A Binary ABC Algorithm Based on Advanced Similarity Scheme for Feature Selection

Emrah Hancer[a,b,*], Bing Xue[b], Dervis Karaboga[a], Mengjie Zhang[b]

[a] *Computer Engineering, Erciyes University, Kayseri 38039, Turkey*
[b] *Evolutionary Computation Research Group, School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6140, New Zealand*

Abstract

Feature selection is the basic pre-processing task of eliminating irrelevant or redundant features through investigating complicated interactions among features in a feature set. Due to its critical role in classification and computational time, it has attracted researchers' attention for the last five decades. However, it still remains a challenge. This paper proposes a binary artificial bee colony (ABC) algorithm for the feature selection problems, which is developed by integrating evolutionary based similarity search mechanisms into an existing binary ABC variant. The performance analysis of the proposed algorithm is demonstrated by comparing it with some well-known variants of the particle swarm optimization (PSO) and ABC algorithms, including standard binary PSO, new velocity based binary PSO, quantum inspired binary PSO, discrete ABC, modification rate based ABC, angle modulated ABC, and genetic algorithms on ten benchmark datasets. The results show that the proposed algorithm can obtain higher classification performance in both training and test sets, and can eliminate irrelevant and redundant features more effectively than the other approaches. Note that all the algorithms used in this paper except for standard binary PSO and GA are employed for the first time in feature selection.

Keywords: Feature selection, artificial bee colony, particle swarm optimization, classification.

1. Introduction

Thanks to the rapid development in computer hardware and software, a huge amount of information can be collected and included in datasets through a large number of features (attributes). However, not all features are relevant to the target concept. In other words, datasets may include irrelevant and redundant features besides relevant ones. Unfortunately, these features may adversely affect the classification performance due to the large search space, known as the

*Corresponding Author. Tel.:+903522076666/32583
E-mail address: emrahhancer@erciyes.edu.tr (E.Hancer)

curse of dimensionality [1, 2]. Furthermore, more features may introduce more noise to the dataset that can be also detrimental to the classification performance. Thus, it is important to select an appropriate feature subset from the available features to achieve similar or even better classification performance than using all features [3]. The task is terminologically known as feature selection. It does not only achieve better classification accuracy, but also improves the efficiency, reduces data complexity, and simplifies the structure of the learnt classifiers [2].

Feature selection is one of the most difficult tasks in data mining and classification due to the feature interaction and the large search space [4, 5]. Feature interaction may be appeared as two-way, three-way or may involve even more features. For instance, a feature by itself may not have a confident effect to the target, but its effect can be increased when used together with other features. Also, a feature which is individually relevant may become redundant when interconnected with others. The other challenging task is the large search space, 2^n, where n is the total number of features. In other words, it is not possible to thoroughly search all possible solutions in most cases. Although a variety of search methods such as sequential forward and backward feature selection (SFS, SBS) [6, 7] have been proposed, they may converge to local minima or cost high computational time.

To address these tasks, evolutionary computation (EC) techniques have been used as a strong alternative to the classical search methods due to their global search potentials. Particle swarm optimization (PSO) [8, 9], genetic algorithms (GAs) [10, 11], genetic programming (GP) [12, 13], and ant colony optimization (ACO) [14, 15] have been widely applied to feature selection. In this study, artificial bee colony (ABC) [16] based on foraging behaviour of honey bees is chosen as the main motivation to address feature selection problems on account of the following advantages when compared to the other well-known EC techniques [17]: 1) It can converge more quickly to the target, 2) It is computationally less expensive, and 3) It is one of the most recent EC techniques. The idea of applying ABC to feature selection is not a novel subject, i.e., there exist some studies concerning the ABC based feature selection [18, 19, 20]. However, the existing studies unfortunately have not demonstrated a comprehensive experimental study, including comparison with recent EC variants, on a variety of datasets or thorough performance evaluation and analysis. Therefore, the potential of ABC for feature selection has not been fully demonstrated and the need for the studies based on ABC has not come to an end.

1.1. Goals

The overall goal of this paper is to propose an improved binary version of the artificial bee colony (ABC) algorithm to address feature selection problems. To achieve this goal, the discrete binary ABC (DisABC) algorithm [21] based on the similarity of Jaccard Coefficient among individuals is further improved by introducing the neighborhood selection mechanism of the differential evolution (DE) strategy. In other words, the similarity based search approach is re-simulated according to the DE mutation, recombination and selection strategies.

The other goal is to put forward a comprehensive comparative study of some variants of the ABC, PSO and GA algorithms on wrapper feature selection in terms of the classification performance and the feature subset size for the future studies of researchers. To establish the second goal, seven algorithms, which are binary PSO (BPSO) [22], new velocity based binary PSO (NBPSO) [23], quantum inspired binary PSO (QBPSO) [24], discrete ABC (DisABC) [21], angle modulated ABC (AMABC) [25], modification rate based ABC (MRABC) [26] and genetic algorithms (GA) [27] are employed, and ten benchmark datasets, including various classes, instances and features are chosen from the UCI machine learning repository [28]. Further, two recently published ACO studies are considered to evaluate the performance of the proposed ABC variant. To our knowledge, the employed algorithms except for BPSO and GA are the first time to be used in feature selection, and a comprehensive comparative analysis on feature selection is not very common in the literature. Specifically, the following points are investigated:

- whether integrating a differential evolution search mechanism to the DisABC algorithm improves its global search ability in feature selection tasks,
- whether the proposed algorithm is able to perform well in both training and test sets in terms of the classification rate when compared with the seven existing algorithms,
- whether the proposed algorithm can more effectively remove redundant or irrelevant features and can obtain better feature subsets than the seven existing algorithms, and
- whether the proposed algorithm performs better than conventional deterministic feature selection approaches.

1.2. The organization of the paper

The rest of the paper is organized as follows. Section 2 gives an outline of the basic ABC algorithm and provides a background on recent studies related to feature selection. Section 3 presents the proposed algorithm and Section 4 describes the experimental design. Section 5 presents the experimental results and discussions. Section 6 concludes the study and provides an insight into the future trends.

2. Background

In this section, background on the artificial bee colony and recent trends of the feature selection are presented.

2.1. Artificial Bee Colony

Artificial bee colony (ABC) that mimics the foraging behaviours of honey bee colony was proposed by Karaboga in 2005 [29]. From the perspective of an optimization problem, the food sources and their nectar amounts represent probable solutions and their corresponding fitness values, respectively. The ABC for a minimization problem can be explained as follows. Employed bees exploit their associated food sources explored before and share the information concerning quality and position of food sources with onlooker bees via waggle dance. Onlooker bees waiting on the hive make decision on the selection of food source to be exploited with the help of the information gained by employed bees. Scout bees are responsible for searching a new food source depending on an internal rule or possible external clues [30, 31]. The basic implementation of ABC comprises of four phases:

1) Initialization Phase. Accepting the search space as the environment of food sources available for the exploration and exploitation processes, the algorithm first randomly produces food sources. Each food source defined as $X_i = \{x_{i1}, x_{i2}, x_{i3}, ..., x_{ij}, ..., x_{iD}\}$ is generated by:

$$x_{ij} = x_j^{min} + U(0,1)(x_j^{max} - x_j^{min}) \tag{1}$$

where $i = \{1, 2, ..., SN\}$ and SN is the number of food sources; $j = \{1, 2, ..., D\}$; $U(0,1)$ is the random variable uniformly distributed between (0,1); D is the dimensionality of the search space; x_j^{min} and x_j^{max} are predefined minimum and maximum values of parameter j.

2) Employed Bee Phase. Between employed bees and food sources, one-to-one and on-to relation is established, i.e., each employed bee is associated with only one food source. An employed bee modifies the position of its concerning food source to find a new richer food source:

$$\upsilon_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{2}$$

where i represents the index of current food source (X_i); k represents the index of neighbor food source (X_k), which is randomly chosen among all sources except for i; j is the randomly selected parameter for modification; V_i is the generated food source determined by modifying one parameter of X_i; and ϕ_{ij} is a random number uniformly distributed within $[-1, 1]$. After V_i is generated, its fitness value is evaluated. If the fitness value of V_i is better than the fitness value of X_i, the employed bee will memorize the new food source position and leave the old one, and its counter holding the number of trials is reset to 0. Otherwise, the current food source is kept in memory and its counter holding the number of trials is increased by 1.

3) Onlooker Bee Phase. After getting the information concerning nectar amount (fitness value) and positions of food sources from employed bees via waggle dance, each onlooker bee selects a food source depending on the probability according to the fitness values through roulette-wheel scheme, where richer food sources have a higher probability than others. The selection scheme based on fitness values is given by:

$$p_i = \frac{fitness_i}{\sum\limits_{i=1}^{SN} fitness_i} \tag{3}$$

where $fitness_i$ is the fitness value of source X_i. After calculation of probability (p_i) value, a random number in the range of 0 and 1 ($rand(0,1)$) is generated for each food source i. If $p_i > rand(0,1)$, X_i is chosen and then the searching-exploiting process on X_i is carried out as in the employed bee phase.

4) Scout bee phase. It is known that after being exploited repeatedly, food sources should be left by bees to avoid waste of energy. In basic ABC, a food source is assumed as abandoned when its counter holding the number of trials exceeds the predefined value, known as the "limit" parameter. Then, a new food source is generated by Eq. (1) to replace the abandoned one.

2.2. Jaccard Similarity Coefficient

The binary similarity/dissimilarity measures play a critical role in many applications, such as classification, clustering and image retrieval [32]. Over the century, there has been a significant effort on consistently measuring the similarity among binary vectors resulting numerous similarity and dissimilarity measures in various fields. One of the most well-known similarity measures is Jaccard Coefficient [33], defined by:

$$Similarity\,(X_i, X_k) = \frac{M_{11}}{M_{11} + M_{10} + M_{01}} \tag{4}$$

where X_i and X_k are D dimensional binary vectors and the dth bit of X_i is represented by x_{id} ($x_{id} \epsilon \{0,1\}$); M_{11} is the number of bits where $x_{id} = x_{kd} = 1$; M_{10} is the number of bits where $x_{id} = 1$ and $x_{kd} = 0$; and M_{01} is the number of bits where $x_{id} = 0$ and $x_{kd} = 1$.

The dissimilarity measure between X_i and X_k is defined by:

$$Dissimilarity\,(X_i, X_k) = 1 - Similarity\,(X_i, X_k) = 1 - \frac{M_{11}}{M_{11} + M_{10} + M_{01}} \tag{5}$$

2.3. Discrete Binary ABC

Kashan et al. [21] introduced a discrete ABC (DisABC) based on the concept of dissimilarity between binary vectors as a measure. Specifically, the substitute operator '$-$' measuring the magnitude of differences between two sources (X_i and X_k) to generate a new neighbor source (V_i) via Eq. (2) is first rewritten in the form of Eq. (6). Eq. (6) is then formed into Eq. (7) by the Jaccard Coefficient based similarity/dissimilarity between vector pairs (Eq. (5)).

$$\upsilon_{ij} - x_{ij} = \phi_{ij}(x_{ij} - x_{kj}) \tag{6}$$

$$Dissimilarity\,(V_i, X_i) \approx \Phi \times Dissimilarity\,(X_i, X_k) \tag{7}$$

where X_i, X_k and V_i are binary sources and Φ is a positive random scaling factor, defined by Eq. (8).

$$\Phi = \Phi_{max} - (\frac{\Phi_{max} - \Phi_{min}}{MCN})iter \tag{8}$$

where Φ_{max} and Φ_{min} are the upper and lower levels of Φ, MCN is the maximum number of cycle, and *iter* is the current cycle.

Eq. (7) reveals that the dissimilarity between V_i and X_i ($Dissimilarity\,(V_i, X_i)$) should be close to the result of $\Phi \times Dissimilarity\,(X_i, X_k)$ as much as possible. According to this information, the number of bits with value 1 in both V_i and X_i (M_{11}), the number of bits with value 1 in V_i and value 0 in X_i (M_{10}), the number of bits with value 0 in V_i and value 1 in X_i (M_{01}) are determined using integer model programming through the following equations;

$$\min\left\{1 - \frac{M_{11}}{M_{11} + M_{10} + M_{01}} - \Phi \times Dissimilarity\,(X_i, X_k)\right\} \tag{9a}$$

$$M_{11} + M_{01} = m_1 \tag{9b}$$

$$M_{10} \leq m_0 \tag{9c}$$

$$M_{11}, M_{10}, M_{01} \geqslant 0 \text{ and they are integers} \tag{9d}$$

where m_1 is the total number of ones and m_0 is the total number of zeros in X_i. Eq.(9b) generates the feasible set of M_{11}, M_{10}, M_{01} is equal to $(m_1 + 1)(m_0 + 1)$ number of combinations. After the determination of the M values between V_i and X_i, the generation task of V_i source is carried out. V_i is first defined as $1 \times D$ zero vector and then the following selection mechanisms are carried out in a possibilistic manner.

1. Random selection. Choose M_{11} number of zero bits where their corresponding values are equal to 1 in X_i and change their values in V_i from 0 to 1. Then, choose M_{10} number of zero bits where their corresponding values are equal to 0 in X_i and change their values in V_i from 0 to 1.
2. Greedy selection. Choose M_{11} number of zero bits where their corresponding values are equal to 1 in both X_i and global best source in population ($GbestParams$), and change their value in V_i from 0 to 1. In some cases, it is not possible to modify M_{11} number of zero bits due to the interaction between X_i and $GbestParams$. If the number of changed bits ($index_0$) is less than M_{11}, choose $(M_{11} - index_0)$ bits of zero where their corresponding values are equal to 1 in X_i and change their values in V_i from 0 to 1. After that, choose M_{10} number of zero bits where their corresponding values are equal to 0 and 1 in X_i and $GbestParams$, respectively, and change their values in V_i from 0 to 1. If the number of changed bits ($index_1$) is less than M_{10}, choose $(M_{10} - index_1)$ number of zero bits where their corresponding values are equal to 0 in X_i and change their values in V_i from 0 to 1.

2.4. Feature Selection

How to determine a feature as relevant is a difficult problem due to the complicated (two-way, three-way or multi-way) interactions between features. A feature may become relevant or irrelevant when used together with other features; thus, an optimal feature subset should comprise complementary features which provide diverse properties of the classes [34]. Meanwhile, a large number of available features lead to increment in the search space, i.e., it is impossible to exhaustively search the whole space in most cases. Although various algorithms have been proposed to address feature selection problems, it still remains a challenge. The factors affecting the performance of a feature selection algorithm are as follows [35]:

1. **Initialization.** The starting feature subset in the space should be first determined, which directly influences the search direction and operators. For instance, one may start with empty set and then iteratively adds features to this subset (known as forward) or one may start with all features and then removes them iteratively (known as backward). Feature set can be also initialized according to some predefined rules [36].
2. **Search strategy.** An exhaustive search is not practically possible due to the large search space in most cases. Therefore, a more realistic approach can be applied. For instance, searching can be made by both forward and backward strategies (known as floating search) or more heuristic global search techniques. In this concept, evolutionary computation based algorithms, including particle swarm optimization (PSO) [37], genetic algorithms (GAs) [27], and ant colony optimization (ACO) [38] get attention due to their search abilities. In recent years, researchers also have started to work on artificial bee colony (ABC) to solve feature selection problems [18, 19].
3. **Evaluation criterion.** An evaluation criterion is expected to measure the quality of a feature subset accurately and inexpensively. Fundamentally, all evaluation criteria are based on the either classification performance or the characteristics of the data itself [39, 40].
4. **Stopping criterion.** Whereas some algorithms complete their processes without any restriction or rules, some of them need to confirm that the reached feature subset is a good one [34]. In addition, the size of obtained feature subset can be also used as a stopping criterion.

Given a dataset Z comprising of N patterns/examples/instances and S feature set. To select a feature subset S_k via a feature selection algorithm, the following issues need to be considered [35]:

1. The size of S_k must be smaller than current S s.t. $S_k \subset S$.
2. The dataset Z within S_k feature subset should achieve the best classification performance.
3. While providing the best classification performance, the size of S_k should be as small as possible.
4. The most appropriate evaluation criterion for the dataset should be selected.

2.4.1. Existing feature selection methods

Feature selection methods can be categorized into two main groups: filter methods and wrapper methods [41]. Filter methods eliminate irrelevant or noisy features from the dataset without applying any classification algorithm. In contrast to filter methods, a wrapper method employs a learning (classification) algorithm to evaluate the goodness of the selected feature subset [42]. Filter methods are argued to be more general than wrapper methods, but wrapper methods are more successful than filter methods in terms of obtaining better classification performance.

1) Deterministic Approaches. One of the simplest filter methods, Relief [43] selects k number of highest correlated features with the target class. However, Relief does not deal with redundant features since it tries to get all relevant features without considering the redundancy of feature. Another basic filter method, FOCUS [44] searches for the smallest possible feature subset. It starts with a single feature and then adds other features to the subset until it finds the subset that splits the training data according to their outputs. However, it is computationally intensive.

Two well-known wrapper methods, sequential forward selection (SFS) [6] and sequential backward selection (SBS) [7] use forward and backward strategies in finding candidate feature subsets. SFS starts with an empty feature subset and then adds features to the subset until the further addition cannot improve the classification performance. On the other hand, SBS starts with a subset including all features and then removes features from this subset until the further removal cannot improve the classification performance. However, the features added or removed cannot be handled or modified later. It can be inferred that these methods are similar to the mechanism of agglomerative and divisive hierarchical methods in terms of non-dynamic structures. Thus, these methods may converge to local minima. To minimize the drawbacks of SFS and SBS, floating forward and backward search mechanisms (SFFS and SFBS) [45] were developed. SFFS (SFBS) performs forward (backward) step after each backward (forward) step. They perform better than SFS and SBS, but they may also converge to local minima.

2) EC (Non-ABC) based Approaches. To overcome the drawbacks of deterministic feature selection methods, researchers have concentrated on solving the feature selection problem using evolutionary and swarm intelligence based algorithms, including GAs, GP, ACO and PSO. Yang and Honavar [10] proposed an objective function based on the feature measurement cost and the classification accuracy to improve the classification performance. Raymer et al. [11] proposed a feature selection method using GA, where feature selection and extraction were carried out simultaneously. The effectiveness of the proposed algorithm was tested by comparing it with the SFFS [45] and linear discriminant analysis methods. However, the obtained feature subset by GA through tuning sets is not much a preferred way to evaluate the feature subsets. Zhu et al. [46] introduced a wrapper-filter feature selection algorithm (WFFSA) based on a combined version of local search and GA. In local search mechanism of

WFFSA, two operators were defined: 1) select a feature from activated feature set (the positions of which represent 1 in chromosome) using linear ranking selection and move it to inactivated features (the positions of which represent 0 in chromosome), and 2) select a feature from inactivated feature set using linear ranking selection and move it to to the activated feature set. Muni et al. [12] introduced a multi-tree GP based feature selection method, in which each classifier has c trees comprising of feature subset for a problem with c classes. Ahmed et al. [47] used GP to combine top-ranked features obtained by the two feature ranking techniques, information gain and Relief to deal with feature selection problems on mass spectrometry. Unler and Murat [8] address feature selection through an adaptive discrete PSO, where a feature subset is selected by the relevance and predictive contribution of each feature. The superiority of the method was demonstrated by comparing it with tabu search, SFS and SBS. Liu et al. [9] proposed an improved feature selection (IFS) method, the aim of which was to reach higher generalization capability. IFS was built on multi swarm PSO (MSPSO), SVM and an improved fitness function based on F-score. The performance analysis of IFS was conducted by comparing it with PSO and GA. However, MSPSO was computationally more expensive than PSO and GA due to its complex structure and large population size. Xue et al. [3] investigated the feature selection problem using PSO with a two stage fitness function, comprising of the error rate and the number of features. This approach improved the feature subset when compared to the PSO using only the error rate as a fitness function. Xue et al. [3] also considered feature selection as a multi objective problem (i.e. maximizing the classification performance and minimizing the number of features simultaneously). In another study, Xue et al. [36] integrated three new initialisation strategies and updating mechanisms to the PSO algorithm motivated by forward selection, backward selection and a combination of them. The detailed survey of PSO on feature selection can be found in [48, 49]. ACO has also been used to solve FS problems, where nodes represent features, and the edges between nodes define the choice of the next feature in graphs [50]. Ming [51] proposed an ACO and rough set theory based feature selection method, which starts with the features in the core of a rough set and uses forward selection. Ding [52] combined ACO and SVM for feature selection problems. In this model, grid based ACO was implemented to optimize the parameters of SVM, and feature subset selection was performed through F-statistics. Nemati et al. [14] introduced a parallel combined version of ACO and GA for feature selection in protein function prediction. Sarac and Ozel [53] proposed an ACO based feature selection approach for web page classification, in which feature extraction is applied before feature selection to group similar HTML tags together, i.e., to reduce the feature space. More information on ACO based feature selection can be found in [50, 54].

3)ABC based Approaches. In recent years, researchers have been trying to develop feature selection approaches using the ABC algorithm after observing the numerous efficient applications based on ABC in various fields [55, 56, 26, 57]. Uzer et al. [19] used the corporation of ABC and SVM in medical dataset classification. Although the performance analysis of the ABC based feature selection approach was conducted by comparing it with the results obtained from existing studies, the feature subset size and standard deviations of the obtained accuracy values were not considered. Subanya and Rajalaxmi [58] applied the combination of the proposed ABC and Naive Bayes to the Cleveland Heart disease dataset. However, no comparative study was reported to show the effectiveness of the proposed approach. Shunmugapriya et al. [59] improved the searching mechanism of ABC with abandoned food source for feature selection. As in [58], no comprehensive comparative study was presented to show the effectiveness of the proposed method. Schiezaro and Pedrini [18] presented a study on feature selection comparing the proposed ABC algorithm with the standard PSO, ABC and GA, where simple modification rate (MR) perturbation is used to select a feature subset. The obtained results indicated that the ABC algorithm was superior to the others. However, the number of evaluations for all algorithms was not chosen as the same value. In [60], an hybrid version of rough set-based attribute reduction (RSAR) and ABC algorithms was applied to feature selection, and the effectiveness of the proposed approach was shown by comparing it with six well-known approaches, including RSAR, PSO and GAs. Although the proposed and some employed algorithms reached the same feature subset size for 3 out of 5 datasets, the obtained features were not clearly evaluated to reflect how the proposed algorithm outperformed the others in terms of accuracy. Consequently, the need and demand for the studies based on ABC and other EC techniques is still expected to be covered.

3. The Proposed Approach

Discrete binary ABC (DisABC) [21] is one of the first binary variants of the ABC algorithm based on the similarity between binary vectors measured by the Jaccard coefficient. It has gained popularity among the researchers, and has been used in the comparative studies of binary problems [61] on account of its simplicity, novelty, and performance. This leads us to choose DisABC as a main motivation for the feature selection problems in this study. Furthermore, the drawbacks including the complicated structure (NP hard) of feature selection and the weakness of DisABC in high-dimensional problems motivate us to improve the search ability of the DisABC algorithm for feature selection problems.

How can we improve the search ability of the DisABC algorithm? For two decades, population-based metaheuristics composed of an evolutionary framework and a set of local algorithms activated within the generation cycle of the external work [62] have attracted attention since they are inspired by the transmission of ideas and combination of multiple operators. By this way, it is expected to achieve good performance in problem solving. Nowadays, it is not difficult to see numerous successful metaheuristic and evolutionary algorithms comprising some forms of lifetime learning [63]. The most common way to use a learning scheme in an EC based algorithm is the hybridization (or modification), which refers to the combination of two or more different methods in an efficient way. In other words, one of the simplest ways to increase the search ability of an algorithm is to modify some parts of the mechanism via internal or external forces or mix of at least two heterogeneous individuals by conscious manipulation or as a natural progressive manipulation [63].

Keeping the above mentioned remarks in mind, we propose a modified DisABC algorithm (MDisABC) using the ideas of mutation, recombination and selection in evolutionary computation, such as DE [45, 64]. The major components/factors of MDisABC are given as follows. Firstly, instead of using one neighbor solution, three neighbors are used to create mutant solution in the search space. In this way, bees can share information obtained from neighbors more effectively in order to improve the search ability of the whole algorithm. In standard DE, a mutant solution ($\Omega_i = \{\omega_{i1}, \omega_{i2}, ..., \omega_{id}, ..., \omega_{iD}\}$) of a current solution X_i is generated by Eq. (10). However, it cannot be directly used in binary problem solving. To form Eq. (10) in the concept of dissimilarity that is suitable to the binary space, Eq. (10) first needs to be rewritten in the form of Eq. (11). Then, the substitute equation '-' in Eq. (11) measuring the magnitude of differences between binary vectors is reformulated into Eq. (12) by the Jaccard Coefficient dissimilarity between vector pairs (Eq. (5)).

$$\Omega_i = X_{r1} + \Phi(X_{r2} - X_{r3}) \tag{10}$$

$$\Omega_i - X_{r1} = \Phi(X_{r2} - X_{r3}) \tag{11}$$

$$\mathit{Dissimilarity}\,(\Omega_i, X_{r_1}) \approx \Phi \times \mathit{Dissimilarity}\,(X_{r_2}, X_{r_3}) \tag{12}$$

where r_1, r_2 and r_3 are randomly selected neighborhoods and Φ (or known as F) is the positive scaling factor.

According to Eq. (12), $\mathit{Dissimilarity}\,(\Omega_i, X_{r_1})$ should be close to $\Phi \times \mathit{Dissimilarity}\,(X_{r_2}, X_{r_3})$ as soon as possible (see Eq. (13)). Eq. (13) is then reformulated into Eq.(14) using mathematical programming as mentioned in Section 2.2, and M values between Ω_i and X_i that are required to generate Ω_i are found by solving Eq. (14).

$$\min \{\mathit{Dissimilarity}\,(\Omega_i, X_{r_1}) - \Phi \times \mathit{Dissimilarity}\,(X_{r_2}, X_{r_3})\} \tag{13}$$

$$\min \left\{1 - \frac{M_{11}}{M_{11} + M_{10} + M_{01}} - \Phi \times \mathit{Dissimilarity}\,(X_{r2}, X_{r3})\right\} \tag{14a}$$

$$M_{11} + M_{01} = m_1 \tag{14b}$$

$$M_{10} \leq m_0 \tag{14c}$$

$$M_{11}, M_{10}, M_{01} \geqslant 0 \text{ and they are integers} \tag{14d}$$

where m_1 is the number of ones in X_{r1} and m_0 is the number of zeros in X_{r1}.

The second important component/factor is the recombination between the current X_i and mutant solution Ω_i, shown by Eq. (15). By this way, the exchange of information is fully provided between the mutant and current solutions to find better solutions, which is known as multiple interaction [29].

$$u_{id} = \begin{cases} \omega_{id}, & \text{if } rand(d) \leq CR, \\ x_{id}, & \text{otherwise,} \end{cases} \tag{15}$$

where CR is the crossover rate and ω_{id} represents the dth dimension of Ω_i. A new solution is then formed as $U_i = \{u_{i1}, u_{i2}, ..., u_{id}, ..., u_{iD}\}$ and u_{id} represents the dth dimension of U_i.

3.1. Major steps to generating a new solution

To further explain the process of generating a new solution in MDisABC, we summarised the major steps as follows:

- Step 1: randomly select three neighbors (i.e. food sources), X_{r1}, X_{r2} and X_{r3} for the current food source X_i;
- Step 2: calculate $\Phi \times \mathit{Dissimilarity}\,(X_{r2}, X_{r3})$ by Eq. (5);
- Step 3: solve Eq. (14) to get M values between Ω_i and X_i;
- Step 4: using the obtained M values, apply random selection or greedy selection in a probabilistic manner to generate Ω_i;
- Step 5: apply recombination between X_i and Ω_i by Eq. (15) to generate a new solution U_i;
- Step 6: choose a better solution between X_i and U_i;

An Example: We also include the following example to illustrate the *first four* steps in detail to show how to generate a mutant solution Ω_i. The fifth and sixth steps are not included in this example since they are easy to understand.

- Step 1: randomly pick up three neighbors, say $X_{r1} = \{1, 0, 0, 0, 0, 1, 1, 0, 1, 0\}$, $X_{r2} = \{0, 0, 1, 1, 0, 0, 0, 1, 1, 1\}$ and $X_{r3} = \{0, 1, 1, 0, 0, 0, 1, 0, 1, 0\}$, and assume that Φ is assigned as 0.8.
- Step 2: calculate $\Phi \times \mathit{Dissimilarity}\,(X_{r2}, X_{r3})$ based on the M values between X_{r2} and X_{r3}, which are $M_{11} = 2$, $M_{10} = 3$ and $M_{01} = 2$ using Eq. (5):

$$\Phi \times \mathit{Dissimilarity}(X_{r2}, X_{r3}) = 0.8 \times (1 - \frac{2}{2 + 3 + 2}) = 0.5714 \tag{16}$$

- Step 3: get the optimal M values between Ω_i and X_i as $M_{11} = 3$, $M_{10} = 3$ and $M_{01} = 1$ by solving Eq. (14):

$$\min \left\{ 1 - \frac{M_{11}}{M_{11} + M_{10} + M_{01}} - 0.5714 \right\} \tag{17a}$$

$$M_{11} + M_{01} = 4 \tag{17b}$$

$$M_{10} \leq 6 \tag{17c}$$

$$M_{11}, M_{10}, M_{01} \geqslant 0 \tag{17d}$$

where $m_1 = 4$ (number of ones in X_{r1}) and $m_0 = 6$ (number of zeros in X_{r1}).

- Step 4: set $\Omega_i = \{0,0,0,0,0,0,0,0,0,0\}$. Select $M_{11} = 3$ bits from $S1_{x_{r1}} = \{1,6,7,9\}$, where $S1_{x_{r1}}$ is the set representing the elements of value 1 in X_{r1}. Assume that sixth, seventh and ninth elements are randomly selected; therefore, $\Omega_i = \{0,0,0,0,0,1,1,0,1,0\}$. After that, select $M_{10} = 3$ number of bits from $S0_{x_{r1}} = \{2,3,4,5,8,10\}$, where $S0_{x_{r1}}$ is the set representing the elements of value 0 in X_{r1}. Assume that third, fifth and last elements are selected; therefore, $\Omega_i = \{0,0,1,0,1,1,1,0,1,1\}$.

3.2. Pseudo code of MDisABC

In MDisABC, the initial binary food sources are generated by Eq. (18) instead of Eq. (1) [21].

$$x_{id} = \begin{cases} 1, & \text{if } rand_{id} \geq 0.75, \\ 0, & \text{otherwise,} \end{cases} \tag{18}$$

where $rand_{id}$ is the uniformly generated number within the range of $[0,1]$ for the dth dimension of source i.

MDisABC follows the basic steps of ABC except for the initialisation and new solution generation mechanisms. The detailed pseudo code of the MDisABC algorithm can be seen in Algorithm 1.

4. Experiment Design

4.1. Datasets

Ten benchmark datasets from the UCI machine learning repository [28] are used in experiments, as shown in Table 1. The datasets comprise of a various number of features, classes and samples, providing a comprehensive analysis of the proposed and employed algorithms. For each dataset, samples are randomly divided into two sets: 70% as the training set and 30% as the test set. As a classifier, k nearest neighbours (KNN) is used to evaluate the classification performance, where k is chosen 5 (5NN) as in [3].

4.2. Employed Algorithms in Comparative Study

To show the effectiveness of the proposed algorithm, the following EC based algorithms are employed:

1. Genetic Algorithms (GA). GA [27] is one of the most popular EC techniques for feature selection, where each chromosome represents a feature subset. Selection, crossover and mutation are the main operators in GA. A sub-population is first selected from initial population for crossover.

```
begin
    Initialize food sources by Eq.(18);
    Find the global best (GbestParams) food source;
    for $cycle \leftarrow 1$ to $MCN$ do
        foreach employed bee $i$ do
            Choose randomly three food sources $X_{r1}$,$X_{r2}$ and $X_{r3}$ in the
            neighbourhood of $X_i$;
            Calculate $Dissimilarity(X_{r2}, X_{r3})$ by Eq. (12);
            Find $M'val$ combination through $Dissimilarity(\Omega_i, X_{r1})$ by Eqs. (14);
            if $rand() < 0.5$ then
                $\Omega_i = randomselection(Mvals, X_{r1})$;
            else
                $\Omega_i = greedyselection(Mvals, X_{r1}, \text{gbest})$;
            end
            for $d = 1$ to $D$ do
                if $rand(d) < CR$ then
                    $u_{id} = \omega_{id}$
                else
                    $u_{id} = x_{id}$
                end
            end
            Apply greedy selection between $U_i$ and $X_i$;
        end
        foreach onlooker bee $i$ do
            Select a food source $X_i$ depending on probability $p_i$ by using Eq.(3);
            Choose randomly three food sources $X_{r1}$,$X_{r2}$ and $X_{r3}$ in the
            neighbourhood of $X_i$;
            Calculate $Dissimilarity(X_{r2}, X_{r3})$ by Eq. (12);
            Find $M'val$ combination through $Dissimilarity(\Omega_i, X_{r1})$ by Eq.(14);
            if $rand() < 0.5$ then
                $\Omega_i = randomselection(Mvals, X_{r1})$;
            else
                $\Omega_i = greedyselection(Mvals, X_{r1}, \text{GbestParams})$;
            end
            for $d = 1$ to $D$ do
                if $rand(d) < CR$ then
                    $u_{id} = \omega_{id}$
                else
                    $u_{id} = x_{id}$
                end
            end
            Apply greedy selection between $U_i$ and $X_i$;
        end
        if there exits an abondoned food source then
            Scout bee determines a new food source;
        end
        Update GbestParams food source;
    end
end
```

Algorithm 1: The pseudocode of the MDisABC

Table 1: Datasets

Dateset	Number of Features	Number of Classes	Number of Examples
Wine	13	3	178
Vehicle	18	4	846
German	24	2	1000
WBCD	30	2	569
Ionosphere	34	2	351
Lung	56	3	32
Hill-Valley	100	2	606
Musk 1	166	2	476
Madelon	500	2	2600
Isolet5	617	26	1559

After crossover is performed, mutation is applied in a probabilistic manner. Then, generated offsprings and current population is sorted and a selection scheme (e.g. elitist or roulette-wheel) is applied among them to generate new population for the next iterations.

2. Binary Particle Swarm Optimization (BPSO). BPSO was proposed by Kennedy and Eberhart for discrete problems in 1997 [22]. In BPSO, velocities are continuous values and are updated as in continuous PSO, but the velocity values represent the probability of the corresponding positions in a particle taking value 1 or 0 in contrast to basic PSO.
3. New Velocity Based Binary Particle Swarm Optimization (NBPSO). Khanesar et al. [23] determined that inertia weight may have negative effects on velocity and particle update mechanism. Thus, they introduced an effective velocity mechanism to update particles, named as new velocity based binary PSO (NBPSO).
4. Quantum Inspired Binary Particle Swarm Optimization (QBPSO). Yun-Won et al. [24] integrated the concept and principles of quantum computing, including quantum bit and superposition of states into basic BPSO, namely quantum inspired BPSO (QBPSO). In QBPSO, a Q-bit individual is integrated as the probability of particles taking value 1 and 0 instead of velocities. Accordingly, some parameters such as inertia weight and balance coefficients are not required to be determined.
5. Angle Modulated Artificial Bee Colony(AMABC). Pampara and Engelbrecht [25] proposed a binary artificial bee algorithm inspired by angle modulation (AMABC). In AMABC, each candidate is represented via a four dimensional continuous vector (a,b,c,d) within [-1,1] and each candidate is transformed into a D-dimensional binary space by the sinusoid

function.

6. Modification Rate Based Artificial Bee Colony (MRABC). Akay and Karaboga [26] introduced a modification rate (MR) perturbation into basic search equation (Eq. 2) to decrease the convergence problems of the standard ABC in high dimensional problems. For each parameter of an individual, a random number (rand(0,1)) is generated. If $rand(0,1) < MR$, that parameter is evolved by Eq. (2); otherwise, it is not changed. Note that [18] used similar mechanism to MRABC in feature selection problems.
7. Discrete Binary Artificial Bee Colony (DisABC). DisABC [21] is based on measuring dissimilarity via Jaccards coefficient between the current and neighborhood candidates. The detailed information concerning the DisABC can be found in Section 2.3.

Among the employed algorithms, BPSO and GA are the most widely used algorithms in feature selection resulting many publications [54, 65]. The other algorithms have been recently proposed and are strongly considered as fundamental approaches for the different variants of ABC and PSO by the researchers. However, they have not been applied to feature selection problems yet. According to the determined points, they are chosen for the comparative studies to show the effectiveness of the proposed approach.

As for the comparison of the proposed algorithm with the deterministic approaches, the linear forward selection (LFS) [66], floating forward selection (LFFS) [66] and greedy stepwise based selection (GSBS) [67] approaches derived from the sequential forward (SFS), sequential floating forward (SFFS) and sequential backward (SBS) feature selection approaches are chosen. While LFS (GSBS) performs forward (backward) search by adding (removing) features, LFFS applies both forward and backward searches in a sequential order. The experiments of LFS, LFFS and GSBS are processed using Waikato Environment for Knowledge Analysis (WEKA) [68], and the results of the deterministic and proposed approaches are presented together in Table 4.

4.3. Parameter Settings

In the experiments, the following parameters are used: the number of individuals in the population is set to 30 as in [4]; the maximum number of iterations is empirically set to 50; the parameters of BPSO are selected as in [69] such that $c_1 = 2$, $c_2 = 2$, initial weight$= 0.9$ and $Vmax = 6$; the crossover and mutation rates of GA are selected as 0.8 and 0.2 as in [70]; the parameters of NBPSO are selected as in [23] such that $w_1 = 0.5$ and $Vmax = 6$; the parameters of QBPSO are selected as in [24] such that $Qmax = 0.05\pi$ and $Qmin = 0.01\pi$; the limit parameter of the AMABC, MRABC, DisABC and MDisABC is experimentally chosen as 50; Φ_{max} and Φ_{min} are set to 0.9 and 0.5 in MDisABC and DisABC as in [21]; the MR parameter of MRABC is set to 0.5; and the CR parameter of MDisABC is experimentally chosen as 0.25.

An individual of an EC based algorithm represents a probable feature subset of S in a feature selection problem. If any dimension of an individual is 1 (or > 0.5), its corresponding feature is selected, which can be illustrated in Fig. 1.

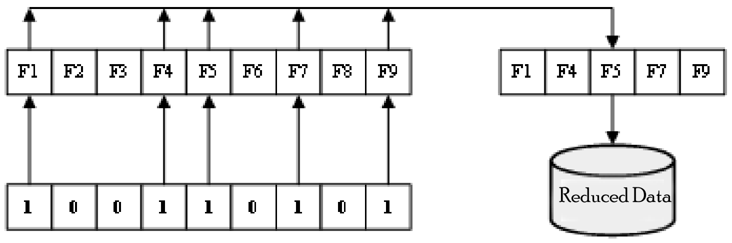

Figure 1: An illustrative representation on how the features are selected

The fitness value (error rate) of the corresponding individual (feature subset) is calculated through 10-fold cross-validation on the training set [71]. The training set is divided into 10 folds. A single fold is used as the sub-test data, and the remaining 9 folds are used as the sub-training data. This process is repeated 10 times with each of the 10 folds as the sub-test data. Then the averaged error rates from the 10 times are used as the fitness value of the corresponding feature subset (individual) during the evolutionary feature selection process. How and why 10-fold cross validation is used in this way is explained in [71] in detail. After the evolutionary process, the best feature subset is evaluated on the test set using with the training set and 5-NN to obtain the classification error rate by Eq. (19).

$$ErrorRate = \frac{FP + FN}{FP + FN + TP + TN} \tag{19}$$

where TP and TN are true positives and negatives, and FP and FN are false positives and negatives.

5. Experimental Study

The experimental results of the classification error percentage over the 30 independent runs are presented in Tables 2, 3 and 4 in terms of mean values, standard deviations and symbols where 'CER' represents the classification error rate, '#NOF' represents the number of selected features and 'T-Sig' shows whether there exists any statistically significant difference obtained by Wilcoxon Rank Sum Test between the proposed and other algorithms.

The symbols used to demonstrate the significant difference between the proposed and the other algorithms on the classification error rate through Wilcoxon Rank Sum Test have the following meanings:

- '+' & '−': The results of the MDisABC algorithm are significantly better or worse than the corresponding algorithm.
- '≈': The results of the MDisABC algorithm are similar to the corresponding algorithm.

In addition, the number of times that each feature is selected over 30 independent runs for the Wine, Vehicle, German, WBCD and Ionosphere datasets are reported in Tables 5 to 9. Due to the large dimensionality, the other datasets could not be used for the analysis.

5.1. Results on the Training Sets

The results of the error rate values on the training sets through EC based algorithms are presented in Table 2. It is shown in Table 2 that the proposed algorithm gets the best performances for five datasets in terms of the mean error rates. Table 2 also shows that MDisABC produces better mean values than DisABC in almost all cases, which shows the improvement on DisABC is well-designed and well-established. Further, between the results of MDisABC and the other ones, there exist mostly significant differences. For instance, MDisABC mostly gets significantly better results than BPSO, which is one of the most widely-used algorithms in feature selection. Only in three cases of Isolet5, the results of MDisABC are statistically worse than the results of the GA, NBPSO and QBPSO algorithms; in two cases of Musk1, NBPSO and GA statistically performs better than MDisABC, and in one case of Wine, GA statistically achieves better results than MDisABC. Except for these six cases, in 64 out of 70 (7 algorithms $\times$ 10 datasets) cases, the existing algorithms cannot obtain significantly better performance than the MDisABC algorithm. Therefore, it can be suggested that the proposed algorithm outperforms the others in terms of minimizing the training error rate.

5.2. Results on the Test Sets

On the test sets, the classification error rate and the feature subset size should be considered together to make fair comparisons of the algorithms. For instance, two algorithms may achieve similar classification performance, but the one selecting a smaller number of features is a better algorithm. Summarizing the average error rate values on the test sets, Table 3 shows that in almost all cases, the EC based algorithms obtain lower classification errors using smaller feature subsets than when all features are used with the 5NN classifier. For instance, while 5NN is able to obtain 22.17% error using all features in Wine, nearly all EC based algorithms except for GA obtain no classification error (0%) just using around half of the available features. Thus, it can be inferred that feature selection plays a vital role in classification performance.

Table 3 also indicates that MDisABC achieves the best performances in terms of the mean error rate values except for four cases and it generally gets the smallest feature subset. Especially in Musk and Madelon, the reduction of the subset size is nearly at least 10% when compared to the others. In Hill-Valley, Vehicle and German, DisABC, MRABC, and AMABC achieve the best mean values, respectively. In terms of the significance test, the proposed MDisABC algorithm only gets significantly worse performance than the other ones in 1 out of 70 cases (10 datasets $\times$ 7 algorithms). Especially in Ionosphere, Madelon, Isolet 5, and WBCD, the statistical performance of the MDisABC algorithm is successful. Although the error rate results of the algorithms are mostly similar to each other especially in Wine, Vehicle and German, MDisABC reduces the feature subset size more effectively than the other algorithms.

Table 2: The obtained error percentages of the algorithms on the training sets

Dataset		**GA**	**BPSO**	**NBPSO**	**QBPSO**	**AMABC**	**MRABC**	**DisABC**	**MDisABC**
Wine	CER	**3.95**±0.50	4.38±0.09	4.38±0.09	4.41±0.18	4.82±0.19	4.45±0.12	4.51±0.16	4.36±0.08
	T-Sig	-	≈	≈	≈	+	+	+	
Vehicle	CER	29.71±0.70	29.12±0.32	29.15±0.56	29.99±1.14	30.39±0.55	29.24±0.50	29.29±0.53	**28.94**±0.15
	T-Sig	+	+	+	+	+	+	+	
German	CER	24.93±0.96	24.80±0.49	24.43±0.78	24.34±0.55	25.49±0.55	24.45±0.61	24.27±0.75	**23.91**±0.39
	T-Sig	+	+	+	+	+	+	≈	
WBCD	CER	4.71±0.41	3.99±0.43	4.58±0.57	4.44±.44	4.12±0.37	4.53±0.39	4.19±0.51	**3.79**±0.33
	T-Sig	+	≈	+	+	+	+	+	
Ionosphere	CER	8.72±1.18	8.89±0.80	7.19±1.12	6.61±.91	7.44±0.68	8.76±1.05	6.04±0.41	**5.74**±0.41
	T-Sig	+	+	+	+	+	+	+	
Lung	CER	25.10±5.58	23.82±3.95	19.63±5.50	**18.66**±4.55	26.32±3.88	23.35±4.36	19.84±3.98	18.83±3.05
	T-Sig	+	+	≈	≈	+	+	≈	
Hill-Valley	CER	40.83±1.48	43.01±0.96	40.18±1.15	40.48±0.84	43.11±0.84	41.79±1.31	40.15±0.80	**40.14**±0.77
	T-Sig	+	+	≈	+	+	+	≈	
Musk 1	CER	7.42±1.02	9.68±0.55	**7.35**±0.94	7.94±0.60	10.67±0.69	8.98±0.85	8.14±1.10	7.85±0.63
	T-Sig	-	+	-	≈	+	+	≈	
Madelon	CER	**18.53**±0.93	21.87±0.49	18.82±1.01	19.42±0.54	22.99±0.85	20.59±0.73	19.63±2.11	18.71±1.03
	T-Sig	≈	+	≈	+	+	+	+	
Isolet5	CER	**11.65**±0.75	14.33±0.53	11.90±0.66	12.25±0.49	15.51±0.49	13.19±0.58	13.63±1.12	12.73±0.61
	T-Sig	-	+	-	-	+	+	+	

Therefore, the MDisABC algorithm can be also regarded as successful in those datasets. Consequently, the proposed MDisABC also outperforms the others in terms of minimizing the error rate and feature subset size on the test sets.

5.3. CPU Time Analysis

The computational time results are presented in Fig. 2. The experiments are implemented in MATLAB 2013a and are executed on a computer with an Intel Core i7-4700HQ 2.40 GHz CPU and 8 GB RAM. The implementation codes of the employed algorithms except for QBPSO are provided from the authors of the corresponding studies [69, 21, 23] or are coded according to the authors' suggestions [25] to demonstrate a reliable analysis. According to Fig. 2, there exists no great difference between the algorithms in term of the CPU computational

Table 3: The obtained error percentages of the algorithms on the test sets

Dataset		**GA**	**BPSO**	**NBPSO**	**QBPSO**	**AMABC**	**MRABC**	**DisABC**	**MDisABC**	**5-NN All**
Wine	CER	9.19±1.64	**0**	0.37±1.02	**0**	**0**	0.12±0.46	**0**	**0**	22.17
	#NOF	6.1	5.8	6.13	5.83	6.63	**5.76**	5.83	**5.76**	13
	T-Sig	+	≈	≈	≈	≈	≈	≈		
Vehicle	CER	20.82±1.71	20.74±1.70	21.32±1.98	22.48±2.19	21.31±1.96	**20.73**±1.67	20.99±1.90	21.22±2.12	23.9
	#NOF	10	9.93	9.76	9.90	11.93	9.73	9.73	**9.30**	18
	T-Sig	≈	≈	≈	≈	≈	≈	≈		
German	CER	29.06±1.84	28.97±1.79	29.17±1.91	29.12±1.84	**28.62**±1.71	29.90±1.69	29.43±2.61	29.85±1.87	32
	#NOF	10.70	11.43	11.30	10.83	13.56	10.40	8.73	**8.03**	25
	T-Sig	≈	≈	≈	≈	-	≈	≈		
WBCD	CER	7.27≈1.46	7.05±1.03	7.54±1.17	7.45±0.91	7.39±1.18	7.84±0.80	7.37±1.06	**6.72**±1.08	7.06
	#NOF	14.96	13.36	13.96	14.30	12.06	14.50	12.23	**11.86**	30
	T-Sig	+	≈	+	+	+	+	+		
Ionosphere	CER	8.47±1.75	8.09±1.45	7.68±1.96	7.96±1.86	7.11±1.90	8.95±1.53	6.69±1.72	**6.38**±1.64	10.48
	#NOF	10.93	10.1	9.63	8.23	**4.9**	10.13	6.03	5.76	34
	T-Sig	+	+	+	+	≈	+	≈		
Lung	CER	40.00±12.58	34.07±13.35	38.88±15.64	35.18±11.70	44.44±15.98	42.96±13.91	39.62±11.91	**32.96**±8.49	44.44
	#NOF	28.03	27.33	27.30	26.70	22.03	26.50	**16.66**	24.36	56
	T-Sig	+	≈	≈	≈	+	+	+		
Hill-Valley	CER	45.95±2.05	45.73±1.84	45.78±1.48	45.53±1.79	45.6±2.68	45.67±2.34	**44.57**±2.23	44.92±2.13	47.25
	#NOF	45.73	46.16	44.86	44.7	31.5	44.73	**24.60**	30.53	100
	T-Sig	+	+	+	≈	≈	≈	≈		
Musk 1	CER	17.19±2.36	16.64±2.78	16.04±2.40	15.40±2.20	18.33±3.17	15.71±2.73	15.88±2.91	**14.71**±2.07	20
	#NOF	81.83	81,2	82,23	81,4	75,96	82.26	86.16	**75.76**	166
	T-Sig	+	+	+	≈	+	≈	≈		
Madelon	CER	23.79±1.28	24.03±1.45	23.96±1.73	22.55±1.47	24.58±1.60	23.95±1.31	23.02±3.84	**21.14**±2.02	28.21
	#NOF	250.03	248.1	246.26	240.56	238.43	248.43	223.86	**195.96**	500
	T-Sig	+	+	+	+	+	+	+		
Isolet5	CER	14.90±1.19	15.33±1.19	15.09±1.04	**14.47**±1.10	16.63±1.14	15.20±1.12	16.02±1.38	14.51±1.17	19.02
	#NOF	306.06	306.6	303.93	305.73	347.4	303.96	378.33	**300.10**	617
	T-Sig	≈	+	+	≈	+	+	+		

times in almost all datasets. For instance, the CPU time difference between the algorithms is at most 1 or 2 seconds from Wine to Musk1. This is not a large difference when compared to the runtime of algorithms on these datasets which was between 18 and 35 seconds. Proportional to the dimensionality and number of patterns, the time complexity is increased in the Madelon and Isolet5 datasets, where the AMABC and DisABC algorithms cannot preserve the scalability as in the other datasets.

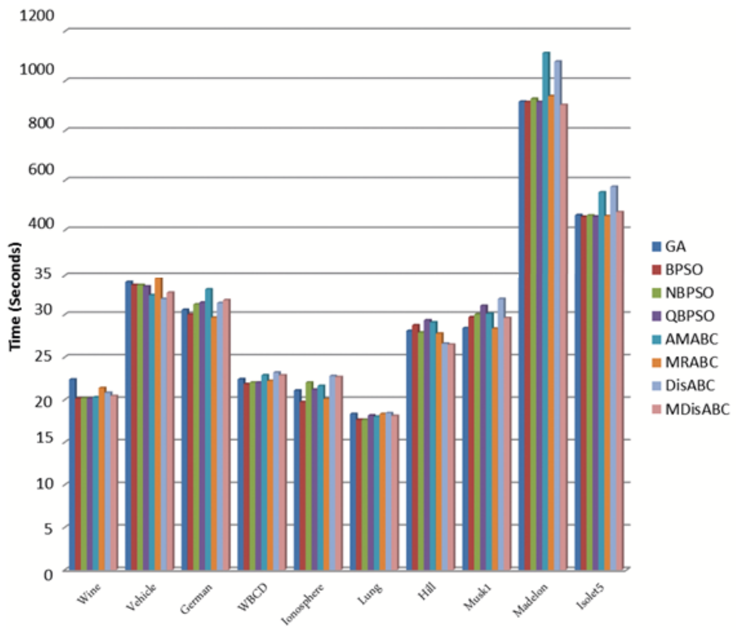

Figure 2: Average CPU computational times of the algorithms

When comparing the proposed MDisABC with deterministic methods, there is no clear pattern since the computational time used by different methods heavily depends on the datasets. In general, for datasets with a small number of features (e.g. Wine), the deterministic methods take shorter time than MDisABC. For datasets with a large number of features (e.g. Madelon), MDisABC is often slower than LFS and LFFS, but is faster than GSBS. The main reason is that most of the computational time in a wrapper feature selection method is spent on the evaluation procedures. On large datasets (e.g. Madelon), GSBS may have a larger number of evaluations than MDisABC and each evaluation in GSBS may take longer time than in MDisABC because GSBS starts evaluations with a large number of features. Thus, it can be concluded that not only in the traning and testing classification performance, but also in the CPU computational time the proposed MDisABC algorithm performs well.

5.4. Analysis of Selected Features

To show the stability/consistency of the features selected by the proposed algorithm over different independent runs, we analyse the number of times for each feature being selected by the algorithms in the 30 independent runs. Due to the space limit, five datasets (Wine, Vehicle, German, WBCD and Ionosphere) with relatively small number of features are used here as examples for the analysis of the selected features.

Presenting the selection times of each feature in Wine, Table 5 shows that F1, F7 and F10 are the most dominant features in classification frequently selected by all algorithms, and F4, F5, F12 and F13 are mostly not preferred features by all algorithms except for the cases of GA and NBPSO in F12. For the other features, whereas F2 is mostly preferred by DisABC and AMABC, it is not preferred frequently by the BPSO variants and the proposed MDisABC algorithm. F8 is mostly preferred by AMABC and is sometimes preferred by DisABC and NBPSO, but it is only selected for once by the others. Therefore, MDisABC is as stable as the other algorithms. Furthermore, the best feature subset obtained by MDisABC comprises of all dominant and two occasionally preferred complementary features ($S_{best} = \{F1, F2, F6, F7, F10\}$).

Considering the count of selection times for each feature in Vehicle, Table 6 shows that F1, F3, F6, F10 and F11 are the dominant features mostly preferred by the algorithms except for the case of AMABC in F1 and the case of QBPSO in F3. As for the least or not preferred features, F4, F7, F12, F13 and F16 can be given as examples. Although F4 is not preferred by MDisABC and binary variants of the PSO, it is occasionally chosen by AMABC, DisABC and GA. Also, although F7 is preferred by AMABC, it is not preferred by the other ones. It can be inferred that the stability of MDisABC also carries on in Vehicle, and AMABC is not as consistent as the other ones. Besides, the best feature subset obtained by MDisABC is $S_{best} = \{F1, F3, F5, F6, F8, F9, F10, F14\}$, comprising of all dominant features (except for F11) and four occasionally selected features.

According to Table 7, F1, F3 and F7 are the most frequently selected features among all of the algorithms in German. Although F18 is also one of the most preferred features among MDisABC, AMABC, NBPSO and BPSO, it is not much preferred by GA, DisABC and QBPSO. For the least preferred features, F2 (except for the case of AMABC), F4 and F10 can be given as samples. Not only F2, but also F14 and F15 are much more frequently picked by AMABC. These features may lead AMABC to obtain higher feature subset size than the others (see Table 3). Therefore, it may be suggested that AMABC is also not good at eliminating redundant or irrelevant features despite its performance in German, and the stability of MDisABC is illustrated in German. The best feature subset obtained by MDisABC is the $\{F1, F3, F6, F8, F12, F17, F21\}$ comprising of dominant features (except for F7).

In WBCD, Table 8 shows that there is no dominant features chosen by all algorithms like the previous datasets, yet F1 and F21 are maybe given as samples. On the other hand, F2, F4, F22 and F24 are the least preferred features by the algorithms. It is difficult to make an analysis of all cases due to the different combinations of feature subsets, but the stability of MDisABC can be illustrated in WBCD. The best feature subset obtained by MDisABC is the $\{F1, F7, F8, F13, F15, F16, F21, F26, F27, F28, F29\}$, which comprises of dominant features, but does not include any least preferred features.

According to Table 9, F5 is the most preferred feature, and F12, F20, F24 (except for GA), F26, F28, F30 and F32 (except for GA) are the least preferred features among all the algorithms in Ionosphere. It is also seen in Table 9 that there exist features such as F15, F23 and F25 which are selected more frequently by the PSO algorithms, GA and MRABC than by the binary variants of ABC. This might be the reason why the size of feature subsets obtained by binary variants of ABC is about half of the feature subset size obtained by the other ones. The best feature subset obtained by MDisABC is the $\{F3, F4, F5, F16, F23, F25, F27\}$, the combination of one available dominant and six occasionally selected features. In conclusion, the proposed MDisABC algorithm is the most stable and robust algorithm.

5.5. Comparisons with Recent ACO Papers

To further test the performance of MDisABC, we compare MDisABC with two ACO based algorithms [72, 73] published in 2015, which use the similar methodology for feature selection to this paper. According to the first study [72], four datasets, including Wine, Vehicle, German and Ionosphere are common with this paper, and the classification results of them for ACO based feature selection are 4.51%, 28.25%, 30.40% and 14.82%, respectively. The results show that MDisABC performs better than ACO [72]. According to the other study [73] with an improved ACO algorithm, Wine and Vehicle datasets are common with this paper, and the classification results of them for an improved version of ACO are 3.10% and 24.70%, respectively. Therefore, MDisABC is also superior to the improved ACO.

5.6. Comparisons with Deterministic Approaches

According to Table 4, LFS finds the smallest feature subsets in most cases. However, it cannot provide the same success in terms of the classification error rate. LFFS performs similar or slightly worse than LFS. On the other hand, GSBS finds the largest feature subsets nearly in all cases, and it performs worse than LFS and LFFS in terms of the classification error rate since it is a greedy backward search starting with the entire subset of features. When comparing the MDisABC algorithm with the deterministic approaches, it is seen that the MDisABC algorithm provides the best performances on 9 out of the 10 datasets and it statistically obtains meaningful results in almost all cases. Only in Hill Valley, the MDisABC algorithm cannot achieve significantly better performance than the deterministic approaches, but their performances are very similar. In terms of the time complexity, the forward approaches (LFS and LFFS) are the cheapest ones, but the backward approach (GSBS) costs higher than the others. Especially in large-scale datasets (Madelon and Isolet5), it may take nearly 1 week. In conclusion, the MDisABC algorithm is also superior to the deterministic approaches and can be used as an alternative in feature selection problems.

6. Conclusions

The main goal of this study was to propose a new variant of the DisABC algorithm for feature selection. This goal was successfully achieved by introducing DE based neighborhood search into the similarity based mechanism of DisABC. The second goal of this study was to demonstrate a comprehensive comparative study for the future studies of researchers. This goal was achieved by comparing the proposed algorithm with the seven different EC based algorithms, including BPSO, NBPSO, QBPSO, DisABC, AMABC, MRABC and GA, and three deterministic approaches, including LFS, LFFS and GSBS. It should be noted that while BPSO and GA are the algorithms most widely applied to the feature selection, the other EC based algorithms are implemented for feature selection for the first time.

The obtained results reveal that the integration of DE based similarity search mechanism into the DisABC algorithm effectively improved the global search ability of the algorithm in feature selection, and the proposed MDisABC algorithm achieved the best classification performance in both training and test sets in almost all cases. Further, the proposed MDisABC algorithm is able to remove redundant features effectively while obtaining the highest classification results. The results also show that the proposed MDisABC algorithm outperformed the deterministic non-EC methods, LFS, LFFS and GSBS in terms of the classification accuracy and selected a much smaller number of features than GSBS. The analysis of the features selected by different algorithms reveals that the proposed MDisABC algorithm is the most stable and consistent algorithm among all the algorithms. Moreover, the analysis of the CPU times of the different methods shows that the proposed MDisABC algorithm achieved better accuracy than the existing methods without taking longer computational time.

In the future, the studies of feature selection based on ABC are expected to increase and more ABC based approaches will be developed. There are also some new metaheuristics [74, 75] which have not been used in feature selection. We will test the proposed method on large dimensional datasets and consider the feature selection problem in filter approaches using ABC.

Acknowledgement

This work is supported in part by the Marsden Funds of New Zealand (VUW1209), the University Research Funds of Victoria University of Wellington (203936/3337), the National Science Foundation of China (NSFC No. 61170180). The authors are also grateful for financial support from the Scientific and Technological Research Council of Turkey (TUBITAK-BIDEB 2214/A) and Turkish Council of Higher Education.

7. References

[1] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach. Learn. Res. 3 (2003) 1157–1182.

[2] M. Dash, H. Liu, Feature selection for classification, Intelligent Data Analysis 1 (14) (1997) 131–156.

[3] B. Xue, Z. Mengjie, W. N. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, IEEE Transactions on Cybernetics 43 (6) (2013) 1656–1671.

[4] B. Xue, Particle swarm optimisation for feature selection in classification, Ph.D. thesis, Victoria University of Wellington, School of Engineering and Computer Science (2013).

[5] M. Lane, B. Xue, I. Liu, M. Zhang, Particle swarm optimisation and statistical clustering for feature selection, in: S. Cranefield, A. Nayak (Eds.), AI 2013: Advances in Artificial Intelligence, Vol. 8272 of Lecture Notes in Computer Science, Springer International Publishing, 2013, pp. 214–220.

[6] A. W. Whitney, A direct method of nonparametric measurement selection, IEEE Transactions on Computers C-20 (9) (1971) 1100–1103.

[7] T. Marill, D. Green, On the effectiveness of receptors in recognition systems, IEEE Transactions on Information Theory 9 (1) (2006) 11–17.

[8] A. Unler, A. Murat, A discrete particle swarm optimization method for feature selection in binary classification problems, European Journal of Operational Research 206 (3) (2010) 528–539.

[9] Y. Liu, G. Wang, H. Chen, H. Dong, X. Zhu, S. Wang, An improved particle swarm optimization for feature selection, Journal of Bionic Engineering 8 (2) (2011) 191–200.

[10] J. Yang, V. G. Honavar, Feature subset selection using a genetic algorithm, IEEE Intelligent Systems 13 (2) (1998) 44–49.

[11] M. L. Raymer, W. F. Punch, E. D. Goodman, L. A. Kuhn, A. K. Jain, Dimensionality reduction using genetic algorithms, IEEE Transactions on Evolutionary Computation 4 (2) (2000) 164–171.

[12] D. Muni, N. Pal, D. J., Genetic programming for simultaneous feature selection and classifier design, IEEE Transactions on Systems Man and Cybernetics, Part B: Cybernetics 36 (1) (2006) 106–117.

[13] R. Ramirez, P. M., An evolutionary computation approach to cognitive states classification, in: IEEE Congress on Evolutionary Computation (CEC'07), 2007, pp. 1793–1799.

[14] S. Nemati, M. E. Basiri, N. Ghasem-Aghaee, M. H. Aghdam, A novel aco-ga hybrid algorithm for feature selection in protein function prediction, Expert Systems with Applications 36 (10) (2009) 12086 – 12094.

[15] L. Wen, Q. Yin, P. Guo, Ant colony optimization algorithm for feature selection and classification of multispectral remote sensing image, in: IEEE International Geoscience and Remote Sensing Symposium (IGARSS2008), Vol. 2, 2008, pp. II–923–II–926.

[16] D. Karaboga, B. Basturk, On the performance of artificial bee colony (ABC) algorithm, Applied Soft Computing 8 (1) (2008) 687–697.

[17] D. Karaboga, B. Gorkemli, C. Ozturk, N. Karaboga, A comprehensive survey: artificial bee colony (ABC) algorithm and applications, Artificial Intelligence Review 42 (1) (2014) 21–57.

[18] M. Schiezaro, H. Pedrini, Data feature selection based on artificial bee colony algorithm, EURASIP Journal on Image and Video Processing 2013 (1) (2013) 1–8.

[19] M. S. Uzer, Y. Nihat, O. Inan, Feature selection method based on artificial bee colony algorithm and support vector machines for medical datasets classification, The Scientific World Journal 2013 (2013) 1–10.

[20] M. Akila, S. S. Kumar, Performance of classification using a hybrid distance measure with artificial bee colony algorithm for feature selection in keystroke dynamics, Int. J. Comput. Intell. Stud. 2 (2) (2013) 187–197.

[21] M. H. Kashan, N. Nahavandi, A. H. Kashan, Disabc: A new artificial bee colony algorithm for binary optimization, Applied Soft Computing 12 (1) (2012) 342–352.

[22] J. Kennedy, R. Eberhart, A discrete binary version of the particle swarm algorithm, in: Systems, Man, and Cybernetics, IEEE International Conference on Computational Cybernetics and Simulation, Vol. 5, 1997, pp. 4104–4108 vol.5.

[23] M. A. Khanesar, M. Teshnehlab, M. A. Shoorehdeli, A novel binary particle swarm optimization, in: Mediterranean Conference on Control&Automation (MED '07), 2007, pp. 1–6.

[24] J. Yun-Won, P. Jong-Bae, J. Se-Hwan, K. Y. Lee, A new quantum-inspired binary pso: Application to unit commitment problems for power systems, IEEE Transactions on Power Systems 25 (3) (2010) 1486–1495.

[25] G. Pampara, A. P. Engelbrecht, Binary artificial bee colony optimization, in: IEEE Symposium on Swarm Intelligence (SIS), 2011, pp. 1–8.

[26] B. Akay, D. Karaboga, A modified artificial bee colony algorithm for real-parameter optimization, Information Sciences 192 (2012) 120–142.

[27] J. H. Holland, Genetic algorithms, Scholarpedia 7 (12) (2012) 1482.

[28] K. Bache, M. Lichman, UCI machine learning repository (2013).
URL `http://archive.ics.uci.edu/ml`

[29] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Tech. rep., Erciyes University, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005).

[30] S. Das, S. Biswas, S. Kundu, Synergizing fitness learning with proximity-based food source selection in artificial bee colony algorithm for numerical optimization, Applied Soft Computing 13 (12) (2013) 4676–4694.

[31] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, Journal of Global Optimization 39 (3) (2007) 459–471.

[32] S. seok Choi, S. hyuk Cha, A survey of binary similarity and distance measures, Journal of Systemics, Cybernetics and Informatics (2010) 43–48.

[33] P. Jaccard, The distribution of the flora in the alpine zone, New Phytologist 11 (3750).

[34] B. Bonev, Feature selection based on information theory, Ph.D. thesis, University of Alicante (2010).

[35] A. Blum, P. Langley, Selection of relevant features and examples in machine learning, Articial Intelligence 97 (1-2) (1997) 245–271.

[36] B. Xue, M. Zhang, W. N. Browne, Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms, Applied Soft Computing 18 (2014) 261–276.

[37] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory (1995).

[38] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm intelligence: from natural to artificial systems, Oxford University Press Inc, New York, NY, USA, 1999.

[39] L. Cervante, B. Xue, L. Shang, M. Zhang, A dimension reduction approach to classification based on particle swarm optimisation and rough set theory, in: AI 2012: Advances in Artificial Intelligence, Springer, 2012, pp. 313–325.

[40] L. Cervante, B. Xue, L. Shang, M. Zhang, A multi-objective feature selection approach based on binary pso and rough set theory, in: Evolutionary Computation in Combinatorial Optimization, Vol. 7832 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2013, pp. 25–36.

[41] B. Xue, M. Zhang, W. N. Browne, New fitness functions in binary particle swarm optimisation for feature selection, in: IEEE Congress on Evolutionary Computation (CEC'2012), IEEE, 2012, pp. 1–8.

[42] H. Bommaganti, Feature boosting: A novel feature subset selection approach, Master thesis, University of Minnesota (2001).

[43] K. Kira, L. A. Rendell, A practical approach to feature selection, in: Proceedings of the Ninth International Workshop on Machine Learning, ML92, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1992, pp. 249–256.

[44] H. Almuallim, T. G. Dietterich, Learning boolean concepts in the presence of many irrelevant features, Artificial Intelligence 69 (1994) 279–305.

[45] P. Pudil, J. Novovicova, J. Kittler, Floating search methods in feature selection, Pattern Recognition Letters 15 (11) (1994) 1119 – 1125.

[46] Z. Zhu, Y.-S. Ong, M. Dash, Wrapper-filter feature selection algorithm using a memetic framework, IEEE Transactions on Systems Man and Cybernetics-Part B 37 (1) (2007) 70–76.

[47] S. Ahmed, M. Zhang, L. Peng, Improving feature ranking for biomarker discovery in proteomics mass spectrometry data using genetic programming, Connection Science 26 (3) (2014) 215–243.

[48] V. Kothari, J. Anuradha, S. Shah, P. Mittal, A survey on particle swarm optimization in feature selection, in: P. V. Krishna, M. R. Babu, E. Ariwa (Eds.), Global Trends in Information Systems and Software Applications, Vol. 270 of Communications in Computer and Information Science, Springer Berlin Heidelberg, 2012, Ch. 22, pp. 192–201.

[49] B. Tran, B. Xue, M. Zhang, Overview of particle swarm optimisation for feature selection in classification, in: Simulated Evolution and Learning, Vol. 8886 of Lecture Notes in Computer Science, Springer International Publishing, 2014, pp. 605–617.

[50] R. Jensen, Performing feature selection with aco, in: A. Abraham, C. Grosan, V. Ramos (Eds.), Swarm Intelligence in Data Mining, Vol. 34 of Studies in Computational Intelligence, Springer Berlin Heidelberg, 2006, pp. 45–73. `doi:10.1007/978-3-540-34956-3_3`. URL `http://dx.doi.org/10.1007/978-3-540-34956-3_3`

[51] H. Ming, A rough set based hybrid method to feature selection, in: International Symposium on Knowledge Acquisition and Modeling (KAM '08), 2008, pp. 585–588.

[52] S. Ding, Feature selection based f-score and aco algorithm in support vector machine, in: Knowledge Acquisition and Modeling, 2009. KAM '09. Second International Symposium on, Vol. 1, 2009, pp. 19–23.

[53] E. Sarac, S. A. Ozel, An ant colony optimization based feature selection for web page classification, The Scientific World Journal (2014) 1–16.

[54] B. de la Iglesia, Evolutionary computation for feature selection in classification problems, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 3 (6) (2013) 381–407.

[55] C. Ozturk, E. Hancer, D. Karaboga, Color quantization: A short review and an application with artificial bee colony algorithm, Informatica 25 (3) (2014) 485–503.

[56] C. Ozturk, E. Hancer, D. Karaboga, Improved clustering criterion for image clustering with artificial bee colony algorithm, Pattern Analysis and Applications (2014) 1–13`doi:10.1007/s10044-014-0365-y`.

[57] C. Ozturk, E. Hancer, D. Karaboga, Automatic clustering with global best artificial bee colony algorithm, Journal of the Faculty of Engineering and Architecture of Gazi University 29 (4) (2014) 677–687.

[58] B. Subanya, R. Rajalaxmi, Artificial bee colony based feature selection for effective cardiovascular disease diagnosis, International Journal of Scientific & Engineering Research 5 (5) (2014) 606–612.

[59] P. Shunmugapriya, S. Kanmani, R. Supraja, K. Saranya, Hemalatha, Feature selection optimization through enhanced artificial bee colony algorithm, in: International Conference on Recent Trends in Information Technology (ICRTIT), 2013, pp. 56–61.

[60] N. Suguna, K. G. Thanushkodi, An independent rough set approach hybrid with artificial bee colony algorithm for dimensionality reduction, American Journal of Applied Sciences 8 (3) (2011) 261–266.

[61] M. S. Kiran, M. Gunduz, Xor-based artificial bee colony algorithm for binary optimization, Turkish Journal of Electrical Engineering & Computer Sciences 21 (2013) 2307–2328.

[62] F. Neri, C. Cotta, Memetic algorithms and memetic computing optimization: A literature review, Swarm and Evolutionary Computation 2 (2012) 1–14.

[63] O. Yew-Soon, L. Meng-Hiot, C. Xianshun, Memetic computation-past, present & future [research frontier], IEEE Computational Intelligence Magazine 5 (2) (2010) 24–31.

[64] M. H. Kashan, A. H. Kashan, N. Nahavandi, A novel differential evolution algorithm for binary optimization, Computational Optimization and Applications 55 (2) (2013) 481–513.

[65] (0,1)-knapsack: 40 object problem, BTL Associates.
URL `http://www.btluke.com/01knt1.html`

[66] M. Gutlein, E. Frank, M. Hall, A. Karwath, Large-scale attribute selection using wrappers, in: IEEE Symposium on Computational Intelligence and Data Mining (CIDM '09), 2009, pp. 332–339.

[67] R. Caruana, D. Freitag, Greedy attribute selection, in: In Proceedings of the Eleventh International Conference on Machine Learning, Morgan Kaufmann, 1994, pp. 28–36.

[68] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The weka data mining software: An update, SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.

[69] S. Mirjalili, A. Lewis, S-shaped versus v-shaped transfer functions for binary particle swarm optimization, Swarm and Evolutionary Computation 9 (2013) 1–14.

[70] S. Sivanandam, S. Deepa, Genetic algorithm implementation using matlab, in: Introduction to Genetic Algorithms, Springer Berlin Heidelberg, 2008, pp. 211–262.

[71] R. Kohavi, G. H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1-22) (1997) 273–324.

[72] N. Sreeja, A. Sankar, Pattern matching based classification using ant colony optimization based feature selection, Applied Soft Computing 31 (0) (2015) 91 – 102.

[73] S. Kashef, H. Nezamabadi-pour, An advanced ACO algorithm for feature subset selection, Neurocomputing 147 (0) (2015) 271 – 279, advances in Self-Organizing Maps Subtitle of the special issue: Selected Papers from the Workshop on Self-Organizing Maps 2012 (WSOM 2012).

[74] B. Javidy, A. Hatamlou, S. Mirjalili, Ions motion algorithm for solving optimization problems, Appl. Soft Comput. 32 (C) (2015) 72–79.

[75] A. Hatamlou, Black hole: A new heuristic optimization approach for data clustering, Information Science 222 (2013) 175–184.

Table 4: The obtained error rates of the deterministic approaches with the proposed approach on the test sets

Dataset		**LFS**	**LFFS**	**GSBS**	**MDisABC**
Wine	CER	25.93	25.93	24.07	**0**
	#NOF	7	8	10	5.8
	T-Sig	+	+	+	N/A
Vehicle	CER	27.89	24.30	24.7	**21.69**±1.85
	#NOF	9	5	16	9.23
	T-Sig	+	+	+	N/A
German	CER	31.67	31.67	30.67	**29.85**±1.87
	#NOF	5	5	20	8.03
	T-Sig	+	+	+	N/A
WBCD	CER	7.65	8.82	7.06	**6.72**±1.08
	#NOF	12	11	29	11.86
	T-Sig	+	+	+	N/A
Ionosphere	CER	9.52	9.52	10.48	**6.38**±1.64
	#NOF	6	6	29	5.76
	T-Sig	+	+	+	N/A
Lung	CER	33.33	33.33	33.33	**32.96±8.49**
	#NOF	6	5	36	24.36
	T-Sig	+	+	+	N/A
Hill-Valley	CER	**44.51**	46.15	45.60	44.92±2.13
	#NOF	9	8	95	30.53
	T-Sig	≈	+	≈	N/A
Musk 1	CER	19.29	20.71	17.14	**14.71**±2.07
	#NOF	12	12	124	75.76
	T-Sig	+	+	+	N/A
Madelon	CER	28.97	32.31	25.12	21.14±2.02
	#NOF	7	6	250	195.96
	T-Sig	+	+	+	N/A
Isolet5	CER	23.72	25.43	19.23	**14.51**±1.17
	#NOF	27	23	585	300.10
	T-Sig	+	+	+	N/A

Table 5: Times of appearance of each feature over 30 runs for Wine

Wine	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**	**F10**	**F11**	**F12**	**F13**
MDisABC	30	12	18	0	0	12	30	1	18	30	18	0	0
DisABC	30	19	16	0	0	17	29	9	12	29	12	2	0
MRABC	30	13	18	0	0	13	28	2	17	30	18	2	0
AMABC	30	27	13	0	0	13	30	23	19	30	13	1	0
QBPSO	30	8	23	0	0	8	30	1	22	30	23	0	0
NBPSO	30	13	17	3	0	13	30	8	15	25	21	8	0
BPSO	30	7	23	0	0	7	30	1	23	30	23	0	0
GA	30	20	17	0	0	6	23	14	15	24	11	23	0

Table 6: Times of appearance of each feature over 30 runs for Vehicle

Vehicle	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**
MDisABC	30	23	30	1	16	30	1	15	15
DisABC	29	22	30	9	21	28	6	19	9
MRABC	28	23	28	2	19	29	5	15	9
AMABC	21	28	30	13	23	23	23	20	18
QBPSO	26	17	21	2	25	27	13	20	10
NBPSO	24	21	30	0	22	30	8	15	17
BPSO	27	26	30	1	20	27	4	11	17
GA	26	19	30	10	24	24	13	12	10

Table 6 Continued: Times of appearance of each feature over 30 runs for Vehicle

Vehicle	**F10**	**F11**	**F12**	**F13**	**F14**	**F15**	**F16**	**F17**	**F18**
MDisABC	29	25	0	0	18	10	0	19	17
DisABC	25	27	0	0	20	12	0	21	17
MRABC	29	25	0	0	24	15	1	20	20
AMABC	28	27	0	1	23	15	12	23	30
QBPSO	24	24	0	0	19	20	11	22	16
NBPSO	29	24	0	1	17	10	2	22	21
BPSO	30	27	0	0	14	17	0	20	27
GA	30	27	0	2	16	15	2	16	24

Table 7: Times of appearance of each feature over 30 runs for German

German	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**	**F10**	**F11**	**F12**
MDisABC	30	0	28	1	11	14	20	11	4	0	3	3
DisABC	30	4	25	5	15	15	24	11	13	0	9	6
MRABC	30	3	28	0	13	19	15	9	9	2	10	5
AMABC	30	17	25	2	14	26	22	17	14	2	17	17
QBPSO	30	5	27	4	13	18	20	15	17	2	12	4
NBPSO	30	4	23	3	15	19	21	21	13	0	9	7
BPSO	30	8	29	2	17	21	23	15	12	1	14	9
GA	30	4	21	6	16	17	18	13	18	3	11	9

Table 7 Continued: Times of appearance of each feature over 30 runs for German

German	**F13**	**F14**	**F15**	**F16**	**F17**	**F18**	**F19**	**F20**	**F21**	**F22**	**F23**	**F24**
MDisABC	8	3	9	11	7	26	16	7	11	8	6	3
DisABC	11	6	8	11	10	14	16	3	5	9	6	6
MRABC	12	10	9	19	15	20	15	11	16	15	16	11
AMABC	18	16	18	22	10	23	13	14	19	16	19	16
QBPSO	10	8	9	13	17	15	16	15	15	16	15	9
NBPSO	10	9	13	15	8	22	19	17	18	18	10	15
BPSO	14	9	9	17	12	23	17	9	14	16	6	16
GA	12	4	8	13	17	16	18	17	16	13	10	11

Table 8: Times of appearance of each feature over 30 runs for WBCD

WBCD	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**	**F10**	**F11**	**F12**	**F13**	**F14**	**F15**
MDisABC	20	0	7	0	14	14	18	12	13	14	13	5	18	7	16
DisABC	18	0	14	0	17	15	11	12	16	15	17	7	16	14	19
MRABC	21	0	23	0	12	20	17	15	15	20	11	20	10	22	15
AMABC	20	2	6	0	11	14	18	14	10	15	14	8	8	6	16
QBPSO	16	1	22	0	19	19	12	14	19	15	19	10	12	22	18
NBPSO	19	3	22	2	11	13	19	14	14	19	13	15	15	22	12
BPSO	21	0	8	0	15	16	15	12	13	13	10	9	16	8	14
GA	15	9	26	2	20	15	15	20	14	16	13	16	14	19	10

Table 8 Continued: Times of appearance of each feature over 30 runs for WBCD

WBCD	**F16**	**F17**	**F18**	**F19**	**F20**	**F21**	**F22**	**F23**	**F24**	**F25**	**F26**	**F27**	**F28**	**F29**	**F30**
MDisABC	9	14	8	10	10	24	0	7	0	13	21	21	17	22	9
DisABC	10	16	12	9	12	19	0	14	0	13	15	18	16	14	8
MRABC	16	12	21	14	15	21	0	23	0	15	19	16	12	16	14
AMABC	18	18	11	12	14	24	0	6	0	11	17	18	17	15	16
QBPSO	13	17	13	19	17	17	1	22	0	13	15	17	18	13	16
NBPSO	17	17	18	13	13	19	3	22	2	13	16	17	15	12	15
BPSO	20	14	19	14	16	28	0	8	0	17	21	26	22	19	7
GA	18	16	18	19	14	15	2	26	2	17	14	21	19	11	13

Table 9: Times of appearance of each feature over 30 runs for Ionosphere

Ionosphere	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**	**F10**	**F11**	**F12**	**F13**	**F14**	**F15**	**F16**	**F17**
MDisABC	13	13	15	1	28	0	2	7	6	1	0	0	1	14	13	14	4
DisABC	10	11	17	9	24	1	7	6	7	1	5	0	4	10	12	12	2
MRABC	17	14	19	0	27	9	3	7	16	6	12	2	7	19	16	15	6
AMABC	12	11	13	14	23	2	4	3	6	1	2	0	7	8	4	4	2
QBPSO	14	13	14	3	30	1	2	6	8	2	6	0	6	18	23	13	6
NBPSO	14	17	13	4	30	5	4	9	11	5	8	0	6	15	20	13	7
BPSO	12	11	14	2	29	7	4	9	14	1	12	3	12	15	18	14	8
GA	17	13	12	4	28	4	1	11	13	9	9	2	11	16	15	13	8

Table 9 Continued: Times of appearance of each feature over 30 runs for Ionosphere

Ionosphere	**F18**	**F19**	**F20**	**F21**	**F22**	**F23**	**F24**	**F25**	**F26**	**F27**	**F28**	**F29**	**F30**	**F31**	**F32**	**F33**	**F34**
MDisABC	2	2	1	0	0	8	0	0	0	20	1	0	0	0	0	5	2
DisABC	0	3	1	1	0	5	2	2	0	18	0	3	0	1	0	5	2
MRABC	2	5	4	7	7	15	3	17	1	11	3	2	3	6	2	11	10
AMABC	2	1	1	1	1	4	2	0	0	6	3	0	2	0	0	4	1
QBPSO	2	5	0	3	2	16	0	10	0	20	2	2	0	4	0	7	5
NBPSO	4	5	2	3	6	19	6	11	0	15	4	7	2	5	2	7	9
BPSO	3	11	2	3	4	15	4	11	0	17	2	6	2	8	5	12	13
GA	8	4	5	4	8	14	12	16	1	15	4	10	2	10	8	11	10