Hybrid binary bat enhanced particle swarm optimization algorithm for solving feature selection problems

Mohamed A. Tawhid
Department of Mathematics and Statistics, Faculty of Science, Thompson Rivers University, Kamloops, Canada and Department of Mathematics and Computer Science, Faculty of Science, Alexandria University, Alexandria, Egypt, and
Kevin B. Dsouza
Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, Canada

Received 15 January 2018
Revised 31 March 2018
Accepted 4 April 2018

Abstract
In this paper, we present a new hybrid binary version of bat and enhanced particle swarm optimization algorithm in order to solve feature selection problems. The proposed algorithm is called Hybrid Binary Bat Enhanced Particle Swarm Optimization Algorithm (HBBEPSO). In the proposed HBBEPSO algorithm, we combine the bat algorithm with its capacity for echolocation helping explore the feature space and enhanced version of the particle swarm optimization with its ability to converge to the best global solution in the search space. In order to investigate the general performance of the proposed HBBEPSO algorithm, the proposed algorithm is compared with the original optimizers and other optimizers that have been used for feature selection in the past. A set of assessment indicators are used to evaluate and compare the different optimizers over 20 standard data sets obtained from the UCI repository. Results prove the ability of the proposed HBBEPSO algorithm to search the feature space for optimal feature combinations.

Keywords Particle Swarm Optimization, Bat algorithm, Binary Algorithms, Hybridization, Meta-heuristics, Feature selection problem
Paper type Original Article

We are grateful to the anonymous 4 reviewers for constructive feedback and insightful suggestions which greatly improved this article. This research was supported partially by Mitacs Canada. The research of the 1st author is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

Applied Computing and Informatics
Vol. 16 No. 1/2, 2020
pp. 117-136
Emerald Publishing Limited
e-ISSN: 2210-8327
p-ISSN: 2634-1964
DOI 10.1016/j.aci.2018.04.001

1. Introduction

Feature selection is a way for identifying the independent features and removing expendable ones from the dataset [1]. The objectives of feature selection are dimensionality reduction of the data, improving accuracy of prediction, and understanding data for different machine learning applications such as clustering, classification, regression and computer vision [2]. It is also widely used in the analysis of economic and trade markets. In the real world, data representation often uses too many features, which means certain independent features can fill in for others and the redundant features can be removed. Moreover, the output is influenced by the relevant features because they contain important information about the data and the results will be obscure if any of them are left out [3]. The classical optimization techniques have some limitations in solving the feature selection problems [4] and hence evolutionary computation (EC) algorithms are the alternative for solving these limitations and searching for the best solution [5]. Evolutionary Computation (EC) algorithms are inspired by nature, group dynamics, social behavior, and biological interaction of species in a group. The binary version of these algorithms allow us to investigate problems like feature selection and arrive at superior results.

Many heuristic algorithms have been used in an attempt to solve the feature selection problem. A survey on evolutionary computation approaches to feature selection is explained in [5]. A binary PSO based method with a mutation operator is introduced in [6] to achieve spam detection using decision trees. A wavelet entropy based feature selection approach is used in [7] to detect abnormal MR brains. Ref. [8] delineates about a Firefly based feature selection approach. A binary bat based feature selection method is shed light upon in [9]. Ref. [35] elaborates a feature subset selection approach by Grey wolf optimization. Even hybrid algorithms have been used to solve feature selection problems. A hybrid genetic algorithm on mutual information is presented in [10]. Ref. [11] expounds a hybrid flower pollination algorithms for feature selection.

Bat algorithm was recently developed by Yang which is based on the ability of bats to use echolocation to sense distance and also to distinguish between prey and background barriers [12]. The bat algorithm and its variants have been used in many computing applications. A binary bat algorithm is suggested in [37] to solve unconstrained optimization bench test problems and compared with binary GA and binary PSO. Also, a binary bat algorithm for feature selection is presented in [9]. A combination of K-means and bat algorithm is used for efficient clustering in [13]. A variant fuzzy bat algorithm is proposed in [14]. Multi-objective optimization problems are dealt with to solve engineering design benchmarks in [15]. A variant of bat algorithm using differential operator and Levy flights to solve function optimization problems is delineated in [16].

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995 [17], inspired by social behavior of bird flocking and fish schooling. A comprehensive survey about PSO and its applications can be found at [18]. In past several years, even though PSO has been successfully applied in many research and application areas like the constrained non-linear optimization problems [19], for optimal design of combinational logic circuits [20] and also to real world hydraulic problems [21], little work is seen in the domain of feature selection [22]. A multi-objective approach using PSO is introduced in [23,24]. Also, a bare bones PSO technique is delineated in [25]. It is demonstrated that PSO gets better results in a faster, cheaper way compared with other methods. Another reason that PSO is attractive is that there are few parameters to tweak. One version, with slight variations, works well in a wide variety of real world applications. Here an enhanced version of the standard PSO is used [26] to solve the feature selection problem.

Hybridization of different algorithmic concepts is a method to obtain better performing systems and is believed to benefit from synergy, i.e. usually it exploits and unites advantages of the individual pure strategies. It is mostly due to the no free lunch theorems [27], that the generalized view of metaheuristics changed and people recognized that there cannot exist a general optimization strategy which is globally better than any other. In fact, to solve a problem at hand most effectively, it almost always requires a specialized algorithm that needs to be compiled of adequate parts. Hybridization is classified into many categories [28,29]. Hybridization of one metaheuristic with another is a popular method to enhance the performance of both the algorithms.

The aim of this work is to propose a new hybrid binary version of bat and enhanced PSO algorithm in order to solve feature selection problems effectively. The hybridization allows us to combine the best aspects of both these algorithms and obtain better performance. In this paper, we propose a new hybrid algorithm, which is called HBBEPSO Algorithm by combining the bat Algorithm with the enhanced PSO algorithm in order to obtain superior results when compared to the respective individual algorithms. The binary HBBEPSO algorithm is tested on 20 standard data sets obtained from the UCI repository [30]. The algorithm is also compared with the HBEPSOB, where the PSO is carried out first and then given to the bat algorithm. A set of assessment indicators is used to evaluate and compare the different optimizers. The experimental results show the ability of the proposed binary HBBEPSO algorithm to search the feature space for optimal feature combinations.

The reminder of this paper is organized as follows. Section 2 presents the definition of the feature selection problem. Section 3 summarizes the main concepts of the bat algorithm. Section 4 describes the main concepts of the enhanced PSO algorithm. Section 5 presents the main structure of the proposed binary HBBEPSO algorithm. The Section 6 provides details about the feature selection problem, evaluation criteria and an insight about the classifier used. Section 7 reports the experimental results and finally, the conclusion and some future work make up Section 8.

2. Definition of the feature selection problem

In real life machine learning applications thousand of features are measured while only handful of them contain useful information. Therefore, we need methods to reduce the dimensionality of our feature set. This can be achieved by two ways, feature reduction and feature selection. Feature reduction is when we apply some sort of transformation on our original feature set of dimension d to produce a new feature set of dimension m with $m < d$. Techniques like Linear Discriminant Analysis (LDA), and Principal Component Analysis (PCA) come under this category. Feature selection is the process of selecting a subset of the original features. In this section, we present the definition of the feature selection problem as follows.

The feature selection problem can be defined as the selection of certain number of features out of the total number of available features in such a way that the *classification performance* is maximum and the *number of selected features* is minimum.

$$fitness = \alpha\gamma_R(D) + \beta\frac{|C-R|}{|C|} \tag{1}$$

where $\gamma_R(D)$ is the classification quality of set R relative to decision D, R is the length of selected feature subset, C is the total number of features, α and β are two parameters corresponding to the importance of classification quality and subset length, $\alpha \in [0, 1]$ and $\beta = 1 - \alpha$. The fitness function maximizes the classification quality, $\gamma_R(D)$, and the ratio of the unselected features to the total number of features is described by $\frac{|C-R|}{|C|}$. The above equation can be easily converted into a minimization problem by using error rate rather than classification quality and using selected features ratio rather than using unselected feature size. The minimization problem can be formulated as in Eq. (2).

$$fitness = \alpha E_R(D) + \beta \frac{|R|}{|C|} \tag{2}$$

where $E_R(D)$ is the error rate of the classifier, R is the length of the selected feature subset and C is the total number of features. $\alpha \in [0, 1]$ and $\beta = 1 - \alpha$ are constants used to control the weights of classification accuracy and feature reduction.

3. Overview of binary bat algorithm

In the following subsection, we will give an overview of the main concepts and structure of the binary bat algorithm.

3.1 Main concepts and inspiration

The binary bat algorithm mimics the concept of echolocation of bats to sense distance and distinguish between prey and background barriers. The bats send out loud, short pulses of sound and can sense the distance by the time it takes for the echo to return to them [31]. This fascinating mechanism also allows bats to distinguish between barrier and prey, thus allowing them to hunt in complete darkness [32].

3.2 Definition of concepts

1. **Loudness:** This parameter (a) is used to eliminate solutions that are too loud and will hinder the bat from reaching the optimum. This mimics the loudness of the bat pulse.
2. **Pulse rate:** This parameter (r) mimics the rate of pulsing of the bat. It randomly assigns the best solution in the previous iteration to the present solution.
3. **Frequency:** This parameter (Q_i) is used to represent the frequency of the bat wave. It varies from a minimum to a maximum and the changes occur randomly. This parameter gives the weight to the separation of the current solution from the best solution in the space. The frequency is represented by a D dimensional vector and is initialized to zero.

$$Q_i = (Q_{i1}, Q_{i2}, \ldots, Q_{iD}) \tag{3}$$

4. **Velocity:** This parameter (v_i) is the resultant velocity of the bat at every iteration. The velocity is represented by a D dimensional vector and is initialized to zero.

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{iD}) \tag{4}$$

3.3 Binary bat algorithm

Using the concepts mentioned in the Section 3.2, the binary bat algorithm distinguishes between barrier and prey. It should also be noted that the bats can change the wavelength of their emitted pulses and the rate of emission based on their relative position with respect to the targets. In the context of feature selection, this gives the algorithm flexibility to adapt the changes in the feature space and explore better solutions.

The details of the algorithm are mentioned in Algorithm 1. We provide a brief overview of the binary bat algorithm. After initializing the position, frequency and velocity vectors, the best solution is noted and updated throughout the algorithm. This is done mainly using the following equations

$$Q_i = Q_{min} + (Q_{min} - Q_{max}) \cdot rand \tag{5}$$

```
1:  Set the initial value of swarm size $SS(N)$, $a$, $r$, $Q_{min}$, $Q_{max}$
    and $max_{iter}$.
2:  Randomly initialize the population as
    $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD}) \in S$ for each solution, the frequency
    vectors $Q$ as D dimensional zero vectors as in Eq. (3) and
    the velocity vectors $v$ as D dimensional zero vectors as in
    Eq. (4).
3:  Evaluate *fitness* of each solution using Eq. (2).
4:  Initialize $x_{temp}$ as D dimensional zero vectors.
5:  Store the best solution in *best* and minimum fitness in $F_{min}$
6:  Set $t := 0$. {Counter initialization}
7:  **for** $(i = 1; j < SS; i++)$ **do**
8:      **for** $(j = 1; j < D; j++)$ **do**
9:          $Q_i = Q_{min} + (Q_{min} - Q_{max}) \cdot rand$
10:         $v(i,j) = v(i,j) + (x(i,j) - best(j)) \cdot Q_i$
11:         $\tilde{x}(i,j) = x(i,j) + v(i,j)$
12:         Binarize $\tilde{x}(i,j)$
13:         $V\_value = \left|\frac{2}{\pi}\arctan\left(\frac{\pi}{2} v(i,j)\right)\right|$
14:         **if** $(rand < V\_value)$ **then**
15:             $x_{temp}(i,j) = \tilde{x}(i,j)$
16:         **else**
17:             $x_{temp}(i,j) = x(i,j)$
18:         **end if**
19:         **if** $(rand > r)$ **then**
20:             $x_{temp}(i,j) = best(j)$
21:         **end if**
22:     **end for**
23:     *fit* = fitness of $x_{temp}$.
24:     **if** $(fit < F_{min}$ & $rand < a)$ **then**
25:         $x(i) = x_{temp}(i)$
26:         update fitness.
27:     **end if**
28:     update *best* and $F_{min}$.
29: **end for**
30: $t = t + 1$ {Iteration counter is increasing}
31: **until**$(t < max_{iter})$ {Termination criteria are satisfied}.
32: Produce the best solution *best*.
```

Algorithm 1. Binary bat algorithm

$$v(i,j) = v(i,j) + (x(i,j) - best(j)) \cdot Q_i \tag{6}$$

$$\tilde{x}(i,j) = x(i,j) + v(i,j) \tag{7}$$

where *rand* denotes a randomly generated number in the interval (0,1) and $\tilde{x}$ represents the new solutions. These new solutions may not always be adopted and are updated depending on certain other parameters in the algorithm. A threshold is selected depending on the value of the velocity of the bat which will control the amount of exploration, the bat is capable of achieving as is given in Eq. (8). If a particular random number is less than this threshold value, the new solutions are updated and the bat moves on to the new solution space.

$$V\_value = \left|\frac{2}{\pi}\arctan\left(\frac{\pi}{2} v(i,j)\right)\right|. \tag{8}$$

The rate of pulse emission decides whether the bat will stick to the previous best solution obtained or adopt the newly updated solution. This is similar to the best global solution adoption step in most meta heuristics and helps to steer and clear off too much unnecessary exploration. The loudness parameter introduces a further filter to the adoption of the solution as the new accepted solution. The solution is only accepted if a random number chosen is lesser than the loudness value and the fitness of the new solution is better than the old solution.

As this solutions will be in the continuous space, a binary map needs to be applied to the solutions so as to make them compatible to feature selection. This map could be a regular squashing function like a sigmoid function or any other function that is capable of taking continuous values into the logistic space.

4. Overview of binary enhanced particle swarm optimization

In the following section, we will give an overview of the main concepts and structure of the binary enhanced PSO algorithm.

4.1 Main concepts and inspiration

The PSO is a population based search method inspired from the swarm behavior (information interchange) of birds [33]. In PSO, initially a random population of particles is initialized and these particles move with certain velocity based on their interaction with other particles in the population. At each iteration the personal best achieved by each particle and the global best of all the particles is tracked and the velocity of all the particles is updated based on this information. Certain parameters are used to give weights to the global and personal best. In the enhanced version of the binary PSO [26], special type of S shaped transfer functions is used to convert a continuous value to a binary value instead of a simple hyperbolic tangent function.

4.2 Movement of particles

Each of the particles is represented by D dimensional vectors and they are randomly initialized with each individual value being binary.

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{iD}) \in S \tag{9}$$

where S is the available search space. The velocity is represented by a D dimensional vector and is initialized to zero,

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{iD}). \tag{10}$$

The best personal(local) position recorded by each particle is maintained as

$$p_i = (p_{i1}, p_{i2}, \ldots, p_{iD}) \in S. \tag{11}$$

At each iteration, each particle changes its position according to its personal best(Pbest) and the global best(gbest) as follows

$$v_i^{(t+1)} = wv_i^{(t)} + c_1 r_{i1} \cdot (Pbest_i(t) - x_i^{(t)}) + c_2 r_{i2} \cdot (gbest - x_i^{(t)}) \tag{12}$$

where c_1 and c_2 are acceleration constants called cognitive and social parameters respectively. r_1 and r_2 are random values $\in[0, 1]$. w is called as the inertia weight. It determines how the previous velocity of the particle influences the velocity in the next iteration. The value of w is determined by the following expression

$$w = w_{max} - iteration \cdot \left(\frac{w_{max} - w_{min}}{Max\_iteration}\right) \quad (13)$$

where $w_{\max}$ and $w_{\min}$ are constants. *Max_iteration* is the maximum number of iterations.

4.3 The continuous to binary map

The position of each particle is determined by the S shaped as a transfer function that maps the continuous velocity value to the position of the particle. This is a special sigmoid function that enhances the PSO.

$$s = \frac{1}{1 + e^{-v_{i,j}}}, \quad i = 1, \ldots, SS, \quad j = 1, \ldots, D \quad (14)$$

$$X(i,j) = \begin{cases} 1 & \text{if } (rand < s) \\ 0 & \text{otherwise} \end{cases}$$

4.4 Enhanced PSO algorithm

In this section, we present in details the main steps of the binary enhanced PSO algorithm as shown in Algorithm 2.

```
1:  Set the initial value of swarm size $SS(N)$, acceleration
    constants $c_1$ and $c_2$, $w_{max}$, $w_{min}$, $v_{max}$ and $max_{iter}$.
2:  Randomly initialize the population as $x$ using Eq. (9)
    for each solution and the velocity vectors $v$ as D
    dimensional zero vectors as in Eq. (10).
3:  Set $t := 0$. {Counter initialization}
4:  $w = w_{max} - t \cdot \left(\frac{w_{max} - w_{min}}{max\_iter}\right)$
5:  Evaluate the fitness function for each of the solutions using
    $f(x_i)$ (Eq. (2)) and assign the values for $Pbest$ and $gbest$.
6:  for $(i = 1; i < SS; i++)$ do
7:      $v_i^{(t+1)} = w v_i^{(t)} + c_1 r_{i1} \cdot (Pbest_i(t) - x_i^{(t)}) + c_2 r_{i2} \cdot (gbest - x_i^{(t)})$
8:  end for {Update the velocities of Particles}
9:  for $(i = 1; i < SS, i++)$ do
10:     for $(j = 1; j < D; j++)$ do
11:         if $(v(i,j) > v_{max})$ then
12:             $v(i,j) = v_{max}$
13:         end if
14:         if $(v(i,j) < -v_{max})$ then
15:             $v(i,j) = -v_{max}$
16:         end if
17:         $s = \frac{1}{1+e^{-v(i,j)}}$
18:         if $(rand < s)$ then
19:             $x(i,j) = 1$
20:         else
21:             $x(i,j) = 0$
22:         end if
23:     end for
24: end for
25: $t = t + 1$ {Iteration counter is increasing}
26: until$(t < max_{iter})$ {Termination criteria are satisfied}.
27: Produce the best solution $gbest$.
```

Algorithm 2. Enhanced PSO Algorithm

After initializing the solutions and the velocity values, the personal and the global best of the particles are noted throughout the algorithm. These values play an important role in directing the velocity values of the particles and need to be revisited in each iteration. The velocity values are updated according Eq. (12) at each stage and these new values are used to change the positions of the particles. It should be noted that, to make sure that the values don't cross the given thresholds its necessary to restrict the velocity values to their assigned maximum and minimum values. These new velocity values are then crushed into the logistic space by applying the binary map as mentioned in Section 4.3 and a threshold is used to update the positions.

This particular algorithm makes the best use of both the personal and the global solutions to arrive at globally optimum solutions. The inertia weight that's updated in every iteration also helps to control the convergence of the algorithm as it progresses. The inertia towards the previous direction is pretty high initially when the algorithm starts but it starts to explore new directions during its progress. This value can be tuned to arrive at better solutions and needs to be tried with grid search over the possible parameter values.

5. Hybrid Binary Bat Enhanced Particle Swarm Optimization (HBBEPSO) algorithm

The main steps of the proposed HBBEPSO algorithm for feature selection are shown in Algorithm 3 and summarized in this section.

The combination of the concepts in the binary bat and the binary PSO algorithms that are described in the previous sections are combined in this section to arrive at an algorithm that can benefit from their amalgamation. In the HBBEPSO algorithm, the decoupling of the velocity vectors of the bats and the particles leads us to a novel formulation. The velocity vectors are updated independently for both the particles according to the weighted combination of the personal and global best solutions and the velocity of the bats is arrived in an instantaneous manner. This is done in order to allow both the algorithms to explore the search space in an alternating fashion and not direct one algorithm with the results obtained from the other algorithm. This form of decoupling is also why the personal and global solutions are not updated after the binary bat algorithm but only after the particle swarm iteration update i.e. once per the whole iteration.

This leads us to some interesting insights that decoupling these variables that accomplish the same goal but in different ways is in fact beneficial for the hybrid algorithm because it benefits from the diversity of the solutions in each iteration which is also the main philosophy behind hybridizing algorithms. It has to be noted that choosing the hyperparameters is important for getting good solutions and can be achieved by a simple grid search or a random search over the hyperparameter space.

6. Feature selection

The feature selection problem is as defined in Section 2. For a feature vector of size N the number of different feature combinations would be 2^N, which is a huge space to search exhaustively. So the proposed hybrid metaheuristic algorithm is used to adaptively search the feature space and produce the best feature combination. The fitness function used is the one given in Eq. (2).

6.1 Classifier

K-nearest neighbor (KNN) [34] is a common simple method used for classification. KNN is a supervised learning algorithm that classifies an unknown sample instance based on the majority vote of its K-nearest neighbors. Here, a wrapper approach to feature selection is used

```
1:  Split the given data set into three equal sizes of training, validation and testing sets.
2:  Set the initial value of swarm size $SS(N)$ and make the dimension $D$ equal to the number of features in the data set.
3:  Set Acceleration constants $c_1$ and $c_2$, $v_{max}$, $w_{max}$, $w_{min}$, $a$, $r$, $Q_{min}$, $Q_{max}$ and $max_{iter}$.
4:  Randomly initialize the population as $x$ using Eq. (9) for each solution, the velocity vectors $v$ as D dimensional zero vectors as in Eq. (10) and the frequency vectors Q as D dimensional zero vectors as in Eq. (3).
5:  Calculate fitness using Eq. (2) and make $fitold = fitness$.
6:  Set $t := 0$. {Counter initialization}
7:  Evaluate the fitness function for each of the solutions using the Eq. (2) and Assign the values for Pbest(using fitold information) and gbest. {The fitness function for feature selection}
8:  Run BBA algorithm as given in Algorithm 1 with initial velocity zero every time. {The bat velocity is instantaneous and is not a resultant from adding with EPSO velocity}.
9:  Update the positions but not the Pbest and gbest.
10: $w = w_{max} - t \cdot \left(\frac{w_{max} - w_{min}}{max\_iter}\right)$
11: for $(i = 1; i < SS; i++)$ do
12:     $v_i^{(t+1)} = wv_i^{(t)} + c_1 r_{i1} \cdot (Pbest_i(t) - x_i^{(t)}) + c_2 r_{i2} \cdot (gbest - x_i^{(t)})$
13: end for {Update the velocities of Particles}
14: for $(i = 1; i < SS, i++)$ do
15:     for $(j = 1; j < D; j++)$ do
16:         if $(v(i,j) > v_{max})$ then
17:             $v(i,j) = v_{max}$
18:         end if
19:         if $(v(i,j) < -v_{max})$ then
20:             $v(i,j) = -v_{max}$
21:         end if
22:         $s = \frac{1}{1+e^{-v(i,j)}}$
23:         if $(rand < s)$ then
24:             $x(i,j) = 1$
25:         else
26:             $x(i,j) = 0$
27:         end if
28:     end for
29: end for
30: $t = t + 1$ {Iteration counter is increasing}
31: until$(t < max_{iter})$ {Termination criteria are satisfied}
32: Produce the best solution gbest.
```

Algorithm 3. HBBEPSO Algorithm

which uses KNN classifier as a guide for the same. Classifiers do not use any model for K-nearest neighbors and are determined solely based on the minimum distance from the current query instance to the neighboring training samples. In this proposed system,the KNN is used as a classifier to ensure robustness to noisy training data and obtain best feature combinations. A single dimension in the search space represents individual feature and hence the position of a particle represents a single feature combination or solution.

7. Experimental results

The proposed binary HBBEPSO algorithm is tested against 20 data sets in Table 1 taken from the UCI machine learning repository [30] and is compared with other algorithms like binary versions of dragonfly, enhanced PSO, GA, bat and greywolf. The algorithm is also compared with HBEPSOB, where the order of implementation of the two algorithms is reversed. The datasets are chosen to have variety in number of instances and features to test for varied data. The datasets are divided into three equal sets: training, validation and testing. The training and validation sets are used for a two fold cross validation on the data. We note that other ways of implementing validation do exist and can be used like stratified K-fold cross validation, group K fold, shuffle split and many others. We take accuracy of the classifier as our main metric and rely on the 2-fold cross validation for weak statistical validation. It should be noted that metrics like uncertainty coefficient which is more robust to the relative sizes of the classes can also be used as a metric along with additional supplements like precision, recall and receiver operating characteristics to analyze the true and false positives of each of the classes.

The value of K (the number of nearest neighbors) is selected as 5 based on the 2-fold cross-validation results of the model. The training set is used to evaluate the KNN on the validation set through this algorithm to guide the feature selection process. The test data is only used for the final evaluation of the best selected feature combination. The global and optimizer-specific parameter setting is given in Table 2. The parameters are set according random search implemented over the hyperparameter space. It should be noted that better values for the hyperparameters are possible and can be obtained by using exhaustive grid search over sufficiently large parameter space assuming computational power is not an issue. The evaluation criteria is explained in Section 7.1.

7.1 Evaluation criteria

The datasets are divided into 3 sets of training, validation and testing. The algorithm is run repeatedly for $M = 10$ times for statistical significance of the results. The following measures [35] are recorded from the validation data:

Table 1.
Datasets.

Dataset	# of attributes	# of instances
Zoo	16	101
WineEW	13	178
IonosphereEW	34	351
WaveformEW	40	5000
BreastEW	30	569
Breastcancer	9	699
Congress	16	435
Exactly	13	1000
Exactly2	13	1000
HeartEW	13	270
KrvskpEW	36	3196
M-of-n	13	1000
SonarEW	60	208
SpectEW	60	208
Tic-tac-toe	9	958
Lymphography	18	148
Dermatology	34	366
Echocardiogram	12	132
hepatitis	19	155
LungCancer	56	32

Parameter	Value
# of iterations (max_{iter})	70
# of search agents (n)	5
Dimension (D)	# of features in the data
Search domain	[0 1]
# of runs (M)	10
a	0.4
r	0.1
Q_{min}	0
Q_{max}	2
w_{max}	0.9
w_{min}	0.4
c_1	2
c_2	2
v_{max}	6
β in fitness function	0.01
α in fitness function	0.99

Table 2.
Parameter setting.

1. *Mean fitness function* is the average of the fitness function value obtained from running the algorithm M times. The mean fitness function is calculated as shown in Eq. (15).

$$Mean = \frac{1}{M}\sum_{i=1}^{M} g_i^* \tag{15}$$

where g_i^* is the best fitness value obtained at run i.

2. *Best fitness function* is the minimum of the fitness function value obtained from running the algorithm M times. The best fitness function is calculated as shown in Eq. (16).

$$Best = \min_{i=1}^{M} \; g_i^* \tag{16}$$

where g_i^* is the best fitness value obtained at run i.

3. *Worst fitness function* is the maximum of the fitness function value obtained from running the algorithm M times. The worst fitness function is calculated as shown in Eq. (17).

$$Worst = \max_{i=1}^{M} \; g_i^* \tag{17}$$

where g_i^* is the best fitness value obtained at run i.

4. *Standard deviation* gives the variation of the fitness function value obtained from running the algorithm M times. It is an indicator of the stability and robustness of the algorithm. Larger values of standard deviation would suggest wandering results where as smaller value suggests the algorithm converges to the same value most of the times. The Standard deviation is calculated as shown in Eq. (18).

$$Std = \sqrt{\frac{1}{M-1}\sum\nolimits_{i=1}^{M}\left(g_i^* - Mean\right)^2} \tag{18}$$

where g_i^* is the best fitness value obtained at run i.

5. *Average Performance* (*CA*) is the mean of the classification accuracy values when an algorithm is run *M* times. The average performance is calculated as shown in Eq. (19).

$$CA = \frac{1}{M}\sum_{i=1}^{M} CA^{i} \tag{19}$$

where CA^{i} is the accuracy value obtained at run *i*

6. *Mean Feature selection ratio* (*FSR*) is the mean of the ratio of the number of selected features to the total number of features when an algorithm is run *M* times. The Mean Feature selection ratio is calculated as shown in Eq. (20).

$$FSR = \frac{1}{M}\sum_{i=1}^{M} \frac{size(g_i^{*})}{D} \tag{20}$$

where g_i^{*} is the best fitness value obtained at run *i*, size(g_i^{*}) gives the number of features selected and *D* is the total number of features.

7. *Average F-score* is a measure that evaluates the performance of a chosen feature subset. It requires that in the data spanned by the feature combination the distance between data points in different classes be large and of those in the same class be as small as possible. The Fischer index for a given feature is calculated as in Eq. (21) [36].

$$F_j = \frac{\sum_{k=1}^{C} n_k\left(\mu_k^j - \mu^j\right)^2}{\left(\sigma_j\right)^2} \tag{21}$$

$$\left(\sigma_j\right)^2 = \sum_{k=1}^{C} n_k\left(\sigma_k^j\right)^2 \tag{22}$$

where F_j is the Fischer index for *j*, μ^j is the mean of the entire data for feature *j*, $\left(\sigma^j\right)_2^2$ is defined as in Eq. (22), n_k is the size of class *k*, μ_k^j is the mean of class *k* for feature *j*, $\left(\sigma_k^j\right)^2$ is the variance of class *k* for feature *j*. The *average F-score* is calculated by taking the average of values obtained from *M* runs for only the selected features.

7.2 Results

The proposed binary version of the HBBEPSO algorithm is compared with the binary bat algorithm, the Enhanced PSO and other optimizers. The results are tabulated as follows.

Table 3 outlines the performance of the algorithms using the fitness function mentioned in Eq. (2) in the minimization mode. The table shows the average fitness obtained over *M* runs and is calculated using Eq. (15). The best performance is achieved by the proposed binary version of the HBBEPSO algorithm proving its ability to search the feature space effectively.

For testing the stability, robustness and the repeatability of convergence of these stochastic algorithms the standard deviation of the fitness values over *M* runs is recorded as per Eq. (18) in Table 4. The table shows that the HBBEPSO algorithm has the ability to converge repeatedly irrespective of the random initialization.

The Best selected feature combinations by the algorithms are also allowed to run on the test data and the average classification accuracy and the average feature selection ratio over *M* runs is recorded using Eqs. (19) and (20), respectively as shown in Tables 5 and 6. As seen from these tables, the HBBEPSO algorithm is able to select the minimum number of features and yet maintain the classification accuracy. This shows the capability of the HBBEPSO algorithm to satisfy both the objectives of optimization.

Dataset	HBBESPO	BBA	EPSO	BGWO2	BDA	BGA	HBEPSOB
Zoo	**0.014**	0.094	0.031	0.11	0.067	0.124	0.053
Wine EW	**0.027**	0.128	0.042	0.092	0.050	0.065	0.038
IonosphereEW	**0.089**	0.146	0.137	0.172	0.130	0.143	0.117
WaveformEW	0.179	0.193	**0.175**	0.185	0.183	0.186	0.176
BreastEW	**0.050**	0.070	0.050	0.080	0.057	0.106	0.052
Breastcancer	**0.031**	0.035	0.032	0.042	0.032	0.036	0.039
Congress	**0.031**	0.053	0.033	0.073	0.042	0.059	0.035
Exactly	0.155	0.303	0.104	0.316	0.178	0.269	**0.064**
Exactly2	**0.224**	0.243	0.234	0.263	0.240	0.243	0.230
HeartEW	**0.140**	0.240	0.153	0.268	0.153	0.250	0.157
KrvskpEW	**0.041**	0.108	0.043	0.080	0.041	0.089	0.043
M-of-n	**0.024**	0.167	0.024	0.154	0.048	0.108	0.047
SonarEW	0.179	0.277	0.192	0.290	0.194	0.262	**0.166**
SpectEW	0.132	0.167	0.160	0.205	0.133	0.168	**0.117**
Tic-tac-toe	**0.214**	0.270	0.222	0.262	0.223	0.241	0.231
Lymphography	**0.343**	0.487	0.531	0.487	0.412	0.466	0.438
Dermatology	**0.015**	0.081	0.016	0.099	0.017	0.031	0.025
Echocardiogram	**0.047**	0.112	0.083	0.200	0.058	0.072	0.080
Hepatitis	0.137	0.175	0.123	0.192	**0.101**	0.152	0.122
LungCancer	**0.129**	0.427	0.220	0.455	0.255	0.318	0.165
Average	**0.110**	0.189	0.123	0.204	0.131	0.169	0.120

Table 3. Mean fitness function obtained from the different algorithms.

Dataset	HBBESPO	BBA	EPSO	BGWO2	BDA	BGA	HBEPSOB
Zoo	**0.020**	0.070	0.033	0.067	0.075	0.066	0.041
Wine EW	**0.014**	0.080	0.018	0.057	0.030	0.026	0.016
IonosphereEW	**0.016**	0.040	0.016	0.057	0.018	0.025	0.019
WaveformEW	**0.003**	0.012	0.008	0.007	0.006	0.008	0.009
BreastEW	0.011	0.017	0.019	0.018	**0.007**	0.755	0.011
Breastcancer	**0.006**	0.009	0.007	0.008	0.005	0.007	0.007
Congress	**0.006**	0.019	0.008	0.015	0.016	0.013	0.009
Exactly	0.064	0.040	0.082	**0.016**	0.119	0.078	0.043
Exactly2	**0.009**	0.017	0.009	0.018	0.015	0.019	0.018
HeartEW	**0.033**	0.064	0.055	0.069	0.036	0.062	0.038
KrvskpEW	**0.005**	0.044	0.007	0.012	0.007	0.051	0.006
M-of-n	0.025	0.036	0.022	**0.019**	0.051	0.032	0.034
SonarEW	**0.025**	0.059	0.029	0.043	0.033	0.030	0.030
SpectEW	**0.013**	0.028	0.027	0.024	0.022	0.029	0.027
Tic-tac-toe	**0.012**	0.025	0.020	0.017	0.012	0.021	0.020
Lymphography	**0.038**	0.047	0.048	0.044	0.049	0.062	0.038
Dermatology	**0.007**	0.075	0.008	0.050	0.014	0.014	0.007
Echocardiogram	**0.023**	0.055	0.030	0.228	0.026	0.024	0.036
Hepatitis	0.054	0.043	0.028	0.052	**0.025**	0.038	0.038
LungCancer	0.113	0.194	0.180	**0.093**	0.151	0.233	0.092
Average	**0.025**	0.049	0.033	0.046	0.036	0.080	0.027

Table 4. Standard deviation of the fitness function obtained from the different algorithms.

To analyze the separability and closeness of the selected features Fischer score of these features is calculated as shown in Eq. (21). The average over *M* runs is recorded in Table 7. As shown in the table, HBBEPSO algorithm achieves superior data compactness in comparison with the other algorithms.

Dataset	HBBESPO	BBA	EPSO	BGWO2	BDA	BGA	HBEPSOB
Zoo	**0.873**	0.799	0.791	0.851	0.788	0.863	0.852
Wine EW	0.901	0.726	0.881	0.896	**0.923**	0.886	0.908
IonosphereEW	**0.831**	0.817	0.829	0.824	0.799	0.828	0.819
WaveformEW	**0.829**	0.779	0.809	0.819	0.807	0.806	0.809
BreastEW	**0.945**	0.842	0.931	0.908	0.944	0.892	0.94
Breastcancer	**0.958**	0.957	0.956	0.957	0.956	0.957	0.952
Congress	**0.944**	0.893	0.943	0.928	0.931	0.915	0.927
Exactly	0.835	0.647	0.884	0.680	0.798	0.687	**0.952**
Exactly2	**0.760**	0.711	0.738	0.732	0.739	0.734	0.726
HeartEW	**0.813**	0.648	0.776	0.702	0.81	0.711	0.783
KrvskpEW	**0.959**	0.772	0.958	0.917	0.954	0.906	0.956
M-of-n	**0.980**	0.719	0.975	0.843	0.949	0.892	0.942
SonarEW	**0.704**	0.678	0.682	0.682	0.658	0.694	0.694
SpectEW	0.764	0.755	0.757	**0.777**	0.752	0.750	0.759
Tic-tac-toe	**0.745**	0.647	0.740	0.713	0.745	0.734	0.733
Lymphography	**0.422**	0.422	0.354	0.379	0.417	0.416	0.422
Dermatology	**0.933**	0.802	0.952	0.908	0.940	0.950	0.957
Echocardiogram	0.888	0.861	**0.906**	0.877	0.893	0.852	0.859
Hepatitis	**0.825**	0.788	0.813	0.788	0.788	0.798	0.792
LungCancer	0.454	0.343	0.390	0.345	0.427	0.409	**0.484**
Average	**0.818**	0.730	0.803	0.776	0.801	0.784	0.813

Table 5.
Average performance of the selected features by different algorithms.

Dataset	HBBESPO	BBA	EPSO	BGWO2	BDA	BGA	HBEPSOB
Zoo	**0.318**	0.512	0.356	0.473	0.331	0.412	0.356
Wine EW	0.392	0.538	0.4	0.516	0.338	**0.315**	0.292
IonosphereEW	**0.376**	0.526	0.388	0.541	0.397	0.402	0.426
WaveformEW	**0.661**	0.664	0.709	1	0.666	0.676	0.742
BreastEW	0.325	0.480	**0.241**	0.470	0.283	0.290	0.270
Breastcancer	0.477	0.511	0.511	0.644	**0.422**	0.566	0.500
Congress	**0.318**	0.493	0.325	0.575	0.337	0.412	0.343
Exactly	**0.500**	0.538	0.507	0.576	0.507	0.561	0.561
Exactly2	**0.392**	0.546	0.492	0.800	0.392	0.400	0.407
HeartEW	**0.384**	0.492	0.407	0.430	0.407	0.415	0.391
KrvskpEW	**0.467**	0.513	0.502	0.633	0.475	0.530	0.541
M-of-n	0.476	**0.446**	0.476	0.923	0.530	0.576	0.546
SonarEW	**0.413**	0.521	0.463	0.533	0.413	0.42	0.45
SpectEW	0.424	**0.481**	0.463	0.529	0.454	0.425	0.415
Tic-tac-toe	**0.511**	0.577	0.666	0.866	0.555	0.511	0.622
Lymphography	**0.355**	0.461	0.416	0.535	0.438	0.4	0.388
Dermatology	0.470	0.494	0.511	0.544	**0.411**	0.479	0.485
Echocardiogram	**0.216**	0.508	0.266	0.483	0.233	0.283	0.218
Hepatitis	**0.231**	0.515	0.321	0.431	0.273	0.231	0.273
LungCancer	**0.348**	0.498	0.423	0.526	0.353	0.380	0.362
Average	**0.403**	0.516	0.442	0.601	0.411	0.434	0.429

Table 6.
Average selected feature ratio by different algorithms.

It should be noted that datasets with comparable number of features and instances outperform the other datasets. Mainly we notice that datasets, if the number of features is f and the number of instances is n, then, datasets with $f < n$ and $n = 10f$ outperform the other

Table 7. Average Fischer index of the selected features by different algorithms.

Dataset	HBBESPO	BBA	EPSO	BGWO2	BDA	BGA	HBEPSOB
Zoo	**165**	112	143	130	105	156	138
Wine EW	1043.6	11,784	648.48	**20,939**	374.67	540.52	90.078
IonosphereEW	**5.783**	4.154	3.870	5.042	3.986	4.769	5.011
WaveformEW	**3.591**	2.029	2.066	3.456	2.314	2.165	2.515
BreastEW	**7.8E+13**	5.7E+12	3.3E+13	6.97E+12	2.5E+11	1.4E+13	6.2E+11
Breastcancer	**1.16**	0.942	1.070	1.105	0.748	0.923	0.740
Congress	**23.050**	11.003	13.996	18.317	31.797	13.045	16.870
Exactly	0.212	0.350	0.131	0.282	0.259	**0.378**	0.106
Exactly2	**0.399**	0.200	0.267	0.237	0.240	0.287	0.354
HeartEW	4.742	161.622	3.357	**430.07**	3.424	140.64	4.721
KrvskpEW	496.94	639.91	940.24	**1187.5**	544.21	1023.2	569.11
M-of-n	**1.828**	1.652	1.735	1.373	1.711	1.786	1.586
SonarEW	5E+6	8.2E+6	8.2E+6	**1.2E+7**	7.3E+6	5.5E+6	5.6E+6
SpectEW	**0.008**	0.006	0.005	0.006	0.006	0.004	0.004
Tic-tac-toe	**0.161**	0.136	0.161	0.117	0.090	0.119	0.120
Lymphography	**9.53**	4.41	9.18	2.73	3.13	2.43	7.51
Dermatology	**384**	210	343	174	269	148	173
Echocardiogram	21,452	**130,939**	1376	53,037	579.06	62,931	152.24
Hepatitis	48.616	132.031	51.80	**53,037**	3.491	14.420	3.078
LungCancer	**47.559**	30.810	40.203	33.615	31.148	29.405	31.574
Average	**3.9E+12**	2.8E+11	1.6E+12	3.4E+12	1.3E+10	6.9E+11	3.1E+10

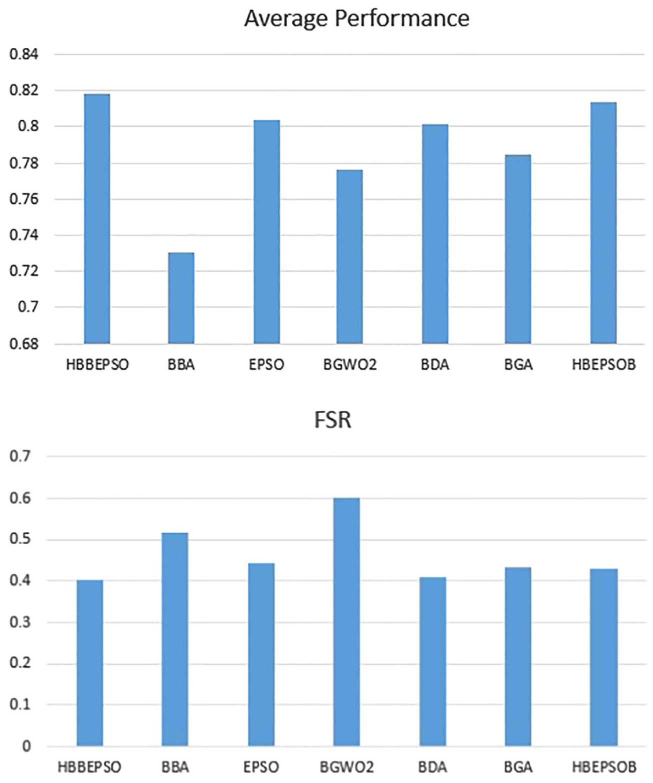

Figure 1.
The Comparison of performance the HBBEPSO algorithm with other optimizers through main objectives of feature selection. The values are averaged over all the datasets.

datasets. This is expected as datasets with too few training samples compared to the number of features will substantially underfit. The tables given above show that the HBBEPSO algorithm outperforms the other algorithms with respect to all of the assessment indicators. It can also be seen that it performs much better when compared to its switched version HBEPSOB algorithm. This leads us to believe that the bat algorithm is powerful in exploring the search space and the enhanced PSO algorithm aids in exploiting the reduced feature space (see Figures 1 and 2).

8. Conclusion and future work

In this paper, a new hybrid binary metaheuristic algorithm with bat algorithm and enhanced PSO algorithm is proposed in order to solve feature selection problems. The proposed algorithm is called hybrid Binary Bat Enhanced Particle Swarm Optimization (HBBEPSO) algorithm. The two algorithms come together to give better solutions than each of them individually. In order to verify the robustness and the effectiveness of the proposed algorithm, we apply it on 20 feature selection problems. The evaluation is performed using a set of evaluation criteria to assess different aspects of the proposed system. The experimental results show that the proposed algorithm is a promising algorithm with its ability to search the feature space effectively. The given algorithm is also run on test data and observations show higher performance of the selected features when compared to the other optimizers. The Fischer index table reveals better separability. It is also noted from the values of standard

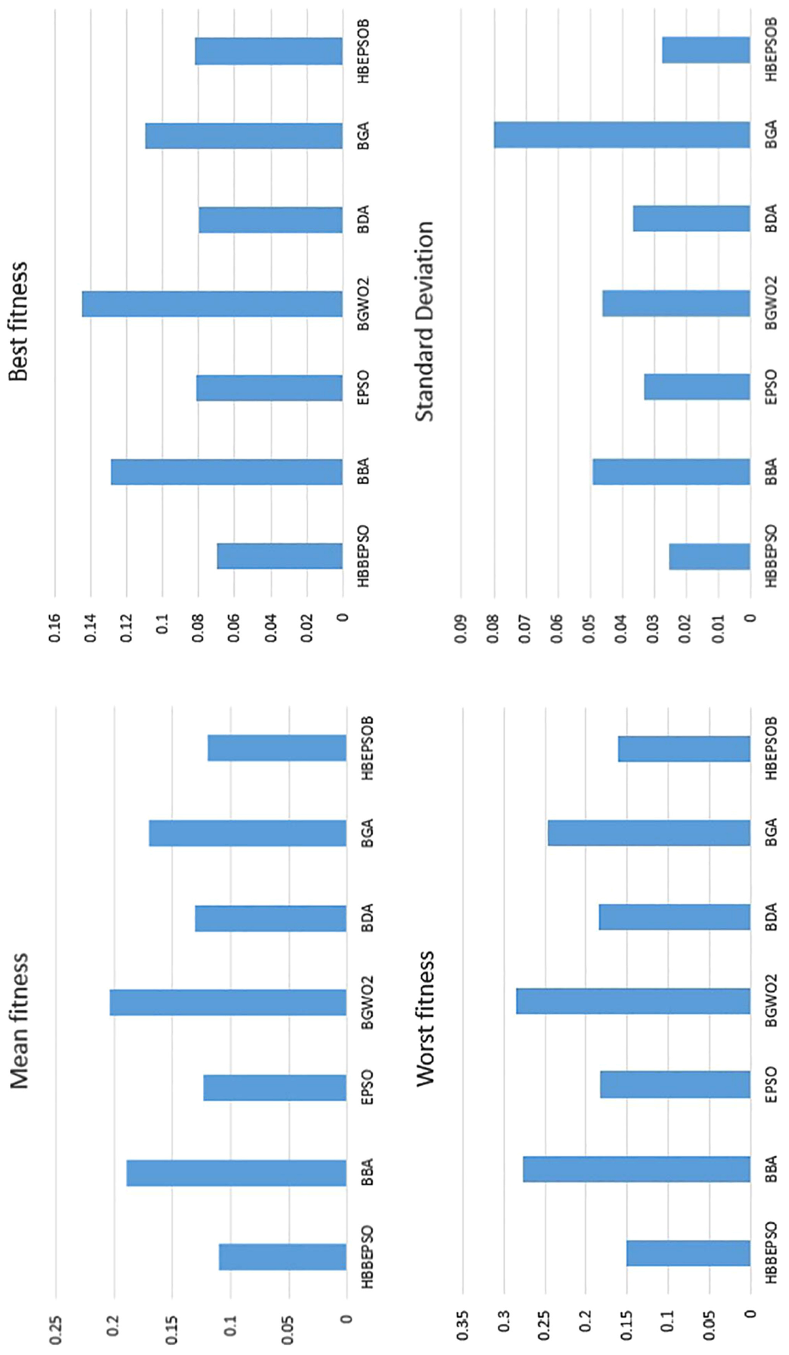

Figure 2. The Comparison of performance the HBBEPSO algorithm with other optimizers through few assessment indicators. The values are averaged over all the datasets.

deviation that the algorithm has the robustness to repeatedly converge to similar solutions therefore a powerful ability to solve feature selection problems better than other algorithms in most cases. This research motivates us for further investigations as future work as follows. The authors in [37] suggested a binary bat algorithm to solve unconstrained optimization bench test problems and compared with binary GA and binary PSO. We would like to apply our proposed algorithm on solving unconstrained optimization problems [39,40], large scale problems and molecular potential energy function [41], and engineering optimization problems [38]. Also, further comparison studies for various binary variants of bat algorithm in the literature need to be done on feature selection problem as suggested by one of the referees.

References

[1] B. Chizi, L. Rokach, O. Maimon, A Survey of Feature Selection Techniques, Encyclopedia of Data Warehousing and Mining, second ed., IGI Global, 2009, pp. 1888–1895.

[2] G. Chandrashekar, F. Sahin, A Survey on Feature Selection Methods, Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY 14623, USA, 2013.

[3] D. Bell, H. Wang, A formalism for relevance and its application in feature subset selection, Mach. Learn. 41 (2) (2000) 175–195.

[4] Samina Khalid, A survey of feature selection and feature extraction techniques in machine learning, in: Science and Information Conference (SAI), 2014.

[5] Bing Xue, Mengjie Zhang, Will Browne, Xin Yao, A survey on evolutionary computation approaches to feature selection, IEEE Trans. Evolution. Comput. (2015).

[6] Y. Zhang, W. Shuihua, P. Preetha, J. Genlin, Binary PSO with mutation operator for feature selection using decision tree applied to spam detection, Knowl.-Based Syst. 64 (2014) 22–31.

[7] X. Zhou, Z. Yudong, J. Genlin, Y. Jiquan, D. Zhengchao, W. Shuihua, Z. Guangshuai, P. Preetha, Detection of abnormal MR brains based on wavelet entropy and feature selection, IEEJ Trans. Electr. Electron. Eng. 11 (3) (2016) 364–373.

[8] H. Banati, M. Bajaj, Fire fly based feature selection approach, IJCSI Int. J. Comp. Sci. Iss. 8 (4) (2011) 2.

[9] R.Y.M. Nakamura, L.A.M. Pereira, K.A. Costa, D. Rodrigues, J.P. Papa, X.-S. Yang, Binary bat algorithm for feature selection, in: Conference on Graphics, Patterns and Images, 2012, pp. 291–297.

[10] Jinjie Huang, Yunze Cai, Xiaoming Xu, A hybrid genetic algorithm for feature selection wrapper based on mutual information, Patt. Recog. Lett. Arch. 28(13) (2007).

[11] Hossam M. Zawbaa, Aboul Ella Hassanien, E. Emary, Waleed Yamany, B. Parv, Hybrid flower pollination algorithm with rough sets for feature selection, in: 11th International Computer Engineering Conference (ICENCO), IEEE, Cairo, Egypt, December, 2015, pp. 278–283.

[12] X.S. Yang, Bat algorithm for multi-objective optimisation, Int. J. Bio-Insp. Comput. 3 (5) (2011) 267–274.

[13] G. Komarasamy, A. Wahi, An optimized K-means clustering technique using bat algorithm, Euro. J. Scient. Res. 84 (2) (2012) 263–273.

[14] K. Khan, A. Nikov, A. Sahai, A fuzzy bat clustering method for ergonomic screening of office workplaces, in: S3T 2011, Adv. Intell. Soft Comput., vol. 101, 2011, pp. 59–66.

[15] X.S. Yang, Bat algorithm for multi-objective optimisation, Int. J. BioInsp. Comput. 3 (5) (2011) 267–274.

[16] J. Xie, Y.Q. Zhou, H. Chen, A novel bat algorithm based on differential operator and Levy flights trajectory, Comput. Intell. Neurosci. 2013, 453812 <http://www.hindawi.com/journals/cin/aip/453812.pdf>.

[17] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan, 1995, pp. 39–43.

[18] Y. Zhang, W. Shuihua, J. Genlin, A comprehensive survey on particle swarm optimization algorithm and its applications, Math. Prob. Eng. (2015) 2015.

[19] G. Coath, S.K. Halgamuge, A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems, in: Proceedings of IEEE Congress on Evolutionary Computation 2003 (CEC 2003), Canbella, Australia, 2003, pp. 2419–2425.

[20] C.A. Coello Coello, E.H.N. Luna, A.H.N. Aguirre, Use of particle swarm optimization to design combinational logic circuits, in: Lecture Notes in Computer Science (LNCS), vol. 2606, 2003, pp. 398–409.

[21] R.A. Krohling, H. Knidel, Y. Shi, Solving numerical equations of hydraulic problems using particle swarm optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2002), Honolulu, Hawaii, USA, 2002.

[22] D.K. Agrafiotis, W. Cedeno, Feature selection for structure-activity correlation using binary particle swarms, J. Med. Chem. 45 (5) (2002) 1098–1107.

[23] B. Xue, Z. Mengjie, N.B. Will, Particle swarm optimization for feature selection in classification: a multi-objective approach, IEEE Trans. Cybernet. 43 6 (2013) 1656–1671.

[24] Y. Zhang, G. Dun-Wei, C. Jian, Multi-objective particle swarm optimization approach for cost-based feature selection in classification, IEEE/ACM Trans. Comput. Biol. Bioinf. (TCBB) 14(1) (2017) 64–75.

[25] Y. Zhang, G. Dunwei, H. Ying, Z. Wanqiu, Feature selection algorithm based on bare bones particle swarm optimization, Neurocomputing 148 (2015) 150–157.

[26] S. Mirjalili, A. Lewis, S-shaped versus V-shaped transfer functions for binary particle swarm optimization, Swarm Evolution. Comput. 9 (2012) 1–14.

[27] D. Wolpert, W. Macready, No free lunch theorems for optimization, IEEE Trans. Evolution. Comput. 1 (1) (1997) 67–82.

[28] C. Cotta, A study of hybridisation techniques and their application to the design of evolutionary algorithms, AI Commun. 11 (3–4) (1998) 223–224.

[29] E.G. Talbi, A taxonomy of hybrid metaheuristics, J. Heur. 8 (5) (2002) 541–565.

[30] A. Frank, A. Asuncion, UCI Machine Learning Repository, 2010.

[31] D.R. Griffin, F.A. Webster, C.R. Michael, The echolocation of flying insects by bats, Animal Behav. 8(34) (1960) 141–154.

[32] H.-U. Schnitzler, E.K.V. Kalko, Echolocation by insect-eating bats, BioScience 51 (7) (2001) 557–569.

[33] J. Kennedy, R.C. Eberhart, Y. Shi, Swarm Intelligence, Morgan Kaufmann, SanMateo, CA, 2001.

[34] L.Y. Chuang, H.W. Chang, C.J. Tu, C.H. Yang, Improved binary PSO for feature selection using gene expression data, Comput. Biol. Chem. 32 (2008) 29–38.

[35] E. Emary, Hossam M. Zawbaa, C. Grosan, A.E. Hassanien, Binary grey wolf optimization approaches for feature selection, Neurocomputing, vol. 172, Elsevier, 2016, pp. 371–381, January.

[36] Quanquan Gu, L. Zhenhui, H. Jiawei, Generalized fisher score for feature selection, in: Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI), Barcelona, Spain, 2011.

[37] S. Mirjalili, S.M. Mirjalili, X.S. Yang, Binary bat algorithm, Neural Comput. Appl. 25 (3–4) (2014) 663–681.

[38] A.F. Ali, M.A. Tawhid, Hybrid simulated annealing and pattern search method for solving minimax and integer programming problems, Pacific J. Optim. 12 (1) (2016) 151–184.

[39] A.F. Ali, M.A. Tawhid, Hybrid particle swarm optimization with a modified arithmetical crossover for solving unconstrained optimization problems, INFOR: Inf. Syst. Operat. Res. 53 (3) (2015) 125–141.

[40] M.A. Tawhid, A.F. Ali, Multi-directional bat algorithm for solving unconstrained optimization problems, OPSEARCH 54 (4) (2017) 684–705.

[41] M.A. Tawhid, A.F. Ali, A hybrid social spider optimization and genetic algorithm for minimizing molecular potential energy function, Soft Comput. 21 (21) (2017) 6499–6514.

Corresponding author
Mohamed A. Tawhid can be contacted at: mtawhid@tru.ca