exactly2	Avg	**0.823**	0.783	0.797	0.780	0.785	0.757	0.674	0.644	0.683	N/A	N/A
	StDev	0.009	0.038	0.015	0.002	0.000	0.014	0.041	0.041	0.04	N/A	N/A
HeartEW	Avg	**0.926**	0.798	0.839	0.901	0.880	0.867	0.788	0.77	0.728	0.854	0.912
	StDev	0.041	0.047	0.011	0.034	0.019	0.009	0.039	0.066	0.061	0.027	0.018
Lymphography	Avg	**0.967**	0.814	0.930	0.913	0.924	0.954	0.842	0.864	0.689	0.761	0.834
	StDev	0.021	0.035	0.021	0.019	0.023	0.016	0.057	0.081	0.103	0.035	0.027
M-of-n	Avg	**1.000**	0.984	**1.000**	**1.000**	0.999	**1.000**	0.913	0.827	0.716	N/A	N/A
	StDev	0.000	0.005	0.000	0.000	0.004	0.000	0.052	0.061	0.083	N/A	N/A
PenglungEW	Avg	**1.000**	0.977	0.959	**1.000**	**1.000**	**1.000**	0.869	0.949	0.816	N/A	N/A
	StDev	0.068	0.000	0.039	0.000	0.000	0.000	0.012	0.054	0.054	N/A	N/A
SonarEW	Avg	**0.995**	0.952	0.964	0.944	0.948	0.993	0.887	0.865	0.814	N/A	N/A
	StDev	0.008	0.015	0.017	0.019	0.012	0.011	0.04	0.047	0.059	N/A	N/A
SpectEW	Avg	**0.941**	0.862	0.894	0.923	0.890	0.925	0.818	0.785	0.756	N/A	N/A
	StDev	0.017	0.007	0.010	0.010	0.013	0.011	0.029	0.034	0.039	N/A	N/A
CongressEW	Avg	**1.000**	0.961	0.976	0.999	0.993	0.975	0.95	0.943	0.869	N/A	N/A
	StDev	0.000	0.014	0.003	0.004	0.006	0.005	0.047	0.026	0.08	N/A	N/A
IonosphereEW	Avg	**0.993**	0.975	0.970	0.970	0.923	0.984	0.891	0.869	0.866	N/A	N/A
	StDev	0.004	0.013	0.013	0.009	0.012	0.011	0.025	0.026	0.027	N/A	N/A
KrvskpEW	Avg	**0.984**	0.962	0.975	0.981	0.968	0.966	0.935	0.898	0.79	N/A	N/A
	StDev	0.007	0.005	0.004	0.006	0.004	0.004	0.019	0.053	0.09	N/A	N/A
Tic-tac-toe	Avg	**0.869**	0.811	0.820	0.839	0.847	0.832	0.806	0.761	0.658	N/A	N/A
	StDev	0.000	0.049	0.005	0.000	0.005	0.005	0.029	0.038	0.081	N/A	N/A
Vote	Avg	**0.998**	0.984	0.996	0.971	0.959	0.972	0.939	0.943	0.856	N/A	N/A
	StDev	0.004	0.038	0.007	0.010	0.008	0.008	0.021	0.025	0.102	N/A	N/A
WaveformEW	Avg	**0.791**	0.724	0.766	0.760	0.738	0.776	0.705	0.697	0.659	N/A	N/A
	StDev	0.013	0.018	0.009	0.010	0.008	0.011	0.015	0.021	0.046	N/A	N/A
WineEW	Avg	**1.000**	0.947	0.991	**1.000**	**1.000**	**1.000**	0.938	0.976	0.838	N/A	N/A
	StDev	0.000	0.049	0.013	0.000	0.000	0.000	0.036	0.035	0.131	N/A	N/A
Zoo	Avg	**1.000**	0.994	**1.000**	**1.000**	**1.000**	**1.000**	0.993	0.995	0.867	N/A	N/A
	StDev	0.000	0.028	0.000	0.000	0.000	0.000	0.023	0.015	0.114	N/A	N/A
COVID-19	Avg	**0.952**	0.894	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.914	0.937
	StDev	0.008	0.028	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.025	0.019

The results in bold point shows the best results in the table

Table 7 Average and standard deviation of average selected features for the proposed IBSCA3 algorithm in comparison to existing algorithms

Dataset	Metric	IBSCA3	BSCA	RBDA	LBDA	QBDA	SBDA	BGWO	BGSA	BBA
Breastcancer	Avg	**2.71**	5.01	5.07	4.93	3.03	5	6.4	4.47	4.1
	StDev	0.12	0.67	1.31	0.25	0.18	0	1.75	1.01	1.27
BreastEW	Avg	**7.86**	8.39	9.07	11.7	13.33	12.2	21.57	14.93	11.77
	StDev	1.69	1.78	1.74	1.97	2.51	2.54	4.8	2	3.94
Exactly	Avg	5.92	8.04	6.07	6.13	7.03	6.13	10.7	7.67	**5.23**
	StDev	1.83	0.09	0.25	0.35	0.85	0.35	2.02	1.49	2.25
Exactly2	Avg	**1.01**	6.38	2.83	1.3	1.03	5.03	6.97	6.13	5.77
	StDev	0.14	2.18	3.11	1.64	0.18	3.76	2.74	2.08	1.57
HeartEW	Avg	5.64	7.03	6.13	6.4	6.33	6.03	9.7	6.63	**5.07**
	StDev	1.82	1.47	1.25	1.28	1.06	0.96	1.99	1.94	1.7
Lymphography	Avg	**5.91**	6.04	9.43	8.07	7.67	6.83	10.6	9	6.87
	StDev	0.77	2.03	1.81	1.51	1.84	0.91	2.63	2.18	1.96
M-of-n	Avg	**5.27**	6.88	6.07	6.07	6.97	6.07	10.43	8.2	5.73
	StDev	1.94	0.54	0.25	0.25	0.67	0.25	1.45	1.16	1.82
PenglungEW	Avg	**84.62**	106.38	110.2	99.9	132.47	117.53	152.33	145.1	126.47
	StDev	9.33	10.39	11.35	8.45	3.82	9.7	7	4.88	15.62
SonarEW	Avg	**22.14**	25.91	23.1	26.53	28.3	24.33	34.87	27.07	23.53
	StDev	2.69	3.57	3.06	4.03	3.62	2.52	7.81	3.64	5.15
SpectEW	Avg	**4.78**	10.13	9.57	5.2	9.4	8.57	13.77	9.77	8.73
	StDev	2.16	2.56	2.37	2.31	1.94	1.63	2.93	2.3	2.29
	StDev	1.72	1.42	1.23	0.86	1.22	1.5	1.88	1.91	2.18
IonosphereEW	Avg	**10.65**	15.73	11	13.63	12.93	12.67	16.17	14.9	12.3
	StDev	1.83	2.81	2.3	3.15	2.99	2.17	2.35	2.89	3.4
KrvskpEW	Avg	**13.07**	21.45	18.93	18.97	20.6	19.57	30.9	19.73	14.97
	StDev	2.71	2.56	2.12	2.83	2.09	2.43	2.93	2.36	2.88
Tic-tac-toe	Avg	**4.1**	5.9	6.7	7	6.93	6.93	8.3	5.6	4.3
	StDev	0.65	1.06	0.47	0	0.37	0.37	1.24	0.97	1.7
Vote	Avg	4.21	7.96	4.3	4.63	6.23	**4**	8.63	7.37	6.1
	StDev	0.87	1.82	0.53	1.45	1.77	0.98	2.63	1.67	2.14
WaveformEW	Avg	18.54	35.94	21.4	21.4	21.77	21.83	34.07	21.6	**16.23**
	StDev	3.49	4.93	3.54	2.3	2.71	2.65	4.48	3.69	4.08
WineEW	Avg	**3.02**	5.94	7.13	3.43	4.07	4.4	7.37	6.57	4.87
	StDev	0.43	1.53	1.43	0.68	0.69	1.07	1.67	1.36	1.87
Zoo	Avg	**1.6**	5.63	3.4	4.2	4.5	1.97	7.37	6.97	6.43
	StDev	0.83	0.92	0.56	0.41	0.73	0.96	1.63	1.25	1.83
COVID-19 dataset	Avg	**2.95**	3.05	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	StDev	0.74	0.183	N/A	N/A	N/A	N/A	N/A	N/A	N/A

The results in bold point shows the best results in the table

performance for classification of gene expression data using Harris Hawks Optimizer with variable neighborhood learning (VNLHHO) [115], Improved equilibrium optimization algorithm using elite opposition-based learning and new

Table 8 Average and standard deviation of the best fitness value for the proposed IBSCA3 algorithm in comparison to existing algorithms

Dataset	Metric	IBSCA3	BSCA	RBDA	LBDA	QBDA	SBDA	BGWO	BGSA	BBA
Breastcancer	Avg	**0.010**	0.029	0.023	0.028	0.011	0.013	0.016	0.027	0.036
	StDev	0.001	0.002	0.002	0.001	0.002	0	0.002	0.007	0.005
BreastEW	Avg	**0.002**	0.022	0.003	0.017	0.025	0.029	0.043	0.039	0.036
	StDev	0.001	0.001	0.001	0.008	0.005	0.006	0.007	0.01	0.009
Exactly	Avg	**0.003**	0.024	0.006	0.005	0.012	0.005	0.185	0.253	0.303
	StDev	0	0.002	0.003	0	0.02	0	0.051	0.094	0.108
Exactly2	Avg	**0.201**	0.29	0.204	0.219	0.214	0.245	0.249	0.288	0.25
	StDev	0.009	0.12	0.18	0	0	0.011	0.014	0.014	0.015
HeartEW	Avg	**0.103**	0.196	0.165	0.104	0.124	0.137	0.128	0.137	0.161
	StDev	0.029	0.025	0.011	0.032	0.019	0.008	0.026	0.03	0.023
Lymphography	Avg	**0.037**	0.11	0.075	0.091	0.079	0.049	0.083	0.081	0.162
	StDev	0.012	0.019	0.02	0.018	0.022	0.016	0.035	0.033	0.053
M-of-n	Avg	**0.005**	0.009	**0.005**	**0.005**	0.007	**0.005**	0.087	0.165	0.165
	StDev	0	0	0.027	0	0.004	0	0.039	0.041	0.044
PenglungEW	Avg	**0.002**	0.048	0.044	0.003	0.004	0.004	0.126	0.004	0.132
	StDev	0	0.019	0.038	0	0	0	0.025	0	0.038
SonarEW	Avg	**0.009**	0.054	0.039	0.059	0.057	0.011	0.104	0.082	0.11
	StDev	0.007	0.025	0.017	0.019	0.011	0.011	0.02	0.023	0.03
SpectEW	Avg	**0.063**	0.136	0.11	0.079	0.113	0.079	0.143	0.153	0.143
	StDev	0.007	0.016	0.009	0.009	0.012	0.01	0.016	0.018	0.021
CongressEW	Avg	**0.003**	0.034	0.028	0.005	0.011	0.029	0.028	0.032	0.032
	StDev	0.001	0.004	0.003	0.003	0.005	0.004	0.01	0.013	0.015
IonosphereEW	Avg	**0.018**	0.091	0.033	0.033	0.081	0.020	0.099	0.127	0.124
	StDev	0.006	0.012	0.013	0.009	0.012	0.01	0.013	0.011	0.019
KrvskpEW	Avg	**0.021**	0.043	0.03	0.024	0.038	0.039	0.051	0.099	0.093
	StDev	0.007	0.005	0.003	0.006	0.004	0.004	0.009	0.049	0.039
Tic-tac-toe	Avg	**0.157**	0.214	0.187	0.169	0.160	0.175	0.177	0.232	0.232
	StDev	0.003	0.005	0.004	0	0.005	0.004	0.008	0.024	0.022
Vote	Avg	**0.004**	0.034	0.007	0.032	0.044	0.030	0.048	0.038	0.063
	StDev	0.003	0.008	0.007	0.01	0.007	0.008	0.009	0.009	0.017
WaveformEW	Avg	**0.209**	0.276	0.237	0.243	0.264	0.227	0.237	0.251	0.251
	StDev	0.009	0.012	0.008	0.009	0.008	0.011	0.008	0.013	0.016
WineEW	Avg	**0.003**	0.008	0.015	**0.003**	**0.003**	0.004	0.045	0.009	0.025
	StDev	0.000	0.009	0.013	0.001	0.001	0.001	0.017	0.012	0.017
Zoo	Avg	**0.001**	0.007	0.002	0.003	0.003	**0.001**	0.007	0.005	0.052
	StDev	0.001	0.002	0	0	0	0.001	0.01	0.001	0.032
COVID-19 dataset	Avg	**0.002**	0.013	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	StDev	0.034	0.072	N/A	N/A	N/A	N/A	N/A	N/A	N/A

The results in bold point shows the best results in the table

Table 9 Parameter settings of ISSA, IHHO, OSACI, VNS-HRS, VNLHHO, IEOA, DSSA, SFS-LARLRM and BGWOPSO

Algorithm	Parameter settings
ISSA	Population size = 10, Number of iterations = 40
IHHO	Population size = 10, Number of iterations = 50, α = 0.99, and β = 0.01
OSACI	Population size = 100, Number of iterations = 50
VNS-MCI	Population size = 10, Number of iterations = 40
VNLHHO	Population size = 30, Number of iterations = 100
IEOA	Population size = 10, Number of iterations = 50, α = 0.99, and β = 0.01
DSSA	Population size = 10, Number of iterations = 100, c2 = rand(), c3 = rand()
SFS-LARLRM	Population size = 10, Number of iterations = 100, k =5, $\sigma = \{10^{-7}, 10^{-5}, 10^{-3}, 10^{-1}, 10^{0}, 10^{1}, 10^{3}, 10^{5}, 10^{7}\}$, p = {0.25, 0.5, 0.75, 1}
BGWOPSO	Population size = 10, Number of iterations = 100, c1 = 0.5, c2 = 0.5, c3 = 0.5, w = 0.5 + rand ()/2, $l \in [0, 1]$

Table 10 Average and standard deviation of classification accuracy for the proposed IBSCA3 algorithm in comparison to BSCA and the other algorithms that incorporate OBL, VNS and Laplace distribution

Dataset	Metric	IBSCA3	BSCA	ISSA	IHHO	OSACI	VNS-HRS	VNLHHO	IEOA	DSSA	SFS-LARLRM	BGWOPSO
Breastcancer	Avg	**0.997**	0.965	0.952	0.986	0.991	0.994	0.935	0.964	0.931	0.995	0.978
	StDev	0.000	0.002	0.007	0.003	0.005	0.013	0.027	0.039	0.022	0.068	0.009
BreastEW	Avg	**1.000**	0.979	0.962	**1.000**	**1.000**	0.983	0.955	0.914	0.913	0.977	0.986
	StDev	0.000	0.005	0.012	0.000	0.000	0.019	0.057	0.041	0.097	0.088	0.062
Exactly	Avg	**1.000**	0.911	0.907	**1.000**	0.903	**1.000**	**1.000**	0.892	**1.000**	**1.000**	
	StDev	0.000	0.037	0.014	0.002	0.004	0.000	0.001	0.051	0.023	0.000	0.000
Exactly2	Avg	**0.823**	0.783	0.724	0.687	0.719	0.753	0.796	0.616	0.698	0.781	0.765
	StDev	0.009	0.038	0.019	0.025	0.042	0.028	0.009	0.077	0.093	0.013	0.005
HeartEW	Avg	**0.926**	0.798	0.887	0.758	0.849	0.803	0.861	0.912	0.829	0.905	0.873
	StDev	0.041	0.047	0.052	0.078	0.066	0.017	0.048	0.034	0.056	0.016	0.037
Lymphography	Avg	**0.967**	0.814	0.930	0.913	0.921	0.954	0.842	0.864	0.689	0.761	0.838
	StDev	0.021	0.035	0.021	0.019	0.023	0.016	0.057	0.081	0.103	0.035	0.027
M-of-n	Avg	**1.000**	0.984	**1.000**	**1.000**	0.999	**1.000**	0.913	0.827	0.716	0.792	0.891
	StDev	0.000	0.005	0.000	0.000	0.004	0.000	0.052	0.061	0.083	0.010	0.007
PenglungEW	Avg	**1.000**	0.977	0.959	**1.000**	**1.000**	**1.000**	0.869	0.949	0.816	0.758	0.896
	StDev	0.068	0.000	0.039	0.000	0.000	0.000	0.012	0.054	0.054	0.025	0.007
SonarEW	Avg	**0.995**	0.952	0.962	0.944	0.948	0.993	0.887	0.865	0.814	0.836	0.923
	StDev	0.008	0.015	0.017	0.019	0.012	0.011	0.04	0.047	0.059	0.016	0.004
SpectEW	Avg	**0.941**	0.862	0.894	0.923	0.890	0.925	0.818	0.785	0.756	0.869	0.927
	StDev	0.017	0.007	0.010	0.010	0.013	0.011	0.029	0.034	0.039	0.062	0.004
CongressEW	Avg	**1.000**	0.961	0.976	0.999	0.993	**1.000**	0.952	0.943	0.869	0.954	0.981
	StDev	0.000	0.014	0.003	0.004	0.006	0.005	0.047	0.026	0.008	0.061	0.017
IonosphereEW	Avg	**0.993**	0.978	0.934	0.978	0.916	0.951	0.984	0.871	0.906	0.982	0.972
	StDev	0.004	0.013	0.006	0.018	0.023	0.007	0.036	0.044	0.012	0.072	0.004
KrvskpEW	Avg	**0.984**	0.962	0.918	0.932	0.946	0.901	0.979	0.813	0.969	0.972	0.955
	StDev	0.007	0.005	0.003	0.010	0.052	0.037	0.026	0.062	0.008	0.017	0.003
Tic-tac-toe	Avg	**0.869**	0.811	0.820	0.839	0.845	0.832	0.806	0.761	0.658	0.728	0.813

Table 10 (continued)

Dataset	Metric	IBSCA3	BSCA	ISSA	IHHO	OSACI	VNS-HRS	VNLHHO	IEOA	DSSA	SFS-LARLRM	BGWOPSO
	StDev	0.000	0.049	0.005	0.000	0.005	0.005	0.029	0.038	0.081	0.061	0.006
Vote	Avg	**0.998**	0.984	0.963	0.971	0.969	0.988	0.922	0.929	0.905	0.891	0.993
	StDev	0.004	0.038	0.009	0.018	0.047	0.014	0.053	0.082	0.005	0.067	0.003
WaveformEW	Avg	0.791	0.724	0.783	**0.854**	0.758	0.772	0.763	0.709	0.684	0.691	0.782
	StDev	0.013	0.018	0.015	0.012	0.009	0.038	0.026	0.047	0.013	0.023	0.019
WineEW	Avg	**1.000**	0.947	0.991	**1.000**	0.985	**1.000**	0.923	**1.000**	0.908	0.862	**1.000**
	StDev	0.000	0.049	0.016	0.000	0.009	0.000	0.027	0.000	0.083	0.099	0.000
Zoo	Avg	**1.000**	0.994	0.991	**1.000**	0.986	**1.000**	0.982	0.919	**1.000**	0.971	**1.000**
	StDev	0.000	0.028	0.007	0.000	0.005	0.000	0.019	0.007	0.000	0.005	0.000
COVID-19	Avg	**0.952**	0.894	0.915	0.949	0.927	0.916	0.893	0.918	0.939	0.872	0.945
	StDev	0.008	0.028	0.006	0.007	0.019	0.048	0.061	0.052	0.013	0.031	0.009

The results in bold point shows the best results in the table

local search strategy for feature selection in medical datasets (IEOA) [116], Dynamic salp swarm algorithm for feature selection (DSSA) [117], Semi-supervised feature selection with minimal redundancy based on local adaptive (SFS-LARLRM) [118] and Binary optimization using hybrid grey wolf optimization for feature selection (BGWOPSO) [119]. Table 9 shows the parameter settings of these algorithms, as in [111–119].

Table 10 shows a comparison of the average classification accuracy achieved by the proposed IBSCA3 algorithm, BSCA and the other algorithms that incorporate OBL, VNS and Laplace distribution. In Table 10, we report the average value and standard deviation of the results. Among all the datasets from UCI and COVID-19 functions (except one, where it is second best), IBSCA3 delivers the best classification accuracy.

5.5 Performance analysis of IBSCA3 compared to state-of-the-art SCA algorithms

A comparison of IBSCA3 with other SCA variants is presented in this section. These variants include: An efficient hybrid sine-cosine Harris Hawks Optimization for low and high-dimensional feature selection (SCHHO) [73], A novel feature selection method for data mining tasks using hybrid Sine Cosine Algorithm and Genetic Algorithm (SCAGA) [75], A Hybrid Feature Selection Framework Using Improved Sine Cosine Algorithm with Metaheuristic Techniques (MetaSCA) [120], A novel hybrid BPSO–SCA approach for feature selection (BPSO–SCA) [78], Boosting Salp Swarm Algorithm by Sine Cosine algorithm and Disrupt Operator for Feature Selection (ISSAFD), and An improved sine cosine algorithm to select features for text categorization (ISCA) [121]. Table 11 shows the parameter settings of these algorithms, as in [72, 73, 75, 78, 120, 121].

Table 12 displays the average classification accuracy of the proposed IBSCA3 algorithm, BSCA and the other state-of-the-art SCA algorithms. In Table 12, we report the average value and standard deviation of the results. IBSCA3 consistently outperforms other algorithms when applied to UCI and COVID-19 datasets. Based on the classification accuracy of IBSCA3 and these algorithms, we determined that it had the best performance.

Table 11 Parameter settings of SCHHO, SCAGA, MetaSCA, BPSO–SCA, ISSAFD and ISCA

Algorithm	Parameter settings
SCHHO	Population size = 10, Number of iterations = 100 , α = 2
SCAGA	Population size = 5, Number of iterations = 80, $pm = 0.02$ α = 0.01, and β = 0.99,
MetaSCA	Population size = 30, Number of iterations = 300
BPSO–SCA	Population size = 50, Number of iterations = 150, e1 = 1.5, e2 =1.5
ISSAFD	Population size = 10, Number of iterations = 100, $\gamma = 0.99$, $\mu = 0.01$, $c1 = 2$, $c2 = 2$,
$ps = 0.7$, $pm = 0.2$, $Rate = 0.8$	
ISCA	Population size = 30, Number of iterations = 0, a = 1, b = 8

Table 12 Average and standard deviation of classification accuracy for the proposed IBSCA3 algorithm in comparison to BSCA and the other SCA variants algorithms

Dataset	Metric	IBSCA3	BSCA	SCHHO	SCAGA	MetaSCA	BPSO–SCA	ISSAFD	ISCA
Breastcancer	Avg	**0.997**	0.965	0.936	0.957	0.921	0.906	0.983	0.891
	StDev	0.000	0.002	0.004	0.007	0.012	0.003	0.001	0.028
BreastEW	Avg	**1.000**	0.979	0.946	0.956	0.961	0.929	**1.000**	0.983
	StDev	0.000	0.005	0.011	0.016	0.007	0.009	0.000	0.031
Exactly	Avg	**1.000**	0.985	0.988	**1.000**	0.961	0.973	**1.000**	0.934
	StDev	0.000	0.037	0.005	0.000	0.003	0.001	0.000	0.014
Exactly2	Avg	**0.823**	0.783	0.751	0.701	0.687	0.723	0.816	0.656
	StDev	0.009	0.038	0.014	0.038	0.015	0.022	0.006	0.043
HeartEW	Avg	**0.926**	0.798	0.812	0.803	0.825	0.813	0.852	0.739
	StDev	0.041	0.047	0.078	0.067	0.082	0.079	0.052	0.066
Lymphography	Avg	**0.967**	0.814	0.918	0.857	0.912	0.931	0.953	0.836
	StDev	0.021	0.035	0.015	0.023	0.036	0.031	0.045	0.025
M-of-n	Avg	**1.000**	0.984	**1.000**	0.932	0.908	0.951	**1.000**	0.881
	StDev	0.000	0.005	0.000	0.011	0.008	0.016	0.000	0.037
PenglungEW	Avg	**1.000**	0.977	0.946	0.981	0.915	0.966	**1.000**	0.904
	StDev	0.068	0.000	0.019	0.023	0.035	0.017	0.006	0.052
SonarEW	Avg	**0.995**	0.952	0.931	0.926	0.951	0.961	0.988	0.917
	StDev	0.008	0.015	0.036	0.022	0.017	0.028	0.013	0.042
SpectEW	Avg	**0.941**	0.862	0.853	0.819	0.779	0.825	0.858	0.841
	StDev	0.017	0.007	0.015	0.027	0.019	0.064	0.019	0.032
CongressEW	Avg	**1.000**	0.961	0.959	0.912	0.942	0.938	0.955	0.917
	StDev	0.000	0.014	0.026	0.039	0.052	0.061	0.017	0.062
IonosphereEW	Avg	**0.993**	0.975	0.963	0.958	0.916	0.937	0.972	0.856
	StDev	0.004	0.013	0.021	0.035	0.042	0.051	0.019	0.063
KrvskpEW	Avg	**0.984**	0.962	0.954	0.936	0.947	0.925	0.961	0.899
	StDev	0.007	0.005	0.007	0.012	0.007	0.005	0.003	0.018
Tic-tac-toe	Avg	**0.869**	0.811	0.831	0.806	0.826	0.802	0.842	0.783
	StDev	0.000	0.049	0.031	0.027	0.016	0.024	0.002	0.053
Vote	Avg	**0.998**	0.984	0.975	0.943	0.922	0.941	0.987	0.916
	StDev	0.004	0.038	0.017	0.023	0.015	0.037	0.013	0.041
WaveformEW	Avg	**0.791**	0.724	0.739	0.718	0.725	0.776	0.748	0.616
	StDev	0.013	0.018	0.015	0.023	0.031	0.026	0.013	0.039
WineEW	Avg	**1.000**	0.947	**1.000**	0.959	0.936	0.920	**1.000**	0.886
	StDev	0.000	0.049	0.000	0.019	0.021	0.028	0.000	0.035
Zoo	Avg	**1.000**	0.994	**1.000**	0.986	0.974	0.938	**1.000**	0.926
	StDev	0.000	0.028	0.000	0.005	0.009	0.016	0.000	0.014
COVID-19	Avg	**0.952**	0.894	0.918	0.872	0.904	0.918	0.932	0.897
	StDev	0.008	0.028	0.016	0.033	0.024	0.018	0.011	0.007

The results in bold point shows the best results in the table

Table 13 Parameter settings of BFFAG, AVOA and GTO

Algorithm	Parameter settings
BFFAG	Population size = 10, Number of iterations = 50 , W = 1, Q = .7, R = 0.9
AVOA	Population size = 30, Number of iterations = 500, L1 = 0.8, L2 = 0.2, w = 2.5, p1 = 0.6, p2 = 0.4, p3 = 0.6
GTO	Population size = 30, Number of iterations = 500, $\beta = 3$, $W = 0.8$, $p = 0.03$

Table 14 Average and standard deviation of classification accuracy for the proposed IBSCA3 algorithm in comparison to BSCA and the other the other new nature-inspired metaheuristic algorithms

Dataset	Metric	IBSCA3	BSCA	BFFAG	AVOA	GTO
Breastcancer	Avg	**0.997**	0.965	0.972	0.985	0.994
	StDev	0.000	0.002	0.005	0.003	0.001
BreastEW	Avg	**1.000**	0.979	0.981	0.995	**1.000**
	StDev	0.000	0.005	0.009	0.006	0.005
Exactly	Avg	**1.000**	0.985	0.991	**1.000**	**1.000**
	StDev	0.000	0.037	0.006	0.000	0.000
Exactly2	Avg	**0.823**	0.783	0.809	0.815	0.822
	StDev	0.009	0.038	0.009	0.001	0.000
HeartEW	Avg	**0.926**	0.798	0.813	0.857	0.913
	StDev	0.041	0.047	0.027	0.018	0.007
Lymphography	Avg	**0.967**	0.814	0.933	0.951	0.959
	StDev	0.021	0.035	0.028	0.015	0.013
M-of-n	Avg	**1.000**	0.984	0.988	**1.000**	**1.000**
	StDev	0.000	0.005	0.005	0.000	0.000
penglungEW	Avg	**1.000**	0.977	0.962	**1.000**	**1.000**
	StDev	0.068	0.000	0.026	0.000	0.000
SonarEW	Avg	**0.995**	0.952	0.971	0.978	0.986
	StDev	0.008	0.015	0.013	0.011	0.009
SpectEW	Avg	**0.941**	0.862	0.885	0.919	0.937
	StDev	0.017	0.007	0.025	0.016	0.007
CongressEW	Avg	**1.000**	0.961	0.974	0.986	0.992
	StDev	0.000	0.014	0.010	0.005	0.002
IonosphereEW	Avg	**0.993**	0.975	0.979	0.988	0.991
	StDev	0.004	0.013	0.011	0.007	0.003
KrvskpEW	Avg	**0.984**	0.962	0.975	0.978	0.981
	StDev	0.007	0.005	0.010	0.008	0.005
Tic-tac-toe	Avg	**0.869**	0.811	0.836	0.852	0.861
	StDev	0.000	0.049	0.0041	0.003	0.001
Vote	Avg	**0.998**	0.984	0.987	0.991	0.997
	StDev	0.004	0.038	0.009	0.007	0.005
WaveformEW	Avg	**0.791**	0.724	0.753	0.779	0.788
	StDev	0.013	0.018	0.015	0.014	0.012
WineEW	Avg	**1.000**	0.947	0.995	**1.000**	**1.000**
	StDev	0.000	0.049	0.009	0.000	0.000

Table 14 (continued)

Dataset	Metric	IBSCA3	BSCA	BFFAG	AVOA	GTO
Zoo	Avg	**1.000**	0.992	0.996	**1.000**	**1.000**
	StDev	0.000	0.028	0.013	0.000	0.000
COVID-19	Avg	**0.952**	0.894	0.931	0.945	0.948
	StDev	0.008	0.028	0.007	0.005	0.004

The results in bold point shows the best results in the table

5.6 Performance analysis of IBSCA3 compared to other new nature-inspired metaheuristic algorithms

This section shows a comparison between IBSCA3 and other new nature-inspired metaheuristic algorithms, including: A novel Binary Farmland Fertility Algorithm (BFFAG) [122], African vultures optimization algorithm (AVOA) [123] and Artificial gorilla troops optimizer (GTO) [124]. Table 13 shows the parameter settings of these algorithms, as in [122–124].

A comparison of the average classification accuracy achieved by the proposed IBSCA3 algorithm, BSCA and the other new nature-inspired metaheuristic algorithms is shown in Table 14, where we report the average value and standard deviation of the results. In all datasets from UCI and COVID-19, IBSCA3 delivers the best classification accuracy.

Table 15 Runtime Performance Comparison for the proposed IBSCA3 algorithm in comparison to existing algorithms

Dataset	IBSCA3	BSCA	RBDA	LBDA	QBDA	SBDA	BGWO	BGSA	BBA
Breastcancer	**9.18E+03**	1.12E+04	1.08E+04	1.03E+04	1.59E+04	1.09E+04	1.21E+04	1.48E+04	1.28E+04
BreastEW	**1.61E+03**	2.74E+03	1.89E+03	1.93E+03	1.86E+03	1.97E+03	2.95E+03	2.34E+03	2.91E+03
Exactly	**1.29E+04**	1.41E+04	1.34E+04	1.37E+04	1.36E+04	1.32E+04	1.49E+04	1.54E+04	1.58E+04
Exactly2	**1.15E+04**	1.46E+04	1.25E+04	1.28E+04	1.23E+03	1.21E+04	1.53E+04	1.62E+04	1.57E+04
HeartEW	**5.13E+03**	6.19E+03	5.68E+03	5.08E+03	5.72E+03	6.02E+03	7.01E+03	8.72E+03	8.31E+03
Lymphography	**3.19E+03**	3.84E+03	3.57E+03	3.40E+03	3.91E+03	3.55E+03	4.16E+03	5.83E+03	4.07E+03
M-of-n	**9.12E+03**	1.29E+04	9.38E+03	8.49E+03	9.54E+03	9.78E+03	1.14E+04	1.09E+04	1.28E+04
PenglungEW	**3.47E+03**	3.98E+03	4.06E+03	4.15E+03	3.83E+03	3.65E+03	4.32E+03	5.79E+03	3.84E+03
SonarEW	**3.56E+03**	4.39E+03	3.83E+03	3.90E+03	4.01E+03	3.16E+03	4.72E+03	5.09E+03	4.87E+03
SpectEW	**4.61E+03**	5.82E+03	4.89E+03	4.93E+03	4.75E+03	4.87E+03	6.08E+03	5.74E+03	5.18E+03
CongressEW	**1.14E+04**	1.31E+04	1.19E+04	1.17E+04	1.18E+04	1.21E+04	1.53E+04	1.74E+04	1.43E+04
IonosphereEW	**4.36E+03**	5.98E+03	5.32E+03	5.74E+03	5.17E+03	5.06E+03	6.04E+03	7.01E+03	6.81E+03
KrvskpEW	**5.73E+04**	6.69E+04	6.06E+04	6.21E+04	5.92E+04	6.17E+04	8.14E+04	7.69E+04	7.83E+04
Tic-tac-toe	**1.26E+04**	1.66E+04	1.49E+04	1.37E+04	1.26E+04	1.39E+04	1.77E+04	1.53E+04	1.68E+04
Vote	**5.96E+03**	6.58E+03	6.97E+03	6.01E+03	6.86E+03	6.38E+03	8.34E+03	7.44E+03	7.81E+03
WaveformEW	**1.48E+04**	1.72E+04	1.51E+04	1.67E+04	1.56E+04	1.53E+04	1.87E+04	1.79E+04	1.78E+04
WineEW	**1.09E+03**	1.57E+03	1.28E+03	1.20E+03	1.14E+03	1.16E+03	2.51E+03	1.86E+03	2.01E+03
Zoo	**4.52E+03**	5.68E+03	4.79E+03	5.02E+03	4.97E+03	5.08E+03	6.42E+03	5.69E+03	6.83E+03

The results in bold point shows the best results in the table

Table 16 Runtime Performance Comparison for the proposed IBSCA3 algorithm in comparison to the other algorithms that incorporate OBL, VNS and Laplace distribution

Dataset	IBSCA3	ISSA	IHHO	OSACI	VNS-HRS	VNLHHO	IEOA	DSSA	SFS-LARLRM	BGWOPSO
Breastcancer	**9.18E+03**	1.03E+04	1.14E+04	1.22E+04	1.31E+04	1.10E+04	1.37E+04	1.61E+04	1.19E+04	1.26E+04
BreastEW	**1.61E+03**	2.48E+03	3.11E+03	3.64E+03	3.92E+03	3.07+03	3.98E+03	3.87E+03	3.25E+03	3.47E+03
Exactly	**1.29E+04**	1.53E+04	1.76E+04	1.88E+04	2.01E+04	1.74E+04	2.16E+04	1.99E+04	1.74E+04	1.82E+04
Exactly2	**1.15E+04**	1.57E+04	1.78E+04	1.91E+04	2.13E+03	1.38E+04	2.36E+04	2.08E+04	1.84E+04	1.93E+04
HeartEW	**5.13E+03**	6.38E+03	6.77+03	7.12E+03	7.49E+03	6.15E+03	7.36E+03	7.23E+03	6.55E+03	6.08E+03
Lymphography	**3.19E+03**	3.51E+03	3.66E+03	3.91E+03	4.02E+03	3.24E+03	3.86E+03	3.41E+03	3.51E+03	3.27E+03
M-of-n	**9.12E+03**	1.35E+04	1.42E+04	1.58E+04	1.71E+04	9.918E+03	1.16E+04	1.02E+04	1.11E+04	1.09E+04
PenglungEW	**3.47E+03**	3.63E+03	4.12E+03	4.27E+03	4.61E+03	3.51E+03	3.62E+03	3.76E+03	3.83E+03	3.57E+03
SonarEW	**3.56E+03**	3.74E+03	3.95E+03	4.28E+03	4.71E+03	3.68E+03	3.71E+03	3.73E+03	3.85E+03	3.79E+03
SpectEW	**4.61E+03**	4.75E+03	4.91E+03	4.95E+03	5.01E+04	4.69E+03	4.70E+03	4.82E+03	4.68E+03	4.65E+03
CongressEW	**1.14E+04**	1.35+04	1.62E+04	1.77E+04	1.91E+04	1.54E+04	1.62E+04	1.67E+04	1.79E+04	1.29E+04
IonosphereEW	**4.36E+03**	4.76E+03	4.91E+03	5.03E+03	5.12E+03	4.88E+03	4.97E+03	5.02E+03	5.11E+03	4.52E+03
KrvskpEW	**5.73E+04**	6.04E+04	6.18E+04	6.44E+04	6.01E+04	5.81E+04	5.96E+04	6.03E+04	6.12E+04	5.98E+04
Tic-tac-toe	**1.26E+04**	1.72E+04	1.89E+04	1.97E+04	1.85E+04	1.91E+04	1.98E+04	2.02E+04	1.94E+04	1.79E+04
Vote	**5.96E+03**	6.06E+03	6.28E+03	6.74E+03	6.91E+03	6.33E+03	6.85E+03	6.91E+03	6.18E+03	6.03E+03
WaveformEW	**1.48E+04**	1.57E+04	1.68E+04	1.79E+04	1.72E+04	1.81E+04	1.66E+04	1.71E+04	1.61E+04	1.59E+04
WineEW	**1.09E+03**	1.27E+03	1.35E+03	1.49E+03	1.58E+03	1.41E+03	1.53E+03	1.62E+03	1.58E+03	1.32E+03
Zoo	**4.52E+03**	5.07E+03	5.18E+03	5.29E+03	5.12E+03	5.02E+03	5.09E+03	5.12E+03	5.07E+03	4.98E+03
COVID-19	**1.06E+03**	1.48E+03	1.59E+03	1.67E+03	1.52E+03	1.64E+03	1.72E+03	1.63E+03	1.56E+03	1.50E+03

The results in bold point shows the best results in the table

Table 17 Runtime Performance Comparison for the proposed IBSCA3 algorithm in comparison to the other SCA variants algorithms

Dataset	IBSCA3	SCHHO	SCAGA	MetaSCA	BPSO-SCA	ISSAFD	ISCA
Breastcancer	**9.18E+03**	1.15E+04	1.36E+04	1.89E+04	1.53E+04	1.94E+04	1.97E+04
BreastEW	**1.61E+03**	1.78E+03	1.95E+03	2.04E+03	1.98E+03	2.14E+03	2.35E+03
Exactly	**1.29E+04**	1.42E+04	1.49E+04	1.45E+04	1.53E+04	1.62E+04	1.73E+04
Exactly2	**1.15E+04**	1.23E+04	1.35E+04	1.41E+04	1.29E+03	1.39E+04	1.72E+04
HeartEW	**5.13E+03**	5.92E+03	6.01E+03	6.15E+03	6.12E+03	6.29E+03	6.43E+03
Lymphography	**3.19E+03**	3.97E+03	4.05E+03	4.16E+03	4.07E+03	4.15E+03	4.38E+03
M-of-n	**9.12E+03**	1.14E+04	1.20E+03	1.32E+03	1.25E+03	1.46E+03	1.76E+04
PenglungEW	**3.47E+03**	3.52E+03	3.75E+03	4.01E+03	3.87E+03	3.99E+03	4.26E+03
SonarEW	**3.56E+03**	4.16E+03	4.52E+03	4.91E+03	5.12E+03	5.16E+03	4.37E+03
SpectEW	**4.61E+03**	5.01E+03	5.13E+03	5.29E+03	5.22E+03	5.36E+03	5.44E+03
CongressEW	**1.14E+04**	1.27E+04	1.38E+04	1.31E+04	1.41E+04	1.59E+04	1.71E+04

Table 17 (continued)

Dataset	IBSCA3	SCHHO	SCAGA	MetaSCA	BPSO-SCA	ISSAFD	ISCA
IonosphereEW	**4.36E+03**	5.02E+03	5.19E+03	5.12E+03	5.35E+03	5.42E+03	5.91E+03
KrvskpEW	**5.73E+04**	5.79E+04	6.16E+04	6.27E+04	6.21E+04	6.39E+04	6.45E+04
Tic-tac-toe	**1.26E+04**	1.39E+04	1.45E+04	1.41E+04	1.47E+04	1.67E+04	1.95E+04
Vote	**5.96E+03**	6.18E+03	6.34E+03	6.27E+03	6.56E+03	6.67E+03	8.78E+03
WaveformEW	**1.48E+04**	1.59E+04	1.68E+04	1.61E+04	1.76E+04	1.83E+04	2.03E+04
WineEW	**1.09E+03**	1.15E+03	1.31E+03	1.46E+03	1.36E+03	1.56E+03	1.96E+03
Zoo	**4.52E+03**	4.70E+03	4.83E+03	4.74E+03	5.03E+03	5.16E+03	5.65E+03
COVID-19	**1.06E+03**	1.32E+03	1.46E+03	1.61E+03	2.01E+03	2.15E+03	2.05E+03

The results in bold point shows the best results in the table

Table 18 Runtime Performance Comparison for the proposed IBSCA3 algorithm in comparison to the other new nature-inspired metaheuristic algorithms

Dataset	IBSCA3	BFFAG	AVOA	GTO
Breastcancer	**9.18E+03**	2.28E+04	1.86E+04	1.12E+04
BreastEW	**1.61E+03**	2.91E+03	1.75E+03	1.67E+03
Exactly	**1.29E+04**	1.92E+04	1.85E+04	1.61E+04
Exactly2	**1.15E+04**	1.81E+04	1.72E+04	1.41E+04
HeartEW	**5.13E+03**	6.25E+03	6.15E+03	5.92E+03
Lymphography	**3.19E+03**	4.59E+03	4.39E+03	3.97E+03
M-of-n	**9.12E+03**	1.96E+04	1.64E+04	1.32E+04
PenglungEW	**3.47E+03**	4.01E+03	3.98E+03	3.62E+03
SonarEW	**3.56E+03**	4.16E+03	4.02E+03	3.77E+03
SpectEW	**4.61E+03**	5.29E+03	5.13E+03	4.84E+03
CongressEW	**1.14E+04**	1.64E+04	1.51E+04	1.37E+04
IonosphereEW	**4.36E+03**	5.06E+03	5.01E+03	4.91E+03
KrvskpEW	**5.73E+04**	6.37E+04	6.24E+04	6.05E+04
Tic-tac-toe	**1.26E+04**	1.72E+04	1.61E+04	1.59E+04
Vote	**5.96E+03**	6.37E+03	6.26E+03	5.99E+03
WaveformEW	**1.48E+04**	1.75E+04	1.71E+04	1.58E+04
WineEW	**1.09E+03**	1.93E+03	1.76E+03	1.41E+03
Zoo	**4.52E+03**	5.23E+03	5.14E+03	4.92E+03
COVID-19	**1.06E+03**	1.48E+03	1.57E+03	1.62E+04

The results in bold point shows the best results in the table

Table 19 Friedman's test when comparing IBSCA3 with existing algorithms based on classification accuracy (Table 6)

	Ranks of the algorithms								
Dataset	BSCA	RBDA	LBDA	QBDA	SBDA	BGWO	BGSA	BBA	IBSCA3
Breastcancer	7	4	5.5	2.5	2.5	5.5	8	9	**1**
BreastEW	5	**1.5**	3	4	6	8	7	9	**1.5**
Exactly	6	**2.5**	**2.5**	5	**2.5**	7	8	9	**2.5**
Exactly2	4	2	5	3	6	8	9	7	**1**
HeartEW	6	5	2	3	4	7	8	9	**1**
Lymphography	8	3	5	4	2	7	6	9	**1**
M-of-n	6	**2.5**	**2.5**	5	**2.5**	7	8	9	**2.5**
PenglungEW	5	6	**2.5**	**2.5**	**2.5**	8	7	9	**2.5**
SonarEW	4	3	6	5	2	7	8	9	**1**
SpectEW	6	4	3	5	2	7	8	9	**1**
CongressEW	6	4	2	3	5	7	8	9	**1**
IonosphereEW	3	4.5	4.5	6	2	7	8	9	**1**
KrvskpEW	6	3	2	4	5	7	8	9	**1**
Tic-tac-toe	6	5	3	2	4	7	8	9	**1**
Vote	3	2	5	6	4	8	7	9	**1**
WaveformEW	6	3	4	5	2	7	8	9	**1**
WineEW	7	5	**2.5**	**2.5**	**2.5**	8	6	9	**2.5**
Zoo	7	**3**	**3**	**3**	**3**	8	6	9	**3**
Sum of ranks	101	63	63	70.5	59.5	130.5	136	160	26.5
Sum of ranks squared	10201	3969	3969	4970.25	3540.25	17030.25	18496	25600	702.25
Average of ranks	5.61	3.5	3.5	3.92	3.31	7.25	7.56	8.89	1.47

The results in bold point shows the best results in the table

Consequently, the overall results summarized in all different sets of experiments indicate the strength of the IBSCA3 algorithm in improving the performance and convergence behavior of the original SCA when solving the FS problem.

5.7 Runtime performance comparison of IBSCA3 to existing algorithms

Tables 15, 16, 17, 18 provide the running time comparison of IBSCA3, BSCA, and the other algorithms described in Tables 6, 10, 12, and 14, respectively. The results are given in milliseconds, representing an average of 30 independent runs. For each algorithm in the tables, the values in the tables represent the run time to obtain the results after 100 iterations. As shown in the tables, IBSCA3 is faster than the other algorithms when applied to all datasets.

The experiments were conducted using an Intel Core i7-3517U, 1.90 GHz CPU with 16 GB RAM running 64-bit Windows. All the algorithms were implemented using Python programming language.

5.8 Statistical test results

An investigation of the significance of the results in Tables 6, 10, 12, and 14 has been conducted. We applied both Friedman's test and Wilcoxon's test [125] to the classification accuracy in the tables with $\alpha = 0.05$. Tables 19, 20, 21 and 22 present the results of the Friedman's test. The best ranks in each row are highlighted in bold. The average ranks of the algorithms were as follows (best to worst): In Table 19: IBSCA3, SBDA, LBDA, RBDA, QBDA, BSCA. BGWO, BGSA, and BBA. In Table 20: IBSCA3, VNS-HRS, IHHO, BGWOPSO, OSACI, ISSA, VNLHHO, SFS-LARLRM, IEOA, and DSSA. In Table 21: IBSCA3, ISSAFD, SCHHO, BPSO-SCA, SCAGA, MetaSCA, and ISCA. In Table 22: IBSCA3, GTO, AVOA, and BFFAG.

It is clear from the results that IBSCA3 achieves the best rank over 12 datasets, and competitive results for the other datasets. Therefore, IBSCA3 is the best in terms of the average of ranks among the other compared algorithms.

Table 20 Friedman's test when comparing IBSCA3 with the other algorithms that incorporate OBL, VNS and Laplace distribution based on classification accuracy (Table 10)

	Ranks of the algorithms									
Dataset	ISSA	IHHO	OSACI	VNS-HRS	VNLHHO	IEOA	DSSA	SFS-LARLRM	BGWOPSO	IBSCA3
Breastcancer	8	5	4	3	9	7	10	2	6	**1**
BreastEW	7	**2**	**2**	5	8	9	10	6	4	**2**
Exactly	8	**4**	9	**4**	**4**	10	**4**	**4**	**4**	**4**
Exactly2	6	9	7	5	2	10	8	3	4	**1**
HeartEW	4	10	7	9	6	2	8	3	5	**1**
Lymphography	3	5	4	2	7	6	10	9	8	**1**
M-of-n	**2.5**	**2.5**	5	**2.5**	6	8	10	9	7	**2.5**
PenglungEW	5	**2.5**	**2.5**	**2.5**	8	6	9	10	7	**2.5**
SonarEW	3	5	4	2	7	8	10	9	6	**1**
SpectEW	5	4	6	3	8	9	10	7	2	**1**
CongressEW	6	3	4	**1.5**	8	9	10	7	5	**1.5**
IonosphereEW	7	4	8	6	2	10	9	3	5	**1**
KrvskpEW	8	7	6	9	2	10	4	3	5	**1**
Tic-tac-toe	5	3	2	4	7	8	10	9	6	**1**
Vote	6	4	5	3	8	7	9	10	2	**1**
WaveformEW	3	**1**	7	5	6	8	10	9	4	2
WineEW	6	**3**	7	**3**	8	**3**	9	10	**3**	**3**
Zoo	6	**3**	7	**3**	8	10	**3**	9	**3**	**3**
COVID-19	8	2	5	7	9	6	4	10	3	**1**
Sum of ranks	31.5	106.5	79	101.5	79.5	123	146	157	132	89
Sum of ranks squared	11342.25	6241	10302.25	6320.25	15129	21316	24649	17424	7921	992.25
Average of ranks	5.47	4.28	5.36	4.03	6.33	7.28	8.5	6.78	4.78	1.69

The results in bold point shows the best results in the table

Table 21 Friedman's test when comparing IBSCA3 with the other SCA variants algorithms based on classification accuracy (Table 12)

	Ranks of the algorithms						
Dataset	SCHHO	SCAGA	MetaSCA	BPSO-SCA	ISSAFD	ISCA	IBSCA3
Breastcancer	4	3	5	6	2	7	**1**
BreastEW	6	5	4	7	**1.5**	3	**1.5**
Exactly	4	**2**	6	5	**2**	7	**2**
Exactly2	3	5	6	4	2	7	**1**
HeartEW	5	6	3	4	2	7	**1**
Lymphography	4	6	5	3	2	7	**1**
M-of-n	**2**	5	6	4	**2**	7	**2**
PenglungEW	5	3	6	4	**1.5**	7	**1.5**
SonarEW	5	6	4	3	2	7	**1**
SpectEW	3	6	7	5	2	4	**1**
CongressEW	2	7	4	5	3	6	**1**
IonosphereEW	3	4	6	5	2	7	**1**
KrvskpEW	3	5	4	6	2	7	**1**
Tic-tac-toe	3	5	4	6	2	7	**1**
Vote	3	4	6	5	2	7	**1**

Table 21 (continued)

Dataset	Ranks of the algorithms						
	SCHHO	SCAGA	MetaSCA	BPSO-SCA	ISSAFD	ISCA	IBSCA3
WaveformEW	4	6	5	2	3	7	**1**
WineEW	**2**	4	5	6	**2**	7	**2**
Zoo	**2**	4	5	6	**2**	7	**2**
COVID-19	3.5	7	5	6	**2**	7	**2**
Sum of ranks	66.5	93	96	89.5	39	124	24
Sum of ranks squared	4422.25	8649	9216	8010.25	1521	15376	576
Average of ranks	3.5	4.89	5.05	4.71	2.05	6.53	1.69

The results in bold point shows the best results in the table

Table 22 Friedman's test when comparing IBSCA3 with the other new nature-inspired metaheuristic algorithms based on classification accuracy (Table 14)

Dataset	Ranks of the algorithms			
	BFFAG	AVOA	GTO	IBSCA3
Breastcancer	4	3	2	**1**
BreastEW	4	3	**1.5**	**1.5**
Exactly	4	**2**	**2**	**2**
Exactly2	4	3	2	**1**
HeartEW	4	3	2	**1**
Lymphography	4	3	2	**1**
M-of-n	4	**2**	**2**	**2**
PenglungEW	4	**2**	**2**	**2**
SonarEW	4	3	2	**1**
SpectEW	4	3	2	**1**
CongressEW	4	3	2	**1**
IonosphereEW	4	3	2	**1**
KrvskpEW	4	3	2	**1**
Tic-tac-toe	4	3	2	**1**
Vote	4	3	2	**1**
WaveformEW	4	3	2	**1**
WineEW	4	**2**	**2**	**2**
Zoo	4	**2**	**2**	**2**
COVID-19	4	3	2	**1**
Sum of ranks	76	52	37.5	24.5
Sum of ranks squared	5776	2704	1406.25	600.25
Average of ranks	4	2.74	1.97	1.29

The results in bold point shows the best results in the table

Table 23 Wilcoxon's test results when comparing IBSCA3 with existing algorithms based on classification accuracy (Table 6)

Algorithm	BSCA	RBDA	LBDA	QBDA	SBDA	BGWO	BGSA	BBA
p-values	0.00328	0.00096	0.00148	0.00064	0.00148	0.00020	0.00020	0.00020

Table 24 Wilcoxon's test results when comparing IBSCA3 with the other algorithms that incorporate OBL, VNS and Laplace distribution based on classification accuracy (Table 10)

Algorithm	ISSA	IHHO	OSACI	VNS-HRS	VNLHHO	IEOA	DSSA	SFS-LARLRM	BGWOPSO
p-values	0.00020	0.01596	0.00030	0.00148	0.00020	0.00020	0.00030	0.00020	0.00044

We also conducted the Wilcoxon's test with $\alpha = 0.05$ as summarized in Tables 23, 24, 25 and 26 to evaluate the data in Tables 6, 10, 12, and 14, respectively. Our purpose here is to evaluate the significance of the values of the classification accuracy of IBSCA3 compared to the other algorithms in the tables. The reported *p-values* indicate that the values of the classification accuracy of IBSCA3 are statistically significant compared to the values of the other algorithms.

In addition, we used Mann-Whitney U test to compare IBSCA3 against all other algorithms. Based on the results, IBSCA3 produces significant results compared to the other algorithms except for IHHO (0.28014), VNS-HRS (0.35758), BGWOPSO (0.0536), AVOA (0.39532), and GTO (0.65272).

Accordingly, the statistical analysis gives evidence that the included modifications of IBSCA3 improve its search strategy, as compared to the original SCA algorithm, and thus achieves the highest accuracy for most of the datasets.

6 Conclusion and future work

This paper introduced three versions of a binary optimization algorithm by the name of Improved Binary Sine Cosine Algorithm with multiple exploration and exploitation approaches (IBSCA) for solving the Feature Selection problem. All versions of IBSCA (IBSCA1, IBSCA2, IBSCA3) employ an opposition-based learning approach in their initialization stage to generate a diverse population of candidate solutions. IBSCA2 and IBSCA3 use a combination of the variable neighborhood search and Laplace distribution to explore the search space using several mutation methods. Further, IBSCA3 improves the best candidate solution using Refraction Learning, which is a novel opposition learning approach that is based on the principle of light refraction. All versions of IBSCA use two-step transfer functions to convert continuous decision variables into binary ones.

The three versions of IBSCA were compared with each other using 18 FS datasets from UCI repository and one COVID-19 dataset. These datasets are suitable for comparison because the numbers of features, objects and classes in these datasets vary significantly. IBSCA3 was found to be the most efficient version of IBSCA. Furthermore, the performance of IBSCA3 was evaluated and compared to several popular binary algorithms (RBDA, LBDA, QBDA, SBDA, BGWO, BGSA, BBA, CHIO, CHIO-GC, ISSA, IHHO, OSACI, VNS-HRS, VNLHHO, IEOA, DSSA, SFS-LARLRM, BGWOPSO, SCHHO, SCAGA, MetaSCA, BPSO–SCA, ISSAFD, ISCA, BFFAG, AVOA, GTO) using the 18 FS datasets from UCI repository and a COVID-19 dataset. The overall simulation results indicate that IBSCA3 outperformed all comparative algorithms in terms of accuracy and number of features selected over most datasets.

It is worth mentioning that the performance of IBSCA is affected by the limitations of its methods. To begin, OBL and RL tend to generate good solutions at the beginning of the optimization process, but the generated solutions may converge to sub-optimality as the optimization process progresses [98]. Besides, every optimization problem requires a special OBL strategy that is suitable for the problem structure. In other words, there are no clear guidelines for designing OBL strategies for different optimization problems [126, 127]. Secondly, if the VNS method is implemented too frequently, the population of solutions could be spread over a larger area than necessary [128].

In the future, we are interested in conducting two research studies based on IBSCA3. We are going to apply IBSCA3 to multi-agent cooperative reinforcement learning

Table 25 Wilcoxon's test results when comparing IBSCA3 with the other SCA variants algorithms based on classification accuracy (Table 12)

Algorithm	SCHHO	SCAGA	MetaSCA	BPSO-SCA	ISSAFD	ISCA
p-values	0.00044	0.00020	0.00014	0.00014	0.00148	0.00014

Table 26 Wilcoxon's test results when comparing IBSCA3 with the other new nature-inspired optimization algorithms based on classification accuracy (Table 14)

Algorithm	BFFAG	AVOA	GTO
p-values	0.00014	0.00096	0.00148

[129, 130] based on the models described in [131, 132]. We also plan to incorporate the island model [96, 133–137] with IBSCA3 to further improve its performance over the FS problem. Applying the proposed methods on other FS applications can also be addressed in future work.

Declarations

Competing interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Conflict of Interests The authors declare that they have no conflict of interest.

Human and animal rights This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Bolón-Canedo V, Sánchez-maroño N, Alonso-Betanzos A (2015) Recent advances and emerging challenges of feature selection in the context of big data. Knowledge-based systems 86:33–45
2. Fayyad U, Piatetsky-Shapiro G, Smyth P (1996) From data mining to knowledge discovery in databases. AI magazine 17(3):37
3. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3(Mar):1157–1182
4. Hua J, Tembe WD, Dougherty ER (2009) Performance of feature-selection methods in the classification of high-dimension data. Pattern Recognit 42(3):409–424
5. Gómez-Verdejo V, Verleysen M, Fleury J (2009) Information-theoretic feature selection for functional data classification. Neurocomputing 72(16-18):3580–3589
6. Al-Abdallah RZ, Jaradat AS, Doush IA, Jaradat YA (2017) A binary classifier based on firefly algorithm. Jordanian J Comput Inf Technol (JJCIT), vol 3(3)
7. Liu H, Yu L (2005) Toward integrating feature selection algorithms for classification and clustering. IEEE Trans Knowl Data Eng, (4):491–502
8. Boutemedjet S, Bouguila N, Ziou D (2008) A hybrid feature extraction selection approach for high-dimensional non-gaussian data clustering. IEEE Trans Pattern Anal Mach Intell 31(8):1429–1443
9. ElMustafa S, Jaradat A, Doush IA, Mansour N (2017) Community detection using intelligent water drops optimisation algorithm. Int J Reasoning-Based Intell Syst 9(1):52–65
10. Ke H, Aviyente S (2008) Wavelet feature selection for image classification. IEEE Trans Image Process 17(9):1709–1720
11. Chen Bo, Chen L, Chen Y (2013) Efficient ant colony optimization for image feature selection. Signal Process 93(6):1566–1576
12. Sawalha R, Doush IA (2012) Face recognition using harmony search-based selected features. Int J Hybrid Inf Technol 5(2):1–16
13. AbuNaser A, Doush IA, Mansour N, Alshattnawi S (2015) Underwater image enhancement using particle swarm optimization. J Intell Syst 24(1):99–115
14. Shang W, Huang H, Zhu H, Lin Y, Qu Y, Wang Z (2007) A novel feature selection algorithm for text categorization. Expert Syst Appl 33(1):1–5
15. Zheng Z, Wu X, Srihari R (2004) Feature selection for text categorization on imbalanced data. ACM Sigkdd Explorations Newsletter 6(1):80–89
16. Liu H, Setiono R (1995) Chi2: feature selection and discretization of numeric attributes. In: Proceedings of 7th IEEE international conference on tools with artificial intelligence. IEEE, pp 388–391
17. Quinlan JR (2014) C4. 5: programs for machine learning. Elsevier
18. Quinlan JR (1986) Induction of decision trees. Machine learning 1(1):81–106
19. Kandaswamy KK, Pugalenthi G, Hazrati MK, Kalies K-U, Martinetz T (2011) Blprot: prediction of bioluminescent proteins based on support vector machine and relieff feature selection. BMC soinformatics 12(1):345
20. Robnik-Šikonja M, Kononenko I (2003) Theoretical and empirical analysis of relieff and rrelieff. Mach Learn 53(1):23–69
21. Le TT, Urbanowicz RJ, Moore JH, McKinney BA (2019) Statistical inference relief (stir) feature selection. Bioinformatics 35(8):1358–1365
22. Huang Z, Yang C, Zhou X, Huang T (2018) A hybrid feature selection method based on binary state transition algorithm and relieff. IEEE J Biomed Health Inf 23(5):1888–1898
23. Deng Z, Chung F-L, Wang S (2010) Robust relief-feature weighting, margin maximization, and fuzzy optimization. IEEE Trans Fuzzy Syst 18(4):726–744
24. Kohavi R, John GH (1997) Wrappers for feature subset selection. Artif Intell 97(1-2):273–324
25. Hossam Faris, Ala'M A-Z, Heidari AA, Aljarah I, Mafarja M, Hassonah MA, Fujita H (2019) An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks. Inf Fusion 48:67–83
26. Chantar H, Mafarja M, Alsawalqah H, Heidari AA, Aljarah I, Faris H (2019) Feature selection using binary grey wolf optimizer with elite-based crossover for arabic text classification. Neural Comput Appl:1–20
27. Zelinka I (2015) A survey on evolutionary algorithms dynamics and its complexity–mutual relations, past, present and future. Swarm Evolution Comput 25:2–14
28. Gharehchopogh FS, Shayanfar H, Gholizadeh H (2020) A comprehensive survey on symbiotic organisms search algorithms. Artif Intell Rev 53(3):2265–2312
29. Mafarja M, Mirjalili S (2017) Whale optimization approaches for wrapper feature selection. Appl Soft Comput 62:441–453
30. Soleimanian GF, Gholizadeh H (2019) A comprehensive survey: whale optimization algorithm and its applications. Swarm Evolution Comput 48:1–24

31. Gharehchopogh FS, Maleki I, Dizaji ZA (2022) Chaotic vortex search algorithm: metaheuristic algorithm for feature selection. Evolution Intell 15(3):1777–1808
32. Turkoglu B, Kaya E (2020) Training multi-layer perceptron with artificial algae algorithm. Eng Sci Technol Int J 23(6):1342–1350
33. Turkoglu B, Uymaz SA, Kaya E (2022) Clustering analysis through artificial algae algorithm. Int J Mach Learn Cybern 13(4):1179–1196
34. Rahnema N, Gharehchopogh FS (2020) An improved artificial bee colony algorithm based on whale optimization algorithm for data clustering. Multimed Tools Appl 79(43):32169–32194
35. Tu Q, Chen X, Liu X (2019) Multi-strategy ensemble grey wolf optimizer and its application to feature selection. Appl Soft Comput 76:16–30
36. Mirjalili S (2015) The ant lion optimizer. Adv Eng Softw 83:80–98
37. Mafarja M, Mirjalili S (2017) Hybrid whale optimization algorithm with simulated annealing for feature selection. Neurocomputing 260:302–312
38. Mafarja M, Mirjalili S (2018) Whale optimization approaches for wrapper feature selection. Appl Soft Comput 62:441–453
39. Mafarja M, Jaber I, Ahmed S, Thaher T (2019) Whale optimisation algorithm for high-dimensional small-instance feature selection. Int J Parallel Emergent Distributed Syst:1–17
40. Hussien AG, Hassanien AE, Houssein EH, Bhattacharyya S, Amin M (2019) S-shaped binary whale optimization algorithm for feature selection. In: Bhattacharyya S, Mukherjee A, Bhaumik H, Das S, Yoshida K (eds) Recent trends in signal and image processing. Springer Singapore, pp 79–87, Singapore
41. Zaman HRR, Gharehchopogh FS (2021) An improved particle swarm optimization with backtracking search optimization algorithm for solving continuous optimization problems. Eng Comput:1–35
42. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. J Global Optimization 39(3):459–471
43. Dorigo M, Maniezzo V, Colorni A (1996) Ant system: optimization by a colony of cooperating agents. IEEE Trans Syst Man Cybern Part B (Cybernetics) 26(1):29–41
44. Turabieh H, Mafarja M, Li X (2018) Iterated feature selection algorithms with layered recurrent neural network for software fault prediction. Expert Syst Appl 122:27–42
45. Taradeh M, Mafarja M, Heidari AA, Faris H, Aljarah I, Mirjalili S, Fujita H (2019) An evolutionary gravitational search-based feature selection. Inf Sci
46. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw
47. Faris H, Heidari AA, Ala'M A-Z, Mafarja M, Aljarah I, Eshtay M, Mirjalili S (2020) Time-varying hierarchical chains of salps with random weight networks for feature selection. Expert Syst Appl 140:112898
48. Neggaz N, Ewees AA, Elaziz MA, Mafarja M (2020) Boosting salp swarm algorithm by sine cosine algorithm and disrupt operator for feature selection. Expert Syst Applications 145:113103
49. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evolution Computat 1(1):67–82
50. Mirjalili S (2016) Sca: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133
51. Gholizadeh S, Sojoudizadeh R (2019) Modified sine-cosine algorithm for sizing optimization of truss structures with discrete design variables. Iran Univ Sci Technol 9(2):195–212
52. Tawhid MA, Savsani P (2019) Discrete sine-cosine algorithm (dsca) with local search for solving traveling salesman problem. Arabian J Sci Eng 44(4):3669–3679
53. Belazzoug M, Touahria M, Nouioua F, Brahimi M (2019) An improved sine cosine algorithm to select features for text categorization. J King Saud Univ-Comput Inf Sci 32(4):454–464
54. Oliva D, Hinojosa S, Elaziz MA, Ortega-sánchez N (2018) Context based image segmentation using antlion optimization and sine cosine algorithm. Multimed Tools Appl 77(19):25761–25797
55. Nenavath H, Jatoth RK, Das S (2018) A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking. Swarm Evolution Computat 43:1–30
56. Reddy KS, Kumar PL, Panigrahi BK, Kumar R (2018) A new binary variant of sine–cosine algorithm: development and application to solve profit-based unit commitment problem. Arabian J Sci Eng 43(8):4041–4056
57. Turgut OE (2017) Thermal and economical optimization of a shell and tube evaporator using hybrid backtracking search—sine–cosine algorithm. Arabian J Sci Eng 42(5):2105–2123
58. Zhang H, Gao Z, Zhang J, Liu J, Nie Z, Zhang J (2020) Hybridizing extended ant lion optimizer with sine cosine algorithm approach for abrupt motion tracking. EURASIP J Image Video Process 2020(1):4
59. Li S, Fang H, Liu X (2018) Parameter optimization of support vector regression based on sine cosine algorithm. Expert Syst Appl 91:63–77
60. Gupta S, Deep K (2019) A hybrid self-adaptive sine cosine algorithm with opposition based learning. Expert Syst Appl 119:210–230
61. Sindhu R, Ngadiran R, Yacob YM, Hanin Zahri NA, Hariharan M (2017) Sine–cosine algorithm for feature selection with elitism strategy and new updating mechanism. Neural Comput Appl 28(10):2947–2958
62. Long W, Wu T, Liang X, Xu S (2019) Solving high-dimensional global optimization problems using an improved sine cosine algorithm. Expert Syst Appl 123:108–126
63. Hao Chen, Heidari AA, Zhao X, Zhang L, Chen H (2020) Advanced orthogonal learning-driven multi-swarm sine cosine optimization: framework and case studies. Expert Syst Appl 144:113113
64. Huiling Chen, Jiao S, Heidari AA, Wang M, Chen X, Zhao X (2019) An opposition-based sine cosine approach with local search for parameter estimation of photovoltaic models. Energy Conversion Manag 195:927–942
65. Liu S, Feng Z-K, Niu W-J, Zhang H-R, Song Z-G (2019) Peak operation problem solving for hydropower reservoirs by elite-guide sine cosine algorithm with gaussian local search and random mutation. Energies 12(11):2189
66. Nenavath H, Jatoth RK (2018) Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking. Appl Soft Comput 62:1019–1043
67. Chegini SN, Bagheri A, Najafi F (2018) Psoscalf: a new hybrid pso based on sine cosine algorithm and levy flight for solving optimization problems. Appl Soft Comput 73:697–726
68. Gupta S, Deep K (2019) Improved sine cosine algorithm with crossover scheme for global optimization. Knowl-Based Syst 165:374–406
69. Al-Betar MA, Awadallah MA, Abu R, Assaleh K (2022) Economic load dispatch using memetic sine cosine algorithm. J Ambient Intell Humanized Comput:1–29
70. Al-Betar MA, Aljarah I, Awadallah MA, Faris H, Mirjalili S (2019) Adaptive β-hill climbing for optimization. Soft Comput 23(24):13489–13512

71. Hafez AI, Zawbaa HM, Emary E, Hassanien AE (2016) Sine cosine optimization algorithm for feature selection. In: 2016 International symposium on innovations in intelligent systems and applications (INISTA). IEEE, pp 1–5
72. Eid MM, El-kenawy E-SM, Ibrahim A (2021) A binary sine cosine-modified whale optimization algorithm for feature selection. In: 2021 National computing colleges conference (NCCC). IEEE, pp 1–6
73. Hussain K, Neggaz N, Zhu W, Houssein EH (2021) An efficient hybrid sine-cosine harris hawks optimization for low and high-dimensional feature selection. Expert Syst Appl 176:114778
74. Elaziz MEA, Ewees AA, Oliva D, Duan P, Xiong S (2017) A hybrid method of sine cosine algorithm and differential evolution for feature selection. In: International conference on neural information processing. Springer, pp 145–155
75. Abualigah L, Dulaimi AJ (2021) A novel feature selection method for data mining tasks using hybrid sine cosine algorithm and genetic algorithm. Cluster Comput:1–16
76. Elaziz MA, Oliva D, Xiong S (2017) An improved opposition-based sine cosine algorithm for global optimization. Expert Syst Appl 90:484–500
77. Sindhu R, Ngadiran R, Yacob YM, Zahri NAH, Hariharan M, Polat K (2019) A hybrid sca inspired bbo for feature selection problems. Math Prob Eng:2019
78. Kumar L, Bharti KK (2021) A novel hybrid bpso–sca approach for feature selection. Natural Comput 20(1):39–61
79. El-Kenawy E-SM, Eid MM, Saber M, Ibrahim A (2020) Mbgwo-sfs: modified binary grey wolf optimizer based on stochastic fractal search for feature selection, vol 8
80. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
81. Elaziz MA, Nabil N, Ewees AA, Lu S (2019) Automatic data clustering based on hybrid atom search optimization and sine-cosine algorithm. In: 2019 IEEE congress on evolutionary computation (CEC). IEEE, pp 2315–2322
82. Uymaz SA, Tezel G, Yel E (2015) Artificial algae algorithm (aaa) for nonlinear global optimization. Appl Soft Comput 31:153–171
83. Turkoglu B, Uymaz SA, Kaya E (2022) Binary artificial algae algorithm for feature selection. Appl Soft Comput 120:108630
84. MiarNaeimi F, Azizyan G, Rashki M (2021) Horse herd optimization algorithm: a nature-inspired algorithm for high-dimensional optimization problems. Knowl-Based Syst 213:106711
85. Awadallah MA, Hammouri AI, Al-Betar MA, Braik MS, Elaziz MA (2022) Binary horse herd optimization algorithm with crossover operators for feature selection. Comput Bio Med 141:105152
86. Hayyolalam V, Kazem AAP (2020) Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems. Eng Appl Artif Intell 87:103249
87. Hu G, Du B, Wang X, Wei G (2022) An enhanced black widow optimization algorithm for feature selection. Knowl-Based Syst 235:107638
88. Khishe M, Mosavi MR (2020) Chimp optimization algorithm. Expert Syst Appl 149:113338
89. Pashaei E, Pashaei E (2022) An efficient binary chimp optimization algorithm for feature selection in biomedical data classification. Neural Comput Appl 34(8):6427–6451
90. Yang Y, Chen H, Heidari AA, Gandomi AH (2021) Hunger games search: visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. Expert Syst Appl 177:114864
91. Devi RM, Premkumar M, Jangir P, Kumar BS, Alrowaili D, Nisar KS (2022) Bhgso: binary hunger games search optimization algorithm for feature selection problem. CMC-Comput Materials Continua 70(1):557–579
92. Mirjalili S, Lewis A (2013) S-shaped versus v-shaped transfer functions for binary particle swarm optimization. Swarm Evolution Computat 9:1–14
93. Mirjalili S (2016) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput Appl 27(4):1053–1073
94. Emary E, Zawbaa HM, Hassanien AE (2016) Binary ant lion approaches for feature selection. Neurocomputing 213:54–65
95. Abed-alguni BH, Alawad NA, Barhoush M, Hammad R (2021) Exploratory cuckoo search for solving single-objective optimization problems. Soft Comput:1–14
96. Alawad NA, Abed-alguni BH (2020) Discrete island-based cuckoo search with highly disruptive polynomial mutation and opposition-based learning strategy for scheduling of workflow applications in cloud environments. Arabian J Sci Eng:1–21
97. Alkhateeb F, Abed-alguni BH, Al-rousan MH (2021) Discrete hybrid cuckoo search and simulated annealing algorithm for solving the job shop scheduling problem. J Supercomput:1–28
98. Tizhoosh HR (2005) Opposition-based learning: a new scheme for machine intelligence. In: International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (CIMCA-IAWTIC'06). IEEE, vol 1, pp 695–701
99. Shishavan ST, Gharehchopogh FS (2022) An improved cuckoo search optimization algorithm with genetic algorithm for community detection in complex networks. Multimed Tools Appl:1–27
100. Luo J, Liu Z (2020) Novel grey wolf optimization based on modified differential evolution for numerical function optimization. Appl Intell 50(2):468–486
101. Tubishat M, Abushariah MA, Idris N, Aljarah I (2019) Improved whale optimization algorithm for feature selection in arabic sentiment analysis. Appl Intell 49(5):1688–1707
102. Hammouri AI, Mafarja M, Al-Betar MA, Awadallah MA, Abu-Doush I (2020) An improved dragonfly algorithm for feature selection. Knowl-Based Syst 203:106131
103. Abed-alguni BH, Paul D, Hammad R (2022) Improved salp swarm algorithm for solving single-objective continuous optimization problem. Appl Intell:1–20
104. Alkhateeb F, Abed-Alguni BH (2019) A hybrid cuckoo search and simulated annealing algorithm. J Intell Syst 28(4):683–698
105. Deep K, Thakur M (2007) A new crossover operator for real coded genetic algorithms. Appl Math Computat 188(1):895–911
106. Boudt K, Galanos A, Payseur S, Zivot E (2019) Multivariate garch models for large-scale applications: a survey. In: Handbook of statistics. Elsevier, vol 41, pp 193–242
107. Taghian S, Nadimi-Shahraki MH (2019) Binary sine cosine algorithms for feature selection from medical data. Adv Comput Int J 235:1–10
108. Lichman M et al (2013) Uci machine learning repository, 2013. http://archive.ics.uci.edu/ml, vol 40. Accessed 14 April 2022
109. Mafarja M, Aljarah I, Heidari AA, Hammouri AI, Faris H, Ala'M A-Z, Mirjalili S (2018) Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems. Knowl-Based Syst 145:25–45
110. Alweshah M, Alkhalaileh S, Al-Betar MA, Bakar AA (2022) Coronavirus herd immunity optimizer with greedy crossover for feature selection in medical diagnosis. Knowl-Based Syst 235:107629
111. Tubishat M, Idris N, Shuib L, Abushariah MA, Mirjalili S (2020) Improved salp swarm algorithm based on opposition based

learning and novel local search algorithm for feature selection. Expert Syst Appl 145:113122
112. Sihwail R, Omar K, Ariffin KAZ, Tubishat M (2020) Improved harris hawks optimization using elite opposition-based learning and novel search mechanism for feature selection. IEEE Access 8:121127–121145
113. Aladeemy M, Adwan L, Booth A, Khasawneh MT, Poranki S (2020) New feature selection methods based on opposition-based learning and self-adaptive cohort intelligence for predicting patient no-shows. Appl Soft Comput 86:105866
114. Ji X, Liao B, Yang S (2022) A variable neighborhood search algorithm for human resource selection and optimization problem in the home appliance manufacturing industry. J Combinatorial Optimization 44(1):223–241
115. Qu C, Zhang L, Li J, Deng F, Tang Y, Zeng X, Peng X (2021) Improving feature selection performance for classification of gene expression data using harris hawks optimizer with variable neighborhood learning. Brief Bioinform 22(5):bbab097
116. Elgamal ZM, Yasin NM, Sabri AQM, Sihwail R, Tubishat M, Jarrah H (2021) Improved equilibrium optimization algorithm using elite opposition-based learning and new local search strategy for feature selection in medical datasets. Computation 9(6):68
117. Tubishat M, Ja'afar S, Alswaitti M, Mirjalili S, Idris N, Ismail MA, Omar MS (2021) Dynamic salp swarm algorithm for feature selection. Expert Syst Appl 164:113873
118. Wu X, Chen H, Li T, Wan J (2021) Semi-supervised feature selection with minimal redundancy based on local adaptive. Appl Intell 51(11):8542–8563
119. Qasem Al-Tashi, Abdul Kadir SJ, Rais HM, Mirjalili S, Alhussian H (2019) Binary optimization using hybrid grey wolf optimization for feature selection. Ieee Access 7:39496–39508
120. Sun L, Qin H, Przystupa K, Cui Y, Kochan O, Skowron M, Su J (2022) A hybrid feature selection framework using improved sine cosine algorithm with metaheuristic techniques. Energies 15(10):3485
121. Belazzoug M, Touahria M, Nouioua F, Brahimi M (2020) An improved sine cosine algorithm to select features for text categorization. J King Saud Univ-Comput Inf Sci 32(4):454–464
122. Hosseinalipour A, Gharehchopogh FS, Masdari M, Khademi A (2021) A novel binary farmland fertility algorithm for feature selection in analysis of the text psychology. Appl Intell 51(7):4824–4859
123. Abdollahzadeh B, Gharehchopogh FS, Mirjalili S (2021) African vultures optimization algorithm: a new nature-inspired metaheuristic algorithm for global optimization problems. Comput Industr Eng 158:107408
124. Abdollahzadeh B, Gharehchopogh FS, Mirjalili S (2021) Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems. Int J Intell Syst 36(10):5887–5958
125. Banerjee T, Sinha S, Choudhury P (2022) Long term and short term forecasting of horticultural produce based on the lstm network model. Appl Intell 52(8):9117–9147
126. Li J, Gao Y, Wang K, Sun Y (2021) A dual opposition-based learning for differential evolution with protective mechanism for engineering optimization problems. Appl Soft Comput 113:107942
127. Goldanloo MJ, Gharehchopogh FS (2022) A hybrid obl-based firefly algorithm with symbiotic organisms search algorithm for solving continuous optimization problems. J Supercomput 78(3):3998–4031
128. Wu G-H, Cheng C-Y, Pourhejazy P, Fang B-L (2022) Variable neighborhood-based cuckoo search for production routing with time window and setup times. Appl Soft Comput 125:109191
129. Abed-alguni BH, Ottom MA (2018) Double delayed Q-learning. Int J Artif Intell^{TM} 16(2):41–59
130. Abed-Alguni BH, Paul DJ, Chalup SK, Henskens FA (2016) A comparison study of cooperative Q-learning algorithms for independent learners. Int J Artif Intell^{TM} 14(1):71–93
131. Abed-alguni BH (2018) Action-selection method for reinforcement learning based on cuckoo search algorithm. Arabian J Sci Eng 43(12):6771–6785
132. Abed-Alguni BH (2017) Bat Q-learning algorithm. Jordanian J Comput Inf Technol(JJCIT) 3(1):56–77
133. Abed-alguni BH, Paul D (2022) Island-based cuckoo search with elite opposition-based learning and multiple mutation methods for solving optimization problems. Soft Comput:1–20
134. Abed-alguni BH, Alawad NA (2021) Distributed grey wolf optimizer for scheduling of workflow applications in cloud environment. Appl Soft Comput J:1–37
135. Abed-alguni BH, Barhoush M (2018) Distributed grey wolf optimizer for numerical optimization problems. Jordanian J Comput Inf Technol (JJCIT), vol 4(03)
136. Abed-Alguni BH (2019) Island-based cuckoo search with highly disruptive polynomial mutation. Int J Artif Intell 17(1):57–82
137. Abed-Alguni BH, Klaib AF, Nahar KM (2019) Island-based whale optimization algorithm for continuous optimization problems. Int J Reasoning-Based Intell Syst:1–11

Dr. Bilal H. Abed-alguni is an Associate Professor in the Department of Computer Science at Yarmouk University, where he has been since 2015. He received a BCS in 2002 and an MCS in 2006 from Yarmouk University. He received his PhD in Computer Science from the University of Newcastle in 2014. His research interests span both machine learning and mathematical optimization. Much of his research work has been on improving the performance of reinforcement learning algorithms, through the incorporation of optimization algorithms. His recent research studies focus on enhancing the performance of state-of-the-art optimization algorithms through mathematical and hybridization techniques.

Dr. Noor Aldeen Alawad has received his PhD in Computer Science, 2017 from Sapienza University of Rome. He is currently an associate professor in the department of computer sciences at Yarmouk University. He was awarded a scholarship funded by Erasmus Mundus to study Ph.D. at Sapienza University of Rome during the period from 2012 - 2015. Then he completed his Ph.D. in 2017. Earlier, he received the bachelor's degree and the master degree in computer science from Yarmouk University in the years 2003, 2005 respectively. He has many published papers in high impact journals and conferences in the area of evolutionary computation that are applied to resource scheduling, cloud computing, feature selection, and also in the area of recommender systems that are applied to social networks. He is a referee for many international journals and served as internal examiner for many theses.

Prof. Mohammed Azmi Al-Betar has received his PhD in Artificial Intelligence, 2010 from the University of Science Malaysia. He was also hired as a postdoctoral research fellow in the same university for 3 years and invited as a visiting researcher two times. He is currently the Head of Artificial Intelligence Research Center (AIRC) and a full-time faculty member in the Master of Artificial Intelligence (MSAI) program at Ajman University since 2020. He is also the Head of the Evolutionary Computation Research Group (ECRG) which publish more than 175 scientific publications in high quality and well-reputed journals and conferences. Therefore, Dr. Al-Betar ranked in Stanford study of the world's top 2% of scientists. Dr. Al-Betar has 15+ years of teaching experience in higher education institutions. He has taught several courses in Computer science and Artificial Intelligence fields. His research is mainly in Scheduling and optimization.

Dr. David Paul is a computer scientist interested in the Internet and distributed computing, service-oriented computing, networking, and privacy and security. Applying computer science to real-world problems, Dr Paul has been heavily involved in eHealth, including working with the Australian Schizophrenia Research Bank to link diverse health-related datasets with a strong need for privacy. He has also worked in agriculture, for example, helping develop the AskBill system to combine multiple models to assist farmers in predicting risk.

Affiliations

Bilal H. Abed-alguni[1] · **Noor Aldeen Alawad**[1] · **Mohammed Azmi Al-Betar**[2] · **David Paul**[3]

Noor Aldeen Alawad
nooraldeen@yu.edu.jo

David Paul
David.Paul@une.edu.au

[1] Department of Computer Sciences, Yarmouk University, Irbid, Jordan

[2] Artificial Intelligence Research Center (AIRC), College of Engineering and Information Technology, Ajman University, Ajman, United Arab Emirates

[3] School of Science and Technology, University of New England, Armidale, Australia