A new penalty-based wrapper fitness function for feature subset selection with evolutionary algorithms

Basabi Chakraborty and Atsushi Kawamura

ABSTRACT
Feature subset selection is an important preprocessing task for any real life data mining or pattern recognition problem. Evolutionary computational (EC) algorithms are popular as a search algorithm for feature subset selection. With the classification accuracy as the fitness function, the EC algorithms end up with feature subsets having considerably high recognition accuracy but the number of residual features also remain quite high. For high dimensional data, reduction of number of features is also very important to minimize computational cost of overall classification process. In this work, a wrapper fitness function composed of classification accuracy with another penalty term which penalizes for large number of features has been proposed. The proposed wrapper fitness function is used for feature subset evaluation and subsequent selection of optimal feature subset with several EC algorithms. The simulation experiments are done with several benchmark data sets having small to large number of features. The simulation results show that the proposed wrapper fitness function is efficient in reducing the number of features in the final selected feature subset without significant reduction of classification accuracy. The proposed fitness function has been shown to perform well for high-dimensional data sets with dimension up to 10,000.

1. Introduction
Rapid progress of internet and communication technologies facilitates day by day generation of huge amount of data leading to the problem of data analysis and knowledge extraction, a big challenge, to the scientists and researchers. For efficient categorization, mining or classification, the data need to be preprocessed to retain the characteristic or discriminatory information while being free from redundant and irrelevant information. Feature selection aims to achieve the necessary preprocessing of the data and it is an important preprocessing task in the area of pattern recognition or data mining (Deviwij & Kittler, 1982; Liu & Motoda, 1998) prior to classification or clustering. A sample data or pattern in the paradigm of pattern recognition or machine learning is represented by a...
n-dimensional vector or a point in a n-dimensional space where individual dimension represents individual feature. Feature subset selection refers to the process of selecting a subset of $d$ features from the set of $n$ features by discarding irrelevant features and retaining discriminatory informative features. Reduction of features facilitates speedy processing of data and improves classification accuracy. Feature extraction process also reduces the dimensionality of the data by projecting original high-dimensional feature set to a lower dimensional set in which the new features are created instead of retaining a subset of original features. In this process the original features are not retained, new features are created. This paper focuses on feature selection paradigm where the selected original features are retained.

Basically feature subset selection process consists of two steps, an evaluation function is needed to be defined to evaluate the goodness of a feature or a feature subset in the first step and a search algorithm has to be decided to find out the best feature subset from all possible feature subsets according to the evaluation function in the final step. Depending on the nature of the evaluation function, the algorithms of feature subset selection are of two types, filter and wrapper. Filter algorithms evaluate the data set without reference to a particular classifier while wrapper algorithms use the classifier accuracy as the evaluation function and thus it is classifier dependent. Though filter algorithms are classifier independent and computationally efficient, they cannot always provide the best result for a particular classifier. The history of pattern recognition is long and early researches on feature selection evolved from the statistical community. A lot of statistical feature selection algorithms have been proposed so far (Duda et al., 2001). However, real world problems are often characterized by vagueness rather than randomness and are difficult to be modelled by rigid framework of mathematics or statistics.

Soft computing technologies emerged to bridge this gap and lots of algorithms based on neural computation, fuzzy logic, rough set theory, evolutionary algorithms have been proposed for feature selection and classification in the area of pattern recognition and data mining (Verikas & Bacauskiene, 2002; Oh et al., 2004; Chakraborty, 2002a). Evolutionary computational (EC) algorithms are well known tools for solving optimization problems and have been efficiently used for the search stage in feature subset selection problem. Among EC-based algorithms, Genetic Algorithms (GAs) (Goldberg, 1989), Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995) and Ant Colony Optimization (ACO) (Dorigo & Stuzle, 2004) are widely used for feature selection. Other less commonly used algorithms are Cuckoo Search (CS) (Yang & Deb, 2009), Gravitational Search Algorithm (GSA) (Reshedi et al., 2009a), Firefly Algorithm (FA) (Yang, 2008), Bat Algorithm (BA) (Yang, 2010) or Dragonfly Algorithm (Mirjalili, 2016). Various fitness functions needed for evolutionary search have been proposed which fall under both the filter and wrapper categories. Wrapper fitness function, mainly classification accuracy, generally produce better classification results than various filter fitness functions but the computational cost is high. Though both types of algorithms having wrapper and filter fitness function, respectively, with GA produce the optimum feature subset with high classification accuracy but none of them take care of the number of features in the final selected feature subset. For high-dimensional data, the number of features in the final selected subset has immense effect on the computational cost of the overall classification task.

In our previous work (Kawamura & Chakraborty, 2017), a comparative study of evolutionary computation (EC)-based feature subset selection algorithms with classification
accuracy as a wrapper fitness function (default) has been done and a preliminary proposal of a new penalty-based wrapper fitness function for simultaneous optimization of data dimension and classification accuracy has been considered with simulation experiments with some benchmark data sets from UCI repository. This work is an extended version of our previous work. In this work the proposal of the new penalty-based wrapper fitness function has been critically examined in terms of minimum number of features in the final selected feature subset with highest possible classification accuracy. The efficiency of the new fitness function over the default function in selecting optimal feature subset with minimum number of features in the final selected feature set by various evolutionary algorithms has been studied by simulation experiments with a larger number of benchmark data sets from UCI repository as well as WCCI 2006 and NIPS 2003 competition data sets. The effect of the parameter tuning of the proposed new function is also extensively studied with simulation experiments and the optimum values of the parameters for different algorithms are noted. The next section represents a brief overview of the algorithms used for study in this work and the related works in this direction of research. The following section describes comparative study of different EC algorithms and the proposal of the new penalty-based fitness function followed by the simulation experiments and results in the next section. The final section contains discussion and conclusion.

2. Related works on EC-based feature subset selection

Evolutionary algorithms are well known for solving optimization problems and are now becoming popular for using as the search algorithm for feature subset selection problem. GA and PSO are the most popular evolutionary algorithms used for feature subset selection. A brief review of the related works with EC algorithms used for feature subset selection in this paper are presented in the following subsections.

2.1. Genetic algorithm

GA, a randomized heuristic and adaptive search technique based on the principal of natural selection and the most popular evolutionary approach is a good candidate for solving optimization problems where the search space is large (Goldberg, 1989). In GA, a population of possible solutions, that is, the possible candidate feature subsets from a feature set of \( n \) features, encoded as a binary string of \( n \) bits, are maintained through several generations. In each generation, genetic operators such as crossover and mutation are used to generate new population from the most elite pairs of the current generation and the good ones are retained after evaluation by a fitness function. Through the generations, the population is led to the better solution space and finally produces the near optimal solution in the final generation. GA requires no domain knowledge and quite robust than other random or local search methods. The main steps of GA are as follows:

- Problem coding: the solution space is properly coded to represent the possible solutions of the problem as a string.
- Initial population generation: a number of solutions are randomly chosen for initial population.
• Genetic operation: three operations namely (a) selection (b) crossover and (c) mutation are applied to the population.
  ○ Selection is done based on evaluation of the population by a fitness function and a number of elite solutions are passed over to the next generation.
  ○ Crossover produces two offsprings from two selected parents from the population in such a way that one parent string is exchanged at some point of the string with the other parent string. The generated offsprings are added to the population instead of their parents to be carried over to the next generation.
  ○ Mutation produces some strings with one or two bits changed with some low probability.
• Fitness evaluation: the population is evaluated by a fitness function until an acceptable solution emerges, otherwise the process is repeated for genetic operation.

In binary version of genetic algorithm BGA, a population of possible solutions, that is, the possible candidate feature subsets from a feature set of $n$ features are encoded as a binary string of $n$ bits, where the features to be included are represented by 1 and the features to be deleted are encoded with 0.

Several research works for solving feature subset selection problem with binary version of genetic algorithm (BGA) have been reported. A survey of various proposals is summarized in (Chakraborty, 2010a). It is the most widely used evolutionary algorithm for feature subset selection. The fitness functions of both filter type and wrapper type are used with GA for feature subset selection. GAs with various filter type fitness functions are reported in Chakraborty (2002b, 2014) and Mahrooghy et al. (2012). Filter type fitness function with GA is used for optimum feature subset selection which is finally evaluated by classification accuracy of several classifiers. These research works mainly differ in the type of filter evaluation function such as probabilistic, correlation based, fuzzy set based, consistency based, etc. Wrapper fitness function with GA for feature subset selection can be found in Zhuo (2008) and Vignolo & Gerard (2017). These works mainly differ on the type of classifier used. In all cases, classifier accuracy is used as the fitness function. Recently multiobjective GA is also a popular candidate for optimal feature subset selection with conflicting objectives and are reported in several works (Khan & Baig, 2015; Chakraborty & Chakraborty, 2013; Kashyap, Das, Bhattacharjee, Halder, & Goswami, 2016).

2.2. Particle swarm optimization

Recently PSO, specially binary particle swarm optimization (BPSO) (Kennedy & Eberhart, 1997) have been also become popular for feature subset selection (Chakraborty, 2008, 2009, 2010b). PSO (Kennedy & Eberhart, 1995) is a population-based evolutionary algorithm. The conventional PSO algorithm begins by initializing a random swarm of $m$ particles in $d$-dimensional space characterizing candidate solution like GA. However PSO is motivated by simulation of social behaviour instead of survival of fittest and each particle is associated with a velocity. The particles fly through the search space, constantly adjusting their velocity according to corresponding particle’s experience and the particle’s neighbours’ experience. Each particle $X_i$ makes use of its individual memory and knowledge gained by the swarm as a whole to find the best solution. At each iteration, the fitness of each particle is evaluated by an appropriate fitness function and the algorithm
progressively stores and replaces two best values, called \( pbest \) and \( gbest \). \( pbest_i \) \((i = 1,2,\ldots,m)\) denotes the best position associated with the best fitness value achieved so far for each individual and \( gbest \) denotes the position corresponding to global best value.

The steps of the PSO algorithm is as follows:

- **Problem coding**: the solution space is properly coded to represent the solution by a point in a multidimensional space. The position \( X_i = (x_{i1}, x_{i2}, \ldots, x_{id}) \) for \( i = 1,2,\ldots,m \) and velocity \( V_i \), for \( i = 1,2,\ldots,m \) of all particles are set randomly within a prespecified range.

- **Updation**: velocity and position of each particle are updated. For each particle \( i \),
  
  For each dimension \( d \),
  
  \[
  V_{id}(t) = w * V_{id}(t-1) + c_1 * r_1 * (pbest_{id} - x_{id}(t-1)) + c_2 * r_2 * (gbest_d - x_{id}(t-1))
  \]
  
  \[
  X_{id}(t) = X_{id}(t-1) + V_{id}(t-1),
  \]

  where \( w \) is inertia weight, \( c_1 \) and \( c_2 \) are acceleration constants and \( r_1 \) and \( r_2 \) are random variables in the range \([0,1]\).

- **Evaluation and memory updation**: fitness of each particle is evaluated and the memory with updated with the best value.
  
  if \( f(x_i) < f(pbest_i) \)
  
  then Update \( pbest_i = x_i \)
  
  For \( k \) neighbourhood of \( x_i \)
  
  if \( f(x_k) < gbest \)
  
  then \( gbest = x_k \)

  where \( f \) represents the Fitness Evaluation function.

- **Termination check**: step 2 and step 3 are repeated until a specified termination condition is met.

BPSO algorithm, also proposed by Kennedy and Eberhart (Kennedy & Eberhart, 1997) is an extension of PSO to solve optimization problems with discrete valued parameters. Here each particle (candidate solution) represents a position in a binary multidimensional space, i.e., components of \( X_i \) can take only binary values instead of continuous values. The velocity vector associated with each particle is real valued. Binary PSO is used for feature subset selection where the population is coded in the same way as GA by binary strings where 0 represents absence of feature and the 1 represents presence of the feature. The position update rule is based on probability produced by normalization of velocity components using sigmoid function.

\[
X_{id}(t) = \begin{cases} 
1 & \text{if } r_3 < \frac{1}{1 + \exp^{-V_{id}(t-1)}} \\
0 & \text{otherwise}
\end{cases}
\]  

(1)

\( r_3 \) is a random number generated in the range \([0,1]\).

Most of the feature subset selection algorithms use BPSO. Depending on the type of the evaluation function used, wrapper based (Liu & Shang, 2013) filter based (Yang et al., 2008) and hybrid (Moradi & Gholampour, 2016) PSO algorithms are proposed for feature subset
2.3. Other EC algorithms

Recently new evolutionary algorithms other than GA and PSO are reported in the literature as search algorithms. These algorithms are also used for feature subset selection. A brief review of the other EC algorithms, which are also used for feature subset selection in this work, are presented here.

2.3.1. Gravitational search algorithm

Gravitational Search algorithm (GSA) is a nature inspired heuristic optimization algorithm based on the law of gravity and mass interactions. The algorithm is comprised of collection of agents which interact with each other through the gravity force. The agents are considered as objects and their performances are measured in terms of their masses. The gravity force causes a global movement where all objects move toward other objects with heavier masses. In GSA, the agent has four parameters which are position, inertial mass, active gravitational mass and passive gravitational mass. The position represent the solution of the problem. the gravitational and inertial masses are determined by fitness function. The algorithm is navigated by adjusting gravitational and inertial mass. Finally the position of the heaviest mass presents the optimum solution. The details are found in Reshedi et al. (2009a). A binary version of GSA, known as BGSA is found in Reshedi et al. (2009b). GSA and BGSA are used for feature subset selection in Han et al. (2014), Nagpal et al. (2017) and Behjat, Mustapha, Nezamabadi-Pour, Sulaiman, and Mustapha (2013), respectively.

2.3.2. Cuckoo search

Cuckoo search is an optimization algorithm belonging to the class of swarm intelligence (SI)-based algorithms like PSO. It is inspired by the obligate interspecific brood parasitism of some cuckoo species that lay their eggs in the nests of other host birds. In this behaviour of reproduction, there are two possible cases for a cuckoo egg dumped into a host bird nest including: the host bird does not recognize the cuckoo egg and the cuckoo egg will hatch and carry over to the next generation or the host bird identifies the cuckoo egg and either throw it away or abandon its nest to build a new one. The two mentioned phenomena have been inspired in the CSA method for two phases of new solution generation including the exploration phase via Levy flights (the first phenomenon) and the exploitation phase via replacement of a fraction of eggs (the second phenomenon). The detail algorithm is presented in Yang & Deb (2009). A binary version of the algorithm is presented in Rodrigues et al. (2013). Cuckoo search algorithm has been used for feature subset selection in Kulshrestha et al. (2015), Pereira et al. (2014) and Alia & Taweel (2017).

2.3.3. Firefly algorithm

Firefly algorithm (FFA) is also another SI-based optimization algorithm inspired by the flashing pattern of tropical fireflies. It is based on three rules: (1) the fireflies are unisex and one is attracted by other irrespective of sex (2) attractiveness is proportional to brightness, less brighter firefly moves to more brighter one, brightness decreases as their
distance increases (3) brightness is determined by landscape of the objective function. The objective function of a given optimization problem is based on differences of light intensity. The fireflies are characterized by light intensity which helps to change their position iteratively to more attracting position in order to obtain optimal solution. The details are in Yang (2008). A binary version of the algorithm BFFA is proposed in Crawford et al. (2014). Firefly and binary firefly algorithms are also used in feature subset selection in sentiment analysis and other problems in Kumar & Khorwal (2017) and Zhang et al. (2017), respectively.

2.3.4. Bat algorithm
Bat algorithm (BA) is a newly proposed swarm intelligence-based metaheuristic optimization algorithm based on echolocation behaviour of bats. Microbats, small bats, use extensive echolocation. They use a type of sonar, to detect prey and to avoid obstacles and locate their resting crevices in the dark. These bats emit a very loud sound pulse and listen for the echo that bounces back from the surrounding objects. Bat algorithm is a modification of PSO in which the position and the velocity of virtual microbats are updated based on frequency of their emitted pulses and loudness. The pseudocode of the algorithm and the details can be found in Yang (2010). A binary version of bat algorithm BBA is proposed in Nakamura et al. (2014). Recent applications of BA in feature subset selection are found in Yang et al. (2017) and Rani & Rajalaxmi (2015).

2.3.5. Dragonfly algorithm
Dragonfly algorithm proposed in Mirjalili (2016) is originated from the static and dynamic swarming behaviours of dragonflies in nature. Two essential phases of optimization, exploration and exploitation, are designed by modelling the social interaction of dragonflies in navigating, searching for foods, and avoiding enemies when swarming dynamically or statistically. Dragonflies create sub swarms and fly over different areas in a static swarm, which is the main objective of the exploration phase. In the static swarm, however, dragonflies fly in bigger swarms and along one direction, which is favourable in the exploitation phase. For simulating the swarming behaviour of dragonflies, the three primitive principles of swarming in insects proposed by Reynolds (1987) as well as two other new concepts. A binary version of the algorithm BDFA is also proposed in the same paper. Recent work on feature selection with dragonfly algorithm is presented in Mafaraja, Hammouri, Eleyan, and Mirjalili (2017).

3. Comparative study of evolutionary algorithms in feature subset selection
In this paper a comparative study of different EC algorithms for feature subset problem has been done. The solution space is considered as a binary multidimensional space and represented by a binary string or a binary vector, a point in binary multidimensional space. Binary versions of the algorithms, Binary Genetic Algorithm (BGA), BPSO, Binary Gravitational Search Algorithm (BGSA), Binary Cuckoo Search (BCS), Binary Firefly Algorithm (BFFA), Binary Bat Algorithm (BBA) and Binary Dragonfly Algorithm (BDFA) are used in this study. The fitness function of the EC algorithms is the classification accuracy of a
linear SVM classifier over test set samples with four-fold cross validation as a default wrapper fitness function which is defined as:

Fitness function $S_1 = \frac{\text{No. of test samples correctly classified (} T_c \text{)} }{\text{Total no. of test samples (} T \text{)}}$.

A new wrapper fitness function in which a penalty term based on number of features is added with classification accuracy is proposed and simulation experiments have been done with proposed fitness function and the results are compared with the default fitness function. The proposal is represented in details in the next subsection.

### 3.1. Proposal of a new fitness function with penalty

The objective of feature subset selection is two fold: to reduce the dimensionality of the data set to lower computational cost as well as to increase the classification accuracy to make the performance higher. But it seems that this two objectives are somewhat contradictory. Reduction of features leads to lower classification accuracy, so use of classification accuracy as the evaluation function of the optimization algorithm is not sufficient for obtaining optimally reduced feature subset. Two contradictory objectives are generally taken care of by using multiobjective GA. Here the contradiction is taken care of by using a single objective function. A new fitness function is proposed with the addition of a penalty term in $S_1$. The new fitness function $S_2$ is given by

$$S_2 = S_1 - \alpha \times \frac{D}{N}, \quad (2)$$

where $D$ and $N$ represent the number of features in the selected feature subset and total number of features, respectively, whereas $\alpha$ is a control parameter used to adjust the weight of the penalty term in the fitness function. The above fitness function is used in conjunction with various evolutionary algorithms to find out the optimal feature subset. The performance of the new fitness function $S_2$ is compared to the performance of the default fitness function $S_1$ in terms of the final reduction of feature set and classification accuracy. The performance of the new fitness function with various EC algorithms (single objective) is also compared with the performance of NSGA II, a popular multiobjective GA (Deb et al., 2002) using classification accuracy and reduction of features in the feature set as two separate objectives.

### 4. Simulation experiments and data sets

Simulation experiments are done with several benchmark data sets from UCI machine learning repository (1980), NIPS (2003) and WCCI (2006) competition data. The details of the data sets used here are presented in Table 1.

The EC algorithms used in the simulation experiments are BGA, BPSO, BGSA, BCS, BFFA, BBA and BDFA. The parameters of different algorithms are set by trial and error so that the maximum number of comparison in the search algorithms are 10,000. Table 2 represents the parameters used. For BGA, two point crossover and rank-based selection is used. $P_c$ and $P_m$ represent probability of crossover and mutation, respectively. For other algorithms,
relevant parameter values (details are omitted here due to lack of space, can be found in the references) are noted in Table 2. BDFA requires no such parameters to be set.

All the data sets are used for experiment with default fitness function and penalty-based fitness function with control parameter $\alpha = 0.05$ to 0.4 for feature subset selection by different EC algorithms. Finally SVM is used for measuring classification accuracy with the final reduced subset. Different training-test ratio of samples are used for experiments. The evaluation of the fitness function is done by the final number of features in the reduced feature subset and average classification accuracy with the final selected subset.

5. Simulation results

Tables 3–12 represent the simulation results for 10 benchmark data sets, respectively, in increasing order of no. of features. Column 2 and column 3 of each table represent the average classification accuracy of the selected feature subset with SVM classifier over test samples with default wrapper fitness function and the proposed new fitness

### Table 1. Data set details.

| Name    | No. of class | No. of features | No. of train samples | No. of test samples |
|---------|--------------|-----------------|----------------------|--------------------|
| Wine    | 3            | 13              | 89                   | 89                 |
| Ada     | 2            | 48              | 4147                 | 415                |
| Cancer  | 2            | 32              | 285                  | 284                |
| Sonar   | 2            | 60              | 104                  | 104                |
| Hill    | 2            | 100             | 606                  | 606                |
| Gas     | 6            | 128             | 6955                 | 6955               |
| Sylva   | 2            | 216             | 13,086               | 1308               |
| Madelon | 2            | 500             | 2000                 | 600                |
| Gina    | 2            | 970             | 3125                 | 315                |
| Arcene  | 2            | 10,000          | 100                  | 800                |

### Table 2. Parameters of EC algorithms.

| Algorithm | Population size | Maximum epoch | Parameter values |
|-----------|-----------------|---------------|------------------|
| BGA       | 8               | 1250          | $P_c = 0.1, P_m = 0.05$ |
| BPSO      | 20              | 500           | $c_1, c_2 = 1, w = 0.5$ |
| BCS       | 20              | 500           | $\alpha = 0.1, \beta = 1.5, \rho = 0.25$ |
| BGSA      | 20              | 500           | $P_f = E, = 1, \text{min-\ flag} = 0$ |
| BFFA      | 20              | 25            | $\alpha = 0.25, \beta = 0.2, \gamma = 1$ |
| BBA       | 20              | 500           | loudness = 0.25, $r = 0.1$ |
| NSGA      | 20              | 250           | same as BGA       |

### Table 3. Simulation results for Wine data.

| Algorithm | Default fitness | Fitness with penalty |
|-----------|-----------------|----------------------|
|           | Accuracy        | No. of features | Accuracy | No. of features |
| BGA       | 0.93034         | 8.2              | 0.91867  | 4.9         |
| BPSO      | 0.92674         | 5.6              | 0.91386  | 3.8         |
| BCS       | 0.92740         | 7.0              | 0.90476  | 4.6         |
| BFFA      | 0.92394         | 5.4              | 0.92242  | 3.7         |
| BBA       | 0.93172         | 6.1              | 0.91921  | 3.9         |
| BGSA      | 0.91530         | 6.7              | 0.91332  | 4.0         |
| BDFA      | 0.92481         | 5.9              | 0.91546  | 3.8         |
function, respectively. Column 4 and 5 of each table represent the number of features in the final selected feature subset with default wrapper fitness function and the proposed fitness function, respectively. All the tables represent simulation results for control parameter $\alpha = 0.15$ for the proposed fitness function as this value seems to be the most appropriate. For Hill and Madelon data sets, simulation results for only three evaluation algorithms are shown, the results for others were not satisfactory probably due to poor parameter setting. For high-dimensional data sets, it is found that the

Table 4. Simulation results for Ada data.
\[
\begin{array}{lcc}
\text{Algorithm} & \text{Default fitness} & \text{Fitness with penalty} \\
& \text{Accuracy} & \text{No. of features} & \text{Accuracy} & \text{No. of features} \\
BGA & 0.77749 & 31.4 & 0.79683 & 10.4 \\
BPSO & 0.83080 & 26.1 & 0.83437 & 5.8 \\
BCS & 0.82462 & 29.7 & 0.80279 & 15.6 \\
BFFA & 0.82859 & 14.8 & 0.82867 & 7.3 \\
BBA & 0.83054 & 27.9 & 0.82831 & 6.8 \\
BGSA & 0.82351 & 27.2 & 0.79651 & 16.9 \\
BDFA & 0.82579 & 25.2 & 0.83554 & 8.6 \\
\end{array}
\]

Table 5. Simulation results for Cancer data.
\[
\begin{array}{lcc}
\text{Algorithm} & \text{Default fitness} & \text{Fitness with penalty} \\
& \text{Accuracy} & \text{No. of features} & \text{Accuracy} & \text{No. of features} \\
BGA & 0.94933 & 14.8 & 0.94476 & 7.6 \\
BPSO & 0.96632 & 13.4 & 0.96059 & 4.2 \\
BCS & 0.94965 & 13.1 & 0.94004 & 8.8 \\
BFFA & 0.96061 & 8.4 & 0.95193 & 3.7 \\
BBA & 0.96488 & 13.3 & 0.95658 & 4.4 \\
BGSA & 0.95173 & 13.5 & 0.94412 & 8.6 \\
\end{array}
\]

Table 6. Simulation results for Sonar data.
\[
\begin{array}{lcc}
\text{Algorithm} & \text{Default fitness} & \text{Fitness with penalty} \\
& \text{Accuracy} & \text{No. of features} & \text{Accuracy} & \text{No. of features} \\
BGA & 0.85392 & 30.16 & 0.94476 & 24.65 \\
BPSO & 0.88224 & 30.51 & 0.96059 & 17.87 \\
BCS & 0.82344 & 30.48 & 0.94004 & 21.23 \\
BFFA & 0.83592 & 18.82 & 0.95193 & 10.56 \\
BBA & 0.88808 & 29.88 & 0.95658 & 16.11 \\
BGSA & 0.84312 & 31.62 & 0.94412 & 22.0 \\
\end{array}
\]

Table 7. Simulation results for Hill data.
\[
\begin{array}{lcc}
\text{Algorithm} & \text{Default fitness} & \text{Fitness with penalty} \\
& \text{Accuracy} & \text{No. of features} & \text{Accuracy} & \text{No. of features} \\
GA & 0.95987 & 54.18 & 0.95234 & 45.7 \\
BFFA & 0.93545 & 47.7 & 0.92666 & 34.66 \\
BBA & 0.97677 & 55.36 & 0.97059 & 39.5 \\
\end{array}
\]
penalty-based fitness function works better than the default fitness function in reducing the number of features in optimal feature subset, though the classification accuracy falls drastically for too high-dimensional data after reducing the dimension to 50% (Madelon data set).

In all the cases, new fitness function produce final feature subset with lesser number of features without much degradation in classification accuracy. Also it is found that BFFA

| Algorithm | Default fitness | Fitness with penalty |
|-----------|-----------------|----------------------|
|           | Accuracy | No. of features | Accuracy | No. of features |
| BGA       | 0.83234  | 95.81           | 0.79315  | 94.66           |
| BPSO      | 0.83777  | 73.00           | 0.91423  | 45.50           |
| BCS       | 0.80874  | 85.80           | 0.86729  | 15.00           |
| BFSA      | 0.96626  | 30.00           | 0.97757  | 30.00           |
| BBA       | 0.83549  | 88.40           | 0.68620  | 59.00           |
| BGSA      | 0.75377  | 108.33          | 0.86679  | 67.25           |
| BDFA      | 0.95508  | 62.80           | 0.87822  | 44.00           |

| Algorithm | Default fitness | Fitness with penalty |
|-----------|-----------------|----------------------|
|           | Accuracy | No. of features | Accuracy | No. of features |
| BGA       | 0.97949  | 182.67          | 0.96244  | 44.50           |
| BPSO      | 0.97643  | 122.00          | 0.96493  | 48.37           |
| BCS       | 0.97783  | 143.17          | 0.85117  | 23.33           |
| BFSA      | 0.97564  | 139.00          | 0.97749  | 24.11           |
| BBA       | 0.98111  | 137.43          | 0.96588  | 26.25           |
| BGSA      | 0.97350  | 169.83          | 0.97129  | 73.88           |
| BDFA      | 0.98191  | 133.83          | 0.97143  | 54.25           |

| Algorithm | Default fitness | Fitness with penalty |
|-----------|-----------------|----------------------|
|           | Accuracy | No. of features | Accuracy | No. of features |
| GA        | 0.65150  | 250.76          | 0.59433  | 246.3           |
| BFSA      | 0.62129  | 125.32          | 0.59141  | 73.31           |
| BBA       | 0.67824  | 249.72          | 0.62487  | 242.38          |

| Algorithm | Default fitness | Fitness with penalty |
|-----------|-----------------|----------------------|
|           | Accuracy | No. of features | Accuracy | No. of features |
| BGA       | 0.79788  | 637.44          | 0.76190  | 307.87          |
| BPSO      | 0.79965  | 457.44          | 0.79286  | 375.75          |
| BCS       | 0.79955  | 545.00          | 0.77302  | 250.37          |
| BFSA      | 0.77778  | 298.50          | 0.78413  | 210.62          |
| BBA       | 0.79524  | 407.17          | 0.80159  | 272.37          |
| BGSA      | 0.80847  | 385.83          | 0.79643  | 373.75          |
| BDFA      | 0.80952  | 459.17          | 0.81270  | 344.86          |
Table 12. Simulation results for Arcene data.

| Algorithm | Default fitness | Fitness with penalty |
|-----------|-----------------|----------------------|
|           | Accuracy        | No. of features      | Accuracy        | No. of features      |
| BGA       | 0.81909         | 5152.72              | 0.81923         | 2498.23              |
| BPSO      | 0.81364         | 4809.36              | 0.82769         | 3775.07              |
| BCS       | 0.82091         | 5435.36              | 0.81714         | 1550.28              |
| BFFA      | 0.79727         | 1224.36              | 0.81929         | 1050.85              |
| BBA       | 0.82182         | 3383.00              | 0.82077         | 1736.77              |
| BGSA      | 0.82091         | 4253.45              | 0.82786         | 4048.28              |
| BDFA      | 0.82545         | 4409.73              | 0.82846         | 3240.00              |

Figure 1. Effect of control parameter on classification accuracy and number of features for Wine data set. (a) Effect of control parameter on classification accuracy and (b) effect of control parameter on number of features.
produced the highest reduction in feature set and it is the most effective EC algorithm. For comparison with multiobjective algorithm, we used NSGA II with two objective functions, classification accuracy and number of features in the selected feature subset. The classification accuracy and number of features in the final selected subset for wine, cancer and sonar data set came out to be 0.95, 0.95, 0.85 and 8.07, 14.65 and 29.5, respectively. It seems that our proposed penalty-based single objective fitness function is better in efficiency than NSGAIi in terms of reducing feature number in the final feature subset without having much degradation in classification accuracy.

Figure 2. Effect of control parameter on classification accuracy and number of features for Ada data set. (a) Effect of control parameter on classification accuracy and (b) effect of control parameter on number of features.
5.1. Effect of control parameter

Simulation experiments are done for different values (from 0.05 to 0.5) of the control parameter $\alpha$ with different data sets and different algorithms. Figures 1–4 represent the simulation results for Wine, Ada, Sylva and Arcene data sets, respectively. From the simulation results it is seen that the value of classification accuracy and number of features become stabilized around $\alpha = 0.15$ for most of the data sets. So this value is considered as the most appropriate.

**Figure 3.** Effect of control parameter on classification accuracy and number of features for Sylva data set. (a) Effect of control parameter on classification accuracy and (b) effect of control parameter on number of features.

5.1. Effect of control parameter $\alpha$

Simulation experiments are done for different values (from 0.05 to 0.5) of the control parameter $\alpha$ with different data sets and different algorithms. Figures 1–4 represent the simulation results for Wine, Ada, Sylva and Arcene data sets, respectively. From the simulation results it is seen that the value of classification accuracy and number of features become stabilized around $\alpha = 0.15$ for most of the data sets. So this value is considered as the most appropriate.
6. Conclusion

Optimal feature subset selection is an extremely important preprocessing step for any pattern recognition or machine learning problem. The successful elimination of redundant and irrelevant information increases the performance of the classifier while retaining important and informative feature is highly needed for improved performance. For high-dimensional data, the judicious selection of feature subset from available features becomes more important as reduction of features having relevance to the class reduces classification accuracy while retaining all the features heavily increases the computational

Figure 4. Effect of control parameter on classification accuracy and number of features for Arcene data set. (a) Effect of control parameter on classification accuracy and (b) effect of control parameter on number of features.
cost. So feature evaluation function should be carefully designed. In search algorithm-based optimal feature subset selection, feature evaluation function is used as the fitness function of the search algorithm. For wrapper method, classification accuracy itself is generally used as the default fitness function of the search algorithm.

In this work various evolutionary algorithm has been used for searching the optimal feature subset from a set of features with classification accuracy as the default fitness function and a newly proposed fitness function in which a penalty term for high number of features in the selected subset is added to classification accuracy. The new penalty function seems to be very effective in reducing the number of features in the selected subset without much degradation in classification accuracy. The proposed fitness function also seems to be effective compared to multiobjective GA. Among the EC algorithms, BFFA produced the best result in terms of reduction of number of features. Simulation experiments with high-dimensional data also show that the algorithms are quite effective for data sets of dimension up to the range of 10,000. As high-dimensional data needs more computation time, proper reduction of dimension without degradation in classification accuracy is very much desirable for reduction of computational time.

Disclosure statement
No potential conflict of interest was reported by the authors.

References
Alia, A. F., & Taweel, A. (2017). Feature selection based on hybrid binary cuckoo search and rough set theory in classification for nominal datasets. *International Journal of Information Technology and Computer Science, 4*, 63–72. Published Online April 2017 in MECS. Retrieved from http://www.mecs-press.org

Behjat, A. R., Mustapha, A., Nezamabadi-Pour, H., Sulaiman, N., & Mustapha, N. (2013). Feature subset selection using binary gravitational search algorithm for intrusion detection system. *Proceedings of ACIIDS, 2013*, 377–386.

Chakraborty, B. (2002a). Genetic algorithm with fuzzy operators for feature subset selection. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E85-A*(9), 2089–2092.

Chakraborty, B. (2002b). Genetic algorithm with fuzzy fitness function. Proceedings of IEEE international symposium on industrial electronics (IEEE-ISIE’2002), pp. 315–331.

Chakraborty, B. (2008). *Feature subset selection by particle swarm optimization with fuzzy fitness function*. Proceedings ISKE 2008 (3rd. international conference on intelligent systems and knowledge engineering), pp. 1038–1042.

Chakraborty, B. (2009). *Binary particle swarm optimization based algorithm for feature subset selection*. Proceedings of ICAPR09, pp. 145–148.

Chakraborty, B. (2010a). Evolutionary computational approaches to feature subset selection. *International Journal of Soft computing and Bioinformatics, 1*(2), 59–65.

Chakraborty, B. (2010b). Feature subset selection using hybrid particle swarm optimization. *ICIC Express Letters, 4*(5), 1161–1168.

Chakraborty, G., & Chakraborty, B. (2013). *Multi-objective optimization using pareto GA for gene-selection from microarray data for disease classification*. Proceedings of the IEEE SMC 2013, pp. 2629–2634, Manchester, UK, October.

Chakraborty, B. (2014). *Rough fuzzy consistency measure with evolutionary algorithm for attribute reduction*. Proceedings IEEE SMC 2014 (pp. 723–728), San Diego, USA, October.
Crawford, B., Ricardo, S., Olivares-Suarez, M., Palma, W., Paredes, F., Olguin, E., & Norero, E., (2014). A binary coded firefly algorithm that solves the set covering problem. *Romanian Journal of Information Science and Technology, 17*(3), 252–264.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 2*(3), 182–197.

Devijver, P. A., & Kittler, J. (1982). *Pattern recognition: A statistical approach*. Upper Saddle River, NJ: Prentice Hall International.

Dorigo, M., & Stutzle, T. (2004). *Ant colony optimization*. Cambridge, MA: The MIT Press.

Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern classification*. Hoboken, NJ: Wiley-Interscience.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Boston, MA: Addison-Wesley.

Han, X., Chang, X., Quan, L., Xiong, X., Li, J., Zhang, Z., & Liu, Y. (2014). Feature subset selection by gravitational search algorithm optimization. *Information Sciences, 281*, 128–146.

Kashyap, H., Das, S., Bhattacharjee, J., Halder, R., & Goswami, S. (2016). Multi-objective genetic algorithm setup for feature subset selection in clustering. *Proceedings of 3rd international conference on recent advances in information technology (RAIT)*, pp. 243–247.

Kawamura, A., & Chakraborty, B. (2017). A comparative study of evolutionary algorithms with a new penalty based fitness function for feature subset selection. In N. Nguyen, S. Tojo, L. Nguyen & B. Trawiński (Eds.), *Lecture notes in computer science: Vol. 10191. Intelligent information and database systems. AICIDS 2017* (pp. 738–747). Cham: Springer.

Kennedy, J., & Eberhart, R. C. (1995). *Particle swarm optimization*. Proceedings IEEE international conference on neural network 4, pp. 1942–1948.

Kennedy, J., & Eberhart, R. C. (1997). *A discrete binary version of the particle swarm algorithm*. Proceedings IEEE international conference on computational cybernetics and simulation, Vol. 5, pp. 4104–4108.

Khan, A., & Baig, A. R. (2015). Multi-objective feature subset selection using non-dominated sorting genetic algorithm. *Journal of Applied Research and Technology, 13*(1), 145–159.

Kulshestha, G., Agarwal, A., Mittal, A., & Sahoo, A. (2015). Hybrid cuckoo search algorithm for simultaneous feature and classifier selection. *Proceedings of cognitive computing and information processing CCIP*, pp. 1–6.

Kumar, A., & Khorwal, R. (2017). Firefly algorithm for feature selection in sentiment analysis. In H. Behera, & D. Mohapatra (Eds.), *Advances in intelligent systems and computing: Computational intelligence in data mining* (Vol. 556, pp. 693–703). Singapore: Springer.

Liu, H., & Motoda, H. (1998). *Feature selection for knowledge discovery and data mining*. Berlin: Springer.

Liu, X., & Shang, L. (2013). A fast wrapper feature subset selection method based on binary particle swarm optimization. *Proceedings of IEEE congress on evolutionary computation, 2013*, pp. 3347–3352.

Mafaraja, M. M., Hammouri, A. I., Eleyan, D., & Mirjalili, S. (2017). Binary dragonfly algorithm for feature selection. *Proceedings of international conference on new trends in computing sciences (ICTCS2017)*, pp. 12–17.

Mahrooghy, M., Younan, N. H., Anantharaj, V. G., Aanstoos, J., & Yarahmadian, S. (2012). On the use of the genetic algorithm filter-based feature selection technique for satellite precipitation estimation. *IEEE Geoscience and Remote Sensing Letters, 9*(5), 963–967.

Mirjalili, S. (2016). Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications, 27*(4), 1053–1073.

Moradi, P., & Gholampour, M. (2016). A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy. *Applied Soft Computing Archive, 43*(C June), 117–130.

Nagpal, S., Arora, S., Dey, S., Shreya, (2017). Feature selection using gravitational search algorithm for biomedical data. *Procedia Computer Science, 115*, 258–265.

Nakamura, Y. M., Pereira, L. A. M., Costa, K. A., Rodrigues, D., Papa, J. P., & Yang, X. S. (2014). Binary bat algorithm. *Neural Computing and Applications, 25*(3), 663–681.
NIPS. (2003). *Workshop on feature extraction*. Whistler, BC. Retrieved from http://clopinet.com/isabelle/Projects/NIPS2003/

Oh, I., Lee, J., & Moon, B. (2004). Hybrid genetic algorithms for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 26*(11), 1424–1437.

Pereira, L. A. M., Rodrigues, D., Almeida, T. N. S., & Papa, J. P. (2014). A binary cuckoo search and its application for feature selection. In X. S. Yang (Ed.), *Studies in computational intelligence: Vol 516. Cuckoo search and firefly algorithm* (pp. 141–154). Cham: Springer.

Rani, A. S. S., & Rajalaxmi, R. R. (2015). Unsupervised feature selection using binary bat algorithm. *Proceedings of IECES 2015*, pp. 451–456.

Reshedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009a). GSA: A gravitational search algorithm. *Information Sciences, 179*(13), 2232–2248.

Reshedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009b). BGSA: Binary gravitational search algorithm. *Natural Computing, 9*(3), 727–745.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM SIGGRAPH Computer Graphics, 21*(4), 25–34.

Rodrigues, D., Pereira, L. A. M., Souza, N., Ramos, C. C. O., & Yang, X. S. (2013). *A binary cuckoo search algorithm for feature selection*. 2013 IEEE international symposium on circuits and systems, pp. 465–468.

Tran, B., Xue, B., & Zhang, M. (2014). Overview of particle swarm optimisation for feature selection in classification. In G. Dick et al. (Eds.), *LNCS: Vol. 8886. SEAL 2014* (pp. 605–617).

UCI Data base. (1980). Machine learning repository. Retrieved from https://archive.ics.uci.edu/ml/datasets.html.

Verikas, A., & Bacauskiene, M. (2002). Feature selection with neural networks. *Pattern Recognition Letters, 23*(11), 1323–1335.

Vignolo, L. D., & Gerard, M. F. (2017). Evolutionary local improvement on genetic algorithms for feature selection. *Proceedings de XLIII CLEI – 46 JAIIO*, pp. 1–8.

WCCI. (2006). Performance prediction challenge. Retrieved from http://www.modelselect.inf.ethz.ch/datasets.php.

Yang, X. S. (2008). *Nature inspired metaheuristic algorithms*. Luniver Press.

Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. In J. R. Gonzalez et al. (Eds.), *Studies in computational intelligence: Vol. 284. Nature inspired cooperative strategies for optimization (NISCO 2010)* (pp. 65–74). Springer Berlin: Springer.

Yang, C. S., Chuang, L. Y., Ke, C. H., & Yang, C. H. (2008). Boolean binary particle swarm optimization for feature selection. *IEEE congress on evolutionary computation (CEC 2008)* (pp. 2093–2098).

Yang, X. S., & Deb, S. (2009). *Cuckoo search via levy flights*. Proceedings NaBIC 2009, pp. 210–214.

Yang, B., Lu, Y., Zhu, K., Yang, G., Liu, J., & Yin, H. (2017). Feature selection based on modified bat algorithm. *IEICE TRANSACTIONS on Information and Systems, E100-D*(8), 1860–1869.

Zhang, Y., Gong, D., Sun, X., & Guo, Y. (2017). A PSO-based multi-objective multi-label feature selection method in classification. *Scientific Reports 7*, Article number: 376.

Zhang, Y., Song, X.-F., & Gong, D.-W. (2017). A return-cost-based binary firefly algorithm for feature selection. *Information Sciences, 418–419*, 561–574.

Zhuo, L. (2008). A genetic algorithm based wrapper feature selection method for classification of hyperspectral images using support vector machine. In *The international archives of the photogrammetry, remote sensing and spatial information sciences* (Vol. XXXVII, pp. 397–402). Part B7. Beijing.