Natural Computing (2021) 20:39–61
https://doi.org/10.1007/s11047-019-09769-z

A novel hybrid BPSO–SCA approach for feature selection

Lalit Kumar[1] · Kusum Kumari Bharti[1]

Published online: 23 October 2019

Abstract
Nature is a great source of inspiration for solving complex problems in real-world. In this paper, a hybrid nature-inspired algorithm is proposed for feature selection problem. Traditionally, the real-world datasets contain all kinds of features informative as well as non-informative. These features not only increase computational complexity of the underlying algorithm but also deteriorate its performance. Hence, there an urgent need of feature selection method that select an informative subset of features from high dimensional without compromising the performance of the underlying algorithm. In this paper, we select an informative subset of features and perform cluster analysis by employing a cross breed approach of binary particle swarm optimization (BPSO) and sine cosine algorithm (SCA) named as hybrid binary particle swarm optimization and sine cosine algorithm (HBPSOSCA). Here, we employ a V-shaped transfer function to compute the likelihood of changing position for all particles. First, the effectiveness of the proposed method is tested on ten benchmark test functions. Second, the HBPSOSCA is used for data clustering problem on seven real-life datasets taken from the UCI machine learning store and gene expression model selector. The performance of proposed method is tested in comparison to original BPSO, modified BPSO with chaotic inertia weight (C-BPSO), binary moth flame optimization algorithm, binary dragonfly algorithm, binary whale optimization algorithm, SCA, and binary artificial bee colony algorithm. The conducted analysis demonstrates that the proposed method HBPSOSCA attain better performance in comparison to the competitive methods in most of the cases.

Keywords Binary artificial bee colony algorithm · Binary particle swarm optimization · Binary dragonfly algorithm · Binary moth flame optimization · Binary whale optimization algorithm · Clustering indices · Feature selection · Sine cosine algorithm

1 Introduction

The high dimensionality of the feature space is a major concern in today's day. Usually, there are so many irrelevant and redundant features in the datasets. These features not only increase computational complexity but also deteriorate performance of the underlying algorithms. Therefore, feature selection is necessary to improve the clustering performance, especially for data sets having very large dimensions. With respect to different selection strategies, feature selection methods are broadly categorized into two categories: Filter Methods and wrapper Methods.

Filter methods (Michaud 1997; Keogh and Mueen 2011; Yang et al. 2015) are classifier independent features. It uses a statistical measure to assign a relevance score to each feature. The computed score is used to rank the features. Here, the user-defined criterion is used to select the subset of features from given high dimensional feature space. As filter methods do not consider interaction with the learning algorithm, it is comparatively faster than the wrapper method. However, the accuracy is comparatively lower than the wrapper algorithm (Fig. 1).

In contrast to filter methods, wrapper methods assess relevance of features subset based on the learning algorithm performance (Keogh and Mueen 2011; Yang et al. 2015; Srivastava et al. 2014; Dash et al. 2002). The filter

✉ Kusum Kumari Bharti
kusum@iiitdmj.ac.in

Lalit Kumar
1611007@iiitdmj.ac.in

[1] PDPM-Indian Institute of Information Technology, Design and Manufacturing, Jabalpur, Dumna Airport Road, P.O. Khamaria, Jabalpur, M.P., India

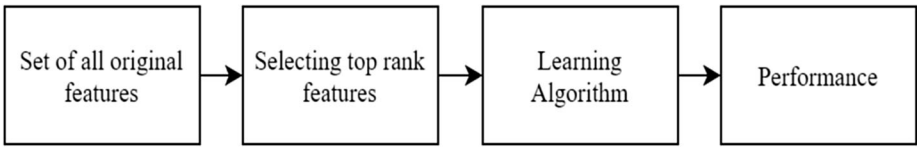

Fig. 1 Process of filter method

method is comparatively faster than the wrapper methods; their major drawback is that features are ranked independently using some statistical measure. It does not consider interaction of features during the feature selection process. On the other hand, the wrapper method considers interaction between features and then select the informative subset of features from the original feature space. In this paper, we propose a wrapper method for feature subset selection (Fig. 2).

The selection of an informative subset of features can be considered as a global combinatorial optimization problem in which the optimum features subset is selected from a high dimensional feature space. Nature inspired algorithms (NIA) gain attention for optimization problem (Agarwal and Mehta 2014). NIA is a mathematical formulation of the living beings present in the environment. Researchers have explored NIA such as genetic algorithm (GA) (Yang and Honavar 1998), particle swarm optimization (PSO) (Xue et al. 2013; Yang 2014), ant colony optimization (ACO) (Yang 2014; Blum 2005; Ali et al. 2017; Ahmed 2005), simulated annealing (SA) (Yang 2014), differential evolution (DE) (Yang 2014; Ali et al. 2017), and bacterial foraging optimization (BFO) (Chen et al. 2017) for feature selection problem. As they consider interaction of the learning algorithm for feature selection, they come under the wrapper method.

Originally, PSO is proposed for continuous problem (Kennedy 1995). Later, it is extended to solve discrete problem (Kennedy and Eberhart 1997). The discrete version of the PSO is named as binary particle swarm optimization. The BPSO is used to solve a wide variety of problems including feature selection (Cervante et al. 2012), cryptography algorithms (Jadon et al. 2011), optimum switching law of inverter (Wu et al. 2010), and classification (Cervantes et al. 2005). Several algorithms have also been developed to improve the performance of BPSO that includes modified BPSO which adopts concepts of the genotype–phenotype representation and the mutation operator of genetic algorithms (Lee et al. 2008), mutation-based binary particle swarm optimization (M-BPSO) for multiple sequence alignment solving (Long et al. 2009), improved binary particle swarm optimization to select the small subset of informative genes (Mohamad et al. 2011), density-based particle swarm optimization algorithm for data clustering (Alswaitti et al. 2018), Particle Swarm Clustering Fitness Evaluation with Computational Centroids (Raitoharju et al. 2017), hybrid binary version of bat and enhanced particle swarm optimization algorithm to solve feature selection problems (Tawhid and Dsouza 2018), hybrid improved BPSO and cuckoo search for review spam detection (Rajamohana and Umamaheswari 2017), and hybrid PSO with grey wolf optimizer (HPSOGWO) (Singh and Singh 2017). In any case, original BPSO (Kennedy 1995) effectively stick into neighborhood optima because of single directional data sharing system by the global best particle in the swarm. Chuang et al. (2008), enhance BPSO by embedding two sorts of chaotic maps, logistic maps and tent maps to evacuate superfluous features and select an informative subset of features. Here, they use a chaotic map to update the value of inertia weight over the number of iterations. This step helps the algorithm to avoid stagnation of the solution at a local optimum solution. Selection of an informative subset of features subset from high dimensional feature space and improvement in the searching capability of the existing NIA is still a great challenge in the area of optimization.

Mirjalili (2016a) was offered sine cosine algorithm (SCA) which is a novel population based optimization technique simply based on Sine and Cosine function. SCA applied for exploitation and exploration phases in global optimization functions. The sine cosine algorithm (SCA) generates different initial random agent solutions using a mathematical model based on sine and cosine functions and requires them to fluctuate outwards or towards the best possible solution.

Several new modified and hybrid variants of SCA algorithm are developed after motivated of this meta-heuristics by the researchers of different areas to improve

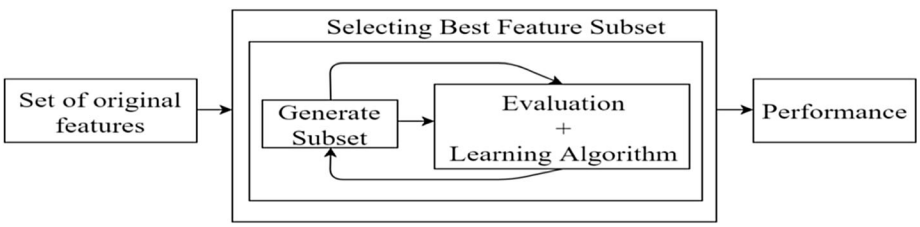

Fig. 2 Process of wrapper method

the convergence performance of SCA algorithm including SCA integrated with differential evolution (Bureerat and Pholdee 2017), Improved SCA based on levy flight (Li N and Deng ZL 2017), Hybrid SCA with multi-orthogonal search strategy (Rizk-Allah 2017), and hybrid back tracking search with sine cosine algorithm (SCA) (Turgut 2017). The researchers are solved numerous real life problems with the help of SCA algorithm including a novel sine cosine algorithm for the feature selection (Hafez et al. 2016), solution of unit commitment problems (Kaur and Prashar 2016), the gear train design problem (Rizk-Allah 2017), Welded beam design (Rizk-Allah 2017), Pressure vessel design problem (Rizk-Allah 2017), Structural Damage Detection (Bureerat and Pholdee 2017), and many other biomedical and mechanical engineering problems.

In this study, we explore capability of NIA for feature selection problem by introducing a hybrid NIA with the combination of BPSO and SCA named as HBPSOSCA. The proposed model HBPSOSCA integrates the exploration capability of the SCA and exploitation capability of the PSO to select an informative subset of features. Here, the V-shaped transfer function is used to convert continuous swarm intelligence technique to binary search space. Next, the K-means algorithm is used to create clusters of data points. The Silhouette Index (SI), Dunn Index (DI) and Davies–Bouldin Index (DBI) are used for cluster assessment. Seven real-life scientific datasets are taken from the UCI machine learning archive and gene expression model selector (GEMS) to test effectiveness of the proposed method compared to other competitive methods. The comparative analysis is performed in terms of (1) number of selected feature subsets and, (2) clustering accuracy measured in terms of SI, DI, and DBI. The comparative analysis of the HBPSOSCA method is compared with state of the arts algorithm BPSO, modified BPSO with chaotic inertia weight (C-BPSO), binary moth flame optimization algorithm (BMFOA), binary dragonfly algorithm (BDA), binary whale optimization algorithm (BWOA), sine cosine algorithm (SCA), and binary artificial bee colony algorithm (Binary-ABC).

The rest of the paper is organized as follows: Sect. 2 describes the foundation of algorithms used as a part of this paper. The detailed descriptions of proposed approach are given in Sect. 3. The experimental results are presented in Sect. 4 while the conclusions and future are presented in Sect. 5.

2 Algorithms background

This section provides a background concerning the optimization algorithms.

2.1 Binary particle swarm optimization (BPSO)

J. Kennedy and R. Eberhart propose Particle swarm optimization (PSO) (Kennedy 1995; Kennedy and Eberhart 1997) in 1995. The PSO is a population-based stochastic approach for solving continuous and discrete optimization problems. The PSO optimization algorithm is inspired by the flocking and schooling patterns of birds and fish. Here, each particle is considered as a potential solution. Every particle is associated with a fitness value, which is used to assess worthiness of the solution compared to others. The particle own best position is named as particle best solution (pbest). The best-fitted solution of the entire swarm is named as a global best solution (gbest). The movement of particle is controlled by its own pbest position and the gbest position of the swarm. Equations 1 and 2 show the velocity and position update of the *ith* particle.

$$vel_i^d(t+1) = vel_i^d(t) + e_1 r_1 * \left(pbest_i^d(t) - pos_i^d(t)\right) + e_2 r_2 * \left(gbest^d(t) - pos_i^d(t)\right) \tag{1}$$

$$pos_i^d(t+1) = pos_i^d(t) + vel_i^d(t+1) \tag{2}$$

where d is the dimension of particle, t is the iteration, r_1 and r_2 are random numbers in the interim (0, 1), and e_1 and e_2 are positive learning constants.

Originally, PSO is developed for continuous problem while numerous optimization problems exist which are discrete/binary in nature, for example, demand side management system (Agneessens et al. 2011), feature selection (Behjat et al. 2014), Knapsack Problems (Bansal and Deep 2012). Kennedy (1995) extend the concept of PSO and develop a binary version of PSO named as binary particle swarm optimization (BPSO) to solve discrete/binary problem. Here, a sigmoidal function $\left(Sig\left(\mathrm{vel}_\mathrm{i}^\mathrm{d}\right)\right)$ is used to convert position (refer Eq. 2) in binary search space (refer Eq. 4) for BPSO. The sigmoidal function is a mathematical function whose curve looks same as the shape of English letter 'S'. The sigmoid function is a kind of function that is real-valued and differentiable. It is defined on all real inputs and returns a positive derivative at every point of it. Position and velocity of ith particle in case of BPSO algorithm are computed by:

$$vel_i^d(t+1) = w * vel_i^d(t) + e_1 r_1 * \left(pbest_i^d(t) - pos_i^d(t)\right) + e_2 r_2 * \left(gbest^d(t) - pos_i^d(t)\right) \tag{3}$$

$$Sig\left(vel_i^d\right) = \frac{1}{1 + e^{-vel_i^d}} \tag{4}$$

$$pos_i^d = \begin{cases} 1 & if \quad rand < Sig\left(vel_i^d\right) \\ 0 & otherwise \end{cases} \tag{5}$$

where w is inertia weight, *rand* is an arbitrary number chose in interim (0, 1), pos_i^d is location of *ith* particle at dimension d and $\left(Sig\left(vel_i^d\right)\right)$ indicates sigmoidal function. The BPSO algorithm is good at exploitation, however, poor at exploration.

2.2 Binary moth flame optimization algorithm (BMFOA)

Seyedali Mirjalili proposes a new nature-inspired algorithm named as Moth Flame Optimization (MFO) (Mirjalili 2015) in 2015 which is inspired by the navigation behavior of moths known as transverse orientation, i.e., keeping a fixed angle on a distant source of light for orientation. It is a population-based swarm intelligence technique which simulates the behaviour of moths at night. Moths attract towards the moonlight or counterfeit lights made by people. Here, a moth flies by using as a settled point with respect to the moon which is an extremely powerful instrument for traveling long separations in a straight way. This system ensures that moths are flying in a straight line in the light. When moths see a man-made light, they maintain a similar angle with the light to fly in a straight line causes a helical route for moths because such man-made lights are very close contrasted with the moon. It might be watched that the moth unsurprisingly meets towards the light. Fundamental segments of MFO calculation are moths and flames. In the MFO algorithm, it is acknowledged that the candidate solutions are moths and the issue's factors are the location of moths in the space. The moths are real hunt specialists that move around the search space. However, flames are in the best position of moths that gets up until this point. As specified over that this advancement calculation is propelled by exceptional route technique, i.e., transverse introduction. The position of every moth which is refreshed with respect to a flame is mathematically defined as follows:

$$Moth^i = SP\left(Moth^i, Flame^j\right) \tag{6}$$

where $Moth^i$ shows the *ith* moth, $Flame^j$ demonstrates the *jth* flame, and SP is the spiral function. The logarithmic spiral function for the MFO algorithm is defined as follows:

$$SP\left(Moth^i, Flame^j\right) = Dis^i *^{bI} * \cos(2\pi I) + Flame^j \tag{7}$$

where Dis^i demonstrates separation of the *ith* moth for the *jth* flame (refer Eq. (8)), b is a constant for characterizing shape of the logarithmic spiral, and I is a random number in [− 1, 1].

$$Dis^i = \left|Flame^j - Moth^i\right| \tag{8}$$

where $Moth^i$ indicates the *ith* moth, $Flame^j$ indicates *jth* flame and Dis^i indicates separation between $Moth^i$ and $Flame^j$.

The parameter I [refer Eq. (7)] decides route of moth around the flame (I = − 1 indicates nearest location to the flame, while I = 1 demonstrates the uttermost). Exploration and exploitation of the binary space can be ensured due to the spiral equation which allows a moth to fly around a flame. I is defined as a random number in [r, 1], where r linearly decreased from − 1 to − 2 over the course of iterations. According to Eq. (7), each moth is constrained to move towards a flame that may incite adjacent perfect stagnation. Flames must be arranged according to their fitness values. The moths by then invigorate their situations with respect to their relating flames. A versatile mechanism for the number of flames (f^{no}) has been proposed as in the following formula due to the degradation in exploitation of the best promising solutions:

$$f^{no} = round\left(N^f - t * \frac{N^f - 1}{T}\right) \tag{9}$$

where t is the present number of iteration, N^f is the maximum number of flames, and T is the total number of iterations.

According to Mirjalili (2015), it can be determined that local optima is high in MFO since MFO utilizes a population of moths to perform optimization. Decreasing the quantity of flames adjusts exploration and exploitation of the search space. The convergence of the MFO algorithm is ensured on the grounds that the moths always tend to update their positions with respect to flames.

In binary search space, the location of moths is confined to twofold factor, i.e., 0 or 1. Here, 1 shows the presence of corresponding feature and 0 demonstrates the absence of that feature. Sigmoidal function (Reddy et al. 2017) $Sig(Moth(t+1))$ is used to change the location of each moth which is invigorated in regards to a flame (refer Eq. (6)) in continuous search space into the new position in binary search space (refer Eq. (10)) for BMFOA.

$$Sig(Moth(t+1)) = \frac{1}{1 + e^{-(Moth(t+1))}} \tag{10}$$

where $Moth(t+1)$ is the location update of the moth for $(t+1)th$ iteration.

In binary space, the position of every moth with respect to a flame is computed (in binary search space) by the given Eq.:

$$Moth^i = \begin{cases} 1 & if \quad rand < Sig(Moth(t+1)) \\ 0 & otherwise \end{cases} \tag{11}$$

2.3 Binary dragonfly algorithm (BDFA)

Dragonfly algorithm (DFA) is introduced in 2015 by Mirjalili (2016b) inspired from dragonflies (small hunters that eat almost all other small insects in nature). Dragonfly has two main swarming behaviours which are the main inspiration of DFA and these behaviours are: static and dynamic swarming behaviours. Exploration and exploitation are two important components of any NIA. In DFA, mathematically these behaviours are expressed by:

- Separation states to avoid the static collision from another sub group of dragonflies. Mathematically, this behaviour is defined by:

$$U_i = -\sum_{j=1}^{N} pos - pos_j \tag{12}$$

  where pos represents current location of an individual, pos_j indicates jth neighbouring individual's location, and N is total quantity of neighbouring individuals.
- Alignment demonstrates matching of an individual's velocity to other neighbourhood individuals. Mathematically, this behaviour is simulated by:

$$O_i = \frac{\sum_{j=1}^{N} vel_j}{N} \tag{13}$$

  where vel_j indicates velocity of jth neighbouring individual.
- The property of an individual to incline towards neighbouring mass center is known as Cohesion. It is computed by:

$$Q_i = \frac{\sum_{j=1}^{N} pos_j}{N} - pos \tag{14}$$

  where N is total number of neighbouring individuals, pos denotes location of current individual, and pos_j indicates location of jth neighbouring dragonfly.
- Attraction behaviour of dragonfly in the direction of food source is computed as:

$$V_i = pos_+ - pos \tag{15}$$

  where pos_+ represents position of food source, and pos represents position of current individual.
- Diversion behaviour from opponent is find as:

$$Z_i = pos_- + pos \tag{16}$$

  where pos_- represents location of enemy, and pos represents location of current individual.

Overall artificial dragonflies consist of five behaviours which are described above. In a continuous space, two vectors are computed to update movement, and location of these dragonflies, i.e., step (Δpos) and position (pos). The step vector and the position vector are calculated as defined in Eqs. (17) and (18).

$$\Delta pos_{t+1} = (uU_i + oO_i + qQ_i + vV_i + zZ_i) + w\Delta pos_t \tag{17}$$

where u is the division weight, U_i demonstrates partition of ith individual, o is an alignment weight, O_i is separation of ith individual, q is the cohesion weight, Q_i is the cohesion of ith individual, v is the sustenance factor, V_i is the food source of the ith individual, z is the adversary factor, Z_i is the location of enemy of ith individual, w is an inertia weight, and t is the iteration counter.

$$pos_{t+1} = pos_t + \Delta pos_{t+1} \tag{18}$$

where pos denotes the current location of an individual and t is the current iteration.

Positions of DA are updated by adding step vectors (similar to velocity vector of PSO) to the position vectors in continuous searching space. In binary space, location of BDFA (Mafarja et al. 2017) is represented in binary form, i.e., 1 or 0 only without using step vector. A transfer function is used according to Mirjalili and Lewis (2013) and Saremi et al. (2015), to convert continuous space to a binary search space without changing the structure of swarm intelligence algorithm. Transfer function accepts velocity (step) values as an input and returns a value in [0, 1], which determine likelihood of changing positions. The transfer function is given below:

$$TF(\Delta pos) = \left| \frac{\Delta pos}{\sqrt{\Delta pos^2 + 1}} \right| \tag{19}$$

$$pos_{t+1} = \begin{cases} \neg pos_t & if \quad r < TF(\Delta pos_{t+!}) \\ pos_t & otherwise \end{cases} \tag{20}$$

where r is a random number in the interval of [0, 1].

According to Mirjalili and Lewis (2013) and Saremi et al. (2015), the algorithm was well-appointed with five parameters to control cohesion, alignment, separation, attraction (towards food sources), and diversion (outwards enemies) of individuals in the swarm. The convergence of the artificial dragonflies towards optimal solutions in continuous and binary search spaces was also observed and confirmed, which are because of the dynamic swarming design behaviour.

2.4 Binary whale optimization algorithm (BWOA)

Whale Optimization Algorithm (WOA) (Mirjalili and Lewis 2016) is introduced in 2016 by Mirjalili and Lewis. Whales are an elegant living being. Humpback whales (one of the largest whale) chase swarms of krill or little fishes near the surface. The foraging is performed by creating distinctive bubbles along a circle. However, Goldbogen

et al. (2013) examine this conduct using label sensors. WOA has two stages: the first stage is scanning arbitrarily for prey which is exploration stage and the second stage is enclosing a prey and spiral bubble-net attacking method which is an exploitation phase.

Movement of whale around a prey is depicted in the exploitation stage. Mathematical formulation of this step is given by:

$$\vec{M} = \left|\vec{G}.\overrightarrow{S^*}(t) - \vec{S}(t)\right| \tag{21}$$

$$\vec{S}(t+1) = \overrightarrow{S^*}(t) - \vec{H}.\vec{M} \tag{22}$$

where t indicates current number of iteration, M shows distance between whale and prey, S^* demonstrates the best solutions acquired up until this point, and S shows the current solution. H and G represent the coefficient vectors which is defined by:

$$\vec{H} = 2\vec{c}.\vec{g} - \vec{c} \tag{23}$$

$$\vec{G} = 2.\vec{g} \tag{24}$$

$$\vec{c} = 2 - t\frac{2}{T} \tag{25}$$

where c gradually decreases from 2 to 0 and g is a random vector in [0, 1], t is the current iteration, and T is the maximum number of iterations.

The distance between the solution (S) and the best solution (S^*) causes the spiral shaped path which is computed by:

$$\vec{S}(t+1) = \vec{M}.e^{bI}.\cos(2\pi I) + \overrightarrow{S^*}(t) \tag{26}$$

where b is the helix's form of the spiral, and I is an irregular number in [− 1, 1].

Shrinking encircling model or spiral model is simulated by Eq. 5:

$$\vec{S}(t+1) = \begin{cases} \overrightarrow{S^*}(t) - \vec{H}.\vec{M} & if\ rand < 0.5 \\ \vec{M}.e^{bI}.\cos(2\pi I) + \overrightarrow{S^*}(t) & if\ rand \geq 0.5 \end{cases} \tag{27}$$

where *rand* is a random number in uniform dissemination.

For exploration phase, the above approach can be utilized to hunt for prey by variation of H vector. A vector H with random values (greater than 1 or less than − 1) is utilized to constrain a solution to move far away from the best-known searched agent. It is computed by:

$$\vec{M} = \left|\vec{G}.\overrightarrow{S_{randm}} - \vec{S}\right| \tag{28}$$

$$\vec{S}(t+1) = \overrightarrow{S_{randm}} - \vec{H}.\vec{M} \tag{29}$$

where S_{randm} is a random whale taken from the present population.

For BWOA (Mafarja and Mirjalili 2017), the sigmoidal function is used to map continuous value into a binary value.

$$Sig\left(\vec{S}(t+1)\right) = \frac{1}{1 + e^{-\vec{S}(t+1)}} \tag{30}$$

$$\overrightarrow{S^{new}} = \begin{cases} 1 & if \quad rand < Sig\left(\vec{S}(t+1)\right) \\ 0 & otherwise \end{cases} \tag{31}$$

2.5 Sine cosine algorithm (SCA)

SCA is proposed in 2016 by Mirjalili (2016a) and Meshkat and Parhizgar (2017). It is based on the functions of Sine and Cosine for exploration and exploitation phases, respectively. SCA starts with random solutions to swing near or away the best possible solution using mathematical expressions defined in Eq. 32. The cyclic condition of sine and cosine enables a solution to be re-arranged around other solution, which facilitates exploitation. In the exploitation phase, however, there are gradual changes in the random solutions, and random variations are considerably less than those in the exploration phase.

In SCA, the position of solution is updated using Eq. 32:

$$\overrightarrow{pos_i^{t+1}} = \begin{cases} \overrightarrow{pos_i^t} + r_3 \times \sin(r_4) \times \left|r_5 \times L_i^t - \overrightarrow{pos_i^t}\right| & if\ r_6 < 0.5 \\ \overrightarrow{pos_i^t} + r_3 \times \cos(r_4) \times \left|r_5 \times L_i^t - \overrightarrow{pos_i^t}\right| & if\ r_6 \geq 0.5 \end{cases} \tag{32}$$

where pos_i^t vector is the current position at *tth* iteration in *ith* dimension, r_3, r_4, r_5 are the random numbers, r_6 are the random number in [0, 1] and L_i is targeted global optimal solution.

$$r_3 = a - t\frac{a}{T} \tag{33}$$

where t is the current iteration, T is the total number of iterations, and a is a constant.

The parameter r_3 demonstrates the direction of movement which could be either in the space between the solution and goal or outside it. The parameter r_4 is a random number in [0, 2π] which exhibits how far the development ought to be towards or outwards the objective and parameter r_5 shows an unpredictable weight for the objective to emphasize ($r_5 > 1$) or deemphasize ($r_5 < 1$) the irregular effect of objective in describing the partition. Finally, the parameter r_6 switches between the sine and cosine segments in Eq. 32.

According to Mirjalili (2016a) and Meshkat and Parhizgar (2017), it can be concluded that the mathematical model of position update equation update the solutions outwards or towards the goal point to ensure exploration and exploitation of the search space, respectively. SCA can

be a very appropriate option compared to the current algorithms for solving different optimization problems.

3 Proposed method

In this paper, we explore capability of NIA for feature selection problem. Traditionally, real-world datasets include a large number of features. However, some of these features are non-informative, redundant and noisy. These features not only increase computational complexity of the underlying algorithm but deteriorate its performance. In literature different nature-inspired algorithms are proposed for feature selection task (Diao and Shen 2015; Kumar 2018). In this paper, a new hybrid method HBPSOSCA is proposed using BPSO and SCA to select an informative subset of features. Here, the V-shaped transfer function is used to convert continuous nature of an algorithm to binary. The movement of a particle in the BPSO algorithm is improved using the sine–cosine algorithm. A detailed description of each of the step is presented in the subsequent sections.

3.1 Initialization of swarm

This Subsection discusses the very first step of the proposed method, i.e., initialization of the swarm. Here, a particle is represented in binary form of size n (as shown in Fig. 3), where n represents the total number of features in the dataset. Here, binary value 1 shows presence of corresponding feature and 0 depicts its absence. We work with the population size of n. For this, the first swarm is randomly initialized with the 2 n particles. Next, top n particles are selected from the 2 n particles with the assistance of fitness value. A pictorial representation of a particle with size n is presented in Fig. 3.

3.2 Clustering algorithm

Clustering is a process of grouping data points into clusters based on defined criteria. Traditionally, clustering methods are categorized into two categories; hierarchical clustering (Xu and Tian 2015), partitional clustering (Xu and Tian 2015). Hierarchical clustering creates clusters either top to bottom, known as divisive clustering or bottom to top known as agglomerative clustering. Initially, a divisive method considers each data point as a part of one cluster and recursively divides the clusters based on defined criteria. On the other hand, the agglomerative clustering method considers each data point as individual clusters and recursively merge the cluster based on defined criteria. On the contrary, partitional clustering creates clusters of the data point at one level.

In this paper, we use the K-mean partitional clustering method. K-means clustering (Xu and Tian 2015; Jain and Dubes 1988; Prakash and Singh 2012) is one of the most commonly used partitional clustering techniques. Here, K is the number of clusters. K-means works iteratively by assigning each data point to one of the K clusters based on some predefined measure. Traditionally, the Euclidian distance measure is used for this purpose. The Euclidian distance is calculated between the cluster center and data point. The data point is assigned to the cluster from which data point has a minimum distance. Next, each data point is assigned to the cluster to which it has smaller Euclidian distance from the cluster center. Assignment of data points to the respective clusters and refinement of cluster centers based on the average of the points assigned to the respective clusters are repeated until the termination criterion is met.

3.3 Evaluation criteria/fitness function

In this subsection, a detailed description of Silhouette Index is presented which is used to evaluate quality of potential solution. The silhouette index is proposed by Kaufman and Rousseeuw (2009) and Desgraupes (2013), which show the similarity of a data point to its own cluster as compared to its similarity with other clusters.

Consider a data point x_i lies in a cluster, where $p(x_i)$ be the normal disparity of x_i with every other datum inside a similar group and $f(x_i)$ be the most reduced normal difference of x_i to some other group. Silhouette index $Sil(x_i)$ for a particular data point x_i is numerically calculated as follows:

$$Sil(x_i) = \frac{f(x_i) - p(x_i)}{max\{p(x_i), f(x_i)\}} \tag{34}$$

The values of Silhouette index lies in the range $[-1, 1]$. Here, high value shows that the data point well matched with own and compared to other clusters.

3.4 Proposed algorithm

As mentioned in Sect. 1, the proposed HBPSOSCA algorithm is a nature-inspired wrapper feature selection algorithm. In this section, we present a detailed description of

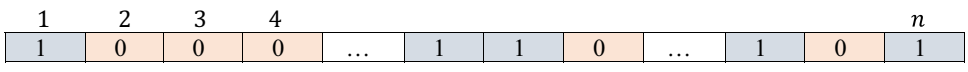

Fig. 3 Particle representation

the proposed method. Firstly, best n particles are selected from the 2 n randomly generated particles based on fitness value (for more details refer, Sect. 3.1). Each particle is the potential solution in the search space. Next, K-means clustering is applied to create clusters of data points on the selected features subspace. Here, the particle's quality is evaluated using fitness function (refer Sect. 3.3). Swarm gbest (best solution of the entire swarm) and pbest (individual best solution) particles are selected from the current population based on their fitness. During the iteration, some particle gets stuck at a local optimum solution. We use a replacement strategy to deal with this problem. This strategy replaces a particle with the opposite solution if a particle does not improve for the defined number of iterations. Flowchart and algorithmic flow of the proposed method are demonstrated in Fig. 4 and Algorithm 1, respectively.

In this paper, an amalgamation of the BPSO with SCA helps the algorithm to expand its searching capability and locate the near global optimum solution. In SCA, if functions of sine and cosine generate a value that is higher than 1 or smaller than -1, then it signifies the exploration of different areas within the search space. Likewise, if sine and cosine functions generate a value within the range between -1 and 1 then it demonstrates that proficient areas of search space are exploited. The SCA algorithm effectively travels from exploration to exploitation utilizing versatile range in the sine and cosine functions. In the hybrid approach, the movement of a particle in the BPSO is improved using the sine–cosine algorithm.

In this paper, we use a V-shaped function to change a real-valued swarm intelligence system to a binary algorithm without altering the knowledge of swarm. As per Saremi et al. (2015), the V-shaped transfer functions [refer Eq. (35)] are superior to S-shaped transfer functions since they don't constrain particles to take estimations of 0 or 1.

$$TF\left(vel_i^d(t+1)\right) = \left| \frac{vel_i^d(t+1)}{\sqrt{1+\left(vel_i^d(t+1)\right)^2}} \right| \tag{35}$$

For proposed HBPSOSCA, the accompanying new technique for updating particle's velocity and position is utilized, respectively to refresh the position of particles in twofold search spaces. Mathematical formulation of these steps is given in Eqs. 36–39.

$$mov = w * vel_i^d(t) + e_1 r_1 * \left(pbest_i^d(t) - pos_i^d(t)\right) + e_2 r_2 * \left(gbest^d(t) - pos_i^d(t)\right) \tag{36}$$

$$A = r_3 * \sin(r_4) \tag{37}$$

$$B = r_3 * \cos(r_4) \tag{38}$$

$$vel_i^d(t+1) = \begin{cases} A * |mov| & if\ rand < 0.5 \\ B * |mov| & if\ rand \geq 0.5 \end{cases} \tag{39}$$

where r_1, r_2 are random numbers in the interim (0, 1), r_3 and r_4 are also the random numbers, pos_i^d is position of *ith* particle at dimension d, and w is the inertia weight (Jain et al. 2017; Bansal et al. 2011). The mathematical formulation of inertia weight is presented in Eq. 40.

$$w = \begin{cases} w_{min} + \dfrac{t.(w_{max} - w_{min})}{\xi.T} & if \quad t \leq \xi.T \\ w_{max} & if \quad \xi.T < t \leq T \end{cases} \tag{40}$$

Here, the inertia weight is increased linearly in every iteration and regulates the velocity of all particles. Here, w_{min} is the initial value and w_{max} is final value of the inertia weight, t represents current iteration, T is the total number of iterations, and ξ is used to control the fraction of iterations when w increases linearly from w_{min} to w_{max}. Here, $\xi = 0.9$ is used to achieve stronger exploration in initial iterations and high exploitation in later iterations (Jain et al. 2017).

The parameter r_3 shows the direction of movement that can be either outside or space in between the solution and destination. It is defined as follows:

$$r_3 = a - t\frac{a}{T} \tag{41}$$

where t is the current iteration, T is the total number of iterations, and a is a constant.

The parameter r_4 is an irregular number in [0, 2π] demonstrates how far the development ought to be towards or outwards the goal.

$$pos_i^d = \begin{cases} 1 & if \quad rand < TF\left(v_i^d(t+1)\right) \\ 0 & otherwise \end{cases} \tag{42}$$

where pos_i^d is the position of *ith* particle at dimension d, *rand* is a random number, and *TF* is the V-shaped transfer function. The mathematical formula is given in Eq. 35.

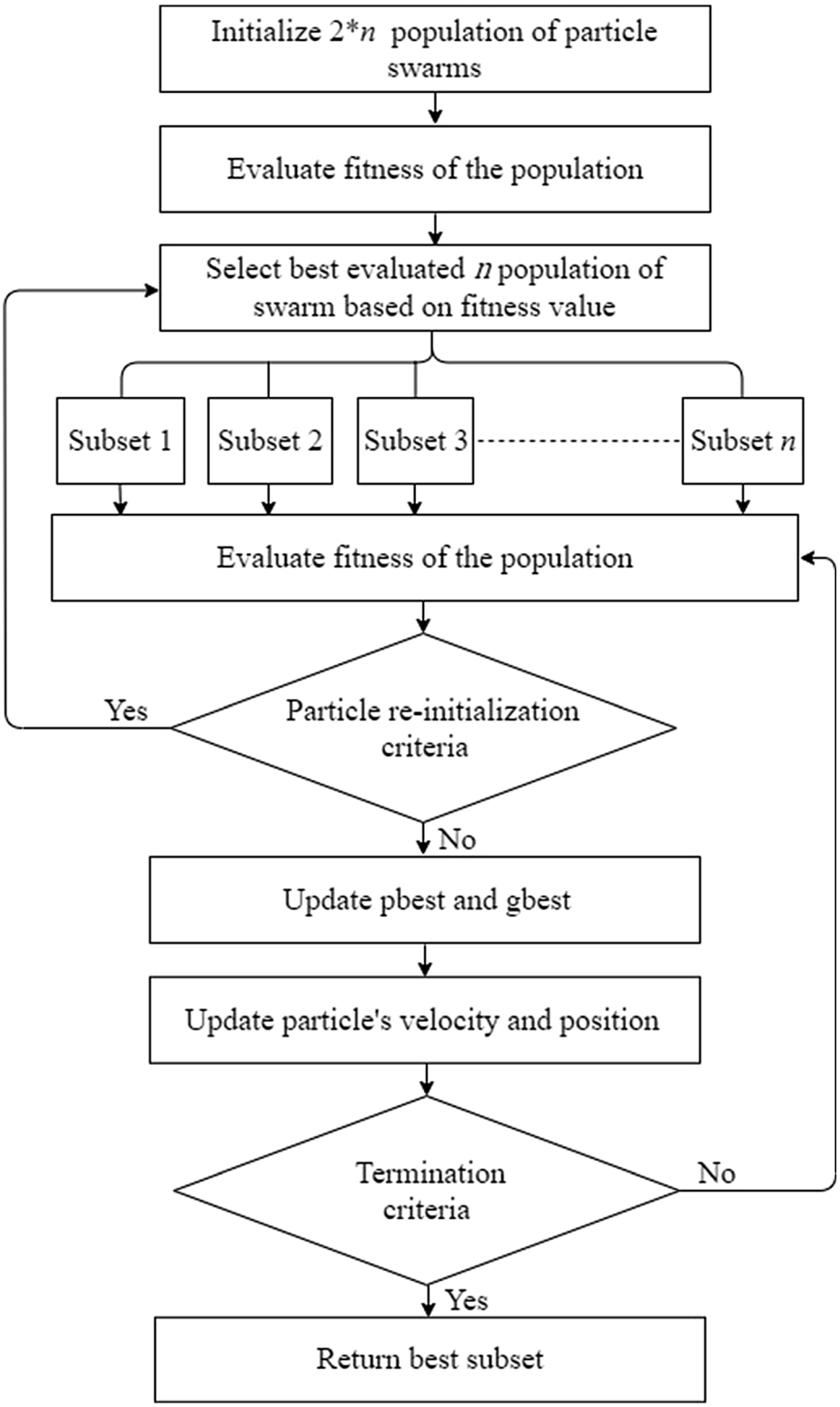

Fig. 4 Flow of proposed method

Algorithm 1: The proposed HBPSOSCA

INPUTS: n Number of particles (swarm size);
T Number of iteration;
K Number of clusters;
Variable count, max_count;

OUTPUT: *gbest* and subset of informative features;

Algorithm:

1) Introduce population of swarm of size 2 * n randomly in binary search space and select best n particles from swarm (Refer, Subsection 3.1);
2) ***For*** each particle do
3) Assess particle fitness by fitness function (Silhouette Index (Eq. 34));
4) Instate *pbest*;
5) ***End for***
6) Select *gbest* (best particle among *pbest* in the swarm);
7) ***While*** maximum iteration is not reached **do**
8) ***For*** each particle **do**
9) Update its velocity and position by utilizing Eq(s) 39-42;
10) Evaluate fitness of new updated particle;
11) ***If*** (fitness of prev. particle ≥ fitness of new particle) **do**
12) Increment the value of count variable;
13) ***Else***
14) set count value to zero;
15) ***If*** (count ≥ max_count) **do**
16) Re-instate the particle or perform supplement of that particle;
17) Set count value to zero;
18) ***End if***
19) ***End if***
20) Update *pbest* of particle;
21) ***End for***
22) Update *gbest* accordingly;
23) ***End while***

4 Experimental results and discussions

4.1 Benchmark functions

Experiments are carried out over a set of 10 widely used benchmark test functions listed in Appendix A. These functions are taken from (Yao et al. 1999) each with different characteristics. Among these benchmarks, F1 to F3 are unimodal functions, F4 is the Rosenbrock function which is a unimodal function for D = 2 and multimodal function for D > 3 (Shang and Qiu 2006), F5 is a step function, F6 is a noisy quartic function, F7 to F10 are multimodal functions.

4.2 Datasets and parameters setup

The effectiveness of the proposed method is demonstrated using several experiments conducted on seven real-life scientific datasets. The datasets are taken arbitrarily from the UCI archive of machine learning and GEMS. The brief description of the datasets is presented below. The statistical details of used datasets are presented in Table 1 (Frank 2010; http://www.gems-system.org).

Ionosphere: This dataset is drawn from the Johns Hopkins University. It comprises of 34 number of features, and 351 number of occurrences belonging to two distinct categories either good or bad. Out of 351 instances, 225 are good and rest 126 are bad occasions.
Breast Cancer Wisconsin: It is acquired from the University of Wisconsin Hospitals Madison, Wisconsin, USA. It consists of 9 features, and each one has ten different values. There is 699 number of instances in which 458 instances belong to the benign class, and the remaining 241 instances belong to the malignant class.
Connectionist Bench: This dataset is developed in Allied-Signal Aerospace Technology Center. There are 60 number of features in this dataset and 208 number of instances which belongs to two different classes (rock and mines). In this, 97 instances belong to rock, and remaining instances belong to mines.
Statlog: It has 18 features and 946 instances which is divided into four classes. The 226 instances belong to

Table 1 Datasets description

Dataset	Number of cluster	Number of features	Number of samples
Ionosphere	2	34	351
Breast Cancer Wisconsin (BCW)	2	9	699
Connectionist Bench (Sonar, Mines vs. Rocks)	2	60	208
Statlog (Vehicle Silhoettes)	4	18	946
Parkinson	2	23	195
9_Tumors	9	5726	60
Leukemia2	3	11,225	72

van, 240 instances belong to saab, 240 instances belong to bus, and the remaining 240 instances belong to Opel.
Parkinson: It is acquired from the Max Little of the University of Oxford. It has 23 features and 195 instances. The 147 instances belong to Parkinson category, and the remaining 48 belongs healthy.

Table 2 Parameter settings

Algorithm	Parameters	Value
BPSO	Population size (swarm size)	50
	e_1 and e_2	1.5
	$[vel_{min}, vel_{max}]$	[− 6, 6]
	T (number of iterations)	150
Chaotic-BPSO	Population size (swarm size)	50
	T (number of iterations)	150
	e_1 and e_2	1.5
	Initial inertia weight	0.48
BMFO	No. of search agents (moths)	50
	b (spiral's shape)	1.0
	T (number of iterations)	150
BDFA	Population	50
	T (number of iterations)	150
	w	0.9 to 0.2
BWOA	Population	50
	T (number of iterations)	150
	b (spiral's shape)	1.0
SCA	Population	50
	T (number of iterations)	150
	A	2
Binary ABC	Population	50
	T (number of iterations)	150
HBPSOSCA	Population size (swarm size)	50
	e_1 and e_2	1.5
	w_{max}	0.9
	w_{min}	0.4
	T (number of iterations)	150
	max_count	10
	ξ	0.9
	a	2

9_Tumors: The dataset comes from a study of 9 human tumor types: NSCLC, colon, breast, ovary, leukemia, renal, melanoma, prostate, and CNS. There are 60 samples, each of which contains 5726 genes.
Leukemia2: The dataset has 72 samples, each of which contains 11,225 features. These samples categorized into AML, ALL, and mixed-lineage leukemia (MLL).

4.3 Parameter setting

The parameters have a large impact on optimizing performance. In BPSO, particle's position and velocity are initialized randomly. Here, the position is in the form of 1 or 0 and velocity is in the range of − 6 to 6, and parameters e_1 ($0 \leq e_1$) and e_2 ($e_2 \leq 2$) are the weighting (acceleration/learning) coefficients for the personal best and global best positions respectively. In Chaotic-BPSO, tent map is preferable to logistics map according to Kennedy and Eberhart (1997). Therefore, the inertia weight parameter is set at to 0.48 for CBPSO. In BMFO, parameter b is a constant value which is used to define shape of the logarithmic spiral when position (refer Eq. (7)) of moth is updated. In BDFA, parameter w is the inertia weight which is used in updating the movement of the dragonfly (refer Eq. (17)) and its value varies from 0.9 to 0.2. In BWOA, parameter b is a constant value which is used to define shape of the logarithmic spiral when position of the whale is updated (refer Eq. (26)). In SCA, parameter a (refer Eq. (33)) is used to compute direction of movement which could be either in the space between the arrangement and objective or outside it. In artificial bee colony, the number of bees is the main factor that affects the speed and quality of the algorithm. In HBPSOSCA, parameters e_1 and e_2 are the acceleration coefficients used in BPSO, parameters w_{min} and w_{max} are the initial and final values of the inertia weight used in finding the linearly increasing inertia weight, i.e., w [refer Eq. (40)]. The parameter max_count is used to re-initialize the particular particle randomly or by taking the complement of that particle if the particle does not change its position for defined number of iteration. The

Table 3 The comparative analysis of BPSO, C-BPSO, BMFO, BDFA, BWOA, SCA, Binary-ABC, and HBPSOSCA on Benchmark Functions F1–F10

Function	BPSO	C-BPSO	BMFO	BDFA	BWOA	SCA	Binary-ABC	HBPSOSCA
F1								
Mean	9.5411	10.1450	23.8187	13.6173	5.2400	2.4120	22.0660	**2.1473**
SD	0.4096	0.3703	0.4758	2.4929	1.1288	0.1705	1.1375	**0.2203**
F2								
Mean	9.2974	10.2464	23.9813	11.2940	6.4200	2.2407	21.6920	**2.1341**
SD	0.4253	0.3157	0.3652	2.9564	0.8354	**0.1741**	0.8724	0.3216
F3								
Mean	160.9358	180.8801	483.4647	269.79	155.3867	37.2620	438.0480	**33.1313**
SD	6.5366	8.7939	8.5697	38.0267	29.6846	**3.3939**	20.0694	7.9122
F4								
Mean	1.1141 e^{+03}	1.1739 e^{+03}	1.7128 e^{+03}	1.4531 e^{+03}	217.1800	416.9060	1.9848 e^{+03}	156.3193
SD	29.9715	30.3347	42.2886	192.1704	0	41.2802	83.0103	26.6875
F5								
Mean	9.4483	10.0278	23.9440	12.3187	8.1467	2.3607	22.2707	**1.9175**
SD	0.3503	0.3381	0.5597	2.8085	1.8724 e^{-15}	0.1384	0.9200	0.2485
F6								
Mean	161.7552	182.3258	481.7741	41.2212	118.9141	36.0881	433.6432	**6.1027**
SD	11.3762	5.5510	7.3281	113.1499	**1.1579**	3.1750	25.4729	1.5730
F7								
Mean	1.6734 e^{+04}	1.6734 e^{+04}	1.6736 e^{+04}	1.6738 e^{+04}	**1.6727** e^{+04}	1.6736 e^{+04}	1.6747 e^{+04}	1.6733 e^{+04}
SD	0.3419	0.2780	0.3743	1.5369	22.4397	0.7677	0.4836	**0.2019**
F8								
Mean	9.4702	10.1709	24.0533	13.2473	7.1200	2.3547	21.8020	**2.1327**
SD	0.4195	0.3226	0.4479	2.8708	0.1232	**0.1213**	0.7693	0.2340
F9								
Mean	1.8526	1.8979	2.8684	2.1322	1.7424	0.5544	2.7508	**0.1391**
SD	0.0455	0.0380	0.0153	**0.1501**	1.0314	0.0261	0.0948	0.0304
F10								
Mean	0.2661	0.2727	0.7199	0.2547	0.1668	0.0596	0.5834	**0.0506**
SD	0.0134	0.0089	0.0123	0.0702	0.2440	**0.0043**	0.0260	0.0085

Bold entries show the best performance recorded with the respective model

parameter ξ varies in $0 \leq \xi \leq 1$. It is used to control portion of iterations where inertia weight increments straightly from w_{min} and w_{max}. Here, $\xi = 0.9$ [refer Eq. (40)] is used to achieve stronger exploration in initial iterations and high exploitation in later iterations, and parameter a [refer Eq. (41)] is used to compute direction of movement which could be either in the space between the arrangement and objective or outside it. Tabulated summary of the parameters adopted in this paper for eight different algorithms namely, BPSO, BMFO, BDFA, BWOA, and HBPSOSCA is presented in Table 2.

4.4 Dunn Index and Davies–Bouldin Index

Silhouette Index (SI), Dunn Index (DI) and Davies–Bouldin Index (DBI) are used to evaluate the quality of created clusters. The higher value of SI (refer Sect. 3.3) and DI, and a lower value of DBI represents better quality of cluster.

DI is proposed in 1974 by Desgraupes (2013) and Bezdek and Pal (1995) that measure maximum cluster diameter and relate it to the minimum cluster distance to judge the clustering performance. DI is calculated by dividing minimum distance between clusters by maximum size of clusters. Therefore, high value of distances between clusters and low value of cluster sizes indicates high value

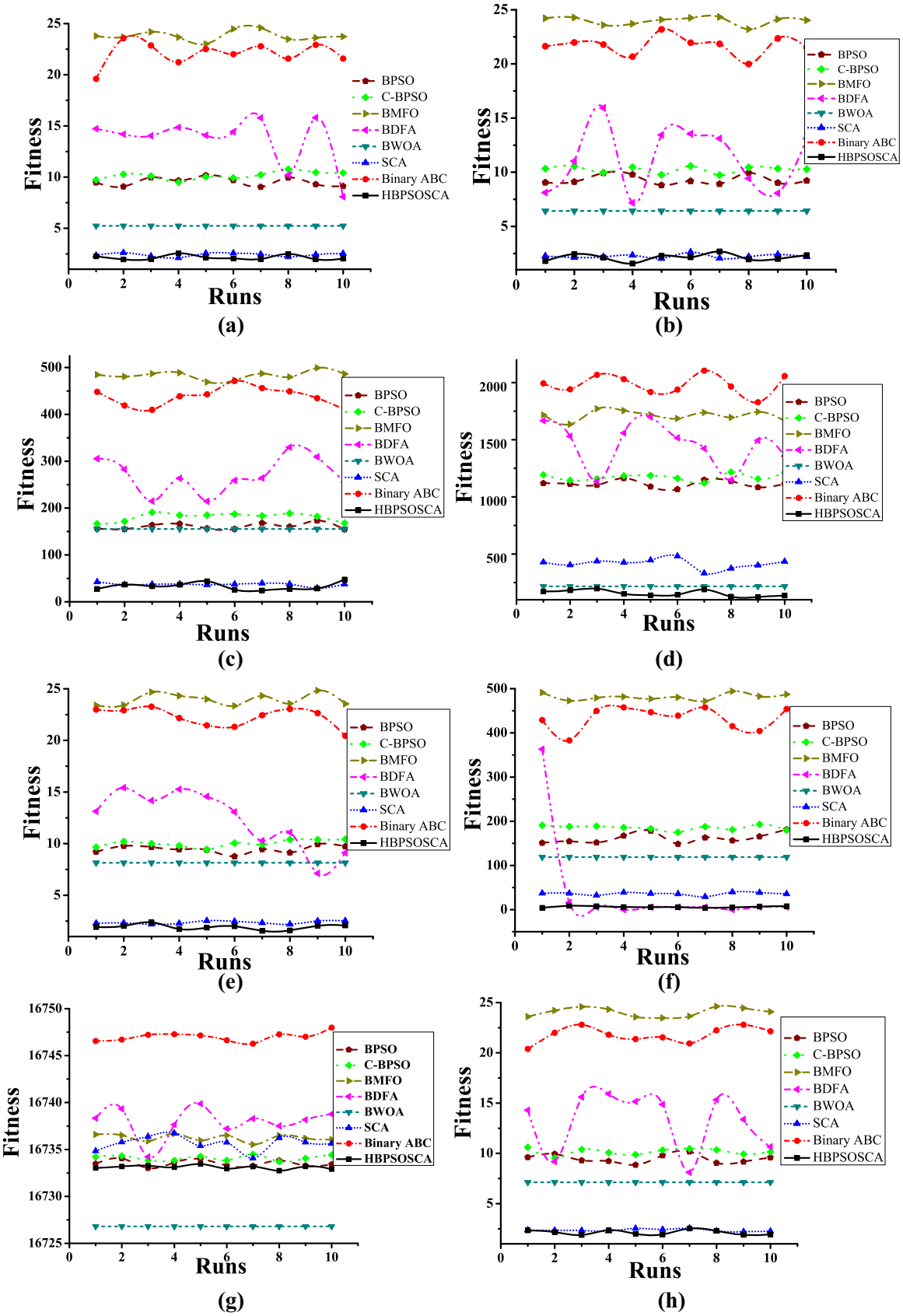

Fig. 5 Comparison of performance over 10 runs for 10 benchmark functions **a** F1, **b** F2, **c** F3, **d** F4, **e** F5, **f** F6, **g** F7, **h** F8, **i** F9, **j** F10

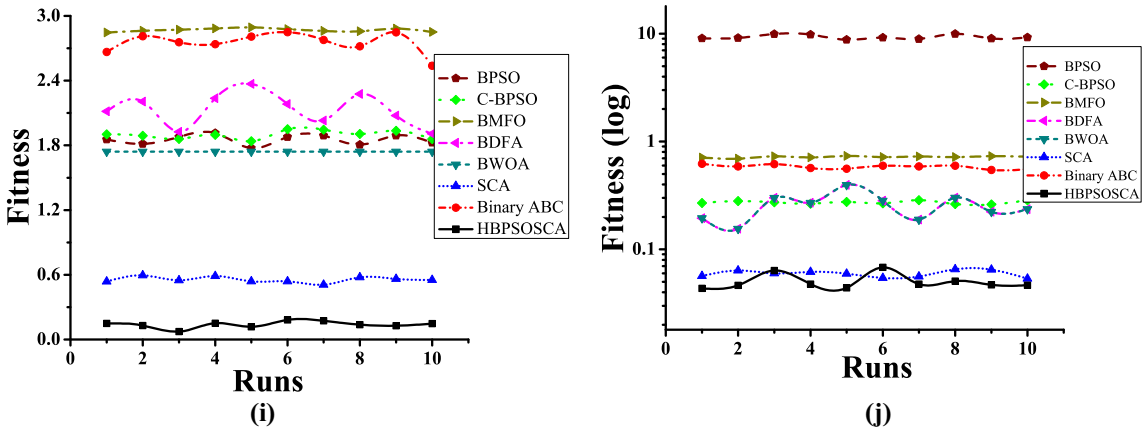

Fig. 5 continued

of DI, leading to increased clustering efficiency. DI for m clusters is denoted as follows:

$$DI = \frac{\min\limits_{1 \le i \le j \le m} \delta(C_i, C_j)}{\max\limits_{1 \le k \le m} \Delta_k} \tag{43}$$

where $\delta(C_i, C_j)$ is the distance among clusters i and j and Δ_k denotes the maximum distance between clusters.

DBI is an evaluation scheme introduced in 1979 by Desgraupes (2013) and Davies and Bouldin (1979). It is the ratio between the within cluster distances and the between cluster distances and the average overall the clusters. The value of DB varies between 0 and 1. The lower value of DBI denotes best partitions of the data point. It is calculated as follows:

$$DB = \frac{1}{m} \sum_{i=1}^{m} D_i \tag{44}$$

where m is the number of clusters and D_i is defined as:

$$D_i = \max_{j \neq i} R_{ij} \tag{45}$$

$$R_{ij} = \frac{P_i + P_j}{l_{ij}} \tag{46}$$

$$l_{ij} = EU(CI_i, CI_j) \tag{47}$$

$$P_i = \frac{1}{||C_i||} \sum_{X \in C_i} EU(X, CI_i) \tag{48}$$

where R_{ij} denotes clustering efficiency (refer Eq. (46)), distinction between *ith* and *jth* clusters is l_{ij} (refer Eq. (47)), within cluster distribution of *ith* cluster is denoted as P_i (refer Eq. (48)), $EU(x, y)$ is the Euclidean separation amongst x and y, center of cluster C_i is CI_i and $||C_i||$ is the convention for C_i.

4.5 Simulation results on benchmark functions

The mean and standard deviations of function values obtained by original BPSO, C-BPSO, BMFO, BDFA, BWOA, SCA, Binary-ABC, and HBPSOSCA for 150 iterations for 10 independent runs are given in Table 3. It is evident from Table 3 that the HBPSOSCA method attains better performance compared to competitive methods in most of the cases.

Graphical summary of the proposed algorithm in comparison to the competitive methods BPSO, C-BPSO, BMFO, BDFA, BWOA, SCA, and Binary ABC is presented in Fig. 5. Here, the fitness value recorded over the number of runs for each algorithm is reported in Fig. 5. The Fig. 5 demonstrates that the proposed method is consistent over the number of run in comparison to competitive methods in most of the case. This behavior depicts stability of the proposed method in comparison to competitive methods.

4.6 Simulation results on real-life datasets

In this subsection, we perform comparative analysis of all models for feature selection problem. The higher value of SI and DI represents better performance whereas lower value of DBI specifies better performance. The statistical results of 10 independent runs, obtained by HBPSOSCA and competitive methods are presented in Tables 4, 5 and 6. Furthermore, we measured the Wilcoxon test for all the algorithms and the results are presented in Tables 7, 8, 9, 10, 11, 12 and 13.

The data obtained from 10 runs is plotted in Fig. 6 to analyze the consistency and overall performance of HBPSOSCA. For the ionosphere dataset, the performance of proposed HBPSOSCA method is higher than the competitive methods with 92.7% average accuracy as shown in Fig. 6a. For BCW dataset, the performance of proposed

Table 4 The results recorded with the BPSO, C-BPSO, BMFO, BDFA, BWOA, SCA, Binary ABC, and HBPSOSCA with 10 independent runs on seven real-life benchmark datasets in term of SI

Algorithms	Dataset	Average length of selected features	Average SI	Best SI	Worst SI	Standard deviation
BPSO	Ionosphere	4.0333	0.6102	0.6132	0.6067	0.0020
	BCW	2.0000	0.8581	0.8669	0.8464	0.0073
	CB	18.1722	0.5697	0.5832	0.5626	0.0069
	Vehicle	8.1325	0.8111	08147	0.8059	0.0026
	Parkinson	2.6093	0.9441	0.9590	0.9354	0.0089
	9_Tumors	$2.8890e^{+03}$	0.2143	0.2196	0.2116	0.0028
	Leukemia2	$5.6196e^{+03}$	0.6256	0.6596	0.5979	0.0165
Chaotic BPSO	Ionosphere	7.5667	0.5856	0.5912	0.5828	0.0026
	BCW	2.0000	0.8749	0.8760	0.8737	$9.9195e^{-04}$
	CB	21.7351	0.5338	0.5432	0.5183	0.0072
	Vehicle	6.6821	0.8082	0.8108	0.8041	0.0022
	Parkinson	3.3113	0.9459	0.9459	0.9459	$1.0937e^{-05}$
	9_Tumors	$2.8666e^{+03}$	0.3670	0.3762	0.3588	0.0056
	Leukemia2	$5.5675e^{+03}$	0.8591	0.8596	0.8584	$4.3881e^{-04}$
BMFO	Ionosphere	21.8267	0.4075	0.3637	0.3536	0.0257
	BCW	5.3800	0.7579	0.7601	0.7532	0.0021
	CB	38.4867	0.3579	0.3637	0.3536	0.0234
	Vehicle	11.4400	0.6109	0.4156	0.4020	0.0296
	Parkinson	13.0800	0.7823	0.7941	0.7715	0.0087
	9_Tumors	$3.6465e^{+03}$	0.1415	0.1466	0.1376	0.0030
	Leukemia2	$7.2229e^{+03}$	0.3205	0.3280	0.3130	0.0051
BDFA	Ionosphere	17.7733	0.2064	0.2649	0.1964	0.0263
	BCW	2.3000	0.5871	0.6202	0.5697	0.0118
	CB	28.0933	0.2405	0.2649	0.1964	0.0123
	Vehicle	6.6067	0.2463	0.2277	0.1817	0.0286
	Parkinson	11.3400	0.5073	0.5161	0.4998	0.0055
	9_Tumors	2833	0.0944	0.1102	0.0717	0.0101
	Leukemia2	5604	0.0345	0.2833	− 0.0624	0.1030
BWOA	Ionosphere	28.1800	0.5783	0.6099	0.5097	0.0213
	BCW	2.0267	0.8701	0.8769	0.8615	0.0058
	CB	22.2800	0.5364	0.6099	0.5097	0
	Vehicle	8.3133	0.8372	0.6086	0.5605	0.0080
	Parkinson	5.8067	0.9349	0.9459	0.9159	0.0129
	9_Tumors	$2.9149e^{+03}$	0.2213	0.2213	0.2213	$5.8514e^{-17}$
	Leukemia2	$6.0829e^{+03}$	0.8029	0.8029	0.8029	0
SCA	Ionosphere	4.2067	0.8378	0.8591	0.7990	0.0213
	BCW	2.0067	0.8409	0.8782	0.8175	0.0232
	CB	9.3733	0.7825	0.7980	0.7505	0.0140
	Vehicle	4.0267	0.8273	0.8659	0.8207	0.0136
	Parkinson	3.0000	0.9543	0.9572	0.9523	0.0020
	9_Tumors	$1.6526e^{+03}$	0.2665	0.2859	0.2487	0.0106
	Leukemia2	$3.6262e^{+03}$	0.7599	0.8152	0.6209	0.0807
Binary ABC	Ionosphere	17.5733	0.4073	0.4221	0.3857	0.0118
	BCW	4.9000	0.6677	0.7471	0.5011	0.0701
	CB	30.8267	0.3920	0.4017	0.3816	0.0079
	Vehicle	7.6533	0.6292	0.6606	0.5989	0.0189
	Parkinson	15.9267	0.8698	0.9209	0.8280	0.0278
	9_Tumors	$2.8651e^{+03}$	0.0603	0.0654	0.0528	0.0043
	Leukemia2	$5.6533e^{+03}$	0.2461	0.3354	0.1724	0.0670

Table 4 (continued)

Algorithms	Dataset	Average length of selected features	Average SI	Best SI	Worst SI	Standard deviation
HBPSOSCA	Ionosphere	2.2400	0.9270	0.9366	0.8809	0
	BCW	2.0000	0.8786	0.8800	0.8771	0.0010
	CB	2.5629	0.9041	0.9366	0.8809	0.0237
	Vehicle	3.5629	0.8461	0.9751	0.8303	0.0013
	Parkinson	2.1325	0.9613	0.9647	0.9589	0.0020
	9_Tumors	188.6644	0.8006	0.8556	0.7421	0.0330
	Leukemia2	932.6980	0.9073	0.9781	0.8237	0.0486

Table 5 The results recorded with the BPSO, C-BPSO, BMFO, BDFA, BWOA, SCA, Binary ABC, HBPSOSCA with 10 independent runs on seven real life benchmark datasets in term of DI

Algorithms	Dataset	Average length of selected features	Average DI	Best DI	Worst DI	Standard deviation
BPSO	Ionosphere	25.0397	0.1379	0.3105	0.1119	0.0162
	BCW	6.0132	0.1770	0.1770	0.1770	0.0376
	CB	30.8940	0.3480	0.3703	0.3272	0.0689
	Vehicle	10.8609	0.5012	0.8443	0.2011	0.2175
	Parkinson	19.5033	0.3409	0.3409	0.3409	0.1472
	9_Tumors	$2.8893e^{+03}$	0.5510	0.5728	0.5425	0.0108
	Leukemia2	$6.1238e^{+03}$	0.7044	0.8235	0.6171	0.0632
Chaotic BPSO	Ionosphere	18.6689	0.0975	0.0993	0.0949	0.0015
	BCW	6.9404	0.1741	0.1746	0.1735	$3.6457e^{-04}$
	CB	31.4437	0.2790	0.2838	0.2695	0.0040
	Vehicle	9.6490	0.2950	0.3184	0.2751	0.0127
	Parkinson	2.5430	0.3513	0.3513	0.3513	$5.2872e^{-08}$
	9_Tumors	$2.8133e^{+03}$	0.4950	0.5036	0.4824	0.0065
	Leukemia2	$5.5576e^{+03}$	1.5373	1.5427	1.5317	0.0030
BMFO	Ionosphere	23.0067	0.0939	0.0980	0.0910	0.0026
	BCW	5.7867	0.1729	0.1770	0.1628	0.0053
	CB	38.8200	0.2662	0.2900	0.2450	0.0092
	Vehicle	11.1933	0.2058	0.2442	0.1713	0.4827
	Parkinson	14.8067	0.3409	0.3409	0.3409	1.0790 e^{-06}
	9_Tumors	$3.6822e^{+03}$	0.4084	0.4159	0.4037	0.0034
	Leukemia2	$7.1904e^{+03}$	0.4107	0.4287	0.3973	0.0085
BDFA	Ionosphere	21.2733	0.0296	0.0357	0.0239	0.0068
	BCW	4.2000	0.0516	0.0570	0.0486	0.0032
	CB	26.0533	0.0607	0.0703	0.0467	0.0074
	Vehicle	2.9600	0.0334	0.0405	0.0244	0.0048
	Parkinson	14.8133	0.0108	0.0130	0.0081	0.0016
	9_Tumors	2834	0.3221	0.3400	0.3026	0.0136
	Leukemia2	5662	0.2215	0.3412	0.1872	0.0462
BWOA	Ionosphere	27.6133	0.1046	0.1046	0.1046	1.4628 e^{-17}
	BCW	6.0867	0.1770	0.1770	0.1770	2.9257 e^{-17}
	CB	32.2600	0.2985	0.3071	0.2876	5.5814 e^{-17}
	Vehicle	12.2200	0.2647	0.2786	0.2483	5.8514 e^{-17}
	Parkinson	19.9667	0.3409	0.3409	0.3409	0
	9_Tumors	$3.2932e^{+03}$	0.5400	0.5400	0.5400	0
	Leukemia2	$4.0762e^{+03}$	0.7654	0.7654	0.7654	$1.1703e^{-16}$

Table 5 (continued)

Algorithms	Dataset	Average length of selected features	Average DI	Best DI	Worst DI	Standard deviation
SCA	Ionosphere	13.2400	0.1040	0.1055	0.1004	0.0016
	BCW	6.2867	0.1692	0.1739	0.1684	0.0017
	CB	21.3067	0.2917	0.3035	0.2801	0.0065
	Vehicle	7.8133	0.2284	0.2496	0.2184	0.0091
	Parkinson	13.2933	0.3409	0.3409	0.3409	$3.5692\ e^{-07}$
	9_Tumors	$2.2772e^{+03}$	0.5143	0.5258	0.5051	0.0068
	Leukemia2	$3.5778e^{+03}$	0.7537	0.7890	0.7104	0.0260
Binary ABC	Ionosphere	20.9667	0.0648	0.0709	0.0614	0.0028
	BCW	4.1533	0.0753	0.0805	0.0709	0.0031
	CB	29.0800	0.1320	0.1611	0.1150	0.0139
	Vehicle	13.4200	0.0727	0.0896	0.0529	0.0107
	Parkinson	4.3600	0.2322	0.3215	0.0977	0.0836
	9_Tumors	$2.9178e^{+03}$	0.3663	0.3732	0.3596	0.0053
	Leukemia2	$5.5821e^{+03}$	0.3111	0.3357	0.2839	0.0186
HBPSOSCA	Ionosphere	12.2450	0.1436	0.2131	0.1089	0.0309
	BCW	5.7067	0.1770	0.1770	0.1770	$2.9257\ e^{-17}$
	CB	11.7947	0.5342	0.6705	0.3158	0.0922
	Vehicle	2.2318	1.9029	3.0612	0.3531	0.9707
	Parkinson	14.2367	0.3409	0.3409	0.3409	$5.6413\ e^{-10}$
	9_Tumors	$2.2312e^{+03}$	0.6019	0.6878	0.5061	0.0863
	Leukemia2	$6.0090e^{+03}$	0.7923	0.8866	0.6828	0.0835

Table 6 The results recorded with BPSO, C-BPSO, BMFO, BDFA, BWOA, SCA, and Binary ABC with 10 independent runs on seven real-life benchmark datasets in term of DBI

Algorithms	Dataset	Average length of selected features	Average DBI	Best DBI	Worst DBI	Standard deviation
BPSO	Ionosphere	13.7550	0.4076	0.2960	0.4790	0.4843
	BCW	2.0000	0.4046	0.4046	0.4046	0.1357
	CB	8.2914	0.3612	0.2575	0.3996	0.5335
	Vehicle	9.2318	0.2689	0.2584	0.3582	0.2311
	Parkinson	2.6358	0.3513	0.3513	0.3513	0.1865
	9_Tumors	$2.8437e^{+03}$	1.2174	1.3379	1.1403	0.0639
	Leukemia2	$5.5800e^{+03}$	0.4942	0.4852	0.5044	0.0066
Chaotic BPSO	Ionosphere	8.2980	0.8413	0.6997	0.8981	0.0627
	BCW	1.9868	0.4057	0.4046	0.4067	$6.8361e^{-04}$
	CB	21.8013	0.9725	0.9725	0.9725	0
	Vehicle	9.6490	0.2950	0.3184	0.2751	0.0127
	Parkinson	2.5430	0.3513	0.3513	0.3513	$5.2872e^{-08}$
	9_Tumors	$2.9017e^{+03}$	1.0080	0.4935	1.7777	0.4577
	Leukemia2	$5.5783e^{+03}$	0.4946	0.4918	0.4971	0.0017
BMFO	Ionosphere	21.4267	0.2967	0.2521	0.3237	0.0368
	BCW	5.1933	0.3134	0.2881	0.3237	0.0486
	CB	38.0200	1.3745	1.3131	1.4308	0.0361
	Vehicle	11.2867	0.4014	0.3970	0.4094	0.0044
	Parkinson	13.6400	0.3513	0.3513	0.3513	$2.8570e^{-06}$
	9_Tumors	$3.6878e^{+03}$	1.9514	1.9111	1.9819	0.0222
	Leukemia2	$7.2183e^{+03}$	1.5994	1.5648	1.6283	0.0163

Table 6 (continued)

Algorithms	Dataset	Average length of selected features	Average DBI	Best DBI	Worst DBI	Standard deviation
BDFA	Ionosphere	16.8867	22.375	14.742	36.383	7.2495
	BCW	2.0067	0.4046	0.4046	0.4046	0
	CB	20.5400	0.4648	0.4001	0.5826	0.0732
	Vehicle	8.8667	0.2918	0.2637	0.3291	0.0212
	Parkinson	5.0467	0.3513	0.3513	0.3513	$2.5809e^{-10}$
	9_Tumors	$2.8659e^{+03}$	2.4811	2.3381	2.5535	0.0551
	Leukemia2	$6.1308e^{+03}$	4.8836	4.7844	5.0287	0.0630
BWOA	Ionosphere	16.0467	1.9234	1.7631	2.1573	$2.3460e^{-16}$
	BCW	4.0000	0.8782	0.8192	0.9075	$1.1703e^{-16}$
	CB	37.6600	1.9696	1.8445	2.1628	0
	Vehicle	9.0867	1.2479	1.1077	1.4317	$2.3406e^{-16}$
	Parkinson	12.8533	0.9786	0.9171	0.9947	0
	9_Tumors	$4.2436e^{+03}$	2.5855	2.5059	2.7404	0.0690
	Leukemia2	$8.3635e^{+03}$	2.9038	2.7576	3.2958	0.1813
SCA	Ionosphere	4.4400	0.4178	0.3574	0.4780	0.0478
	BCW	2.0000	0.4046	0.4046	0.4083	0.0012
	CB	8.9133	1.3552	1.3107	1.4451	0.0437
	Vehicle	7.1258	0.3659	0.3627	0.3707	0.0022
	Parkinson	2.9333	0.3513	0.3513	0.3513	$3.4508e^{-06}$
	9_Tumors	$1.5022e^{+03}$	1.3520	1.2649	1.4017	0.0468
	Leukemia2	$3.6995e^{+03}$	0.5701	0.5166	0.6938	0.0633
Binary ABC	Ionosphere	17.1133	0.3694	0.2311	0.4223	0.0691
	BCW	2.0000	0.4032	0.3692	0.4094	0.0120
	CB	4.5800	0.5338	0.4805	0.5850	0.0366
	Vehicle	2.8133	0.2755	0.2636	0.2982	0.0107
	Parkinson	2.3533	0.3513	0.3513	0.3513	$2.6403e^{-07}$
	9_Tumors	$2.8847e^{+03}$	1.2358	1.1981	1.3040	0.0334
	Leukemia2	$5.6059e^{+03}$	0.5365	0.5320	0.5405	0.0030
HBPSOSCA	Ionosphere	3.8800	0.2960	0.2960	0.2960	0.0894
	BCW	2.0000	0.3046	0.3046	0.3046	0
	CB	1.1000	0.2614	0.2575	0.2922	0.0466
	Vehicle	6.5828	0.2596	0.2584	0.2629	0.0157
	Parkinson	2.0000	0.3513	0.3513	0.3513	$5.8514e^{-17}$
	9_Tumors	21.3067	0.3425	0.2717	0.3929	0.0403
	Leukemia2	92.4000	0.3243	0.2160	0.4293	0.0716

method is higher than BPSO, BMFO, BDFA, SCA, and binary ABC methods as shown in Fig. 6b and also proposed method achieves a little bit better average accuracy with 87.86% than an average accuracy of C-BPSO and BWOA, i.e., 87.49% and 87.01% respectively. For CB dataset, the proposed method performs best as compared to other methods with 90.41% average accuracy. For Statlog dataset, the performance of BWOA and HBPSOSCA is nearly same up to four iterations. After 4th iterations, performance of the proposed method is higher than BWOA and other method with 84.61% average accuracy as shown in Fig. 6d. For Parkinson dataset, SCA performed better with average accuracy 95.43% with respect to other methods, but the proposed method achieves better average accuracy as shown in Fig. 6e as compared to stated methods with 96.13% average accuracy. Figure 6f shows performance of the proposed method for the 9_Tumors dataset with other methods used in this thesis. This figure describes that proposed method achieves higher performance as compared to other methods with 80.06% average accuracy. For Leukemia2 dataset, the proposed method achieves 90.73% average accuracy which is

Table 7 Wilcoxon test for ionosphere dataset

Optimizers	Wilcoxon test value	Comment
HBPSOSCA—BPSO	1.8267 e^{-04}	Significant
HBPSOSCA—C-BPSO	1.8267 e^{-04}	Significant
HBPSOSCA—BMFO	1.8267 e^{-04}	Significant
HBPSOSCA—BDFA	1.8267 e^{-04}	Significant
HBPSOSCA—BWOA	1.8267 e^{-04}	Significant
HBPSOSCA—SCA	0.0013	Significant
HBPSOSCA—Binary-ABC	1.8267 e^{-04}	Significant

Table 8 Wilcoxon test for BCW dataset

Optimizers	Wilcoxon test value	Comment
HBPSOSCA—BPSO	1.8165 e^{-04}	Significant
HBPSOSCA—C-BPSO	1.8165 e^{-04}	Significant
HBPSOSCA—BMFO	1.8165 e^{-04}	Significant
HBPSOSCA—BDFA	1.8063 e^{-04}	Significant
HBPSOSCA—BWOA	1.8165 e^{-04}	Significant
HBPSOSCA—SCA	4.2880 e^{-04}	Significant
HBPSOSCA—Binary-ABC	1.8165 e^{-04}	Significant

Table 9 Wilcoxon test for CB dataset

Optimizers	Wilcoxon test value	Comment
HBPSOSCA—BPSO	1.8267 e^{-04}	Significant
HBPSOSCA—C-BPSO	1.8267 e^{-04}	Significant
HBPSOSCA—BMFO	1.8267 e^{-04}	Significant
HBPSOSCA—BDFA	1.8267 e^{-04}	Significant
HBPSOSCA—BWOA	1.8267 e^{-04}	Significant
HBPSOSCA—SCA	1.8267 e^{-04}	Significant
HBPSOSCA—Binary-ABC	1.8267 e^{-04}	Significant

Table 10 Wilcoxon test for vehicle dataset

Optimizers	Wilcoxon test value	Comment
HBPSOSCA—BPSO	1.8267 e^{-04}	Significant
HBPSOSCA—C-BPSO	1.8267 e^{-04}	Significant
HBPSOSCA—BMFO	1.8267 e^{-04}	Significant
HBPSOSCA—BDFA	1.8267 e^{-04}	Significant
HBPSOSCA—BWOA	0.0539 e^{-04}	Insignificant
HBPSOSCA—SCA	5.8284 e^{-04}	Significant
HBPSOSCA—Binary-ABC	1.8267 e^{-04}	Significant

Table 11 Wilcoxon test for Parkinson dataset

Optimizers	Wilcoxon test value	Comment
HBPSOSCA—BPSO	2.4480 e^{-04}	Significant
HBPSOSCA—C-BPSO	8.7450 e^{-05}	Significant
HBPSOSCA—BMFO	1.8267 e^{-04}	Significant
HBPSOSCA—BDFA	1.8267 e^{-04}	Significant
HBPSOSCA—BWOA	1.6118 e^{-04}	Significant
HBPSOSCA—SCA	1.8267 e^{-04}	Significant
HBPSOSCA—Binary-ABC	1.8267 e^{-04}	Significant

Table 12 Wilcoxon test for 9_Tumors dataset

Optimizers	Wilcoxon test value	Comment
HBPSOSCA—BPSO	1.8267 e^{-04}	Significant
HBPSOSCA—C-BPSO	1.8267 e^{-04}	Significant
HBPSOSCA—BMFO	1.8267 e^{-04}	Significant
HBPSOSCA—BDFA	1.8267 e^{-04}	Significant
HBPSOSCA—BWOA	6.3864 e^{-05}	Significant
HBPSOSCA—SCA	1.8267 e^{-04}	Significant
HBPSOSCA—Binary-ABC	1.8165 e^{-04}	Significant

Table 13 Wilcoxon test for Leukemia2 dataset

Optimizers	Wilcoxon test value	Comment
HBPSOSCA—BPSO	1.8267 e^{-04}	Significant
HBPSOSCA—C-BPSO	0.0257	Significant
HBPSOSCA—BMFO	1.8267 e^{-04}	Significant
HBPSOSCA—BDFA	1.8267 e^{-04}	Significant
HBPSOSCA—BWOA	6.3864 e^{-05}	Significant
HBPSOSCA—SCA	1.8267 e^{-04}	Significant
HBPSOSCA—Binary-ABC	1.8267 e^{-04}	Significant

comparatively higher than the competitive method as shown in Fig. 6g. A possible reason of this improvement is good exploration and exploitation of the given feature space.

We can see that the score obtained by HBPSOSCA is significantly better compared to other stated methods in Tables 7, 8, 9, 10, 11, 12 and 13. The comparative performance of BPSO, C-BPSO, BMFO, BWOA, SCA, binary ABC, and HBPSOSCA in term of number of selected features is shown in Fig. 7. Here, the proposed method HBPSOSCA selects 6.5882%, 22.2222%, 4.715%, 19.794%, 9.693%, 3.2949%, and 8.3091% of features for Ionosphere, BCW, CB, Vehicle, Parkinson, 9_Tumors, and Leukemia2 datasets, respectively, which is comparatively

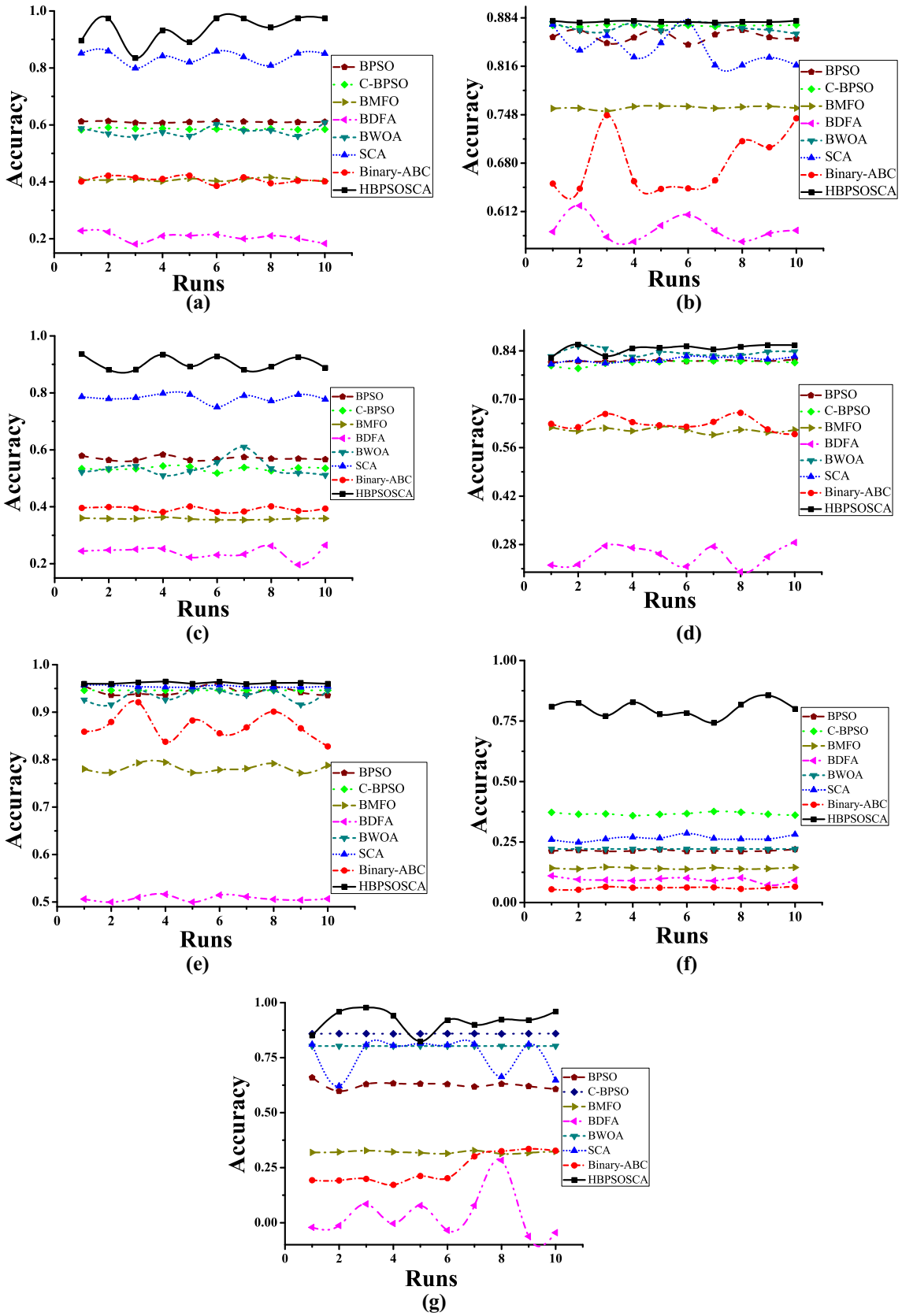

Fig. 6 Comparison of performance over 10 runs for seven datasets **a** Ionosphere, **b** BCW, **c** CB, **d** Vehicle, **e** Parkinson, **f** 9_Tumors, **g** Leukemia2

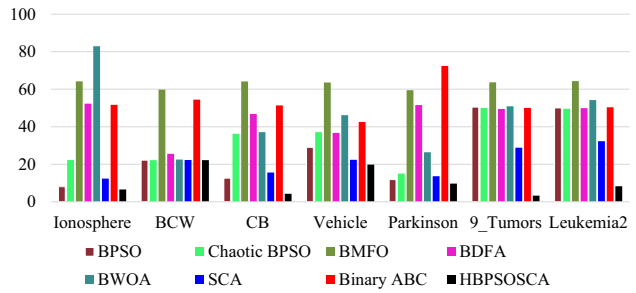

Fig. 7 The comparative performance of all algorithms in terms of features reduction rate

less than the other stated methods. It is clear from the conducted analysis that the HBPSOSCA outperforms the other methods in most of cases.

5 Conclusions and future directions

A feature selection method is used to select an informative subset of features from high dimensional irrelevant, redundant, and noisy feature space. The irrelevant, redundant, and noisy feature not only increases computational complexity but also deteriorate performance of the underlying algorithms. In this paper, a nature-inspired algorithm is used to select an informative subset of features from given feature space. Each algorithm has its own advantages and disadvantage. We introduce a new hybrid method to take advantage of one method and lessen the disadvantage of others for feature selection task. Here, we integrate the BPSO with the SCA, named as HBPSOSCA for this task. The integration of the two algorithms provides global search ability and local exploitation ability to the HBPSOSCA by improving the movement of a particle in the BPSO with the SCA. The proposed algorithm is tested on ten benchmark test functions and seven well-known scientific datasets. Experimental results show that the proposed algorithm obtains the near global minimum compared to other competitive nature-inspired algorithms for most of the test functions. Moreover, it achieves better clustering accuracy compare to the competitive methods for all real-life datasets. The Wilcoxon Test confirms that the results obtained by the HBPSOSCA are significantly better than the competitive methods.

In the future, we intend to combine other feature selection methods with the nature-inspired algorithm to select an informative subset of features from high dimensional space without much increasing computational complexity of the algorithm. As the parameters' value significantly affects performance of the underlying algorithm, we plan to develop a model to adaptively set the parameter's values.

Appendix: Benchmark functions

1. $F1(X) = \sum_{i=1}^{D} X_i^2; -100 \le X_i \le 100$
2. $F2(X) = \sum_{i=1}^{D} |X_i| + \prod_{i=1}^{D} X_i; -10 \le X_i \le 10$
3. $F3(X) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} X_j \right)^2; -100 \le X_i \le 100$
4. $F4(X) = \sum_{i=1}^{D-1} \left[100\left(X_{i+1} - X_i^2\right)^2 + (X_i - 1)^2 \right]; -30 \le X_i \le 30$
5. $F5(X) = \sum_{i=1}^{D} (X_i + 0.5)^2; -100 \le X_i \le 100$
6. $F6(X) = \sum_{i=1}^{D} iX_i^4 + rand[0, 1); -1.28 \le X_i \le 1.28$
7. $F7(X) = \sum_{i=1}^{D} -X_i * \sin\left(\sqrt{|X_i|}\right) + D * 418.98288727243369; -500 \le X_i \le 500$
8. $F8(X) = \sum_{i=1}^{D} \left[X_i^2 - 10\cos(2\pi X_i) + 10 \right]; -5.12 \le X_i \le 5.12$
9. $F9(X) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} X_i^2} \right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos 2\pi X_i \right) + 20 + e; -32 \le X_i \le 32$
10. $F10(X) = \frac{1}{4000}\sum_{i=1}^{D} X_i^2 - \prod_{i=1}^{D} \cos\left(\frac{X_i}{\sqrt{i}}\right) + 1; -600 \le X_i \le 600$

References

Agarwal P, Mehta S (2014) Nature-inspired algorithms: state-of-art, problems and prospects. Int J Comput Appl 100(14):14–21

Agneessens J, Vandoorn T, Meersman B, Vandevelde L (2011) The use of binary particle swarm optimization to obtain a demand side management system. In: IET renewable power generation conference, Proceedings. IET events

Ahmed AA (2005) Ant colony optimization for feature subset selection. WEC 2:35–38

Ali MZ, Awad NH, Suganthan PN, Reynolds RG (2017) An adaptive multipopulation differential evolution with dynamic population reduction. IEEE Trans Cybern 47(9):2768–2779

Alswaitti M, Albughdadi M, Isa NA (2018) Density-based particle swarm optimization algorithm for data clustering. Expert Syst Appl 91:170–186

Bansal JC, Deep K (2012) A modified binary particle swarm optimization for knapsack problems. Appl Math Comput 218(22):11042–11061

Bansal JC, Singh PK, Saraswat M, Verma A, Jadon SS, Abraham A (2011) Inertia weight strategies in particle swarm optimization. In: 2011 Third world congress on nature and biologically inspired computing (NaBIC). IEEE, pp 633–640

Behjat AR, Mustapha A, Nezamabadi PH, Sulaiman MN, Mustapha N (2014) A new binary particle swarm optimization for feature subset selection with support vector machine. In: Recent advances on soft computing and data mining. pp 47–57

Bezdek JC, Pal NR (1995) Cluster validation with generalized Dunn's indices. In: Second New Zealand international two-stream conference on artificial neural networks and expert systems. IEEE, pp 190–193

Blum C (2005) Ant colony optimization: introduction and recent trends. Phys Life Rev 2(4):353–373

Bureerat S, Pholdee N (2017) Adaptive sine cosine algorithm integrated with differential evolution for structural damage detection. In: International conference on computational science and its application, lecture notes computer science (LNCS) 10404, pp 71–86

Cervante L, Xue B, Zhang M, Shang L (2012) Binary particle swarm optimisation for feature selection: a filter based approach. In: 2012 IEEE congress on evolutionary computation (CEC). pp 1–8

Cervantes A, Galván IM, Isasi P (2005) Binary particle swarm optimization in classification. Neural Network World, pp 229–241

Chen YP, Li Y, Wang G, Zheng YF, Xu Q, Fan JH, Cui XT (2017) A novel bacterial foraging optimization algorithm for feature selection. Expert Syst Appl 83:1–17

Chuang LY, Li JC, Yang CH (2008) Chaotic binary particle swarm optimization for feature selection using logistic map. In: Proceedings of the international multi conference of engineers and computer scientists, vol 1

Dash M, Choi K, Scheuermann P, Liu H (2002) Feature selection for clustering-a filter solution. In: 2002 IEEE international conference on data mining, 2002. ICDM 2003. Proceedings. IEEE, pp 115–122

Davies DL, Bouldin DW (1979) A cluster separation measure. IEEE Trans Pattern Anal Mach Intell 2:224–227

Desgraupes B (2013) Clustering indices, vol 1. University of Paris Ouest-Lab ModalX, Paris, p 34

Diao R, Shen Q (2015) Nature inspired feature selection meta-heuristics. Artif Intell Rev 44(3):311–340

Frank A (2010) UCI machine learning repository. Expert Systems with Applications. http://archive.ics.uci.edu/ml

Gene Expression Model Selector. http://www.gems-system.org. 26 Apr 2018

Goldbogen JA, Friedlaender AS, Calambokidis J, Mckenna MF, Simon M, Nowacek DP (2013) Integrative approaches to the study of baleen whale diving behavior, feeding performance, and foraging ecology. Bioscience 63(2):90–100

Hafez AI, Zawbaa HM, Emary E, Hassanien AE (2016) Sine cosine optimization algorithm for feature selection. In: Proceeding of international symposium on innovations in intelligent systems and applications (INISTA), Sinaia, Romania, IEEE Xplore (2016)

Jadon SS, Sharma H, Kumar E, Bansal JC (2011) Application of binary particle swarm optimization in cryptanalysis of DES. In: Proceedings of the international conference on soft computing for problem solving (SocProS). Springer, India, pp 1061–1071

Jain AK, Dubes RC (1988) Algorithms for clustering data. Prentice Hall, Englewood Cliffs

Jain I, Jain VK, Jain R (2017) Correlation feature selection based improved-binary particle swarm optimization for gene selection and cancer classification. Appl Soft Comput 62:203–215

Kaufman L, Rousseeuw PJ (2009) Finding groups in data: an introduction to cluster analysis, vol 344. Wiley, Hoboken

Kaur S, Prashar S (2016) A novel sine cosine algorithm for the solution of unit commitment problem. Int J Sci Eng Technol Res 5(12):3298–3310

Kennedy J (1995) Particle swarm optimization. In: Proceedings of ICNN'95—international conference on neural networks, Perth, WA, Australia, vol 4. pp 1942–1948

Kennedy J, Eberhart RC (1997) A discrete binary version of the particle swarm algorithm. In: 1997 IEEE international conference on systems, man, and cybernetics, 1997. Computational cybernetics and simulation, vol 5. IEEE, pp 4104–4108

Keogh E, Mueen A (2011) Curse of dimensionality. In: Sammut C, Webb GI (eds) Encyclopedia of machine learning. Springer, Boston

Kumar S (2018) A comprehensive review of nature-inspired algorithms for feature selection. In: Handbook of research on modeling, analysis, and application of nature-inspired meta-heuristic algorithms. IGI Global, pp 331–345

Lee S, Soak S, Oh S, Pedrycz W, Jeon M (2008) Modified binary particle swarm optimization. Prog Nat Sci 18(9):1161–1166

Li N, L, G, Deng ZL (2017) An improved sine cosine algorithm based on levy flight. In: Proceedings of SPIE 10420, ninth international conference on digital image processing (ICDIP 2017), 104204R, https://doi.org/10.1117/12.228207

Long HX, Xu WB, Sun J (2009) Binary particle swarm optimization algorithm with mutation for multiple sequence alignment. In: Biology Forum/Rivista di Biologia, vol 102, no 1. pp 75–94

Mafarja M, Mirjalili S (2017) Whale optimization approaches for wrapper feature selection. Appl Soft Comput 62:441–453

Mafarja MM, Eleyan D, Jaber I, Hammouri A, Mirjalili S (2017) Binary dragonfly algorithm for feature selection. In: 2017 international conference new trends in computing sciences (ICTCS). IEEE, pp 12–17

Meshkat M, Parhizgar M (2017) A novel sine and cosine algorithm for global optimization. In: 2017 7th International conference on computer and knowledge engineering (ICCKE). IEEE, pp 60–65

Michaud P (1997) Clustering techniques. Future Gener Comput Syst 13(2–3):135–147

Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl Based Syst 88:228–249

Mirjalili S (2016a) SCA: a sine cosine algorithm for solving optimization problems. Knowl Based Syst 96:120–133

Mirjalili S (2016b) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput Appl 27(4):1053–1073

Mirjalili S, Lewis A (2013) S-shaped versus V-shaped transfer functions for binary particle swarm optimization. Swarm Evolut Comput 9:1–14

Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67

Mohamad MS, Omatu S, Deris S, Yoshioka M (2011) A modified binary particle swarm optimization for selecting the small subset of informative genes from gene expression data. IEEE Trans Inf Technol Biomed 15(6):813–822

Prakash J, Singh PK (2012) An effective hybrid method based on de, ga, and k-means for data clustering. In: Proceedings of the second international conference on soft computing for problem solving (SocProS 2012). Springer, New Delhi, pp 1561–1572

Raitoharju J, Samiee K, Kiranyaz S, Gabbouj M (2017) Particle swarm clustering fitness evaluation with computational centroids. Swarm Evolut Comput 34:103–118

Rajamohana SP, Umamaheswari K (2017) Hybrid optimization algorithm of improved binary particle swarm optimization (iBPSO) and cuckoo search for review spam detection. In: Proceedings of the 9th international conference on machine learning and computing. ACM, pp 238–242

Reddy S, Panwar LK, Panigrahi BK, Kumar R (2017) Solution to unit commitment in power system operation planning using binary coded modified moth flame optimization algorithm (BMMFOA): a flame selection based computational technique. J Comput Sci 25:298–317

Rizk-Allah RM (2017) Hybridizing sine cosine algorithm with multi-orthogonal search strategy for engineering design problems. J Comput Des Eng 5:249–273 (**in Press, accepted for publication**)

Saremi S, Mirjalili S, Lewis A (2015) How important is a transfer function in discrete heuristic algorithms. Neural Comput Appl 26(3):625–640

Shang YW, Qiu YH (2006) A note on the extended Rosenbrock function. Evolut Comput 14(1):119–126

Singh N, Singh SB (2017) Hybrid algorithm of particle swarm optimization and grey wolf optimizer for improving convergence performance. J Appl Math. https://doi.org/10.1155/2017/2030489

Srivastava MS, Joshi MN, Gaur M (2014) A review paper on feature selection methodologies and their applications. IJCSNS 14(5):78

Tawhid MA, Dsouza KB (2018) Hybrid binary bat enhanced particle swarm optimization algorithm for solving feature selection problems. Appl Comput Inf. https://doi.org/10.1016/j.aci.2018.04.001

Turgut OE (2017) Thermal and economical optimization of a shell and tube evaporator using hybrid backtracking search-sine-cosine algorithm. Arab J Sci Eng. https://doi.org/10.1007/s13369-017-2458-6

Wu H, Sun Y, Peng L (2010) Binary particle swarm optimization algorithm for control of single-phase full-bridge inverter. In: Power and energy engineering conference (APPEEC), Asia-Pacific. IEEE, pp 1–4

Xu D, Tian Y (2015) Ann. Data. Sci. 2:165. https://doi.org/10.1007/s40745-015-0040-1

Xue B, Zhang M, Browne WN (2013) Particle swarm optimization for feature selection in classification: a multi-objective approach. IEEE Trans Cybern 43(6):1656–1671

Yang XS (2014) Nature-inspired optimization algorithms. Elsevier, Amsterdam

Yang J, Honavar V (1998) Feature subset selection using a genetic algorithm. In: Feature extraction, construction and selection. Computer Science Technical Reports, Springer, Boston, MA, pp 117–136

Yang XS, Lee S, Lee S, Theera-Umpon N (2015) Information analysis of high-dimensional data and applications. Math Probl Eng. https://doi.org/10.1155/2015/126740

Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evolut Comput 3(2):82–102