Expert Systems With Applications 200 (2022) 116834

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Binary Simulated Normal Distribution Optimizer for feature selection: Theory and application in COVID-19 datasets

Shameem Ahmed [a], Khalid Hassan Sheikh [a], Seyedali Mirjalili [b,c,d,*], Ram Sarkar [a]

[a] Department of Computer Science and Engineering, Jadavpur University, Kolkata 700032, India
[b] King Abdulaziz University, Jeddah, Saudi Arabia
[c] Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, Fortitude Valley, Brisbane, 4006 QLD, Australia
[d] Yonsei Frontier Lab, Yonsei University, Seoul, Republic of Korea

ARTICLE INFO

Keywords:
Meta-heuristic
Feature selection
Generalized Normal Distribution Optimizer
Simulated annealing
COVID-19
Optimization
Algorithm

ABSTRACT

Classification accuracy achieved by a machine learning technique depends on the feature set used in the learning process. However, it is often found that all the features extracted by some means for a particular task do not contribute to the classification process. Feature selection (FS) is an imperative and challenging pre-processing technique that helps to discard the unnecessary and irrelevant features while reducing the computational time and space requirement and increasing the classification accuracy. Generalized Normal Distribution Optimizer (GNDO), a recently proposed meta-heuristic algorithm, can be used to solve any optimization problem. In this paper, a hybrid version of GNDO with Simulated Annealing (SA) called Binary Simulated Normal Distribution Optimizer (BSNDO) is proposed which uses SA as a local search to achieve higher classification accuracy. The proposed method is evaluated on 18 well-known UCI datasets and compared with its predecessor as well as some popular FS methods. Moreover, this method is tested on high dimensional microarray datasets to prove its worth in real-life datasets. On top of that, it is also applied to a COVID-19 dataset for classification purposes. The obtained results prove the usefulness of BSNDO as a FS method. The source code of this work is publicly available at https://github.com/ahmed-shameem/Feature_selection.

1. Introduction

Data mining and Machine learning are some of the fastest-growing research topics in the information industry due to the availability of ample amounts of data that can be converted to potentially useful information. These fields are essential and integral part of the knowledge discovery (KDD) process which consists of a set of iterative sequences of tasks such as data cleaning, data reduction, data integration, and data transformation etc. Han et al. (2011). These pre-processing steps have a major impact on the performance of data mining and machine learning algorithms. Data can be considered as the 'new currency' in this decade, which simply states the importance of data. Hence, handling data properly for our needs is a new adventure. With the growing popularity of these fields, we are receiving data in abundance, which is making our job difficult as the dimensions of these data are very high. Now, any data mining and machine learning algorithm take a huge amount of time during training because of this. To solve this problem of the 'curse of dimensionality', researchers have come up with various techniques. Feature selection (FS) is one such most popular technique, which removes the unnecessary and irrelevant features, thereby reducing the number of attributes that do not help in the classification purpose, rather act as a noise and increase the space requirement as well as the computational cost.

Generally speaking, there are two ways to perform FS: filter and wrapper (Liu & Motoda, 2012). Filter methods try to evaluate the feature subset using some designated methods such as Information gain (IG), Chi-square (Zheng et al., 2004), Laplacian score (He et al., 2006) etc. Whereas, wrapper methods use a learning algorithm to evaluate the selected feature subset. Filter methods are usually faster than wrapper methods but generally wrapper methods produce better classification accuracy (Liu & Motoda, 2012).

Some of the recent and promising ones like the column-subset selection problem (Boutsidis et al., 2014; Cortinovis & Kressner, 2020; Drineas et al., 2008; Tripathi & Reza, 2020) are known to perform FS with provable theoretical bounds. These methods have been used to perform FS on k-means (Boutsidis et al., 2009), SVM (Paul et al., 2016)

[*] Corresponding author at: Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, Fortitude Valley, Brisbane, 4006 QLD, Australia.

E-mail addresses: shameemahmed20apr2000@gmail.com (S. Ahmed), skkhalidhassan7@gmail.com (K.H. Sheikh), ali.mirjalili@gmail.com (S. Mirjalili), ramjucse@gmail.com (R. Sarkar).

https://doi.org/10.1016/j.eswa.2022.116834
Received 13 December 2020; Received in revised form 25 February 2022; Accepted 3 March 2022
Available online 15 March 2022

which provide a significant performance enhancement. These methods are known to outperform existing methods like mutual information, recursive feature elimination, etc.

Finding the most functional feature subset or necessary features is a challenging task. For the last few years, meta-heuristic algorithms have been employed to address the FS problem. These works have widened the way to FS in an efficient manner. If any dataset consists of 'N' features/attributes, then there are 2^N-1 number of feature combinations. Evaluating all these feature subsets is a hectic task, i.e., it is very time consuming and hence inefficient. To solve this problem random search is another possible solution (Lai et al., 2006). However, meta-heuristic procedures are considered to be more appropriate as they can handle the worst case scenario (Talbi, 2009). There are many such meta-heuristic algorithms in literature like genetic algorithm (GA) (Davis, 1991), particle swarm optimization (PSO) algorithm (Kennedy & Eberhart, 1995), artificial bee colony (ABC) algorithm (Karaboga & Basturk, 2007), harmony search (HS) algorithm (Geem et al., 2001), sine–cosine algorithm (SCA) (Mirjalili, 2016) etc. The search process of any meta-heuristic algorithm depends on the balance between its exploration and exploitation phases. Exploration simply means diversification of solutions, i.e., evaluate the candidate solutions which are not neighbouring solutions. Exploitation, on the other hand, means intensification, i.e., searching the neighbourhood for possible better solutions. These two traits become the deciding factor in finding an optimal solution. Hence, proper tuning between these two is very important. In this paper, we have tried to maintain a fine balance between these two phases of Generalized Normal Distribution Optimizer (GNDO) (Zhang et al., 2020) with the help of Simulated Annealing (SA) (Kirkpatrick et al., 1983) which acts as a local search to enhance the exploitation capabilities of GNDO. The proposed method has been applied over various datasets to prove its worth and effectiveness.

The rest of the paper is organized in the following manner: Section 2 discusses some popular and recent meta-heuristic algorithms found in the literature, Section 3 presents the motivation and contribution of this work, Section 4 describes the search process of GNDO and SA, Section 5 discusses the fitness function and transfer function used here, as well as the time complexity of the method, Section 6 reports the detailed experiments that have been performed to prove the effectiveness of the proposed method, Section 7 proves the robustness of the model, Section 8 shows the effectiveness of the proposed method in COVID detection and finally Section 9 concludes the paper along with its future work.

2. Related work

In recent times, optimization algorithms have attracted a lot of attention from researchers. In particular, meta-heuristic algorithms have seen numerous improvements over the years. Meta-heuristic is a genre of randomized algorithms where the algorithm learns to find the optimal solution through the iteration process. Meta-heuristic algorithms can be divided into different categories: single solution based and population-based (Gendreau & Potvin, 2005), nature-inspired and non-nature inspired (Abdel-Basset et al., 2018; au2 et al., 2013), etc. From the 'inspiration' point of view, these algorithms can broadly be divided into four categories (Nematollahi et al., 2019): Evolutionary, Swarm inspired, Physics based, and Human related.

- **Evolutionary Algorithms:** These algorithms are basically inspired by the biological process of evolution. In an evolutionary endeavour, the fittest individual is generated through crossover and mutation in each generation, which inspired the pioneer algorithm in this field, Genetic Algorithm (GA) (Davis, 1991). Other evolutionary algorithms are Genetic programming (Koza, 1994), Co-evolving algorithm (Hillis, 1990), Cultural algorithm (Xue et al., 2011), Biogeography-Based Optimization (Simon, 2008), Grammatical evolution (Ryan et al., 1998) etc.
- **Swarm inspired algorithms:** This genre of algorithm mimics the individual and social behaviour of swarm, herd, schools, groups, and teams. The key idea behind such algorithms in the optimization field is that in swarms, each individual has a certain behaviour but with the collective effort, the swarm can solve very complex optimization problems. One of the most popular algorithms in this field is PSO (Kennedy & Eberhart, 1995), which is inspired by the behaviour of flock of birds. The other famous swarm-based algorithms are the Shuffled frog-leaping algorithm (Eusuff et al., 2006), Bacterial foraging Passino (2002), ABC (Karaboga & Basturk, 2007), Firefly Algorithm (Yang, 2009), Grey Wolf Optimizer (GWO) (Mirjalili et al., 2014), Crow search algorithm (Askarzadeh, 2016), The Whale Optimization Algorithm (Mirjalili & Lewis, 2016), Grasshopper Optimization Algorithm (Saremi et al., 2017), Squirrel Search Algorithm (Jain et al., 2019).
- **Physics based algorithms:** This type of algorithms is inspired by the working principle of the physical world. Music, metallurgy to mathematics, physics, chemistry, and complex dynamic systems, are some of the physical processes which inspire Physics based meta-heuristic algorithms. Some noted algorithms are the Gravitational Search algorithm (GSA) (Rashedi et al., 2009), SA (Kirkpatrick et al., 1983), Self propelled particles (Vicsek et al., 1995), HS algorithm (Geem et al., 2001), Black hole optimization (Hatamlou, 2013), Multi-verse optimizer (Mirjalili et al., 2015), Find-Fix-Finish-Exploit-Analyze (Kashan et al., 2019) etc.
- **Human related algorithms:** These are developed based on human behaviour Teaching–Learning-Based optimization (Rao et al., 2011), Society and civilization (Ray & Liew, 2003), Fireworks algorithm (Tan & Zhu, 2010), are some algorithms in this genre.

However, one of the issues with meta-heuristic algorithms is their premature convergence that leads to finding less optimal solutions. Therefore, these algorithms are often coupled with other techniques (e.g. local search algorithms). In this case, the local search algorithm tries to find solutions that are locally adjacent to an existing solution, which can outperform the existing solutions. Some of the commonly used local search algorithms are Hill Climbing (HC), SA (Kirkpatrick et al., 1983), Tabu Search (TS) (Glover & Laguna, 1998), Late acceptance hill climbing (LAHC). Some modifications to HC algorithm are βHC (Al-Betar, 2016) and Adaptive βHC (Al-Betar et al., 2019). Some of the work based on hybridization of local search and meta-heuristic algorithms are Elgamal et al. (2020), Kurtuluş et al. (2020) and Mafarja and Mirjalili (2017).

3. Motivation and contributions

For the past few decades, meta-heuristic algorithms have proved their utilities in several research fields. Because of its immense usefulness, researchers are investing more time in it to come up with better-performing algorithms. At the end of the day, we want to find the most optimal solution to such NP-hard problems. So there is not really any best result. We can always improve our findings with new or modified algorithms. Moreover, according to *the No Free Lunch* (NFL) theorem (Wolpert & Macready, 1997), any two algorithms produce equivalent results when they are evaluated on all possible optimization problems. It has been observed that an algorithm may achieve superior results on some problems, but that does not ensure the same on other problems. Hence, we can say that there is no such universal algorithm that is qualified enough to be used in all the optimization problems and produce the best results. These inferences keep the research resilient in this field. As FS is considered as an optimization problem (Ghosh et al., 2020), so researchers are also coming up with new and efficient FS methods using meta-heuristic algorithms.

This is the motivation of our proposed work where we have designed a new algorithm by modifying the GNDO (Zhang et al., 2020).

The GNDO algorithm is inspired by the generalized normal distribution model, where each individual uses a generalized normal distribution curve to update its current position in the hope of finding a better position. GNDO is employed to increase the accuracy of extracting the unknown parameters of the single diode model, double diode model and photovoltaic module model. There are two ways of hybridizing meta-heuristic algorithms (Talbi, 2009): low-level and high-level. A low-level approach en routes one algorithm in the other, whereas the algorithms are executed in succession in the high-level accession. This work follows the high-level version to hybridize GNDO and SA, maintaining a pipeline model where the output of one meta-heuristic algorithm is considered as the input to the other. To the best of our knowledge, this is the first time, GNDO is hybridized with SA for solving FS problems.

The present work proposes an improved version of the binary form of GNDO (BGNDO), known as Binary Simulated Normal Distribution Optimizer (BSNDO), hybridized with another meta-heuristic algorithm called SA (Kirkpatrick et al., 1983). Recently, some hybrid FS methods have been proposed (Ahmed et al., 2021, 2020b; Bhattacharyya et al., 2020; Sheikh et al., 2020), which have demonstrated their effectiveness and superiority over other methods. This has also motivated us to come up with hybrid version of GNDO. Also, COVID-19 is a threat to the humanity as many people are suffering from it as well as many have died. Our normal day-to-day life is destroyed because of this uncertainty. Many works have been proposed for detection of COVID, few of them are Das et al. (2021), Garain et al. (2021), Karbhari et al. (2021) etc. We have performed FS on a publicly available COVID-19 dataset for classification purpose of COVID-19. In a nutshell, the main contributions of this work are as follows:

- A new FS method called BSNDO is introduced using BGNDO and another popular meta-heuristic called SA.
- The proposed hybrid FS method is assessed on 18 standard UCI datasets (Dua & Graff, 2017) using K-nearest Neighbours (KNN) classifier, Random Forest classifier as well as Naive Bayes classifier.
- BSNDO is also applied on high-dimensional microarray datasets to prove its effectiveness.
- It is also applied on a publicly available COVID-19 dataset for classification purposes.
- The proposed FS method is compared with many state-of-the-art meta-heuristic based FS methods.

4. Preliminaries

4.1. Generalized normal distribution optimizer

GNDO (Zhang et al., 2020) is inspired from normal distribution (Gaussian distribution) theory. This distribution is used to narrate the natural phenomenon. A normal distribution is described as follows: assume a random variable 'r', obeys a probability distribution having location parameter α and scale parameter β, and its probability density function can be written as:

$$f(r) = \frac{1}{\sqrt{2\pi}\beta} \exp(-\frac{(r-\alpha)^2}{2\beta^2}) \tag{1}$$

Then 'r' is a random variable and this distribution is called normal distribution, i.e., r$\tilde{N}(\alpha,\beta)$.

Any population-based optimization algorithm starts with the random initialization, and then all solutions converge towards the global optima following the rules of exploration and exploitation. In the end, all individuals assemble around the achieved the best solution. Now, we can visualize this search process as multiple normal distributions. The position of every individual is regarded as random variables which are subject to normal distribution. Exploration of GNDO is dependent on three randomly selected agents. And the exploitation of GNDO is based on the generalized normal distribution model, which is accompanied by the current mean position and the current optimal position.

Based on the correspondence between the distribution of the solutions in the population and the normal distribution, a generalized distribution model can be built by:

$$v_i^t = \alpha_i + \beta_i \times \gamma, i = 1, 2, 3, \dots, D \tag{2}$$

where v_i^t is the trail vector of the ith agent at time t, α_i is generalized mean position of the ith agent, β_i is generalized standard variance and γ is a penalty factor. Also, α, β and γ can be defined as follows:

$$\alpha_i = \frac{1}{3}(r_i^t + r_{best}^t + M) \tag{3}$$

$$\beta_i = \sqrt{\frac{1}{3}[(r_i^t - \alpha)^2 + (r_{best}^t - \beta)^2 + (M - \alpha)^2]} \tag{4}$$

$$\gamma = \begin{cases} \sqrt{-\log(\lambda_1)} \times \cos(2\pi\lambda_2), \text{if } a \leq b \\ \sqrt{-\log(\lambda_1)} \times \cos(2\pi\lambda_2 + \pi), \text{otherwise} \end{cases} \tag{5}$$

where a, b, λ_1 and λ_2 are random numbers $\in$ [0, 1], r_{best}^t is the current best position and M is the mean position of the current population, which is calculated by:

$$M = \frac{\sum_{i=1}^{D} r_i^t}{D} \tag{6}$$

As current best individual r_{best}^t contains useful information related to the global optimal solution, the ith individual r_i^t is pulled towards the direction of r_{best}^t. It is to be noted that when r_{best}^t gets confined into local optima, all agents still move towards the direction of r_{best}^t that will lead the algorithm to premature converge. To resolve this concern, the mean position of the current population M is introduced. Although the position of r_{best}^t may not change in some generations, however, the mean position M is changed over the generation that becomes useful for finding better solutions. Thus, the mean position M is introduced in the searching process, which increases the probability to avoid the local optima.

β_i is employed to amplify the local search ability of GNDO. Further, β_i can be interpreted as a random progression to perform the local search around the generalized mean position α_i. Moreover, the distance between the position of the ith individual r_i^t and the mean position M and the position of the best individual r_{best}^t is larger, the oscillation of the generated random sequence is prominent. Hence, the probability to find a better solution around an individual is very minimal when r_i^t has a very bad fitness value which can help the individual to search better solution. On the contrary, there is a large probability for the individual to find a better solution around it when the individual r_i^t has good fitness. Thus, a random sequence with weak oscillation may help the individual to achieve a better solution.

In the GNDO algorithm, the penalty factor γ is used to increase the randomness of the generated generalized standard variance. Most penalty factors are located $\in$ [−1, 1]. As the generated generalized standard variances are all positive, the penalty factor can increase the search directions of GNDO, which can enhance its searchability.

Now, the global exploration of GNDO is dependent on three randomly selected individuals, which is given by:

$$v_i^t = r_i^t + \delta \times (|\lambda_3| \times v_1) + (1 - \delta) \times (|\lambda_4| \times v_2) \tag{7}$$

where λ_3 and λ_4 are randomly generated numbers subject to standard normal distribution, δ is the adjust parameter which is a random number $\in$ [0, 1], v_1 and v_2 are trail vectors which are calculated as follows:

$$v_1 = \begin{cases} r_i^t - r_{p1}^t, \text{if } f(r_i^t) < f(r_{p1}^t) \\ r_{p1}^t - r_i^t, \text{otherwise} \end{cases} \tag{8}$$

$$v_2 = \begin{cases} r_{p2}^t - r_{p3}^t, \text{if } f(r_{p2}^t) < f(r_{p3}^t) \\ r_{p3}^t - r_{p2}^t, \text{otherwise} \end{cases} \tag{9}$$

Table 1
Brief description of parameters used in GNDO.

Parameter	Description	Value
α	Generalized mean position	NA
β	Generalized standard variance (enhances local search ability)	NA
γ	Penalty factor (enhances randomness of generated generalized standard variance)	[−1,1]
a,b, λ_1, λ_2	Random numbers	[0,1]
λ_3, λ_4	Random numbers subject to standard normal distribution	[0,1]
δ	Adjust parameter	[0,1]
D	Dimension of search space	NA

where p1, p2 and p3 are three randomly generated integers ∈ [1, D], following $p1 \neq p2 \neq p3 \neq i$. The ith individual is given information by p2 and p3. The solution p1 shares information with ith solution . The adjust parameter δ is used to balance the two information-sharing procedures. Moreover, λ_3 and λ_4 are random numbers with standard normal distribution, which can make GNDO has a larger search space in the process of performing the global search. In order to bring the better solution into the next generation population, a mechanism is designed, which is represented as :

$$r_i^{t+1} = \begin{cases} v_i^t \text{if } f(v_i^t) < f(r_i^t) \\ r_i^t \text{otherwise} \end{cases} \tag{10}$$

A brief description of the parameters used in GNDO are summarized in Table 1.

The pseudocode of the GNDO algorithm is given in Algorithm 1.

Algorithm 1 Pseudo-code of GNDO algorithm

```
Input: Problem dependent Information Output: Best solution
Initialize population
Calculate the fitness value of every individual using Eq. (11) and achieve the optimal solution $r_{best}$
while $t < T_{max}$ do
    for i = 1:N do
        Generate a random number 'rnd' ∈ [0, 1]
        if rnd > 0.5 then
            Select the current optimal solution $r_{best}$ and calculate the mean position M using Eq. (6)
            Compute generalized mean position $\alpha$, generalized standard variance $\beta$ and penalty factor by Eqs. (3)–(5) respectively
            Perform local exploitation using Eqs. (2) and (10)
        else
            Perform global exploration using Eqs. Eqs. (7)–(10)
        end if
    end for
    Update the current number of iteration, t = t + 1
end while
Return $r_{best}$
```

4.2. Simulated annealing

SA, proposed by Kirkpatrick et al. (1983), is inspired by an analogy between simulation of annealing of solids and large combinatorial optimization problems. Often meta-heuristic algorithms can fail to find the global optima, rather they can stagnate in local optima. To overcome this issue, SA uses a probabilistic approach to accept a poor solution. By accepting poor solutions with a certain probability, exploration increases. The algorithm starts with a randomly generated initial solution, in each iteration, a solution neighbouring with respect to the current solution is generated on a random basis based on the existing neighbouring structure. Then, the neighbouring solution is evaluated based on the fitness function. There may occur two possibilities:

1. Neighbouring solution is a better performing solution than the existing solution: in this case, a new solution is always accepted.

Table 2
Brief description of parameters used in SA.

Parameter	Description	Value
P	Boltzmann probability	[0, 1]
T	Temperature	NA
$\|N\|$	Number of attributes for each dataset	NA

2. Neighbouring solution is worse performing solution in comparison with the existing solution: in this case, the worse solution can be accepted with a certain probability determined by the Boltzmann probability, $P = e^{-\frac{\delta}{T}}$. Here δ is the difference of fitness value of the neighbouring solution and existing best solution, and T is the temperature of "simulated annealing process". The temperature is periodically reduced over the iterations. The temperature is initialized to 2 $*$ $\|N\|$ where $\|N\|$ represents the feature-length. The temperature reducing scheme can be represented by the following equation: $T(t+1) = 0.93 * T(t)$. It is to be noted that the temperature decay and the probability of exploration/exploitation are taken from the work done by Kirkpatrick et al. (1983) and Zhang et al. (2020) respectively.

A brief description of the parameters used in SA are summarized in Table 2.

Algorithm 2 shows the pseudo-code of SA.

5. Proposed method

This section elaborates the fitness function and transfer function used, and the computational complexity of the proposed algorithm. At every iteration, the agents update their position following the rules of GNDO and at the end, they try to find a better solution in their neighbourhood using SA.

5.1. Fitness function

Selecting the relevant features from a dataset that actually helps the classifier to identify the class of a sample is the main challenge. Now, during the process of selecting relevant features, we have to automatically rule out the redundant ones and maximize the classification accuracy of a classification problem when the selected feature subset is used for classification purposes (Pudil et al., 1994). This work applies BSNDO to find the best feature subset and calculate the classification accuracy of this subset using a classifier. Let $\mathcal{A}_c$ be the classification accuracy of the model calculated using a classifier, d_s be the dimension of the feature subset and D_t be the total number of attributes present in the original dataset. So, (1 - $\mathcal{A}_c$) is the classification error and $\frac{d_s}{D_t}$ is the fraction of features selected from the original dataset. We define the fitness function as:

$$\downarrow Fitness = \mu \cdot (1 - \mathcal{A}_c) + (1 - \mu) \cdot \frac{d_s}{D_t} \tag{11}$$

where $\mu \in [0, 1]$ denotes weightage given to the classification error.

Algorithm 2 Pseudo-code of SA

```
Input: Problem dependent information
Output: BestSol
$T_0 = 2 * |N|, |N|$ is number of attributes
$BestSol := S'_i$
$\delta(BestSol) := \delta(S_i)$, $\delta$ indicates the quality of the solution
while T > $T_0$ do
    Generate a random new solution, TrialSol in the neighborhood of $S'_i$
    Calculate $\delta(TrialSol)$
    if $\delta(TrialSol) > \delta(BestSol)$ then
        $S'_i := TrialSol$
        $BestSol := TrialSol$
        $\delta(S'_i) := \delta\ (TrialSol)$
        $\delta(BestSol) := \delta(TrialSol)$
    else if $\delta(TrialSol) == \delta(BestSol)$ then
        Calculate $|TrialSol|$ and $|BestSol|$

        if $|TrialSol| < |BestSol|$ then
            $S'_i := TrialSol$
            $BestSol := TrialSol$
            $\delta(S'_i) := \delta\ (TrialSol$
            $\delta(BestSol) := \delta(TrialSol)$
        end if
    else
        Calculate $\theta = \delta(TrialSol) - \delta(BestSol)$
        Generate a random number, rnd $\in$ [0,1]
        if rnd $\leq exp^{-\frac{\theta}{T}}$ then
            $S'_i := TrialSol$
            $\delta(S'_i) := \delta\ (TrialSol$
        end if
    end if
    T = 0.93 * T
end while
```

5.2. Transfer function

As FS is a binary optimization problem (Ghosh et al., 2020), its output is ∈ {0, 1} where zero represents that the feature is rejected as it is redundant and one represents that the feature is useful and hence it is selected. However, we cannot discard the possibility of the obtained result going out of the desired range. To ensure that the output always stays within the expected range, we have to apply a binarization function on each agent. Here, this task is performed by the sigmoid (S-shaped) transfer function (Mirjalili & Lewis, 2013). The S-shaped transfer function, depicted in Fig. 1, is given by -

$$T(x) = \frac{1}{1+e^{-x}} \tag{12}$$

$$X^d(t) = \begin{cases} 1 & \text{if } rnd < T(X^d(t)) \\ 0 & \text{if } rnd \geq T(X^d(t)) \end{cases} \tag{13}$$

The range of this function ∈ [0,1]. If the transfer function produces output > rnd, where rnd is a random number with uniform distribution in the range (0, 1), we set the value to be 1 i.e., we consider that attribute is useful and if it is ≤ rnd, we set the value to be 0 i.e., the attribute is redundant, hence it will not be considered (Mafarja et al., 2019).

5.3. Computational complexity

For any meta-heuristic algorithm, the computation complexity depends on the time taken by each individual to update their positions, the maximum number of iterations and some other operations like comparison/sorting and the time to update variables. The computational complexity of BSNDO is $O(Max_{iter} \times Pop_{size} \times Dim_s \times T_{fitness})$, where Max_{iter} represents the maximum number of iterations, Pop_{size} represents the number of agents, Dim_s represents the dimension of the search space, and $T_{fitness}$ indicates the required time for calculating the fitness of a particular solution using a classifier. The usage of SA is to find a better solution if available in the neighbourhood of the current solution. SA does not affect the computational cost in terms of O-notation.

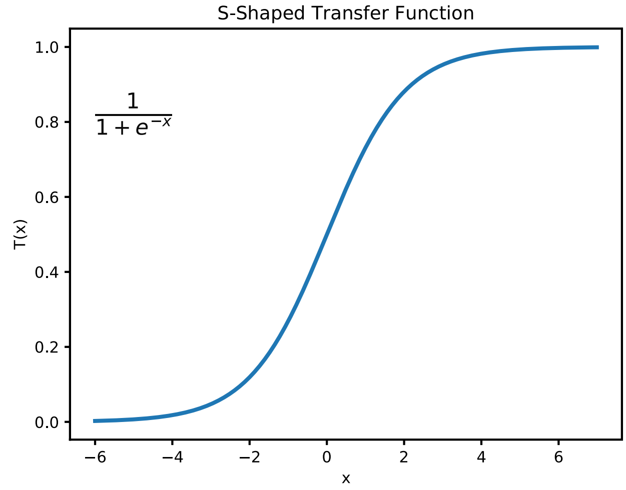

Fig. 1. S-shaped transfer function.

Table 3
Brief idea of the datasets employed here to assess the proposed FS method.

Sl. No.	Dataset	#Attributes	#Samples	#Classes	Domain
1	Breastcancer	9	699	2	Biology
2	BreastEW	30	569	2	Biology
3	CongressEW	16	435	2	Politics
4	Exactly	13	1000	2	Biology
5	Exactly2	13	1000	2	Biology
6	HeartEW	13	270	2	Biology
7	IonosphereEW	34	351	2	Electromagnetic
8	KrvskpEW	36	3196	2	Game
9	Lymphography	18	148	4	Biology
10	M-of-n	13	1000	2	Biology
11	PenglungEW	325	73	2	Biology
12	SonarEW	60	208	2	Biology
13	SpectEW	22	267	2	Biology
14	Tic-tac-toe	9	958	2	Game
15	Vote	16	300	2	Politics
16	WaveformEW	40	5000	3	Physics
17	WineEW	13	178	3	Chemistry
18	Zoo	16	101	6	Artificial

6. Experiments

6.1. Dataset details

To investigate the performances of BGNDO and BSNDO, 18 standard UCI datasets (Dua & Graff, 2017) are considered here. These datasets are from diverse domains. Some basic information regarding these datasets is provided in Table 3. As the datasets used here are assorted in terms of the number of features and instances, so it helps us to understand the robustness of the proposed FS method.

6.2. Parameter settings

For any multi-agent evolutionary algorithm, the parameters always play an important role in determining the outcome. Specially, the

Table 4
Achieved classification accuracy obtained by BGNDO and BSNDO with different population sizes.

Pop_size	10		20		30		40		50	
Dataset	BGNDO	BSNDO	BGNDO	BSNDO	BGNDO	BSNDO	BGNDO	BSNDO	BGNDO	BSNDO
Breastcancer	98.57	100	97.14	100	99.28	98.57	99.28	99.28	98.57	100
BreastEW	96.49	97.36	97.37	98.25	96.49	98.24	97.36	97.36	95.61	99.122
CongressEW	96.55	100	97.7	100	98.85	98.85	96.55	97.7	98.85	98.85
Exactly	100	100	100	100	100	100	100	99.5	100	100
Exactly2	77	78.5	80.5	80.5	80	78.5	79	80	80	79.5
HeartEW	85.18	83.33	90.74	94.44	88.88	90.74	85.18	94.44	85.18	90.74
IonosphereEW	91.43	92.86	95.71	95.74	91.43	94.28	94.29	92.86	92.85	94.28
KrvskpEW	97.65	98.12	98.12	98.44	98.43	97.49	98.59	97.81	97.96	97.33
Lymphography	93.33	93.33	96.67	96.67	90	96.67	96.67	90	96.67	93.33
M-of-n	100	100	100	100	100	100	100	100	100	100
PenglungEW	93.33	93.33	100	100	93.33	93.33	86.67	100	93.33	100
SonarEW	95.23	88.09	97.62	95.24	92.85	92.86	97.62	95.23	92.85	97.62
SpectEW	90.56	94.44	92.45	96.22	92.45	94.44	94.33	90.74	92.45	88.89
Tic-tac-toe	83.33	86.46	89.58	87.5	84.89	86.46	83.85	86.46	88.54	84.89
Vote	98.33	100	100	100	98.33	98.33	100	100	98.33	100
WaveformEW	84.3	84.4	84.5	87	83.4	84.6	85.3	85.6	85.7	83.8
WineEW	97.22	100	100	100	100	100	100	100	97.22	100
Zoo	100	100	100	100	100	100	100	100	100	100

Table 5
Number of selected features by BGNDO and BSNDO for different population sizes.

Pop_size	10		20		30		40		50	
Dataset	BGNDO	BSNDO	BGNDO	BSNDO	BGNDO	BSNDO	BGNDO	BSNDO	BGNDO	BSNDO
Breastcancer	4	4	7	4	4	6	4	3	4	4
BreastEW	13	8	14	4	15	11	16	5	13	12
CongressEW	6	6	9	7	7	9	8	10	7	8
Exactly	7	6	10	6	7	6	6	7	7	6
Exactly2	6	4	9	8	8	6	9	12	7	6
HeartEW	5	5	6	4	6	4	5	4	4	5
IonosphereEW	17	12	26	16	15	16	20	12	10	8
KrvskpEW	24	24	22	22	21	25	26	24	22	17
Lymphography	10	5	11	5	8	8	6	6	9	6
M-of-n	7	6	8	6	7	6	7	6	7	6
PenglungEW	48	132	209	187	129	132	171	179	177	139
SonarEW	30	24	39	27	33	31	30	36	27	28
SpectEW	9	13	14	6	10	7	11	6	11	12
Tic-tac-toe	6	9	9	9	7	9	9	9	9	9
Vote	10	7	10	3	7	3	8	6	6	7
WaveformEW	34	25	27	33	27	26	31	28	26	4
WineEW	6	4	9	3	7	1	4	4	4	4
Zoo	8	6	11	5	6	1	7	6	5	8

population size and the total number of iterations (number of generations) always affect the outcome of the algorithm heavily. Hence, we have performed some experiments to determine the approximate ideal population size and a total number of iterations. The achieved classification accuracy by BGNDO and BSNDO for different population sizes varying from 10 to 50 are provided in Table 4. Similarly, the numbers of selected features for different population sizes varying from 10 to 50 by BGNDO and BSNDO are depicted in Table 5. To observe the convergence of the solution to the optimal position, convergence graphs have been plotted (which are given in Fig. 2) over 50 iterations. To maintain the fairness of the comparison, we have run each dataset 10 times and taken the average over these runs.

From the initial experiments, we have found that a population size of 20 leads to noteworthy results. Keeping the computational cost in min,d this population size is considered for further experiments. At the same time, from the convergence graphs, approximately after 30 iterations, the best solution is almost at the optimal position. Hence, it has been used for further experiments.

6.3. Result and discussion

This section discusses the outcomes produced by BGNDO and BSNDO evaluated on UCI datasets whose details are given in Table 3 while evaluated using KNN, Random Forest and Naive Bayes classifiers. These results establish the superiority of BSNDO over BGNDO.

Tables 6–8 present the outcomes obtained by the proposed BSNDO algorithm while evaluated by KNN, Random Forest and Naive Bayes classifiers respectively. Compared to the BGNDO algorithm, the obtained results clearly depicts the effect of BSNDO in finding a better solution. Observing these results, we can conclude that BSNDO performs better than BGNDO on UCI datasets. Furthermore, KNN is used widely in the references for FS on UCI datasets (Emary et al., 2016; Mafarja & Mirjalili, 2017; Mafarja et al., 2019), hence, for further experiments and discussion, we have utilized KNN classifier with $K = 5$.

Inspecting the results in these tables, we can observe that BSNDO provides better results than BGNDO in every dataset while evaluated using different classifiers. From Table 6 we can see that BSNDO achieves >90% accuracy in 15 datasets (83.33%), out of which it produces 100% classification accuracy in 8 datasets (44.44%) while evaluated using KNN classifier. It produces better classification accuracy than BGNDO except Exactly, Exactly2, Lymphography, M-of-n, PenglungEW, Vote, WineEW and Zoo, where both produces equivalent accuracy. Considering the number of selected features, BSNDO beats BGNDO in every dataset except WaveformEW. They select the same number of features in the case of KrvskpEW.

Similarly, from Table 7 we can see that BSNDO achieves >90% accuracy in 16 datasets (88.89%) while evaluated using Random Forest classifier. It achieves 100% accuracy in 6 datasets (33.33%). In the case of Exactly, Exactly2, M-of-n, Vote, WineEW and Zoo, BSNDO and BGNDO produces the same classification accuracy. BSNDO has

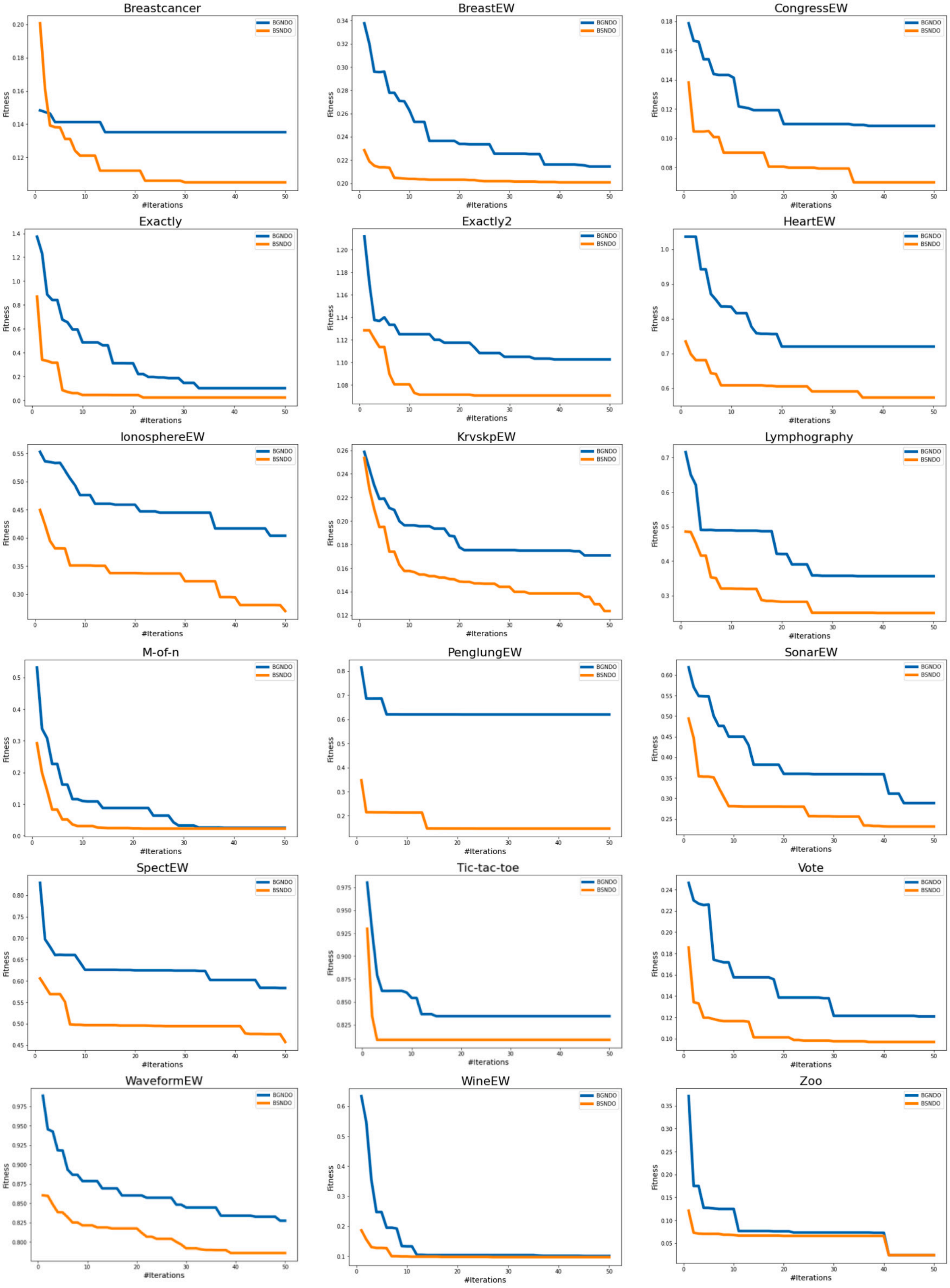

Fig. 2. Convergence graphs depicting the convergence of best individual at every iteration for 18 UCI datasets using BGNDO and BSNDO.

the upper hand over BGNDO over the rest cases. Talking about the number of selected features, only in the case of BreastEW, CongressEW, PenglungEW and Tic-tac-toe, BGNDO produces better results. They select the same number of features in cases of Exactly, Exactly2 and HeartEW. BSNDO selects fewer features than BGNDO in the rest cases.

While evaluated using Naive Bayes classifier, BSNDO produces >90% classification accuracy in 15 datasets (83.33%) Table 8. It achieves 100% accuracy in 7 datasets (38.89%). BSNDO and BGNDO produce equivalent results in the case of Exactly2, Vote, WineEW and Zoo. BSNDO produces better results in the rest of the cases. It also selects fewer features than BGNDO in almost every dataset except KrvskpEW and PenglungEW. Both of them selects the same number of features in the case of M-of-n and SpectEW.

From the above discussion, we can say that BSNDO is superior to BGNDO while evaluated using KNN, Random Forest and Naive Bayes classifiers. The results achieved by BSNDO using these classifiers

Table 6
Achieved classification accuracy and number of selected features by BGNDO and BSNDO using KNN classifier (highest classification accuracies and lowest no. of selected features are highlighted in bold font).

Sl. No.	Dataset	Original		BGNDO		BSNDO	
		Accuracy	Features	Accuracy	Features	Accuracy	Features
1	Breastcancer	96	9	97.14	7	**100**	**4**
2	BreastEW	92.63	30	97.37	14	**98.25**	**4**
3	CongressEW	92.18	16	97.7	9	**100**	**7**
4	Exactly	72.3	13	**100**	10	**100**	**6**
5	Exactly2	73.3	13	**80.5**	9	**80.5**	**8**
6	HeartEW	68.15	13	90.74	6	**94.44**	**4**
7	IonosphereEW	83.43	34	95.71	26	**95.74**	**16**
8	KrvskpEW	96.1	36	98.12	**22**	**98.44**	**22**
9	Lymphography	81.33	18	**96.67**	11	**96.67**	**5**
10	M-of-n	87.4	13	**100**	8	**100**	**6**
11	PenglungEW	81.33	325	**100**	209	**100**	**187**
12	SonarEW	80.95	60	94.62	39	**95.24**	**27**
13	SpectEW	82.22	22	92.45	14	**96.22**	**6**
14	Tic-tac-toe	81.1	9	83.854	8	**87.5**	**8**
15	Vote	92.33	16	**100**	10	**100**	**3**
16	WaveformEW	81.44	40	84.5	**27**	**87**	33
17	WineEW	66.67	13	**100**	9	**100**	**3**
18	Zoo	87	16	**100**	11	**100**	**5**

Table 7
Achieved classification accuracy and number of selected features by BGNDO and BSNDO using Random Forest classifier (highest classification accuracies and lowest no. of selected features are highlighted in bold font).

Sl no.	Dataset	Original		BGNDO		BSNDO	
		Accuracy	Features	Accuracy	Features	Accuracy	Features
1	Breastcancer	97.8	9	97.14	7	**97.86**	**2**
2	BreastEW	98.2	30	95.61	**2**	**100**	4
3	CongressEW	97.7	16	96	**1**	**98.85**	**5**
4	Exactly	78.5	13	**100**	**6**	**100**	**6**
5	Exactly2	74	13	**76**	**1**	**76**	**1**
6	HeartEW	81.5	13	88.89	**5**	**94.44**	**5**
7	IonosphereEW	91.4	34	95.71	24	**98.57**	**20**
8	KrvskpEW	99.5	36	98.12	**28**	**99.53**	17
9	Lymphography	90	18	93.33	8	**96.67**	**4**
10	M-of-n	**100**	13	**100**	**8**	**100**	**6**
11	PenglungEW	86.7	325	93.33	140	**100**	**193**
12	SonarEW	90.7	60	92.86	42	**95.24**	**14**
13	SpectEW	88.9	22	90.74	15	**96.3**	**7**
14	Tic-tac-toe	**95.8**	9	82.94	**5**	94.37	8
15	Vote	95	16	**98.33**	12	**98.33**	**6**
16	WaveformEW	85.8	40	83	33	**86.2**	**29**
17	WineEW	**100**	13	**100**	4	**100**	**3**
18	Zoo	**100**	16	**100**	4	**100**	**3**

establish the fact that BSNDO produces noteworthy and impressive results while evaluated using different classifiers.

6.4. Comparison

We have established this claim that BSNDO produces better results than BGNDO beforehand. This section provides the performance comparison of BSNDO with eight state-of-the-art FS methods. These state-of-the-art methods consist of few popular methods and some recently proposed hybrid methods. They are: GA, PSO, adaptive switching grey-whale optimizer (ASGW), serial grey-whale optimizer (HSGW), random switching grey-whale optimizer (RSGW), social ski driver algorithm and late acceptance hill-climbing (SSDs+LAHC) (Chatterjee et al., 2020), electrical harmony based meta-heuristic (Sheikh et al., 2020) and embedded chaotic whale survival algorithm (ECWSA-4) (Guha et al., 2020).

From Table 9 we can say that BSNDO produces the overall best result. In the case of Breastcancer, BSNDO and EHHM produce 100% accuracy. In BreastEW, BSNDO holds the second position along with SSDs+LAHC after ASGW and EHHM. BSNDO holds the top position along with SSDs+LAHC in CongressEW producing 100% accuracy. In the case of Exactly, BSNDO produces the best result with SSDs+LAHC, HSGW, BGA, BPSO and EHHM. HSGW beats BSNDO in Exactly2 with a very narrow margin. It stands at third position in the case of HeartEW. It stands at fifth position in IonosphereEW and SonarEW. In the case of KrvskpEW and Lymphography, it attains the second position after BGA and EHHM respectively. In the case of M-of-n, PenglungEW, Vote, WineEW and Zoo, BSNDO stands at first position along with few other methods. It achieves the highest classification accuracy in the case of SpectEW and Tic-tac-toe. In the case of WaveformEW, it produces the second best result after EHHM.

Table 10 gives the comparison of BSNDO with state-of-the-art FS methods based on the number of selected features. It selects the least number of features in BreastEW, Exactly along with SSDs+LAHC, BGA and BPSO, Lymphography along with BGA and BPSO. It also produces the best result in the case of Vote along with BPSO. It stands at second position in the case of Breastcancer along with BGA and BPSO, HeartEW, M-of-n with SSDs+LAHC, BGA and BPSO, SpectEW with BPSO, WineEW along with SSDs+LAHC. BSNDO stands at third position in the case of Exactly2 with SSDs+LAHC and Zoo along with BPSO. It stands at fourth position in the case of IonosphereEW and Tic-tac-toe along with ECWSA-4. It selects the same number of features as EHHM in the case of CongressEW attaining the fifth position. In the case of PenglungEW and WaveformEW, it stands at ninth position.

Table 8
Achieved classification accuracy and number of selected features by BGNDO and BSNDO using Naive Bayes classifier (highest classification accuracies and lowest no. of selected features are highlighted in bold font).

Sl. No.	Dataset	Original		BGNDO		BSNDO	
		Accuracy	Features	Accuracy	Features	Accuracy	Features
1	Breastcancer	89.28	9	97.87	7	**99.24**	**5**
2	BreastEW	96.49	30	97.36	14	**98.2**	**5**
3	CongressEW	98.85	16	98.85	8	**100**	**6**
4	Exatly	69.5	13	96	8	**100**	**6**
5	Exactly2	76	13	**76**	9	**76**	**6**
6	HeartEW	94.44	13	92.3	9	**94.44**	**4**
7	IonosphereEW	95.71	34	92.88	18	**95.74**	**15**
8	KrvskpEW	65.88	36	95.31	**9**	**97.18**	12
9	Lymphography	86.67	18	90	15	**100**	**8**
10	M-of-n	96.5	13	98.5	**6**	**100**	**6**
11	PenglungEW	60	325	73.33	120	**93.33**	**157**
12	SonarEW	80.95	60	80.95	23	**97.61**	**21**
13	SpectEW	72.22	22	90.24	**14**	**92.59**	**14**
14	Tic-tac-toe	75.52	9	72.92	6	**82.92**	**5**
15	Vote	98.33	16	**100**	8	**100**	**3**
16	WaveformEW	82.2	40	82.5	25	**85.8**	**18**
17	WineEW	**100**	13	**100**	3	**100**	**2**
18	Zoo	**100**	16	**100**	5	**100**	**3**

Table 9
Comparison of BSNDO with state-of-the-art FS methods based on achieved classification accuracy tested on UCI datasets (highest classification accuracies are highlighted).

Dataset	BSNDO	SSDs+LAHC	HSGW	RSGW	ASGW	BGA	BPSO	EHHM	ECWSA-4
Breastcancer	**100**	98.93	98.6	97.1	98.5	97.43	96.29	**100**	95.21
BreastEW	98.25	98.25	98.1	98.2	**100**	97.54	97.19	**100**	97.38
CongressEW	**100**	**100**	97.5	96.1	99.4	96.79	96.33	98.85	96.23
Exactly	**100**	**100**	**100**	99.7	99.9	**100**	**100**	**100**	78.09
Exactly2	80.5	79	**81.5**	77.9	77.7	77	76.8	79.1	78.9
HeartEW	90.74	91.67	**92.3**	84.8	83.1	87.41	83.7	90.7	85.63
IonosphereEW	95.74	96.43	94.4	97.8	97.2	94.89	94.89	**98.6**	86.79
KrvskpEW	98.44	97.81	97.3	97.2	97.1	**98.5**	97.31	97.81	93.53
Lymphography	96.67	96.67	93.4	89.3	88.4	83.78	89.19	**96.9**	87.02
M-of-n	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	92.47
PenglungEW	**100**	**100**	94.2	**100**	**100**	91.89	91.89	**100**	87.63
SonarEW	95.24	97.62	96.4	97.9	94.8	**99.04**	94.23	92.85	76.84
SpectEW	**96.22**	95.15	86.2	81.5	87	89.55	88.81	90.74	79.84
Tic-tac-toe	**87.5**	87.24	82.8	85.9	86.5	79.96	79.96	85	78.75
Vote	**100**	**100**	98.3	99.6	98.4	97.33	96	98.4	95.08
WaveformEW	85	84.4	74.8	75.7	74.6	78.36	75.6	**86.8**	80.18
WineEW	**100**	**100**	**100**	**100**	**100**	98.88	97.75	**100**	98.02
Zoo	**100**	**100**	**100**	**100**	**100**	90.2	96.08	100	98.95
Avg rank	**1.833**	2	3.5	3.944	4	4.22	5.33	2.33	5.944
Ass rank	**1**	2	4	5	6	7	8	3	9

Table 10
Comparison of BSNDO with state-of-the-art methods based on number of selected features tested on UCI datasets (least number of selected features are highlighted).

Dataset	BSNDO	SSDs+LAHC	HSGW	RSGW	ASGW	BGA	BPSO	EHHM	ECWSA-4
Breastcancer	4	**2.5**	5	5.933	4.867	4	4	4	7
BreastEW	**4**	9	16.667	17.5	15.833	8	9	13	15
CongressEW	7	5.5	8.867	9.7	8.833	**2**	3	7	4
Exactly	**6**	**6**	6.7	7.1	6.867	**6**	**6**	7	7
Exactly2	8	8	9.033	9.2	7.933	**1**	**1**	5	9
HeartEW	4	5	8.767	6.133	6.367	5	**3**	8	9
IonosphereEW	16	12	18.167	20.5	17.3	**7**	**7**	**7**	10
KrvskpEW	22	20	24.8	24.8	24.5	**11**	12	15	16
Lymphography	**5**	6.5	10.567	10.567	11.2	**5**	**5**	6	10
M-of-n	6	6	6.8	7.1	6.867	6	6	7	**5**
PenglungEW	187	140	135.33	181.2	170.3	84	130	**74**	93
SonarEW	27	23.5	34.3	36.433	35.5	**19**	22	22	23
SpectEW	6	9	10.233	13.3	10.167	**5**	6	11	7
Tic-tac-toe	8	9	7	7	7	**5**	6	6	8
Vote	**3**	4.5	7.567	8.8	8.967	5	**3**	5	6
WaveformEW	33	22.5	26.933	27.533	25.833	**15**	**15**	20	**15**
WineEW	3	3	4.533	5.867	5.933	4	5	**1**	7
Zoo	5	4.5	5.533	5.3	7.6	4	5	**1**	7
Avg rank	3.5	3.22	5.277	6.277	5.33	**1.61**	2.055	2.944	4.16
Ass rank	5	4	7	9	8	**1**	2	3	6

Table 11
p-values generated *via* pairwise Wilcoxon test using the results obtained from 20 independent runs of the proposed BSNDO method and state-of-the-art FS methods used for comparison.

Dataset	SSDs+LAHC	HSGW	RSGW	ASGW	BGA	BPSO	EHHM	ECWSA-4
Breastcancer	0.031623	0.000212	0.000292	0.013724	0.000126	0.000392	0.000455	0.000392
BreastEW	0.275234	0.00029	0.000119	0.284088	0.003088	0.001373	1.91E−06	1.91E−06
CongressEW	0.599266	0.000138	0.000297	0.017474	0.000161	0.008211	0.176853	0.000212
Exactly	0.022958	8.83E−05	8.73E−05	0.000131	8.54E−05	8.72E−05	0.000131	0.000618
Exactly2	7.88E−05	8.72E−05	8.66E−05	0.000127	0.000153	7.99E−05	0.00021	0.974353
HeartEW	0.359797	8.81E−05	0.000102	0.00023	0.000153	8.72E−05	0.521673	8.72E−05
IonosphereEW	0.925575	0.000291	0.000127	0.003191	0.000845	0.047031	0.294252	1.91E−06
KrvskpEW	0.000155	8.86E−05	8.83E−05	0.000515	8.81E−05	0.000132	1.91E−06	0.009436
Lymphography	0.510917	0.000127	8.34E−05	0.001458	0.000285	0.000213	3.62E−05	0.000119
M-of-n	0.072789	8.77E−05	8.79E−05	8.77E−05	8.82E−05	0.000131	0.000618	8.72E−05
PenglungEW	0.054977	0.000234	0.014786	0.145537	0.213681	0.474082	0.009436	8.77E−05
SonarEW	0.264962	0.000115	0.00013	0.000527	0.937537	0.011806	1.91E−06	0.000127
SpectEW	0.262646	8.77E−05	0.000212	0.011068	0.000178	0.00013	1.91E−06	0.000127
Tic-tac-toe	0.155787	8.78E−05	8.83E−05	0.021748	0.000187	8.84E−05	0.452375	8.84E−05
Vote	0.166793	0.000128	0.000127	0.029028	0.00019	0.000406	4.77E−05	9.02E−05
WaveformEW	0.000182	8.86E−05	0.000103	8.77E−05	8.86E−05	8.83E−05	1.91E−06	0.000297
WineEW	0.365712	9.95E−05	8.71E−05	0.000859	0.05349	0.000269	0.001341	0.000269
Zoo	0.763025	9.02E−05	0.000104	0.0455	0.001689	0.000147	0.974353	8.77E−05

Table 12
Description of datasets used to check the robustness of BSNDO.

Dataset	Number of features	Number of samples	Number of classes
AMLGSA2191	12 616	54	2
DLBCL	7070	77	2
Leukaemia	5147	72	2
Prostate	12 533	102	2
MLL	12 533	72	3
SRBCT	2308	83	4

To make a quantitative decision about a process, we perform a statistical test. The goal of this test is to determine whether there is enough clarity to “reject” a conjecture about the process. The conjecture is called the null hypothesis. In our case, the null hypothesis states that the two sets of results have the same distribution, which implies that if the distribution of two results is statistically different, then the generated *p*-value from the test statistics will be <0.05 when the test is performed at 0.05% significance level. This will result in the rejection of the null hypothesis. So, to determine the statistical significance of the BSNDO algorithm, Wilcoxon rank-sum test (Wilcoxon, 1992) has been performed. It is a non-parametric statistical test where a pairwise comparison is performed. Individual meta-heuristic algorithms were run 20 times for each UCI dataset used here to perform the statistical test. From the test results provided in Table 11, we can conclude that the results of the proposed BSNDO algorithm is found to be statistically significant.

7. Additional testing on microarray datasets

The reported results, mentioned above, establish the fact that BSNDO performs better than the state-of-the-art methods considered here for comparison. To check the robustness of the proposed method, we have applied it on several high-dimensional microarray datasets (Ahmed et al., 2020a). The description of the datasets is given in Table 12. To confirm the superiority of the proposed method, it is compared with some state-of-the-art methods, namely: GA (Ghosh et al., 2018a), Memetic algorithm (MA) (Ghosh et al., 2019b, 2018b), WFACOFS (Ghosh et al., 2019a) and ECWSA (Guha et al., 2020). The comparison table is given in Table 13

As microarray datasets consist of a very high number of attributes, it becomes a challenging task for us to rule out the irrelevant ones. The obtained results again demonstrate the effectiveness of BSNDO. From this Table 13, we can see that BSNDO produces noteworthy results as compared to the other methods considered here for comparison. It produces 100% accuracy in every dataset except Prostrate. It also selects the least features in the case of AMLGSE2191 and SRBCT.

8. Testing on COVID-19 dataset

COVID-19 is a contagious disease, which is caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). The common symptoms of COVID-19 are fever, cough, fatigue, breathing difficulties, and loss of smell and taste etc. Symptoms begin 1 to 14 days after exposure to the virus. While most people have mild symptoms, some people develop acute respiratory distress syndrome (ARDS). Because of its nature, the accurate result of the COVID-19 test has become a challenging task. Some recent COVID-19 screening techniques are Bandyopadhyay et al. (2021), Barnes et al. (2021), Dey et al. (2021), Ismael and Şengür (2021), Kundu et al. (2021) and Nigam et al. (2021). More than 190 million people is suffering from COVID-19, and more than 4 million people have died because of it. So, detecting the COVID-19 and keeping those people in quarantine have become one of the topmost priorities of every country. Though vaccination process has started, it will take time to reach everyone, especially in the under-development countries.

We have tested our FS method on a COVID-19 dataset, which is publicly available at https://github.com/Atharva-Peshkar/Covid-19-Patient-Health-Analytics in csv format. This dataset contains 1086 instances and 74 attributes. The obtained results are compared with some meta-heuristics based FS methods: SSDs+LAHC, ASGW, HSGW, RSGW, GA, PSO and Adaptive β-coral reefs optimization (AβCRO) (Ahmed et al., 2020a). The comparison table shows the achieved classification accuracy and the number of selected features (in brackets) (see Table 14).

9. Conclusion and future work

In this work, a new FS method, called BSNDO, which is based on GNDO and SA has been proposed. SA has been used as a local search to enhance the exploitation of the GNDO and to create a proper balance between exploration and exploitation of the overall method. The proposed method shows significant improvement in achieved classification accuracy while FS is performed using BSNDO than BGNDO and some state-of-the-art methods. Primarily the method has been tested on various UCI datasets. To prove the robustness of the model, the proposed method is also applied on high dimensional microarray datasets. Furthermore, it is experimented on a COVID-19 dataset for detecting the COVID-19 cases. This is to be noted that all the datasets used here are publicly available. The obtained results show the applicability of BSNDO in varied datasets. One of the limitations of this method may be the computational complexity due to addition of the local search to the GNDO algorithm. As a future scope of this work, a deeper analysis of the gene selections done by BSNDO and their biological impact can be studied. The proposed FS method can also be applied to some

Table 13
Comparison of the results of BSNDO on microarray with state-of-the-art methods. The number of features selected is provided in brackets at the side of the accuracy.

Dataset	GA	MA	WFACOFS	ECWSA-1	ECWSA-2	ECWSA-3	ECWSA-4	BSNDO
AMLGSE2191	100(98)	100(91)	96.3(17)	96.67(17)	**100(9)**	95.83(16)	95.83(18)	**100(9)**
DLBCL	100(88)	100(105)	**100(3)**	100(29)	100(24)	100(26)	100(31)	100(10)
Leukaemia	100(85)	100(65)	100(5)	97.22(7)	100(8)	**100(4)**	97.22(5)	100(12)
Prostrate	**100(99)**	100(107)	100(22)	96.3(16)	98.15(16)	96.3(9)	96.3(19)	95.24(20)
MLL	100(94)	100(80)	100(25)	100(16)	100(17)	**100(8)**	100(15)	100(16)
SRBCT	100(78)	100(50)	100(19)	100(45)	100(32)	100(34)	100(30)	**100(11)**

Table 14
Comparison of achieved classification accuracy evaluated on mentioned COVID-19 dataset.

GNDO+SA	SSDs+LAHC	ASGW	HSGW	RSGW	GA	PSO	AβCRO
98.61 (26)	97.69 (23)	97.69 (40)	96.31 (50)	97.75 (54)	94.9 (23)	97.24 (31)	98.2(20)

other real-world problems like handwritten word or digit recognition, and face recognition etc., where researchers sometimes use very high dimensional feature vectors without knowing the importance of all the features.

CRediT authorship contribution statement

Shameem Ahmed: Conceptualization, Methodology, Writing – original draft, Software, Investigation. **Khalid Hassan Sheikh:** Writing – review & editing, Software, Investigation, Conceptualization. **Seyedali Mirjalili:** Writing – review & editing, Supervision, Project administration, Validation. **Ram Sarkar:** Writing – review & editing, Supervision, Project administration, Validation, Funding acquisition.

Declaration of competing interest

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.

Acknowledgement

We would like to thank the Centre for Microprocessor Applications for Training, Education and Research (CMATER) laboratory of the Computer Science and Engineering Department, Jadavpur University, Kolkata, India for providing us with the infrastructural support.

References

Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Metaheuristic algorithms: A comprehensive review. In *Computational intelligence for multimedia big data on the cloud with engineering applications* (pp. 185–231). Elsevier, http://dx.doi.org/10.1016/b978-0-12-813314-9.00010-4.

Ahmed, S., Ghosh, K. K., Garcia-Hernandez, L., Abraham, A., & Sarkar, R. (2020). Improved coral reefs optimization with adaptive β-hill climbing for feature selection. *Neural Computing and Applications*, http://dx.doi.org/10.1007/s00521-020-05409-1.

Ahmed, S., Ghosh, K. K., Mirjalili, S., & Sarkar, R. (2021). AIEOU: Automata-based improved equilibrium optimizer with U-shaped transfer function for feature selection. *Knowledge-Based Systems*, *228*, Article 107283. http://dx.doi.org/10.1016/j.knosys.2021.107283.

Ahmed, S., Ghosh, K. K., Singh, P. K., Geem, Z. W., & Sarkar, R. (2020). Hybrid of harmony search algorithm and ring theory-based evolutionary algorithm for feature selection. *IEEE Access*, *8*, 102629–102645. http://dx.doi.org/10.1109/access.2020.2999093.

Al-Betar, M. A. (2016). β-hill climbing: an exploratory local search. *Neural Computing and Applications*, *28*(S1), 153–168. http://dx.doi.org/10.1007/s00521-016-2328-2.

Al-Betar, M. A., Aljarah, I., Awadallah, M. A., Faris, H., & Mirjalili, S. (2019). Adaptive β-hill climbing for optimization. *Soft Computing*, *23*(24), 13489–13512. http://dx.doi.org/10.1007/s00500-019-03887-7.

Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Computers & Structures*, *169*, 1–12. http://dx.doi.org/10.1016/j.compstruc.2016.03.001.

au2, I. F. J., Yang, X.-S., Fister, I., Brest, J., & Fister, D. (2013). A brief review of nature-inspired algorithms for optimization. arXiv:1307.4186.

Bandyopadhyay, R., Basu, A., Cuevas, E., & Sarkar, R. (2021). Harris hawks optimisation with simulated annealing as a deep feature selection method for screening of COVID-19 CT-scans. *Applied Soft Computing*, *111*, Article 107698. http://dx.doi.org/10.1016/j.asoc.2021.107698.

Barnes, S. J., Diaz, M., & Arnaboldi, M. (2021). Understanding panic buying during COVID-19: A text analytics approach. *Expert Systems with Applications*, *169*, Article 114360. http://dx.doi.org/10.1016/j.eswa.2020.114360.

Bhattacharyya, T., Chatterjee, B., Singh, P. K., Yoon, J. H., Geem, Z. W., & Sarkar, R. (2020). Mayfly in harmony: A new hybrid meta-heuristic feature selection algorithm. *IEEE Access*, *8*, 195929–195945. http://dx.doi.org/10.1109/access.2020.3031718.

Boutsidis, C., Drineas, P., & Magdon-Ismail, M. (2014). Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, *43*(2), 687–717. http://dx.doi.org/10.1137/12086755x.

Boutsidis, C., Drineas, P., & Mahoney, M. W. (2009). Unsupervised feature selection for the k-means clustering problem. *Advances in Neural Information Processing Systems*, *22*, 153–161.

Chatterjee, B., Bhattacharyya, T., Ghosh, K. K., Singh, P. K., Geem, Z. W., & Sarkar, R. (2020). Late acceptance hill climbing based social ski driver algorithm for feature selection. *IEEE Access*, *8*, 75393–75408. http://dx.doi.org/10.1109/access.2020.2988157.

Cortinovis, A., & Kressner, D. (2020). Low-rank approximation in the frobenius norm by column and row subset selection. *SIAM Journal on Matrix Analysis and Applications*, *41*(4), 1651–1673. http://dx.doi.org/10.1137/19m1281848.

Das, S., Roy, S. D., Malakar, S., Velásquez, J. D., & Sarkar, R. (2021). Bi-level prediction model for screening COVID-19 patients using chest X-Ray images. *Big Data Research*, *25*, Article 100233. http://dx.doi.org/10.1016/j.bdr.2021.100233.

Davis, L. (1991). *Handbook of genetic algorithms*. CumInCAD.

Dey, S., Bhattacharya, R., Malakar, S., Mirjalili, S., & Sarkar, R. (2021). Choquet fuzzy integral-based classifier ensemble technique for COVID-19 detection. *Computers in Biology and Medicine*, *135*, Article 104585. http://dx.doi.org/10.1016/j.compbiomed.2021.104585.

Drineas, P., Mahoney, M. W., & Muthukrishnan, S. (2008). Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, *30*(2), 844–881. http://dx.doi.org/10.1137/07070471x.

Dua, D., & Graff, C. (2017). UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.

Elgamal, Z. M., Yasin, N. B. M., Tubishat, M., Alswaitti, M., & Mirjalili, S. (2020). An improved harris hawks optimization algorithm with simulated annealing for feature selection in the medical field. *IEEE Access*, *8*, 186638–186652. http://dx.doi.org/10.1109/ACCESS.2020.3029728.

Emary, E., Zawbaa, H. M., & Hassanien, A. E. (2016). Binary grey wolf optimization approaches for feature selection. *Neurocomputing*, *172*, 371–381. http://dx.doi.org/10.1016/j.neucom.2015.06.083.

Eusuff, M., Lansey, K., & Pasha, F. (2006). Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. *Engineering Optimization*, *38*(2), 129–154. http://dx.doi.org/10.1080/03052150500384759.

Garain, A., Basu, A., Giampaolo, F., Velasquez, J. D., & Sarkar, R. (2021). Detection of COVID-19 from CT scan images: A spiking neural network-based approach. *Neural Computing and Applications*, http://dx.doi.org/10.1007/s00521-021-05910-1.

Geem, Z. W., Kim, J. H., & Loganathan, G. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation*, *76*(2), 60–68. http://dx.doi.org/10.1177/003754970107600201.

Gendreau, M., & Potvin, J.-Y. (2005). Metaheuristics in combinatorial optimization. *Annals of Operations Research*, *140*(1), 189–213. http://dx.doi.org/10.1007/s10479-005-3971-7.

Ghosh, M., Adhikary, S., Ghosh, K. K., Sardar, A., Begum, S., & Sarkar, R. (2018). Genetic algorithm based cancerous gene identification from microarray data using ensemble of filter methods. *Medical & Biological Engineering & Computing*, *57*(1), 159–176. http://dx.doi.org/10.1007/s11517-018-1874-4.

Ghosh, K. K., Ahmed, S., Singh, P. K., Geem, Z. W., & Sarkar, R. (2020). Improved binary sailfish optimizer based on adaptive β-hill climbing for feature selection. *IEEE Access*, *8*, 83548–83560. http://dx.doi.org/10.1109/access.2020.2991543.

Ghosh, M., Guha, R., Sarkar, R., & Abraham, A. (2019). A wrapper-filter feature selection technique based on ant colony optimization. *Neural Computing and Applications*, *32*(12), 7839–7857. http://dx.doi.org/10.1007/s00521-019-04171-3.

Ghosh, M., Kundu, T., Ghosh, D., & Sarkar, R. (2019). Feature selection for facial emotion recognition using late hill-climbing based memetic algorithm. *Multimedia Tools and Applications*, *78*(18), 25753–25779. http://dx.doi.org/10.1007/s11042-019-07811-x.

Ghosh, M., Malakar, S., Bhowmik, S., Sarkar, R., & Nasipuri, M. (2018). Feature selection for handwritten word recognition using memetic algorithm. In *Studies in computational intelligence* (pp. 103–124). Springer Singapore, http://dx.doi.org/10.1007/978-981-10-8974-9_6.

Glover, F., & Laguna, M. (1998). Tabu search. In *Handbook of combinatorial optimization* (pp. 2093–2229). Springer.

Guha, R., Ghosh, M., Mutsuddi, S., Sarkar, R., & Mirjalili, S. (2020). Embedded chaotic whale survival algorithm for filter-wrapper feature selection. *Soft Computing*, *24*(17), 12821–12843. http://dx.doi.org/10.1007/s00500-020-05183-1.

Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Hatamlou, A. (2013). Black hole: A new heuristic optimization approach for data clustering. *Information Sciences*, *222*, 175–184. http://dx.doi.org/10.1016/j.ins.2012.08.023.

He, X., Cai, D., & Niyogi, P. (2006). Laplacian score for feature selection. In *Advances in neural information processing systems* (pp. 507–514).

Hillis, W. (1990). Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena*, *42*(1–3), 228–234. http://dx.doi.org/10.1016/0167-2789(90)90076-2.

Ismael, A. M., & Şengür, A. (2021). Deep learning approaches for COVID-19 detection based on chest X-ray images. *Expert Systems with Applications*, *164*, Article 114054. http://dx.doi.org/10.1016/j.eswa.2020.114054.

Jain, M., Singh, V., & Rani, A. (2019). A novel nature-inspired algorithm for optimization: Squirrel search algorithm. *Swarm and Evolutionary Computation*, *44*, 148–175. http://dx.doi.org/10.1016/j.swevo.2018.02.013.

Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, *39*(3), 459–471. http://dx.doi.org/10.1007/s10898-007-9149-x.

Karbhari, Y., Basu, A., Geem, Z. W., Han, G.-T., & Sarkar, R. (2021). Generation of synthetic chest X-ray images and detection of COVID-19: A deep learning based approach. *Diagnostics*, *11*(5), 895. http://dx.doi.org/10.3390/diagnostics11050895.

Kashan, A. H., Tavakkoli-Moghaddam, R., & Gen, M. (2019). Find-fix-finish-exploit-analyze (F3EA) meta-heuristic algorithm: An effective algorithm with new evolutionary operators for global optimization. *Computers & Industrial Engineering*, *128*, 192–218. http://dx.doi.org/10.1016/j.cie.2018.12.033.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks, Vol. 4* (pp. 1942–1948). IEEE.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680. http://dx.doi.org/10.1126/science.220.4598.671.

Koza, J. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, *4*(2), http://dx.doi.org/10.1007/bf00175355.

Kundu, R., Basak, H., Singh, P. K., Ahmadian, A., Ferrara, M., & Sarkar, R. (2021). Fuzzy rank-based fusion of CNN models using gompertz function for screening COVID-19 CT-scans. *Scientific Reports*, *11*(1), http://dx.doi.org/10.1038/s41598-021-93658-y.

Kurtuluş, E., Yıldız, A. R., Sait, S. M., & Bureerat, S. (2020). A novel hybrid harris hawks-simulated annealing algorithm and RBF-based metamodel for design optimization of highway guardrails. *Materials Testing*, *62*(3), 251–260. http://dx.doi.org/10.3139/120.111478.

Lai, C., Reinders, M. J., & Wessels, L. (2006). Random subspace method for multivariate feature selection. *Pattern Recognition Letters*, *27*(10), 1067–1076. http://dx.doi.org/10.1016/j.patrec.2005.12.018.

Liu, H., & Motoda, H. (2012). *Feature selection for knowledge discovery and data mining, Vol. 454*. Springer Science & Business Media.

Mafarja, M. M., & Mirjalili, S. (2017). Hybrid whale optimization algorithm with simulated annealing for feature selection. *Neurocomputing*, *260*, 302–312. http://dx.doi.org/10.1016/j.neucom.2017.04.053.

Mafarja, M., Qasem, A., Heidari, A. A., Aljarah, I., Faris, H., & Mirjalili, S. (2019). Efficient hybrid nature-inspired binary optimizers for feature selection. *Cognitive Computation*, *12*(1), 150–175. http://dx.doi.org/10.1007/s12559-019-09668-6.

Mirjalili, S. (2016). SCA: A Sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, *96*, 120–133. http://dx.doi.org/10.1016/j.knosys.2015.12.022.

Mirjalili, S., & Lewis, A. (2013). S-shaped versus V-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation*, *9*, 1–14. http://dx.doi.org/10.1016/j.swevo.2012.09.002.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67. http://dx.doi.org/10.1016/j.advengsoft.2016.01.008.

Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2015). Multi-verse optimizer: a nature-inspired algorithm for global optimization. *Neural Computing and Applications*, *27*(2), 495–513. http://dx.doi.org/10.1007/s00521-015-1870-7.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46–61. http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.

Nematollahi, A. F., Rahiminejad, A., & Vahidi, B. (2019). A novel meta-heuristic optimization method based on golden ratio in nature. *Soft Computing*, *24*(2), 1117–1151. http://dx.doi.org/10.1007/s00500-019-03949-w.

Nigam, B., Nigam, A., Jain, R., Dodia, S., Arora, N., & Annappa, B. (2021). COVID-19: Automatic detection from X-ray images by utilizing deep learning methods. *Expert Systems with Applications*, *176*, Article 114883. http://dx.doi.org/10.1016/j.eswa.2021.114883.

Passino, K. (2002). Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Systems*, *22*(3), 52–67. http://dx.doi.org/10.1109/mcs.2002.1004010.

Paul, S., Magdon-Ismail, M., & Drineas, P. (2016). Feature selection for linear SVM with provable guarantees. *Pattern Recognition*, *60*, 205–214. http://dx.doi.org/10.1016/j.patcog.2016.05.018.

Pudil, P., Novovičová, J., & Kittler, J. (1994). Floating search methods in feature selection. *Pattern Recognition Letters*, *15*(11), 1119–1125. http://dx.doi.org/10.1016/0167-8655(94)90127-9.

Rao, R., Savsani, V., & Vakharia, D. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, *43*(3), 303–315. http://dx.doi.org/10.1016/j.cad.2010.12.015.

Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences*, *179*(13), 2232–2248. http://dx.doi.org/10.1016/j.ins.2009.03.004.

Ray, T., & Liew, K. (2003). Society and civilization: an optimization algorithm based on the simulation of social behavior. *IEEE Transactions on Evolutionary Computation*, *7*(4), 386–396. http://dx.doi.org/10.1109/tevc.2003.814902.

Ryan, C., Collins, J., & Neill, M. O. (1998). Grammatical evolution: Evolving programs for an arbitrary language. In *Lecture notes in computer science* (pp. 83–96). Springer Berlin Heidelberg, http://dx.doi.org/10.1007/bfb0055930.

Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper optimisation algorithm: Theory and application. *Advances in Engineering Software*, *105*, 30–47. http://dx.doi.org/10.1016/j.advengsoft.2017.01.004.

Sheikh, K. H., Ahmed, S., Mukhopadhyay, K., Singh, P. K., Yoon, J. H., Geem, Z. W., & Sarkar, R. (2020). EHHM: Electrical harmony based hybrid meta-heuristic for feature selection. *IEEE Access*, *8*, 158125–158141. http://dx.doi.org/10.1109/access.2020.3019809.

Simon, D. (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation*, *12*(6), 702–713. http://dx.doi.org/10.1109/tevc.2008.919004.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation, Vol. 74*. John Wiley & Sons.

Tan, Y., & Zhu, Y. (2010). Fireworks algorithm for optimization. In *Lecture notes in computer science* (pp. 355–364). Springer Berlin Heidelberg, http://dx.doi.org/10.1007/978-3-642-13495-1_44.

Tripathi, R., & Reza, A. (2020). A subset selection based approach to structural reducibility of complex networks. *Physica A: Statistical Mechanics and its Applications*, *540*, Article 123214. http://dx.doi.org/10.1016/j.physa.2019.123214.

Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., & Shochet, O. (1995). Novel type of phase transition in a system of self-driven particles. *Physical Review Letters*, *75*, 1226–1229. http://dx.doi.org/10.1103/PhysRevLett.75.1226, URL: https://link.aps.org/doi/10.1103/PhysRevLett.75.1226.

Wilcoxon, F. (1992). Individual comparisons by ranking methods. In *Springer series in statistics* (pp. 196–202). Springer New York, http://dx.doi.org/10.1007/978-1-4612-4380-9_16.

Wolpert, D., & Macready, W. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67–82. http://dx.doi.org/10.1109/4235.585893.

Xue, X., Yao, M., & Cheng, R. (2011). A novel selection operator of cultural algorithm. In *Advances in Intelligent and Soft Computing* (pp. 71–77). Springer Berlin Heidelberg, http://dx.doi.org/10.1007/978-3-642-25661-5_10.

Yang, X.-S. (2009). Firefly algorithms for multimodal optimization. In *Stochastic algorithms: foundations and applications* (pp. 169–178). Springer Berlin Heidelberg, http://dx.doi.org/10.1007/978-3-642-04944-6_14.

Zhang, Y., Jin, Z., & Mirjalili, S. (2020). Generalized normal distribution optimization and its applications in parameter extraction of photovoltaic models. *Energy Conversion and Management*, *224*, Article 113301. http://dx.doi.org/10.1016/j.enconman.2020.113301.

Zheng, Z., Wu, X., & Srihari, R. (2004). Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter*, *6*(1), 80–89. http://dx.doi.org/10.1145/1007730.1007741.