Applied Soft Computing Journal 93 (2020) 106341

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

Introducing clustering based population in Binary Gravitational Search Algorithm for Feature Selection

Ritam Guha [a], Manosij Ghosh [a,*], Akash Chakrabarti [a], Ram Sarkar [a], Seyedali Mirjalili [b]

[a] Computer Science and Engineering Department, Jadavpur University, 188, Raja S.C. Mallick Road, Kolkata 700032, West Bengal, India
[b] Institute of Integrated and Intelligent Systems, Griffith University, Nathan, Brisbane, QLD 4111, Australia

ARTICLE INFO

Article history:
Received 23 October 2018
Received in revised form 8 December 2019
Accepted 25 April 2020
Available online 8 May 2020

Keywords:
Gravitational search algorithm
Feature selection
Initial population clustering
UCI dataset

ABSTRACT

Feature Selection (FS) is an important aspect of knowledge extraction as it helps to reduce dimensionality of data. Among the numerous FS algorithms proposed over the years, Gravitational Search Algorithm (GSA) is a popular one which has been applied to various domains. However, GSA suffers from the problem of pre-mature convergence which affects exploration leading to performance degradation. To aid exploration, in the present work, we use a clustering technique in order to make the initial population distributed over the entire feature space and to increase the inclusion of features which are more promising. The proposed method is named Clustering based Population in Binary GSA (CPBGSA). To assess the performance of our proposed model, 20 standard UCI datasets are used, and the results are compared with some contemporary methods. It is observed that CPBGSA outperforms other methods in 12 out of 20 cases in terms of average classification accuracy.

The relevant codes of the entire CPBGSA model can be found in the provided link: https://github.com/ManosijGhosh/Clustering-based-Population-in-Binary-GSA.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

Technological advances in recent times have fueled an increase in application of machine learning in a variety of domains. Classification, an import aspect of machine learning, is being widely adopted. In an effort to improve the classification accuracy, many features descriptors are used for feature extraction. With the increase in feature dimension, interpretation and use of features to accomplish any task become cumbersome due to the curse of dimensionality. As a result, several sophisticated techniques are employed in order to efficiently extract useful features. These techniques use various characteristics of datasets for feature extraction. Amidst this huge pool of features, not all are useful or relevant. These unnecessary and misleading pieces of information make the classification models inefficient and in turn degrade the performance of the employed methods both in terms of accuracy as well as computation time. This makes direct feeding of data (or features) to classification models an improper practice and hence, data pre-processing becomes a pre-requisite. One well-regarded pre-processing technique is Feature Selection (FS) which searches for an optimal subset of features from the entire feature vector under consideration; thereby increasing the efficiency of the classification model.

Searching for an optimal subset of features becomes a vital issue in FS problems. The main objective when selecting these features is to find an optimal set of N features from the total set of M features where $N \leq M$, such that there is no degradation in classification performance. The brute-force solution of this problem is to exhaustively generate all possible feature subsets and selecting the one with the best performance [1]. However, this approach is impractical for large datasets because for a dataset with M features, 2^M solutions need to be generated and evaluated. This results in an extremely computationally expensive solution with exponential growth.

Another strategy is to search for this optimal feature subset randomly, or using heuristic information about the search space [2–4] to guide the search. Unlike the brute-force approach, a heuristic strategy does not guarantee a globally optimal feature subset. But it usually finds a near optimal solution within a reasonable amount of time, resulting in acceptable solutions for most practical problems. Meta-heuristics are a class of general-purpose heuristic algorithms which are applicable to a wide range of problems including FS. In recent years, a number of meta-heuristic algorithms [5–9] inspired by the interaction and behavior of physical and biological systems in nature have been proposed in the literature. They exhibit satisfactory performance when applied to FS problems.

* Corresponding author.
E-mail addresses: ritamguha16@gmail.com (R. Guha), manosij1996@gmail.com (M. Ghosh), chkaksh22@gmail.com (A. Chakrabarti), raamsarkar@gmail.com (R. Sarkar), seyedali.mirjalili@griffithuni.edu.au (S. Mirjalili).

https://doi.org/10.1016/j.asoc.2020.106341
1568-4946/© 2020 Elsevier B.V. All rights reserved.

Since no one algorithm can be found to be best for FS on all kinds of datasets, a number of meta-heuristic algorithms have been proposed by the researchers over time. Methods for FS can be of two types namely, single entity-based and population-based. The former revolves around improvement of a single feature subset over the iterations, while the latter improves the results of a set of feature subsets. Most of the population-based FS methods use many candidate solutions which interact with each other globally as well as locally to find the optimal subset of features. While searching, each candidate uses history (positions visited in previous iterations) along with the information gained from the others to improve its own accuracy. Single entity based FS approaches include Simulated Annealing (SA) [2], Hill climbing [10], etc and population-based approaches include Genetic Algorithm (GA) [7], Particle Swarm Optimization (PSO) [6], Gravitational Search Algorithm (GSA) [8], etc.

FS algorithms can also be divided based on the way they evaluate features. Filter methods perform FS based on the intrinsic properties derived from the data. They mostly assess statistical (e.g. chi-square [11]), distance (e.g. relief [12]) or entropy (e.g. symmetrical uncertainty) aspects of the features. Despite being fast, they are unable to provide high accuracy due to the absence of a learning model or a classifier. Wrapper methods, on the other hand, employ a classifier and are much more useful in generating highly accurate feature subsets. They are widely used and some of the most popular wrapper methods are GA [7], PSO [6], Ant Colony Optimization (ACO) [5] and GSA [13] to name a few. Hybrid methods [14,15] use both filter and wrapper methods, but a suitable combination of filter and wrapper methods is challenging as well as application-specific at times. Although wrapper methods are computationally expensive, they improve accuracy to a significant extent when compared to their filter counterparts. Hence, in this paper, we have proposed a wrapper-based meta-heuristic called Clustering based Population in Binary Gravitational Search Algorithm (CPBGSA). To ensure that particles explore the search space a better way, the initial population taken in Binary GSA (BGSA) is modified so that it becomes more distributed over the search space. Enhancement in explorability of search processes can be attempted through this initial population initialization technique. Moreover, since this is a generalized procedure, it is can be amalgamated with any other wrapper-based FS problem. However, to show the utility of this method, we have chosen GSA.

GSA was first proposed in 2009 by Rashedi et al. [8] as a general-purpose optimizer. In 2010 [13], it was employed to solve FS problems. GSA uses the concept of Newtonian law of gravitational forces among heavenly bodies. According to this theory, every body attracts every other body in the universe with a force which is dependent on their masses and the distance between their centers. When applied to FS, each body becomes a candidate solution which attracts other solutions depending on various parameters under consideration. The force of attraction generates some velocity in the candidates allowing them to move in the search space. Thus, this concept became applicable to FS. The problem with this approach is that whenever a candidate becomes a good solution, it produces a large force of attraction and converges rapidly. Due to this high convergence rate, candidates of GSA cannot explore the search space properly i.e. it suffers from pre-mature convergence. To avoid this problem we propose CPBGSA, which starts with a population which is more distributed over the search space. This helps in diversifying the initial candidate solutions so that GSA can achieve the required exploration of the search space.

The rest of the paper is organized as follows: Section 2 provides a literature review of meta-heuristics and FS methods. The CPBGSA is detailed in Section 3. Section 4 presents and discusses the results. Finally, Section 5 concludes the work and suggests some future plans.

2. Related work

The nature-inspired evolutionary algorithms can be broadly classified into three categories based on their sources of inspiration: physical phenomenon-based algorithms (e.g. GSA [8], SA [2], etc.), Evolutionary algorithms (EA) (e.g. GA [16], Differential Evolution (DE) [17], etc.), and Swarm-based algorithms (e.g. PSO [6], ACO [5], Artificial Bee Colony (ABC) Algorithm [18], Grey Wolf Optimizer (GWO) [19], etc.).

GA [16] was one of the first EAs proposed in the literature mimicking the natural process of evolution through reproduction and natural selection. Various GA approaches have been proposed over the years to deal with the FS problems [7,20,21]. GA has been applied in various domains of pattern recognition like handwritten digit recognition [21], biomedical data [22], handwritten word recognition [23], etc. Memetic Algorithm (MA) [24] is a variation of GA where a local search is added to enhance the exploitation ability of GA. This EA has also been applied to various pattern recognition problems [25,26]. In general, such EAs do not possess good balance of exploitation and exploration as compared to swarm-based algorithms. This is primarily because EAs change the population at the same rate.

An interesting subclass of the population-based meta-heuristic algorithms is the swarm-based algorithms, commonly known as Swarm Intelligence (SI) algorithms. These algorithms are inspired from and imitate the social behavior of swarms, flocks and herds in nature. Unlike the EAs, the SI algorithms utilize the information gathered from the search space over the previous iterations to guide their search.

One of the most popular SI techniques is the PSO algorithm, which was first proposed by Kennedy and Eberhart in 1995 [6]. Here, each solution is thought of as a particle characterized by its position, fitness and velocity. To deal with binary optimization problems, a binary version of the PSO algorithm (BPSO) was proposed in [27]. PSO has been effectively applied to FS problem in [28]. A hamming distance based BPSO algorithm (HDBPSO) for FS on high dimensional datasets was proposed in [29]. Some of the reasons for the popularity of PSO and other SI algorithms in general, are the requirement for tuning fewer parameters and also employing fewer operators as opposed to elitism, crossover and mutation of EAs. PSO, however, converges only to positions the swarm can reach i.e. in many cases PSO converges to a local optimum [30]. Another limitation of PSO when applied to the FS problem is that the personal and global best updating mechanisms may miss some feature sets with small number of features and high classification accuracy [31].

Binary Bat Algorithm (BBA) [32] creates binary solutions mimicking the use of ultrasonic sounds by bats to locate prey (here best solution). A v-shaped transfer function is used to allow the flipping of bit values when velocities are high. This allows for a balanced combination of extensive local search capacity and search ability of PSO. However, since only the best bat is used to direct exploration, this hampers exploration capacity. In Binary Grey Wolf Optimization (BGWO) [19], the three best wolves are used to direct exploration. The sigmoid function is used to convert the continuous values to a probability distribution for determining the state of a feature. However, use of the whole population for exploration gives added information about the search space, leading to a more informed search.

BGSA [13] uses the *tanh* function for flipping the state of a feature. So, if the velocity is high it calls for a change of the value representing the feature (‘0’ becomes ‘1’ or vice versa). While this is applied to discrete functions, the same algorithm has been applied to FS in [31]. GSA frequently stagnates and to counter this in [33] the transfer function is modified to have a minimum output value which depends on number of times the

best particle remains unchanged in consecutive iterations. Also, an elitism strategy is used to allow for better retention of good particles in the population and a normalized hamming distance is used for better distance estimation.

Operators have also been suggested to enhance exploration and exploitation in GSA such as the disruption operator [34]. Here, the ratio of the distance from a particle to its nearest neighbor and the distance to the best particle is measured. If this ratio is less than a threshold, either exploration (if the particle is far away from the best) or exploitation (if the particle is close to the best) is performed. Another approach to enhance exploitation is to include the best particle's (*gbest*'s) position in the velocity equation to accelerate movement towards the optimum point [35]. Similarly, in [36] both the *gbest* and the best position of the particle in its history (*pbest*) are used in the velocity equation. Therefore, GSA is enhanced by borrowing the style of movement of particles from PSO and combining that with its own movement style during the calculation of particle velocity.

A few different approaches have also been proposed in the literature. In Quantum GSA [37], the particles are represented with a complex number having unit modulus and the value of the angle varies. The velocity of the particles is considered equal to angular velocity which provides the change in angles of the quantum state of the particles. The square of the cosine of the angle is the probability of the state of the feature being '1'. A hybrid approach is reported in [38] where after performing BGSA, Mutual Information-based FS (MIFS) is used to further refine the feature subsets.

For quick reference, a tabular summary of the above-mentioned nature-inspired evolutionary algorithms is provided in Table 1.

3. Proposed CPBGSA algorithm

Any population-based algorithm, like GSA, PSO or ACO, introduces a random initial population from where they start their search for the optimal feature subset. But if the random initial population is not generated properly, it may lead to poor exploration of the search space. Moreover, GSA has a higher convergence rate compared to other algorithms like PSO or ACO. GSA suffers from premature convergence and to avoid this, we have made an attempt to aid exploration by creating a more distributed initial population. In GSA, each point moves towards the best points. The movement is synonymous to the movement of particles in space. The mass of each particle is defined using the fitness of that particle. In BGSA [13], the particles are encoded by binary strings of length m where each '1' ('0') represents inclusion (exclusion) of the corresponding feature in the feature subset, respectively. The number of features in the population is denoted by m. The particles (x_i) are also referred to as agents. The flowchart showing the steps of the proposed FS model is depicted in Fig. 1.

Fitness function used in case of any wrapper-based FS is generally a learning model or a classifier and fitness value is the ability of the particle (features selected by the particle) to predict the classes i.e. classification accuracy of the feature set fed to the classifier. BGSA refers its candidate solutions as masses. Each mass, therefore, has a fitness value and BGSA tries to optimize these fitness values. The mass of the ith particle at time t ($mass_i(t)$) is assigned based on the fitness values using Eq. (1).

$$mass_i(t) = \frac{fitness_i\,(t) - minimum(t)}{maximum\,(t) - minimum(t)} \tag{1}$$

The maximum fitness value ($maximum\,(t)$) and minimum fitness value ($minimum(t)$) of the population at time t are used to normalize the masses. The masses are re-calculated after one unit of time (t). From the value of $mass_i(t)$, the value of $Mass_i(t)$ is calculated by dividing $mass_i(t)$ by sum of all masses as shown in Eq. (2) where n represents total number of particles. The masses represent the goodness of a particle in BGSA. Therefore, as t increases, BGSA tries to improve (increase) the masses assigned to each candidate.

$$Mass_i\,(t) = \frac{mass_i(t)}{\sum_{j=1}^{n} mass_j(t)} \tag{2}$$

One mass applies force on other masses, where the force (force of jth particle on ith particle - $F_{ij}(t)$) is a vector of size m. The force has a component for every feature. The kth position of the vector stores the force exerted on the kth feature. Therefore, $F_{ij}^k(t)$ denotes the force exerted on the kth feature of the ith particle by the jth particle and is calculated using Eq. (3). The distance between two particles ($dist\,(x_i, x_j)$) is the hamming distance between them where x_i and x_j denote current positions of ith and jth particles respectively in the search space.

$$F_{ij}^k\,(t) = G\,(t) * \frac{Mass_i\,(t) * Mass_j\,(t)}{dist\,(x_i, x_j)} * (x_j^k(t) - x_i^k\,(t)) \tag{3}$$

Where $G(t)$ refers to the gravitational constant which is calculated using Eq. (4).

$$G\,(t) = e^{\frac{\Omega * t}{total\,time}} \tag{4}$$

where the value of Ω is taken as -20, t is the current time while *total time* is the maximum time till which the algorithm runs.

The net force on the ith particle's kth feature denoted as $F_i^k\,(t)$ is calculated using Eq. (5). Here $random_j$ is random number in the range of 0 to 1 both inclusive.

$$F_i^k\,(t) = \sum_{j=1,\ j\neq i}^{m} random_j * F_{ij}^k\,(t) \tag{5}$$

Following the rules of Physics, the velocity of the ith particle's kth feature is affected by the force being applied on it. The new velocity is calculated using Eq. (6).

$$v_i^k\,(t+1) = random_j * v_i^k\,(t) + \frac{F_i^k\,(t)}{Mass_i\,(t)} \tag{6}$$

For FS, the velocity of a feature is interpreted as the probability of changing state of that feature (whether feature is included in subset - '1' or excluded - '0') in a particle. If velocity for a feature is high, that means the state of that feature in that particle is different from that of the better particles (at that time) and therefore the value needs to be changed. Eq. (7) calculates the probability of flipping using the velocity values. If a random number generated is less than the value of probability, the corresponding feature state is flipped i.e. a '1' becomes a '0' and vice versa. It should be noted that in this step, sigmoid function is widely utilized whose problems were pointed out in [39]. So, instead of that we have used *tanh function* to overcome those problems.

$$probability = \tanh(v_i^k\,(t+1)) \tag{7}$$

BGSA has a high rate of convergence and therefore it is possible for GSA to get stuck in local optima. Moreover, if a particle achieves high fitness value, it exerts enormous force on other particles, thereby attracting all other particles towards itself. Due to this massive external influence, the particles start converging till all particles have similar values i.e. till all of them represent very similar feature sets. This harms the exploration ability of BGSA severely. To prevent the occurrence of this problem, the key would be selection of an initial population which is sufficiently distributed.

To avoid premature convergence of the candidate solutions, p random initial particles are created. Then the number of clusters

Table 1
Summary of some related nature-inspired evolutionary algorithms.

Meta-heuristic FS methods	Algorithm	Variants and Application Domains	Reference(s)
Physical phenomenon-based algorithms	GSA	BGSA for binary optimization problems	[13], [31]
		Quantum GSA or QGSA for feature selection	[37]
		Hybrid of BGSA and mutual information for feature selection in intrusion detection systems	[38]
	SA	Various application for SA	[2]
Evolutionary algorithms	GA	GA for feature selection	[7,20,21]
		GA for classification of biomedical data	[22]
		GA for handwritten word recognition	[23]
		Memetic Algorithm or MA for PR	[24], [25], [26]
	DE	Application of fuzzy theory in numeral function optimization	[17]
Swarm-based algorithms	PSO	Binary PSO or BPSO for binary optimization problems	[27]
		PSO for FS	[28]
		Hamming distance based BPSO or HDBPSO for FS on high dimensional datasets	[29]
	ACO	Various applications of ACO in routing, assignment, scheduling, etc.	[5],
	ABC	Numeral function optimization	[18]
	GWO	BGWO for binary optimization problems	[19],

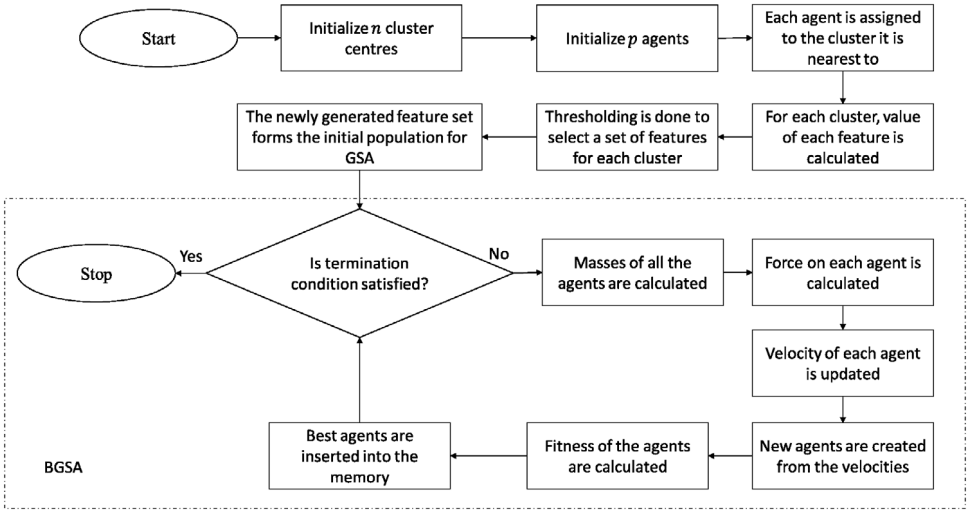

Fig. 1. Flowchart of the proposed FS model – CPBGSA.

n is calculated using Eq. (8). The clustering procedure starts by creating n cluster centers which are randomly created candidate solutions. Then the similarity (defined by Eq. (9)) of each particle with the cluster centers are calculated and each particle is allocated to the center for which it has maximum similarity. There are two terms in the expression of the similarity. The first term is the inverse of the hamming distance of a particle from a cluster center. The second term is the inverse of the difference in accuracies of the particle and a cluster center. The first term represents how easily a particle can be brought to the cluster center and the second term represents the similarity between the classification abilities of the current particle and cluster center. Both of these terms are used with some weightages to generate the similarity of a particle to all the cluster centers.

$$n = \lfloor \alpha * p \rfloor; \tag{8}$$

$$S_i = \beta * (1/H_d) + \eta * (1/D_a) \tag{9}$$

Where α is a value in $(0, 1)$, p is the number of particles used to generate the initial population, H_d represents the hamming distance, D_a shows the difference in accuracies (between particle and cluster center) and S_i shows the similarity of a cluster center to an agent. β and η represent the weightage of the two terms present in the expression of S_i where $\beta = 1 - \eta$. In order to calculate D_a, a classifier is used to compute the classification accuracies of every particle and cluster center. Then D_a is computed using the differences in the classification accuracies between every particle and every cluster center.

Each cluster now has a set of particles allocated to it. The next step is to generate a feature set from each cluster. It can be assumed that better features lead to a rise in accuracy. Suppose we have q particles in a cluster. Considering the dth cluster, the features in the particles of that cluster are evaluated to select the

useful ones. Eq. (10) is proposed to measure goodness measure (h_i^d is how good the ith feature of the dth cluster is). The best features are the features whose goodness values are greater than that of the mean of all goodness measures in a cluster.

$$h_i^d = \sum_{j=1}^{q} k_{ji}^d \times Acc_j^d \tag{10}$$

where k_{ji}^d is position of jth particle in dth cluster and Acc_j^d represents accuracy of the jth particle in dth cluster.

The entire steps of our proposed FS model, namely CPBGSA are listed hereafter with a suitable example presented in Tables 2–5:

1. Randomly create a set of n cluster centers and compute the fitness of each cluster center using a classifier. Consider Table 2 which contains the representation and fitness (i.e. classification accuracy) values of 4 randomly created cluster centers.
2. Initialize a random population (of size p) and find the fitness of each particle. Table 3 contains representation and fitness values of a population of 10 agents (randomly initialized).
3. Find the hamming distance between the cluster centers and each particle. For each particle, similarity (S_i) to the cluster center is calculated. The distance is defined in Eq. (9), where i varies from 1 to n. One field in Table 4 contains hamming distance values between each agent and each cluster center.
4. Each particle is assigned to a cluster from which S_i is maximum. The assignment of each agent to one of the cluster centers is shown in Table 4.
5. Each cluster now contains a related set of particles. In each cluster, we proceed to assign a value (h_i) to the ith feature using the Eq. (10).
6. For each cluster, a threshold (cut-off) is calculated which is the mean of the values assigned to the features as h_i.
7. In each cluster, the features whose h_i values are above cut-off are taken into a new particle i.e. if the ith feature has value greater than cut-off then the ith position for the particle is made 1, 0 otherwise. For each cluster, the final cluster centers derived via histogram are represented in Table 5.
8. Now we have a set of n particles which is then used as the initial population for BGSA. BGSA is then performed to optimize the initial population.

The method takes some random search agents as its input and produces the final agents by application of clustering and GSA. The shapes of clusters are totally dependent on how different agents or candidate solutions scattered over the search space are assigned to specific clusters.

Most of the population based meta-heuristic algorithms start with a random initial population and hence they suffer from the problem of premature convergence. The main problem lies in the randomness of the procedures. Random procedures may or may not produce expected results every time. But our proposed model tries to reduce the degree of randomness by converting the random initial population to a guided one. The proposed model selects its initial population from different portions of the search space which increases its exploration ability. Hence, theoretically CPBGSA is expected to give better results than other state-of-the-art algorithms which use unguided random initial population.

4. Results and analysis

This section contains all the experimental outcomes obtained by the proposed model and its comparison with other well-established FS techniques. Here, experimentations are done on a machine with 4 GB RAM and Intel Core-i3 (5th gen.) processor and MATLAB as programming platform.

4.1. Dataset description

We have evaluated our FS model on 20 UCI datasets which are publicly available [40]. We have classified the datasets into 3 categories — small, medium and large depending on the number of features present therein. The description of the datasets is provided in Table 6. We have selected 2 small, 15 medium and 3 large datasets for the evaluation of our model.

Brief descriptions of the popular datasets (2 small, 10 medium and 3 large datasets) are given below:

(1) Small Datasets:

- BreastCancer: The Original Wisconsin Breast Cancer dataset is widely used for classification of breast cancer. The goal is to distinguish between two classes of breast cancer — benign and malignant and for that it uses attributes such as clump thickness, uniformity of cell size and shape, etc.
- Tic-tac-toe: The Tic-tac-toe Endgame dataset encodes the complete set of possible board configurations at the end of a tic-tac-toe game. The aim is to distinguish between two outcomes – win or loss – for the player who made the first move. There are nine attributes, each corresponding to one of the nine squares of the board, representing the state of that square at the end of a game.

(2) Medium Datasets:

- BreastEW: The goal is to distinguish between two classes of breast cancer diagnosis — benign and malignant. The features are extracted from a digitized image and they describe the characteristics of the cell nuclei present in the image. It is called the Diagnostic Wisconsin Breast Cancer dataset.
- CongressEW: The Congressional Voting Records dataset contains information about the votes for each of the U.S. House of Representatives Congressmen based on different key issues. The attributes are categorical. The goal is to classify each instance into either a democrat or a republican.
- HeartEW: The target is to predict the presence or absence of heart disease in an individual based on certain characteristics represented by the attributes such as age, sex, chest pain type, etc. It is also called the Statlog (Heart) dataset.
- IonosphereEW: The Ionosphere dataset is used for the binary classification of radar returns from the ionosphere into 'good' returns, which show evidence of some sort of structure in the ionosphere, and 'bad' returns, which do not.
- Lymphography: Lymphography refers to the use of X-rays for visualizing the body's lymphatic system. This is a multi-class dataset. The aim is to divide the data into four classes, namely, normal, metastases, malign lymph and fibrosis.
- Zoo: This dataset contains information about the characteristic features of various animals. The target is to distinguish between seven classes of animals based on the given boolean attributes, indicating the presence or absence of various features such as hair, feathers, milk, etc.
- WineEW: The wine dataset encodes the results of a chemical analysis of wines to determine their origins. They are grown in the same region but have different origins and the goal is to distinguish the data with respect to their origin based on the quantities of some key constituents found in each instance.

Table 2
Details of 4 random cluster centers used in worked out example.

Cluster center #	Representation of cluster centers										Classification accuracy
1	1	1	0	0	0	1	0	0	1	1	0.9599
2	1	1	1	0	0	1	1	0	1	0	0.9766
3	1	0	0	1	0	0	0	0	1	1	0.9599
4	1	0	1	0	1	0	0	0	1	1	0.9732

Table 3
Details of initial agents of the population in the example.

Agent #	Representation of initial population										Classification accuracy
1	1	0	0	1	1	0	1	1	1	0	0.9833
2	1	1	1	1	1	0	1	0	1	0	0.9799
3	1	0	1	1	0	0	1	0	1	1	0.9732
4	0	1	1	1	1	1	0	1	1	0	0.9766
5	1	1	1	1	1	0	0	0	1	0	0.9732
6	1	0	1	1	1	0	1	0	0	1	0.9732
7	0	1	0	1	1	1	0	1	1	0	0.9766
8	1	1	1	0	1	1	1	1	0	0	0.9833
9	1	0	1	1	1	1	0	0	0	0	0.9732
10	0	1	1	1	0	0	1	1	1	1	0.9833

Table 4
Values of hamming distance, difference in classification accuracy and S_i of every agent corresponding to each cluster center in the example.

Agent #	Cluster center number	Hamming distance (H_d)	Difference in classification accuracy (D_a)	Similarity measure (S_i)	Cluster assignment
1	1	7	0.024	12.4	Cluster 2
	2	6	0.0077	39.1	
	3	4	0.024	12.46	
	4	5	0.011	27.33	
2	1	6	0.021	12.36	Cluster 2
	2	3	0.004	69.29	
	3	5	0.021	14.38	
	4	4	0.008	39.19	
3	1	5	0.014	21.005	Cluster 4
	2	4	0.0043	69.23	
	3	2	0.014	21.22	
	4	3	0.001	300.233	
4	1	6	0.018	17.04	Cluster 2
	2	5	0.001	300.14	
	3	7	0.017	17.028	
	4	6	0.0043	69.17	
5	1	5	0.014	21.005	Cluster 4
	2	4	0.0043	69.228	
	3	4	0.014	21.0403	
	4	3	0.001	300.233	
6	1	7	0.0144	20.9653	Cluster 4
	2	6	0.0043	69.17	
	3	4	0.0144	21.0403	
	4	3	0.001	300.233	
7	1	5	0.0172	17.07	Cluster 2
	2	6	0.001	300.116	
	3	6	0.0172	17.044	
	4	7	0.0043	69.153	
8	1	6	0.0244	12.406	Cluster 2
	2	3	0.0077	39.2502	
	3	9	0.024	12.37	
	4	6	0.011	27.31	
9	1	6	0.0144	20.982	Cluster 4
	2	5	0.0043	69.193	
	3	5	0.0143	21.0053	
	4	4	0.001	300.175	
10	1	6	0.0244	12.406	Cluster 2
	2	5	0.0077	39.16	
	3	5	0.0244	12.43	
	4	6	0.011	27.31	

Table 5
Representation of agents present in each cluster and final agents produced via histogram for each cluster.

Cluster #	Agents present in the cluster										Final cluster center derived via histogram									
1	1	1	0	0	0	1	0	0	1	1	1	1	0	0	0	1	0	0	1	1
	1	1	1	0	0	1	1	0	1	0										
	1	0	0	1	1	0	1	1	1	0										
	1	1	1	1	1	0	1	0	1	0										
2	0	1	1	1	1	1	0	1	1	0	0	1	0	1	1	0	1	1	1	0
	0	1	0	1	1	1	0	1	1	0										
	1	1	1	0	1	1	1	1	0	0										
	0	1	1	1	0	0	1	1	1	1										
3	1	0	0	1	0	0	0	0	1	1	1	0	0	1	0	0	0	0	1	1
	1	0	1	0	1	0	0	0	1	1										
	1	0	1	1	0	0	1	0	1	1										
4	1	1	1	1	1	0	0	0	1	0	1	0	1	1	1	0	0	0	0	0
	1	0	1	1	1	0	1	0	0	1										
	1	0	1	1	1	1	0	0	0	0										

Table 6
Description of 20 UCI datasets used for evaluation of CPBGSA.

Type	Dataset	Number of features	Number of instances	Number of classes
Small (No. of features $<$ 10)	BreastCancer	9	699	2
	Tic-tac-toe	9	958	2
Medium (10 $\leq$ No. of features $<$ 100)	BreastEW	30	569	2
	CongressEW	16	435	2
	Exactly	13	1000	2
	Exactly2	13	1000	2
	HeartEW	13	270	2
	IonosphereEW	34	351	2
	KrVsKpEW	36	3196	2
	Lymphography	18	148	4
	m-of-n	13	1000	2
	Vote	16	300	2
	Zoo	16	101	7
	WineEW	13	178	3
	Waveform	40	5000	3
	SpectEW	22	267	2
	SonarEW	60	208	2
Large (No. of features $\geq$ 100)	PenglungEW	325	73	7
	Arrhythmia	279	452	16
	Madelon	500	4400	2

- Waveform: he Waveform dataset is used for the purpose of classifying waves into three waveform domains based on certain noisy attributes.
- SpectEW: The SPECT Heart dataset classifies each patient into two categories — normal and abnormal, based on features extracted from the analysis of cardiac Single Proton Emission Computed Tomography (SPECT) images.
- SonarEW: The Connectionist Bench (Sonar, mines vs. Rocks) dataset encodes various patterns which are obtained by bouncing sonar signals off a metal cylinder and rocks at various angles and under various conditions. The goal is to discriminate between these sonar signals and classify them into two classes.

(3) Large Datasets:

- PenglungEW: This dataset is a large dataset with 325 attributes and 73 samples. It is also a multiclass dataset with 7 classes. It has been widely used as a large dataset to test feature selection algorithms.
- Arrythmia: This dataset is also multiclass. The aim is to distinguish between the presence and absence of cardiac arrhythmia and to classify the results into 16 classes.
- Madelon: This is an artificial dataset which is used for binary classification purpose. The data points are grouped into 32 clusters corresponding to the vertices of a five-dimensional hypercube and randomly labeled $+1$ or -1. Each dimension represents a feature and to these 15 linear combinations of these features were added to get a total of 20 features, which are then used to classify the data into two classes corresponding to the +1 or -1 labels.

4.2. Parameter selection

- Our proposed method mainly has three parameters for which we need to set the optimal values and those are — population size, number of iterations, the value of β, η *and* α. We have varied all the parameters while experimentation and finally we have found an optimal set of values for the three parameters. The obtained optimal set is-
- Population size — 30
- Number of iterations — 20
- α – 0.4
- β – 0.7
- η – 0.3

For rest of the experimentations, we have used this set of values for the parameters. The detailed parameter variation results are presented in a graphical manner in Figs. 2–17. Figs. 2–6 represent results for parameter variations in small datasets. Figs. 7–12 show the same for medium datasets and Figs. 13–17 for large datasets. Sections 4.3 and 4.4 explain the experimental results obtained by parameter variation in detail. This section contains the graphs representing the results obtained by the variation of parameters used in CPBGSA. These graphs illustrate variation of

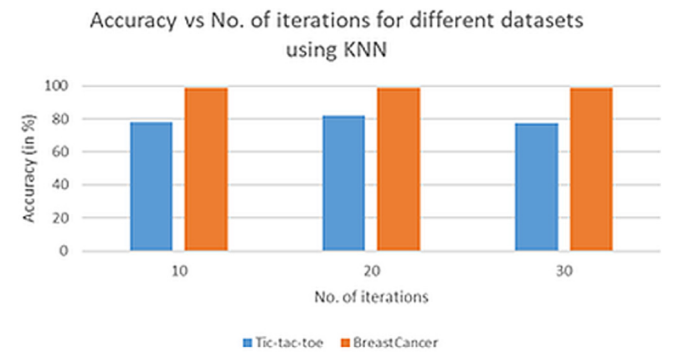

Fig. 2. Accuracy (using KNN) versus no. of iterations for CPBGSA over 2 small datasets.

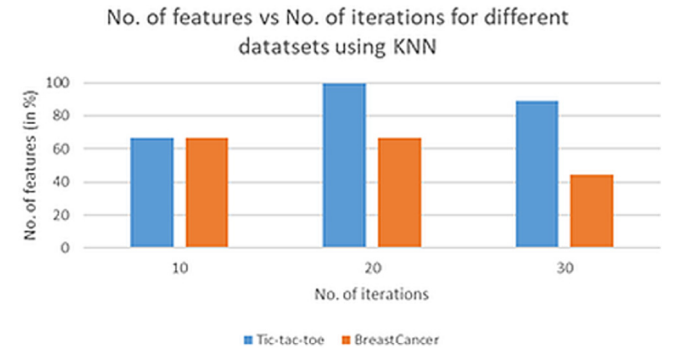

Fig. 3. No. of features selected (in %) versus no. of iterations for CPBGSA over 2 small datasets.

all the parameters used in the process — no. of iterations, population size, β (or η) and α. The interpretation and conclusions drawn from the graphs are provided in Section 4.2.1 to 4.2.3.

4.2.1. Group 1 (small datasets)

Figs. 2 and 3 represent the variation of the classification accuracy and the percentage of total features selected respectively by CPBGSA using the KNN classifier against the number of iterations, keeping the population size fixed at 30, over the 2 small datasets. For the Tic-tac-toe dataset, although our proposed model achieves the best classification accuracy when the number of iterations is set to 20, it selects all of the features under those conditions, as can be seen from Fig. 2. In case of BreastCancer dataset, the proposed model achieves almost 100% classification accuracy for all iteration values while the percentage of selected features is considerably low for the iteration value of 30.

Figs. 4 and 5 show the variations of the classification accuracy and the percentage of features selected respectively by CPBGSA using the KNN classifier against the population sizes, keeping the number of iterations constant at 20, over the small datasets. Fig. 4 illustrates that our proposed FS model achieves the best accuracy for both the small datasets corresponding to population size of 30. This initial population size also results into minimum number of selected features for the BreastCancer dataset as can be seen from Fig. 5.

Based on the above discussion and taking both the small datasets into consideration, it can be concluded that the proposed model gives the best overall performance for iteration count of 20 and initial population size of 30.

In addition to the initial population size and no. of iterations, further experimentations have been performed to find a suitable value for β (or η). For these experimentations, the values of initial population size and no. of iterations are fixed at 30 and 20

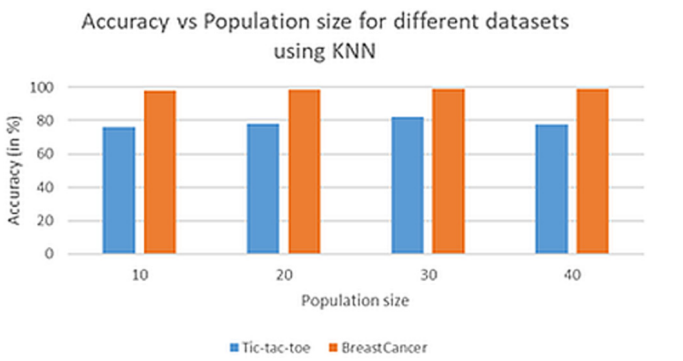

Fig. 4. Accuracy (using KNN) versus population size for CPBGSA over 2 small datasets.

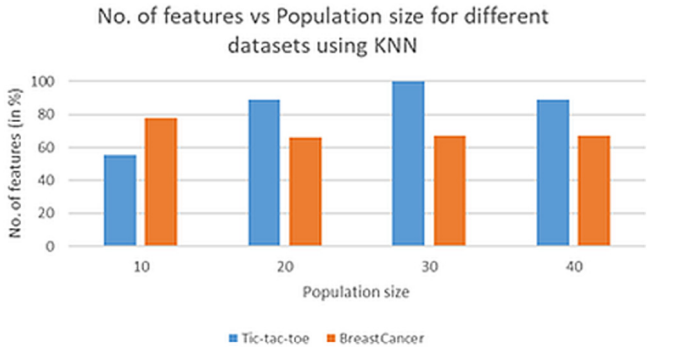

Fig. 5. No. of features selected (in %) versus population size for CPBGSA over 2 small datasets.

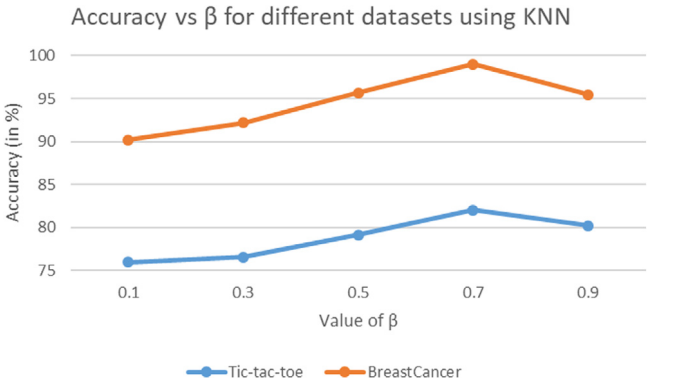

Fig. 6. Accuracy variation for different values of β for CPBGSA over 2 small datasets.

respectively. Fig. 6 provides the classification accuracy obtained by varying the value of β as 0.1, 0.3, 0.5, 0.7, 0.9 for two small datasets. The graph presented in Fig. 6 clearly illustrates that 0.7 is the most suitable value for β in case of small datasets.

4.2.2. Group 2 (medium datasets)

Figs. 7 and 8 illustrate the variation of the classification accuracy and the percentage of total features selected respectively by CPBGSA using the KNN classifier against the number of iterations, keeping the population size fixed at 30, over the medium datasets. Fig. 8 shows that our proposed model achieves greater than 95% classification accuracy in 10 out of the 15 datasets for iteration count of 20. This value also results in less than 60% feature selection in 12 datasets, thus exhibiting better overall performance than other iteration values.

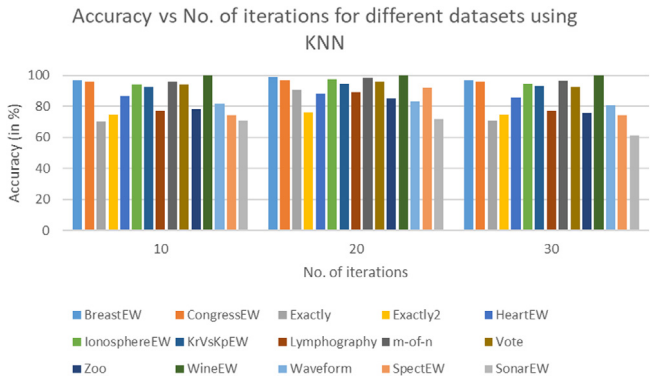

Fig. 7. Accuracy (using KNN) versus no. of iterations for CPBGSA over 15 medium datasets.

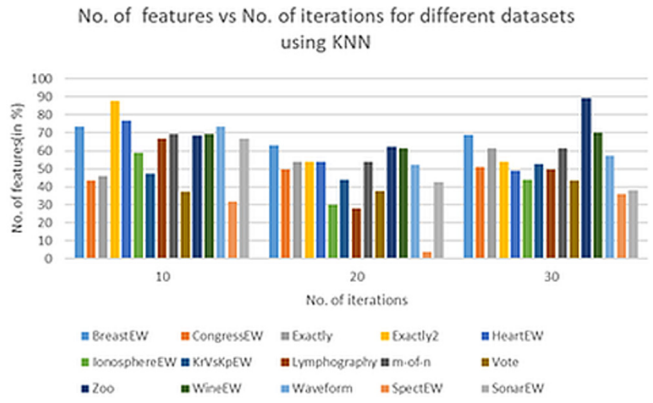

Fig. 8. No. of features (in %) versus no. of iterations for CPBGSA over 15 medium datasets.

Figs. 9 and 10 demonstrate the variation of the classification accuracy and the percentage of features selected respectively by CPBGSA using the KNN classifier against the population size, keeping the number of iterations constant at 20, over the medium datasets. These figures, on being subjected to similar analysis as Figs. 7 and 8, exhibit similar trends. The initial population size of 30 leads to greater than 90% accuracy for 11 out of the 15 datasets, while 12 datasets have less than 60% of their features selected.

Therefore, a similar conclusion can be reached, that the proposed model exhibits the best performance for the medium datasets, using the same parameter values applied for small datasets.

Figs. 11 and 12 provide the classification accuracy obtained by varying the value of β as 0.1, 0.3, 0.5, 0.7, 0.9 for 8 and 7 medium datasets respectively. In Fig. 11, we can see that the algorithm could not achieve highest accuracy for $\beta = 0.7$ in case of Exactly and HeartEW. On the other hand, the same value of β does not result into maximum accuracy in Fig. 12 for Zoo and SonarEW. But for other 11 out of 15 medium datasets, assigning 0.7 to β produces highest classification accuracy. Hence, we conclude that 0.7 is the most suitable value for β for medium datasets as well.

4.2.3. Group 3 (large datasets)

We have used 3 large datasets namely PenglungEW, Arrhythmia and Madelon for evaluation of the proposed model. Figs. 13 and 14 display accuracy obtained and no. of features selected respectively by CPBGSA with respect to changing no. of iterations (keeping population size fixed at 30) over all the 3 large datasets. Similarly, Figs. 15 and 16 represent accuracy obtained

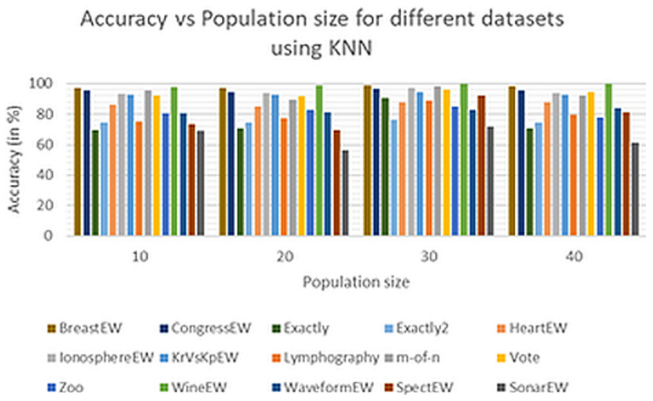

Fig. 9. Accuracy (using KNN) versus population size for CPBGSA over 15 medium datasets.

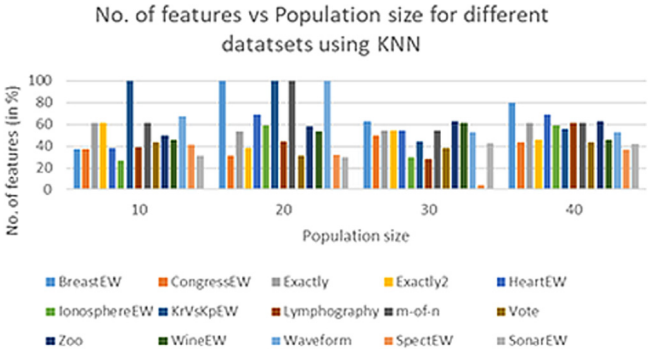

Fig. 10. No. of features (in %) versus population size for CPBGSA over 15 medium datasets.

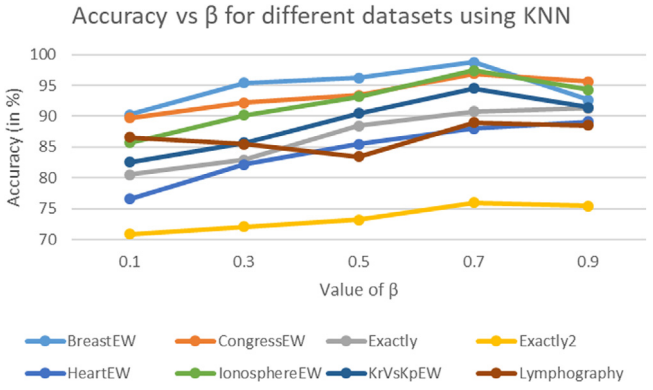

Fig. 11. Accuracy variation for different values of β for CPBGSA over 8 medium datasets.

and no. of features selected respectively by CPBGSA with respect to changing population size (keeping no. of iterations fixed at 20) over the same datasets. For the large datasets, slight deviations can be observed. Although accuracy obtained by the model over PenglungEW and Madelon have reached their maximum value for population size 30 and no. of iterations 20, it could not achieve the best feature reduction for the same pair of values.

So, as a conclusion to these graphical representations, it is clearly visible that keeping population size and no. of iterations fixed at 30 and 20 respectively give the best output in terms of classification accuracy but not in terms of percentage of selected features. However, as increase in classification accuracy is a more important parameter than number of selected features, for rest

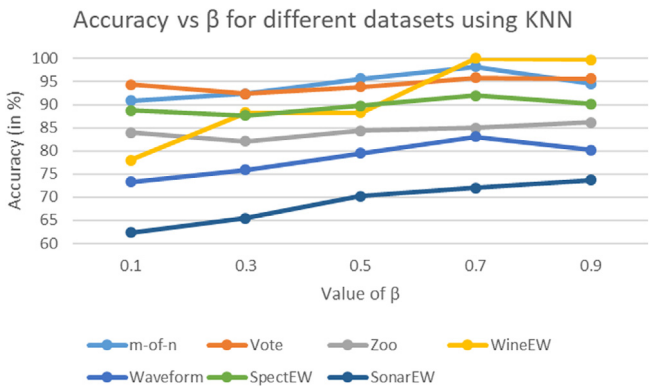

Fig. 12. Accuracy variation for different values of β for CPBGSA over other 7 medium datasets.

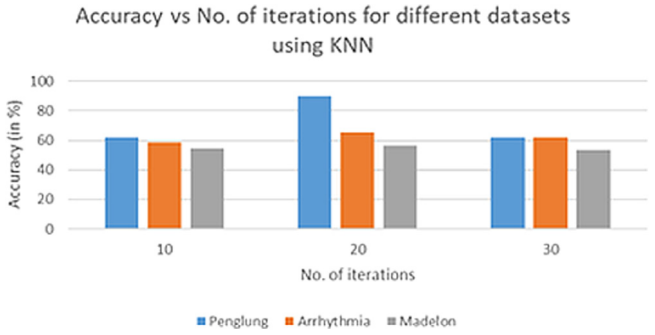

Fig. 13. Accuracy (using KNN) versus no. of iterations for CPBGSA over 3 large datasets.

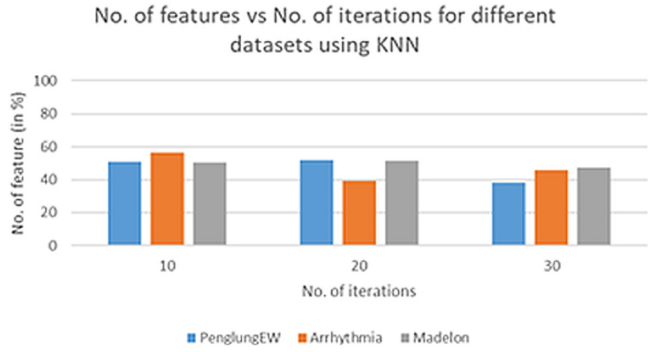

Fig. 14. No. of features (in %) versus no. of iterations for CPBGSA over 3 large datasets.

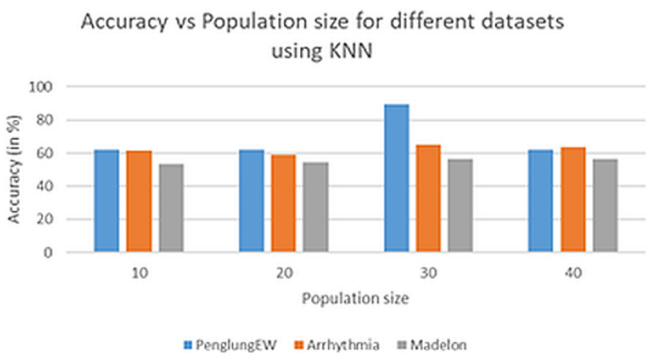

Fig. 15. Accuracy (using KNN) versus population size for CPBGSA over 3 large datasets.

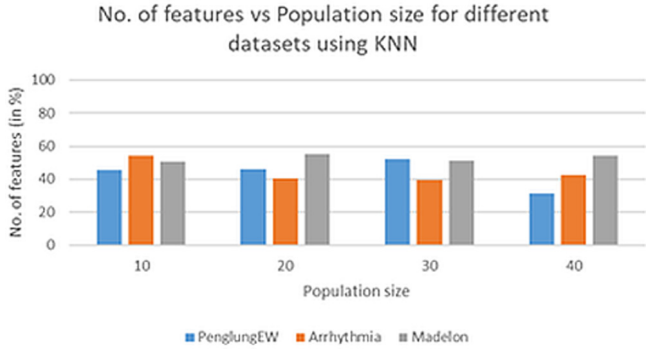

Fig. 16. No. of features (in %) versus population size for CPBGSA over 3 large datasets.

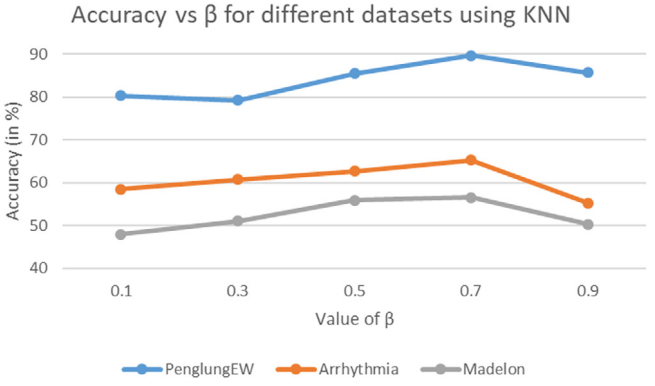

Fig. 17. Accuracy variation for different values of β for CPBGSA over 3 large datasets.

of the experimentations, we have used the above-mentioned parameter values.

Fig. 17 provides the classification accuracy obtained by varying the value of β as 0.1, 0.3, 0.5, 0.7, 0.9 for three large datasets. The graph presented in Fig. 17 clearly illustrates that 0.7 is the most suitable value for β in case of large datasets too.

4.2.4. Cluster validity analysis

After fixing the values of population size, no. of iterations and β (and η), we have performed another set of experimentations to find a proper value for α mentioned in Eq. (8). The number of clusters used in the proposed guidance method is decided by the value of α which is a very important parameter in the process. To evaluate the impact of different values of α on the formation of clusters, we have used a popular cluster validation index called Davies–Bouldin (DB) index [41]. The index value is computed using Eq. (11) shown below:

$$DB(C) = \frac{1}{K}\sum_{i=1}^{K}\max_{i\neq j}\left\{\frac{\Delta(C_i)+\Delta(C_j)}{\delta(C_i,C_j)}\right\} \tag{11}$$

where $C = (C_1, C_2, \ldots.., C_K)$ is the set of all clusters and K is the total number of clusters. $\Delta(C_i)$ is the intra-cluster distance of the ith cluster and $\delta(C_i, C_j)$ is inter-cluster distance between ith and jth cluster where $i \neq j$.

From Eq. (11), we can clearly see that a clustering technique is said to perform well if the DB value of the clusters is low because it will result into small intra-cluster distance and high inter-cluster distance which are the most important objectives of clustering. The value of α is varied as 0.1, 0.2, 0.3, 0.4 and 0.5.

Table 7
DB values for the proposed clustering technique for varying α values over 20 datasets.

Dataset	Davies–Bouldin (DB) Value				
	α=0.1	α=0.2	α=0.3	α=0.4	α=0.5
BreastCancer	4.414196	2.994526	2.35515	1.938426	2.004465
BreastEW	3.623333	2.720222	2.421556	2.158087	2.400567
CongressEW	3.737727	2.554261	1.861572	2.279896	2.138538
Exactly	3.404304	3.075041	3.500292	2.184364	2.473248
Exactly2	4.798773	3.116076	3.009258	2.478885	2.477764
HeartEW	4.514785	3.546698	3.291837	2.023946	2.488591
IonosphereEW	3.002876	2.964739	2.839696	2.218197	2.321103
KrVsKpEW	4.239587	3.535245	2.59886	2.126689	2.268521
Lymphography	3.639455	3.0927	2.806083	2.368344	2.534663
M-of-n	3.4112	3.333034	2.922217	2.28907	2.483904
PenglungEW	4.546997	3.1936	2.635774	2.078518	2.073403
SonarEW	4.245935	3.133103	2.646555	2.167747	2.170887
SpectEW	3.491593	3.345392	2.855128	1.970907	2.216221
Tic-tac-toe	2.800961	2.877239	2.608736	2.321402	2.343808
Vote	3.7066	3.286053	2.238519	1.878741	2.050918
Waveform	3.978715	2.953695	3.061707	2.117501	2.271444
WineEW	3.826538	3.178374	3.069182	2.226068	2.184569
Zoo	3.739837	3.515495	2.567063	2.037598	2.192491
Arrhythmia	4.57179	3.41216	2.908513	2.101353	2.300187
Madelon	4.441881	3.144074	2.79312	2.238099	2.211511

The DB values of the proposed clustering technique for varying α values over all the datasets is presented in Table 7. From the table, we can clearly see that the proposed clustering technique has performed the best when $\alpha = 40$. That is why we have selected the most optimal value of α as 0.4.

4.3. Experimental results

The detailed results obtained by our model are recorded in Tables 8 and 9. The experiments have been run using three popular classifiers *namely* KNN, MLP (Multi-layer Perceptron) and RF (Random Forest - a decision tree-based classifier). The classifier settings used to find the results are also provided in Tables 8 and 9. Table 8 contains the results for first 10 UCI datasets and Table 9 contains the second 10 UCI dataset results.

The following observations can be made by inspecting the results given in Tables 8 and 9:

- For KNN classifier, CPBGSA obtains greater than 90% accuracy for 10 datasets, greater than 80% but less than 90% accuracy for 6 datasets, greater than 70% but less than 80% accuracy for 2 datasets and between 50% and 70% accuracy for 2 datasets. Similarly, using MLP classifier, CPBGSA obtains greater than 90% accuracy for 13 datasets, greater than 80% but less than 90% accuracy for 5 datasets and between 60% and 80% for remaining 2 datasets. For RF classifier, CPBGSA achieves more than 90% accuracy in 8 datasets, between 80 and 90% accuracy for 3 datasets, between 70 and 80% in 6 datasets and less than 70% accuracy in remaining 3 datasets. Note that, CPBGSA obtains 100% (full) accuracy for 1 dataset using KNN, for 1 dataset using RF and for 5 datasets using MLP. Such high values of accuracies over such diverse datasets certainly prove the applicability of the proposed model.
- Best, worst and average accuracies are almost equal and very low value of standard deviation indicates the ability of the proposed model to increase the accuracy of every candidate solution in the population which is a very important evaluation criterion of any population-based model.
- Using MLP as a classifier gives the best results in term of accuracy among all three classifiers but it takes much more time as compared to the rest of the classifiers. RF takes moderate amount of time which is in between the time requirements of KNN and MLP classifiers. From Tables 8 and 9, it is evident that RF classifier is not able to achieve better "Average accuracy" than the other classifiers (KNN and MLP). We can draw another interesting conclusion from the results which is that the standard deviation for RF classifier is on the higher side compared to KNN and MLP. Hence, although even in some cases, RF classifier obtains better "Best accuracy" when compared to KNN but could not get better "Average accuracy".

4.4. Comparison of results

For establishing the superiority of our proposed model, we have selected a variety of classical as well as recently proposed FS algorithms having different kinds of exploration–exploitation trade-offs. In Tables 10–12, we have provided comparison of our proposed model — CPBGSA in terms of best classification accuracy, average classification accuracy and average number of selected features respectively. CPBGSA has been compared with Ant-Lion Optimization (ALO), Binary Ant-Lion Optimization (BALO-1), BALO-S, BALO-V, BBA, GA, PSO, GSA, Deluge based Genetic Algorithm (DGA) [42], Wrapper Filter based Ant Colony Optimization for Feature Selection (WFACOFS) [43] and Histogram based Multi-Objective Genetic Algorithm (HMOGA) [21]. GA is well-known in the literature for its exploration ability through crossover. Though GA achieves exploitation to some extent using mutation, its weak local search capability makes GA inefficient in terms of exploitation. PSO on the other hand, achieves impressive local search which in turn increases its exploitation ability. BBA is a combination between PSO and intensive local search but balance between the two methods depend on the values of loudness and pulse emission rate. A good balance between exploitation and exploration is shown by all the versions of ALO. Adaptive boundary shrinking mechanism and elitism applied in ALO result in good exploration and high convergence rate. On the other hand, roulette wheel selection and random walk procedures enable ALO to achieve good exploration of the search space. Binary version of ALO is proposed in BALO replacing the idea of average in ALO with crossover. BALO-S and BALO-V use squashing functions of shape-S (sigmoidal function) and shape-V (hyperbolic tan function) respectively which force search agents to move in binary space. DGA uses Great Deluge Algorithm (GDA) to enhance the local search capability of GA which in turn improves the extent of exploitation in GA. WFACOFS introduces a filter-based evaluation method in the system of search agents (ants in ACO) which

Table 8
Results obtained by our proposed FS model on first 10 UCI datasets. The best, worst and average accuracies along with the number of features are given. Computation time for execution is also provided.

Dataset	Type	CPBGSA (with KNN)			CPBGSA (with MLP)			CPBGSA (with RF)		
		Accuracy	No. of features	k-value	Accuracy	No. of features	No. of neurons in hidden layer	Accuracy	No. of features	No. of trees
BreastCancer	Best	0.9899	6	6	0.9966	7	110	0.9900	6	70
	Worst	0.986	5		0.9833	6		0.9766	5	
	Average	0.9895	5.87		0.9866	6		0.9869	6.667	
	STD	0.001	0.34		0.004	1.142		0.0044	2.309	
	Time (s)	24.46			486.84			44.2266		
BreastEW	Best	0.988	19	6	1	21	50	0.9588	13	70
	Worst	0.976	15		0.9882	30		0.9235	17	
	Average	0.986	18.75		0.9907	21.46		0.9461	14.25	
	STD	0.003	1.224		0.003	4.032		0.0083	2.417	
	Time (s)	22.58			502.563			39.1012		
CongressEW	Best	0.969	8	4	0.9769	11	50	0.9308	6	100
	Worst	0.961	7		0.969	4		0.8615	5	
	Average	0.968	7.79		0.971	7.17		0.8994	9.333	
	STD	0.002	0.658		0.003	2.353		0.0175	2.348	
	Time (s)	23.34			443.531			25.6159		
Exactly	Best	0.9075	7	5	1	8	50	0.6925	4	85
	Worst	0.695	13		0.7125	7		0.6875	8	
	Average	0.881	7.75		0.9651	7.96		0.6921	7.83	
	STD	0.072	2.027		0.094	0.36		0.0014	1.80	
	Time (s)	24.942			490.76			40.8252		
Exactly2	Best	0.76	7	5	1	11	110	0.7400	5	70
	Worst	0.74	3		0.74	5		0.7400	13	
	Average	0.752	5.542		0.8852	9		0.7400	8.667	
	STD	0.009	1.933		0.115	3.426		0.0000	2.774	
	Time (s)	24.51			512.194			59.5465		
HeartEW	Best	0.88	7	6	0.9012	13	50	0.8395	7	115
	Worst	0.84	9		0.852	8		0.7778	6	
	Average	0.87	7.25		0.873	9.75		0.8220	7.833	
	STD	0.013	0.68		0.014	3.22		0.0170	1.697	
	Time (s)	19.73			431.13			23.5446		
IonosphereEW	Best	0.974	10	5	0.9867	19	50	0.9073	14	70
	Worst	0.947	18		0.9669	21		0.7815	15	
	Average	0.969	11.33		0.974	19.792		0.8676	18	
	STD	0.01	3.045		0.006	1.769		0.0346	6.715	
	Time (s)	17.78			431.162			25.2219		
KrVsKpEW	Best	0.9452	16		0.995	36	110	0.9171	24	70
	Worst	0.9201	36		0.982	36		0.7496	20	
	Average	0.9421	18.5		0.9875	36		0.8633	24.25	
	STD	0.008	6.75		0.0031	0		0.0510	7.724	
	Time (s)	79.624			884.89			256.3547		
Lymphography	Best	0.89	5	5	0.8863	18	50	0.7273	9	85
	Worst	0.84	7		0.7954	9		0.4773	18	
	Average	0.85	7.58		0.8106	13.42		0.5549	9.917	
	STD	0.019	2.16		0.0229	4.51		0.0831	4.100	
	Time (s)	16.61			411.05			22.6799		
Arrhythmia	Best	0.652	92	4	0.717	165	50	0.4934	130	70
	Worst	0.578	136		0.664	144		0.4934	279	
	Average	0.617	99.3		0.675	165.71		0.4934	162.9	
	STD	0.017	16.75		0.012	21.12		0.0000	47.013	
	Time (s)	63.204			1608.75			52.5925		

reduces the overall time requirement of the model. HMOGA uses histogram to calculate relative importance of the features and performs thresholding to select the final set of features. Thus, we conclude that the algorithms we have selected for the comparison with CPBGSA are from a large variety of families and have different exploration–exploitation capabilities. For comparison of CPBGSA with other well-established FS methods, we have chosen KNN as our classifier. KNN is much faster than MLP and RF as can be observed from the recorded time in Tables 8–9.

4.4.1. Comparison based on highest accuracy

Table 10 consists of the comparison of the highest classification accuracies achieved by different models. Out of the 20 datasets, the proposed one works performs best for 9 datasets. GSA performs best for 6 datasets, which is the highest for any other algorithm, followed by BALO-1 which performs best for 4 datasets and WFACOFS which performs best in 1 dataset. For remaining 11 datasets, in which CPBGSA failed to achieve the best results, it can be seen that the proposed method is able to get comparable accuracies. For further clarification, the rank of CPBGSA accuracy among the accuracies obtained by all the methods

Table 9
Results obtained by our proposed FS model on second 10 UCI datasets. The best, worst and average accuracies along with the number of features are given. Time for execution is also provided.

Dataset	Type	CPBGSA (with KNN)			CPBGSA (with MLP)			CPBGSA (with RF)		
		Accuracy	No. of features	*k*-value	Accuracy	No. of features	No. of neurons in hidden layer	Accuracy	No. of features	No. of trees
M-of-n	Best	0.9825	7	4	1	9	50	0.910	13	70
	Worst	0.8625	13		1	13		0.738	7	
	Average	0.9675	7.75		1	11		0.803	8	
	STD	0.04			0	2.043		0.062	2.730	
	Time (s)	2.027			435.14			73.757		
PenglungEW	Best	0.8966	169	4	0.8965	186	50	0.724	205	100
	Worst	0.862	156		0.8276	186		0.69	161	
	Average	0.878	131.958		0.8577	174.33		0.698	181.75	
	STD	0.018	36.823		0.0614	14.062		1.724	19.033	
	Time (s)	26.134			445.97			264.425		
SonarEW	Best	0.72	26	14	0.866	29	140	0.746	16	70
	Worst	0.66	6		0.761	25		0.493	27	
	Average	0.71	23.5		0.787	24.79		0.600	26.5	
	STD	0.02	6.76		0.03	8.06		0.067	4.359	
	Time (s)	18.8			450.47			54.012		
SpectEW	Best	0.92	1	13	0.904	15	50	0.797	13	70
	Worst	0.73	8		0.743	11		0.75	22	
	Average	0.89	2.16		0.786	12.92		0.77	17.5	
	STD	0.071	2.66		0.042	3.99		1.820	5.196	
	Time (s)	19.95			408.6			213.459		
Tic-tac-toe	Best	0.82	9	5	0.943	9	140	0.739	6	85
	Worst	0.77	5		0.804	6		0.708	4	
	Average	0.82	8.5		0.848	8.375		0.724	5.5	
	STD	0.02	1.35		0.037	1.244		0.010	1.382	
	Time (s)	26.64			511.96			42.898		
Vote	Best	0.9583	6	6	0.9583	16	50	0.933	10	70
	Worst	0.925	5		0.9417	10		0.833	12	
	Average	0.9486	5.67		0.9483	11.083		0.874	9	
	STD	0.016	0.761		0.006	3.549		0.028	2.486	
	Time (s)	21.498			418.493			64.942		
Waveform	Best	0.831	21	10	0.875	40	50	0.845	40	115
	Worst	0.82	20		0.857	30		0.739	22	
	Average	0.83	20.87		0.869	38.75		0.814	26.75	
	STD	0.001	0.34		0.005	3.38		0.031	8.935	
	Time (s)	177.85			866.2807			306.221		
WineEW	Best	1	6	5	1	9	50	1.000	10	70
	Worst	0.98	5		1	13		0.894	6	
	Average	0.99	5.88		1	9.333		0.973	8.25	
	STD	0.007	0.34		0	2.28		0.030	2.864	
	Time (s)	21.1			410.114			34.191		
Zoo	Best	0.85	9	5	0.878	9	50	0.854	9	85
	Worst	0.83	10		0.805	13		0.854	11	
	Average	0.83	10.06		0.83	9.92		0.854	9	
	STD	0.008	1.06		0.015	2.083		0.000	2.892	
	Time (s)	19.54			442.75			207.160		
Madelon	Best	0.565	252	5	0.605	290	70	0.595	156	70
	Worst	0.561	242		0.58	225		0.523	238	
	Average	0.564	250.75		0.587	290.458		0.584	287.3333	
	STD	0.001	3.38		0.0061	66.91		0.019	115.715	
	Time (s)	1163.006			8367.328			932.976		

used for comparison is also provided in Table 10. Even though CPBGSA could not achieve the best classification accuracy for all the datasets, for most of the datasets, the difference between the highest accuracy and CPBGSA accuracy is within 2%.

4.4.2. Comparison based on average accuracy

In Table 11, the comparison of average accuracies of the methods is given. Best accuracies are not always indicative of the robustness of the model because some algorithms may increase the accuracy of only few candidate solutions from the population but the applicability of a model lies in the ability to increase the accuracy of every candidate. So, to establish the effectiveness of CPBGSA, we have compared its average accuracy with that of other methods in Table 11. In case of average accuracy, CPBGSA outperforms the rest of the algorithms 12 out of 20 times. Again, for remaining 8 datasets, CPBGSA achieves comparable average accuracies. So, we can see that the number of datasets in which CPBGSA achieves best result in terms of average accuracy is more than that in terms of highest accuracy. This happens because CPBGSA focuses on improving all of its candidates by passing information gained by one candidate to other which is clear by the low values of standard deviation provided in Tables 8 and 9. Unlike most of the other methods used for comparison, which start with random candidates, CPBGSA clubs together similar

Table 10
Comparison of CPBGSA with state-of-the-art FS methods in terms of highest classification accuracy achieved by them. The highest accuracy for each dataset is made bold.

Dataset	Methods											Rank of CPBGSA
	ALO	BALO-1	BALO-S	BALO-V	BBA	GA	PSO	GSA	DGA	WFACOFS	HMOGA	CPBGSA (Proposed)	
BreastCancer	0.971	0.974	0.969	0.97	0.975	0.95	0.9554	**0.9933**	0.99	0.99	0.962	0.9899	4
BreastEW	0.972	0.979	0.979	0.979	0.976	0.9379	0.9232	**0.9941**	0.98	0.978	0.981	0.988	2
CongressEW	0.961	0.97	0.961	0.963	0.967	0.931	0.9366	**0.984**	0.97	0.97	0.98	0.969	6
Exactly	0.701	0.856	0.723	0.856	0.748	0.7357	0.7381	0.855	0.86	0.88	0.72	**0.9075**	1
Exactly2	0.764	**0.766**	**0.766**	**0.766**	0.765	0.7538	0.7598	0.75	0.74	0.75	0.75	0.76	6
HeartEW	0.869	0.872	0.867	0.878	0.872	0.7778	0.7756	**0.901**	0.89	0.89	0.9	0.88	5
IonosphereEW	0.885	0.889	0.877	0.892	0.899	0.9	0.879	0.97	0.96	0.97	0.96	**0.974**	1
KrVsKpEW	0.948	**0.967**	0.946	0.966	0.957	0.9431	0.9423	0.961	0.95	0.95	0.95	0.9452	10
Lymphography	0.824	0.875	0.844	0.861	0.851	0.748	0.684	0.8863	0.89	0.84	0.85	**0.89**	1
M-of-n	0.93	0.994	0.917	0.99	0.95	0.8781	0.8799	**1**	0.94	0.95	0.95	0.9825	4
PenglungEW	0.788	0.774	0.754	0.781	0.774	0.589	0.6	0.8965	0.79	0.878	0.86	**0.8966**	1
SonarEW	0.827	**0.868**	0.825	0.846	0.865	0.825	0.837	0.742	0.69	0.57	0.68	0.72	9
SpectEW	0.85	0.89	0.863	0.891	0.875	0.8	0.8045	0.9197	0.84	0.88	0.74	**0.92**	1
Tic-tac-toe	0.772	0.787	0.779	0.787	0.781	0.765	0.766	0.8015	0.82	0.81	0.78	**0.82**	1
Vote	0.95	0.955	0.953	0.96	**0.962**	0.914	0.908	0.9583	0.94	0.94	0.94	0.9583	3
Waveform	0.786	0.8	0.778	0.805	0.79	0.7759	0.7745	0.83	0.81	0.8	0.8	**0.831**	1
WineEW	0.989	0.989	0.989	0.994	0.992	0.983	0.983	**1**	**1**	**1**	0.976	**1**	1
Zoo	0.891	**0.931**	0.906	0.921	0.916	0.86	0.8788	0.878	0.8	0.83	0.84	0.85	9
Arrhythmia	0.61	0.638	0.62	0.625	0.618	0.53	0.51	0.57	0.61	0.64	0.61	**0.652**	1
Madelon	0.54	0.573	0.56	0.542	0.568	0.56	0.5	0.56	0.57	**0.64**	0.61	0.565	6

Table 11
Comparison of CPBGSA with state-of-the-art FS methods in terms of average classification accuracy. The highest accuracy for each dataset is made bold.

Dataset	Methods											Rank of CPBGSA
	ALO	BALO-1	BALO-S	BALO-V	BBA	GA	PSO	GSA	DGA	WFACOFS	HMOGA	CPBGSA (Proposed)	
BreastCancer	0.956	0.962	0.945	0.9531	0.9601	0.9326	0.9521	0.9875	0.98	0.987	0.954	**0.9895**	1
BreastEW	0.9591	0.962	0.9542	0.9532	0.9714	0.937	0.9295	0.9772	0.96	0.982	0.974	**0.986**	1
CongressEW	0.9433	0.9528	0.9446	0.957	0.9417	0.93	0.8967	0.9597	0.92	0.96	0.946	**0.968**	1
Exactly	0.682	0.8489	0.7164	0.8392	0.7303	0.7028	0.6992	0.7146	0.71	0.75	0.72	**0.881**	1
Exactly2	0.7462	0.7571	0.7491	0.757	**0.765**	0.7462	0.7481	0.7424	0.71	0.74	0.75	0.752	4
HeartEW	0.8432	0.8544	0.8414	0.8519	0.8464	0.7423	0.7562	0.8589	0.84	0.86	0.85	**0.87**	1
IonosphereEW	0.8672	0.8722	0.8592	0.8749	0.883	0.8705	0.8674	0.9477	0.93	0.95	0.96	**0.969**	1
KrVsKpEW	0.9344	**0.9512**	0.9311	0.9504	0.9421	0.9361	0.9334	0.9201	0.92	0.94	0.95	0.9421	4
Lymphography	0.8064	**0.8574**	0.8272	0.8439	0.8346	0.7114	0.6756	0.8015	0.79	0.8	0.85	0.85	2
m-of-n	0.9162	**0.9788**	0.9029	0.9751	0.9351	0.8765	0.8688	0.8528	0.85	0.91	0.95	0.9675	3
PenglungEW	0.7669	0.7556	0.7348	0.7625	0.7565	0.679	0.7018	0.7991	0.74	0.86	0.86	**0.878**	1
SonarEW	0.8081	**0.8511**	0.8077	0.8291	0.8486	0.8209	0.7938	0.7231	0.49	0.53	0.68	0.71	9
SpectEW	0.8336	0.8718	0.8471	0.8738	0.8574	0.8465	0.7987	0.7406	0.7	0.76	0.74	**0.89**	1
Tic-tac-toe	0.7567	0.7714	0.763	0.7712	0.7646	0.7499	0.7498	0.7622	0.77	0.8	0.78	**0.82**	1
Vote	0.9323	0.9378	0.9354	0.9429	0.9459	0.8987	0.8768	0.9334	0.9	0.93	0.94	**0.9486**	1
WaveformEW	0.7723	0.7839	0.7623	0.7887	0.774	0.7752	0.7735	0.7914	0.76	0.74	0.8	**0.83**	1
WineEW	0.9742	0.9733	0.9736	0.9778	0.9756	0.9709	0.9683	**0.9957**	0.96	0.976	0.96	0.99	2
Zoo	0.8743	**0.9139**	0.8896	0.9042	0.8996	0.859	0.8403	0.8091	0.78	0.8	0.84	0.83	9
Arrhythmia	0.6	0.595	0.58	0.601	0.571	0.51	0.51	0.57	0.56	0.6	0.61	**0.617**	1
Madelon	0.51	0.542	0.53	0.532	0.537	0.54	0.49	0.56	0.52	**0.62**	0.61	0.564	3

feature attributes (from the random initial candidates) to remove redundancy from the system of solutions. Thus, when the GSA phase starts, the solutions it gets are dissimilar in nature and have information about different parts of the search space. So, it can be assumed that when a candidate in CPBGSA receives information (here velocity) from other candidates, they are likely to be new information unknown to the candidate beforehand. On the other hand, if only random agents or candidate solutions are used, then the information received by the agent may be similar to the information which it already has. So, it will not let the agents to collectively explore the search space and hence will limit their classification ability.

4.4.3. Comparison based on average number of selected features

Table 12 contains comparison of average number of features selected by different FS approaches. In terms of average reduction of feature dimension, it can be seen that CPBGSA selects a moderate number of features from the entire feature dimension. CPBGSA selects neither too many features, nor too less which helps it to achieve a stable feature dimension along with an impressive classification accuracy.

Therefore, from all the comparative results given in Tables 10 and 11, we can conclude that CPBGSA is stable, capable of avoiding premature convergence and clearly suitable for dimensionality reduction of feature sets. Table 13 contains p-values for Wilcoxon test for 20 pairs (one for each dataset) of average accuracy values. All the values are below 0.05 which show the effectiveness of our method.

Taken together, the results of this work showed that the proposed method is beneficial in improving the performance of BGSA when applying to FS problems. The initial population is uniformly spread around the search space, which eventually leads to a better local optima avoidance and better exploration of search space.

The key advantage of the proposed method is that it does not work with completely random population. Even though it allows initial population to be randomized, it creates a population which is better distributed across the search space out of the

Table 12
Comparison of CPBGSA with state-of-the-art FS methods in terms of average number of selected features.

Dataset	Methods												Rank of CPBGSA
	ALO	BALO-1	BALO-S	BALO-V	BBA	GA	PSO	GSA	DGA	WFACOFS	HMOGA	CPBGSA (Proposed)	
BreastCancer	4.671	3.771	4.617	3.987	4.77	4.446	4.869	5.94	4.55	6.03	5.65	5.87	10
BreastEW	15.57	14.82	16.11	16.11	16.86	15.75	17.22	16.026	14.9	19.45	14	18.75	10
CongressEW	8.13	9.39	11.97	9.27	14.49	12.39	18.12	6.306	8.15	7.95	11.14	7.79	2
Exactly	5.278	5.668	6.721	6.006	6.331	5.005	5.551	8.04	7.25	9	10.2	7.75	8
Exactly2	5.941	4.394	7.111	4.329	5.33	5.941	5.837	5.973	5.95	6.58	6.5	5.542	4
HeartEW	8.45	6.162	7.449	6.552	7.332	7.163	7.059	7.28	8.3	8.75	8.2	7.25	5
IonosphereEW	11.39	14.654	11.9	13.566	17	15.946	18.904	15.706	15.1	9.55	19	11.33	2
KrVsKpEW	24.768	16.452	20.124	16.848	19.872	19.836	21.276	20.906	22.35	23.58	21	18.5	3
Lymphography	7.506	6.498	8.55	7.776	9.162	9.216	9.558	9.706	10.35	12.13	13.25	7.58	3
m-of-n	8.892	6.279	8.775	6.669	7.228	8.112	7.332	8.106	8.85	10.23	9	7.75	5
PenglungEW	136.075	127.725	106.925	121.55	154.05	138.775	149.5	138.1	161.15	207.3	194.15	131.958	4
SonarEW	17.16	26.58	26.64	27.72	29.58	27.36	30.36	29.56	32.05	36.3	27.36	23.5	2
SpectEW	11.176	8.602	11.946	8.558	10.164	10.824	9.284	8.72	9.85	10.7	13	2.16	1
Tic-tac-toe	6.219	5.886	5.553	5.67	5.22	6.498	5.607	5.756	6.55	7.8	8.22	8.5	12
Vote	4.56	5.216	4.672	5.28	6.896	6.832	8.56	6.947	6.85	9.15	7	5.67	5
WaveformEW	33.88	23.72	26.16	22.68	24.44	28.28	26.28	22.8786	19.3	23.75	25.3	20.87	2
WineEW	6.838	5.772	6.11	5.057	6.11	6.552	7.11	5.893	11.25	7.48	8.22	5.88	3
Zoo	7.056	5.776	7.328	6.336	7.056	6.224	7.552	8.413	8.25	8.58	10.35	10.06	11
Arrhythmia	234.5	219.8	178.2	220.3	80.83	217.76	147.28	81	139.9	27.6	166	99.3	3
Madelon	256.4	267.4	251.4	298.4	122.167	276.08	173.24	218.3	238.95	315.08	297	250.74	5

Table 13
P-values of Wilcoxon test for comparison between the proposed algorithm and other methods.

Methods	ALO	BALO-1	BALO-S	BALO-V	BBA	GA	PSO	GSA	DGA	WFACOFS	HMOGA
p values of Wilcoxon test	4.05E−03	3.33E−02	5.73E−03	3.04E−02	1.58E−02	1.71E−03	7.79E−04	4.49E−04	8.83E−05	1.51E−03	0

initial population. This helps the system to get rid of redundancy at initial stage. When this distinct set of candidates enters into the GSA phase, it is able to successfully explore the entire search space. Thus, the speedy convergence problem faced by GSA is removed in the proposed method. As CPBGSA is able to explore a larger portion of the entire search space in comparison to other methods, it has a better probability of providing the best candidate solution which is clearly proved by the obtained results.

5. Conclusion

Most of the population-based FS approaches suffer from premature convergence which lead to a lack of exploration. As a remedy to this problem, a guiding strategy is used in which the members of the initial population are picked up from all over the search space. This guidance strategy clusters the initial population to a number of groups depending on the similarity of the members of the population. From each cluster, a representative member can be generated. A set of such representative members are used as the population of the FS algorithm under consideration. As the primary population of the FS algorithm are selected from different parts of the search space, the chances of exploration get increased which in turn reduce the possibility of premature convergence. In this paper, our proposed model CPBGSA uses this very idea to guide the initial population of GSA. One of the advantages of this guidance procedure is the fact that it is totally independent of the underlying FS approach. So, it can be used in conjunction with any population-based FS method without any modification in the FS strategy. Moreover, it reduces the need for a large initial population. Due to the diverse set of backgrounds of the initial population, the FS algorithm can work effectively even with a low-size population using this guidance procedure. The proposed guidance strategy, however, suffers from a drawback. It uses difference in accuracies as a measure of similarity between the members of the population which requires classification using a learning algorithm resulting in some additional time requirement. This makes the overall model computationally a bit expensive if a complex classifier is used for classification. The effectiveness of this strategy is verified by the obtained results. Evaluation outcomes of CPBGSA over small, medium and large datasets confirm the robustness of the method. An interesting future scope of our work is application of this model to any population-based FS algorithm, we plan to amalgamate this model with other FS techniques.

Declaration of competing interest

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

References

[1] J.C. Culberson, On the Futility of Blind Search, 1996.
[2] A.E.H. Van Laarhoven, P.J., Simulated annealing, in: Simulated Annealing: Theory and Applications, Springer, Dordr, 1987, pp. 7–15.
[3] F. Glover, M. Laguna, Tabu search, in: Handb. Comb. Optim., Springer, 1998, pp. 2093–2229.
[4] Zong Woo Geem, Joong Hoon Kim, G.V. Loganathan, A new heuristic optimization algorithm: Harmony search, Simulation 76 (2001) 60–68, http://dx.doi.org/10.1177/003754970107600201.
[5] M. Dorigo, M. Birattari, Ant colony optimization, in: Encycl. Mach. Learn., Springer, 2011, pp. 36–39.
[6] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Micro Mach. Hum. Sci. 1995. MHS'95. Proc. Sixth Int. Symp., IEEE, 1995, pp. 39–43.
[7] J. Yang, V. Honavar, Feature subset selection using a genetic algorithm, IEEE Intell. Syst. Their Appl. 13 (1998) 44–49.
[8] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: A gravitational search algorithm, Inf. Sci. (Ny) 179 (2009) 2232–2248, http://dx.doi.org/10.1016/j.ins.2009.03.004.
[9] M. Ghosh, T. Kundu, D. Ghosh, R. Sarkar, Feature selection for facial emotion recognition using late hill-climbing based memetic algorithm, Multimed. Tools Appl. (2019) http://dx.doi.org/10.1007/s11042-019-07811-x.
[10] L. Davis, Bit-climbing, representational bias, and test suit design, in: Proc. Intl. Conf. Genetic Algorithm, 1991, pp. 18–23.

[11] R.J. Tallarida, R.B. Murray, Chi-square test, in: Manual of Pharmacologic Calculations, Springer, New York, NY, 1987, pp. 140–142.
[12] H. Liu, H. Motoda, Non-myopic feature quality evaluation with (R) Relieff, in: Computational Methods of Feature Selection, Chapman and Hall/CRC, 2007, pp. 174–197.
[13] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, BGSA: Binary gravitational search algorithm, Nat. Comput. 9 (2010) 727–745, http://dx.doi.org/10.1007/s11047-009-9175-3.
[14] Z. Zhu, Y.S. Ong, M. Dash, Markov Blanket-embedded genetic algorithm for gene selection, Pattern Recognit. 40 (2007) 3236–3248, http://dx.doi.org/10.1016/j.patcog.2007.02.007.
[15] B. Duval, J.-K. Hao, J.C. Hernandez Hernandez, A memetic algorithm for gene selection and molecular classification of cancer, in: Proc. 11th Annu. Conf. Genet. Evol. Comput. - GECCO '09, 2009, p. 201, http://dx.doi.org/10.1145/1569901.1569930.
[16] J.H. Holland, Genetic algorithms, Sci. Am. 1 (1992) 66–73.
[17] J. Liu, J. Lampinen, A fuzzy adaptive differential evolution algorithm, Soft Comput. 9 (6) (2005) 448–462.
[18] D. Karaboga, B. Akay, A comparative study of artificial bee colony algorithm, Appl. Math. Comput. 214 (2009) 108–132.
[19] E. Emary, H.M. Zawbaa, A.E. Hassanien, Binary grey wolf optimization approaches for feature selection, Neurocomputing 172 (2016) 371–381, http://dx.doi.org/10.1016/j.neucom.2015.06.083.
[20] Il-Seok Oh, Jin-Seon Lee, Byung-Ro Moon, Hybrid genetic algorithms for feature selection, IEEE Trans. Pattern Anal. Mach. Intell. 26 (2004) 1424–1437, http://dx.doi.org/10.1109/TPAMI.2004.105.
[21] M. Ghosh, R. Guha, R. Mondal, P.K. Singh, R. Sarkar, Feature Selection using Histogram Based Multi-Objective GA for Handwritten Devanagari Numeral Recognition, 2017.
[22] M. Ghosh, S. Adhikary, K.K. Ghosh, A. Sardar, S. Begum, R. Sarkar, Genetic algorithm based cancerous gene identification from microarray data using ensemble of filter methods, Med. Biol. Eng. Comput. 57 (2019) 159–176.
[23] S. Malakar, M. Ghosh, S. Bhowmik, R. Sarkar, M. Nasipuri, A GA based hierarchical feature selection approach for handwritten word recognition, Neural Comput. Appl. (2019) 1–20.
[24] P. Moscato, C. Cotta, A. Mendes, Memetic algorithms, New Optim. Tech. Eng. (2005) 53–85, http://dx.doi.org/10.4156/ijiip.vol1.
[25] M. Ghosh, S. Malakar, S. Bhowmik, R. Sarkar, M. Nasipuri, Memetic algorithm based feature selection for handwritten city name recognition, in: Int. Conf. Comput. Intell. Commun. Bus. Anal., Springer, 2017, pp. 599–613.
[26] M. Ghosh, S. Begum, R. Sarkar, D. Chakraborty, U. Maulik, Recursive memetic algorithm for gene selection in microarray data, Expert Syst. Appl. 116 (2019) 172–185.
[27] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on, vol. 5, IE, 1997, pp. 4104–4108.
[28] B. Xue, M. Zhang, S. Member, W.N. Browne, Particle swarm optimization for feature selection in classification?: A multi-objective approach, IEEE Trans. Cybern. (2012) 1–16, http://dx.doi.org/10.1109/TPWRD.2003.813641.
[29] H. Banka, S. Dara, A hamming distance based binary particle swarm optimization (HDBPSO) algorithm for high dimensional feature selection, classification and validation, Pattern Recognit. Lett. 52 (2015) 94–100.
[30] van den Bergh, A.P. Engelbrecht, A new locally convergent particle swarm optimiser, in: Systems, Man and Cybernetics, 2002 IEEE International Conference on, vol. 3, IEEE, 2002, p. 6.
[31] W.N. Xue, B. Zhang, M. Browne, Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms, Appl. Soft Comput. 18 (2014) 261–276.
[32] S. Mirjalili, S.M. Mirjalili, X.S. Yang, Binary bat algorithm, Neural Comput. Appl. 25 (2014) 663–681, http://dx.doi.org/10.1007/s00521-013-1525-5.
[33] E. Rashedi, H. Nezamabadi-Pour, Feature subset selection using improved binary gravitational search algorithm, J. Intell. Fuzzy Syst. 26 (2014) 1211–1221, http://dx.doi.org/10.3233/IFS-130807.
[34] S. Sarafrazi, H. Nezamabadi-Pour, S. Saryazdi, Disruption: A new operator in gravitational search algorithm, Sci. Iran. 18 (2011) 539–548, http://dx.doi.org/10.1016/j.scient.2011.04.003.
[35] S. Mirjalili, A. Lewis, Adaptive gbest-guided gravitational search algorithm, Neural Comput. Appl. 25 (2014) 1569–1584, http://dx.doi.org/10.1007/s00521-014-1640-y.
[36] B. Gu, F. Pan, Modified gravitational search algorithm with particle memory ability and its application, Int. J. Innov. Comput. Inf. Control. 9 (2013) 4531–4544.
[37] H. Nezamabadi-Pour, A quantum-inspired gravitational search algorithm for binary encoded optimization problems, Eng. Appl. Artif. Intell. 40 (2015) 62–75, http://dx.doi.org/10.1016/j.engappai.2015.01.002.
[38] H. Bostani, M. Sheikhan, Hybrid of binary gravitational search algorithm and mutual information for feature selection in intrusion detection systems, Soft Comput. 21 (2017) 2307–2324, http://dx.doi.org/10.1007/s00500-015-1942-8.
[39] J. Wei, R. Zhang, Z. Yu, R. Hu, J. Tang, C. Gui, Y. Yuan, A BPSO-SVM algorithm based on memory renewal and enhanced mutation mechanisms for feature selection, Appl. Soft Comput. 58 (2017) 176–192.
[40] UCI Repository, 2019, https://archive.ics.uci.edu/ml/datasets.html (Accessed 7 January 2019).
[41] S. Petrovic, A comparison between the silhouette index and the davies-bouldin index in labelling ids clusters, in: Proc. 11th Nord. Work. Secur. IT Syst. sn, 2006, pp. 53–64.
[42] R. Guha, M. Ghosh, S. Kapri, S. Shaw, S. Mutsuddi, V. Bhateja, R. Sarkar, Deluge based genetic algorithm for feature selection, Evol. Intell. (2019) 1–11.
[43] M. Ghosh, R. Guha, R. Sarkar, A. Abraham, A wrapper-filter feature selection technique based on ant colony optimization, Neural Comput. Appl. (2019) 1–19, http://dx.doi.org/10.1007/s00521-019-04171-3.