Information Sciences 507 (2020) 67–85

Contents lists available at ScienceDirect

Information Sciences

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

Binary differential evolution with self-learning for multi-objective feature selection

Yong Zhang [a,*], Dun-wei Gong [a], Xiao-zhi Gao [b], Tian Tian [c], Xiao-yan Sun [a]

[a] School of Information and Control Engineering, China University of Mining and Technology, Xuzhou, China
[b] School of Computing, University of Eastern Finland, Kuopio, Finland
[c] School of Computer Science and Technology, Shandong Jianzhu University, Jinan, China

ARTICLE INFO

Article history:
Received 23 December 2018
Revised 11 August 2019
Accepted 14 August 2019
Available online 15 August 2019

Keyword:
Differential evolution
Multi-objective optimization
Feature selection
Self-learning

ABSTRACT

Feature selection is an important data preprocessing method. This paper studies a new multi-objective feature selection approach, called the Binary Differential Evolution with self-learning (MOFS-BDE). Three new operators are proposed and embedded into the MOFS-BDE to improve its performance. The novel binary mutation operator based on probability difference can guide individuals to rapidly locate potentially optimal areas, the developed One-bit Purifying Search operator (OPS) can improve the self-learning capability of the elite individuals located in the optimal areas, and the efficient non-dominated sorting operator with crowding distance can reduce the computational complexity of the selection operator in the differential evolution. Experimental results on a series of public datasets show that the effective combination of the binary mutation and OPS makes our MOFS-BDE achieve a trade-off between local exploitation and global exploration. The proposed method is competitive in comparison with some representative genetic algorithm-, particle swarm-, differential evolution-, and artificial bee colony-based feature selection algorithms.

1. Introduction

Classification is an important topic in machine learning. By removing irrelevant or redundant features, a Feature Selection (FS) algorithm can effectively reduce the dimension of data, shorten the learning time, and improve the classification performance [13]. During the past decade, numerous FS algorithms have been proposed [30,31]. Among them, meta-heuristic methods have shown a lot of advantages in dealing with feature selection problems, due to their powerful exploration capabilities. Meta-heuristic feature selection methods include genetic algorithms [7,10], ant colony optimization [33], particle swarm optimization [4,42], firefly algorithm [46], memetic algorithm [27,47], artificial bee colony [45], grasshopper optimization algorithm [24], evolutionary gravitational search [3,35], etc.

Generally, feature selection can be modeled as a multi-objective combinatory optimization problem, which mainly contains two objectives, i.e., the number of the selected features and the classification accuracy. In some cases, removing irrelevant and redundant features from the original datasets can improve the classification accuracy of classifiers. In other words, the feature size can be reduced, while the classification accuracy is improved. However, when a dataset only contains the key features that must be used by classifiers, deleting any feature may increase the error rate of these classifiers. Here, the two objectives are conflicting with each other. Moreover, obtaining the report values of different features often needs dif-

* Corresponding author.
E-mail address: yongzh401@126.com (Y. Zhang).

https://doi.org/10.1016/j.ins.2019.08.040
0020-0255/

ferent level costs, such as time, money, or other resources. A large number of features usually mean a high cost. Therefore, reducing the number of the selected features is also an important indicator. Moreover, formulating a feature selection problem as a multi-objective optimization one is beneficial in obtaining a set of optimal feature subsets so as to meet various requirements of decision-makers. As we know that multi-objective evolutionary algorithms can seek multiple solutions lying in the Pareto optimal front in a single run. The aforementioned algorithms have been widely applied in handling the feature selection problems [17,38].

Because of its simplicity and efficiency [8,36], Differential Evolution (DE) is a very popular evolutionary algorithm used in the feature selection problems [1,2,16]. However, most of the current study focus on the single-objective case, i.e., maximizing the classification accuracy. There is little work on applying the DE to the multi-objective case. Xue et al. [37] firstly used the multi-objective DE in feature selection, and designed a DE-based multi-objective feature selection algorithm. After that, the multi-objective DE methods have been successfully employed to the multi-label feature selection [44], entity extraction in biomedical texts [32], and facial expression recognition systems [25]. The results obtained show the effectiveness of the multi-objective DE in handling the feature selection problems. However, these approaches have the following disadvantages. Firstly, most of them adopt the DE/rand/1/bin strategy to generate candidate individuals, where the base vector in the mutation is randomly chosen from the population. Due to the randomness of the base vector, this strategy makes the population yield a good exploration performance, but slows down the overall convergence. Secondly, the elite individuals in the population only take responsibility for guiding the search of other individuals. Improving the self-learning ability of the elite individuals may enhance the exploration of the whole population. Therefore, how to boost the global exploration of the DE without sacrificing its convergence needs new effective strategies to be developed.

To improve the capability of the original DE in dealing with the multi-objective feature selection, the present paper studies a new binary differential evolution with a self-learning strategy, namely MOFS-BDE. In our algorithm, the DE is responsible for exploring the search space and finding potential regions, while a self-learning strategy is utilized to effectively exploit these potential areas. The main contributions of this paper are as follows:

(1) A binary differential evolution algorithm with a self-learning strategy, MOFS-BDE, is proposed to attack the multi-objective feature selection problems.
(2) A new binary mutation operator based on the probability difference is designed to generate fresh solutions. Since the base vector is always the best one among the three randomly generated vectors, this operator can guide individuals to locate potentially optimal areas in a fast way.
(3) A new problem-specific self-learning strategy, namely one-bit purifying search, is proposed to refine the elite individuals in the population. Thus, the elite individuals not only take the responsibility of guiding the search of other individuals, but also have the self-learning capability.
(4) An efficient non-dominated sorting combined with crowding distance is employed to select appropriate parent individuals so as to reduce the time consumption of the selection operator in the regular differential evolution.

The structure of our paper is as follows. Section 2 introduces the background of the feature selection problems, and Section 3 reviews some of the existing evolutionary feature selection approaches. The standard and modified DE are discussed in details in Sections 4 and 5, respectively. Section 6 presents experimental results of the proposed MOFS-BDE. Finally, a few conclusions and remarks are given in Section 7.

2. Problem formulation

Suppose that S is a data set containing K samples and D features and *Fset* is the set of all the features, a FS problem can be described as follows: to select d features ($d \le D$) from all the features so that some objective functions, such as the classification error rate and classification accuracy, are optimized. Since the number of the selected features determines the computational cost of a classification algorithm, it is also a key objective function [42]. This paper considers the following two objective functions: minimizing the classification error rate (Err) and the number of the selected features.

We use a binary string to encode a solution to the FS problems:

$$X = (x_1, x_2, \ldots, x_D), \quad x_j \in \{0, 1\} \tag{1}$$

where $x_j = 1$ indicates that the jth feature is selected into the subset X; otherwise, it is not. A multi-objective FS problem is therefore formulated as:

$$\begin{aligned} &\min\ Err(X), \ \ \min\ |X| \\ &\text{s.}t.\ \ X = (x_1, x_2, \ldots, x_D),\ x_j \in \{0, 1\},\ j = 1, 2, \ldots, D, \\ &\qquad 1 \le |X| \le D. \end{aligned} \tag{2}$$

where $|X|$ is the number of the features within set X.

Generally, the two objectives, $Err(X)$ and $|X|$, are conflicting with each other. Take the image analysis problem as an example, the computational expense of the features refers to the time and space complexities of the feature acquisition process. That is, the larger the number of the selected features is, the higher the computational cost of acquiring the values of the features. However, a good value of $Err(X)$ usually depends on more features. The purpose of the multi-objective FS

problems is to minimize $Err(X)$ and $|X|$ simultaneously. Since this case has two conflicting objectives, there does exist a unique solution to make these two objectives minimal at the same time. Thus, the algorithm proposed in this paper aims at acquiring a group of the non-dominated optimal solutions or feature subsets.

3. EA-based feature selection algorithms

In principle, feature selection is an NP-hard combination problem, because the size of the search space increases exponentially as the number of features increases [31,48]. Since the Evolutionary Algorithm (EA) can find the best solutions with the global search strategies, it has been widely applied in dealing with the FS problems [12]. A detailed survey on these approaches can be found in [39]. This section only reviews some typical DE-based feature selection and multi-objective feature selection methods.

(1) DE-based single-objective feature selection approaches. Applying a float number-based DE algorithm in feature selection, Khushaba et al. [19] proposed a roulette wheel structure to make the solutions generated by the float-optimizer suitable for the discrete feature selection. However, this algorithm lacks the ability of effectively reducing the size of features, because it is limited to select feature subsets only with a predefined cardinality [37]. Al-Ani et al. [1] studied a novel wheel-structure approach to narrowing down the search space without eliminating any feature, and Kumar et al. [21] introduced a DE-based feature selection algorithm so as to solve the anaphora resolution in a resource-poor language. These work indeed verified the effectiveness of the DE in handling feature selection problems. However, the authors considered only one single objective, i.e., the classification accuracy. As mentioned above, formulating a feature selection problem as a multi-objective optimization one is beneficial to obtaining a set of the optimal feature subsets in order to meet various requirements of decision makers.

(2) Multi-objective feature selection approaches. For multi-objective feature selection problems, Oliveira et al. [28] made the first attempt by using a multi-objective genetic algorithm to generate a set of alternative classifiers. Based on the initial version of the NSGA, this method needs the sharing parameter to be specified. Following that, Hamdani et al. [15] introduced the NSGA-II into feature selection, but the performance of their method has not been compared with any other EA-based algorithms. During the recent years, more and more evolutionary algorithms have been used to multi-objective feature selection problems. For example, Xue et al. [38] developed two improved multi-objective feature selection algorithms by combining the crowding distance and Pareto dominance relationship together in the particle swarm optimization. Hancer et al. [17] proposed a multi-objective feature selection approach using a new artificial bee colony algorithm integrated with the non-dominated sorting and genetic operators (MOABCFS), and implemented two different versions, i.e., binary version B-MOABCFS and continuous version C-MOABCFS. Their experimental results showed that the former outperformed the latter in most cases. Additionally, the EA-based multi-objective feature selection schemes have been applied to deal with real-world problems, such as facial expression recognition [25], prediction of warfarin dosage [34], online sales forecasting problems [18], etc.

The applications of the DE have been extended to the multi-objective optimization cases in the past decade. Xue et al. [37] proposed a DE-based multi-objective method, namely DEMOFS. Sikdar et al. [32] studied a binary multi-objective feature selection algorithm using the DE (B-DEMOFS) for the entity extraction in biomedical texts. In this algorithm, a binary coding strategy was introduced to transfer a continuous DE into the binary version. However, these approaches usually suffer from the disadvantage of stagnating in the local optima, because they use the traditional DE operators, e.g., the DE/rand/1/bin strategy, and generally are lack of the problem-oriented operators.

4. Differential evolution

In the DE, each individual represents a possible solution to the optimization problem. For a population with N individuals, the first step of the DE is randomly generating N individuals (target vectors). For each target vector, a trial vector is next obtained by using the mutation and crossover. The selection operator further generates a new parent population for the successive generation based on these trial and target vectors.

Mutation: In the mutation, the DE generates a new reference (mutation) vector by using the difference between two random vectors. For the ith target vector $X_i(t)$, a new mutation vector $V_i(t)$ is generated as follows:

$$V_i(t) = X_{r1}(t) + F \cdot (X_{r2}(t) - X_{r3}(t)), \tag{3}$$

where t is the iteration step. The three vectors, $X_{r1}(t)$, $X_{r2}(t)$, and $X_{r3}(t)$, are randomly selected from the population, and $r1 \neq r2 \neq r3 \neq i$. The parameter, $F \in (0, 2]$, is a scale that amplifies the difference between $X_{r2}(t)$ and $X_{r3}(t)$.

Crossover: The DE follows a discrete recombination approach, i.e., elements from the target vector $X_i(t)$ are combined with those from the new mutation vector $V_i(t)$ to produce a trial vector $U_i(t)$,

$$\begin{aligned} U_i(t) &= (u_{i,1}(t), u_{i,2}(t), \ldots, u_{i,D}(t)) \\ u_{i,j}(t) &= \begin{cases} v_{i,j}(t), & \text{if } U(0,1) < CR \text{ or } j = h \\ x_{i,j}(t), & \text{otherwise} \end{cases} \end{aligned} \tag{4}$$

where D is the variable dimension, $U(0, 1)$ is a random value between 0 and 1, and $CR \in [0, 1]$ is the crossover probability. A random index h is applied here to guarantee that $U_i(t)$ always receives at least one element from $V_i(t)$.

Selection: The DE uses a very simple selection procedure. If the fitness of the trial vector $U_i(t)$ is better than that of the target vector $X_i(t)$, the target vector is set to be the trial vector $U_i(t)$ in the next generation, i.e., $X_i(t+1) = U_i(t)$; otherwise, $X_i(t+1) = X_i(t)$.

5. A new multi-objective feature selection algorithm

In this section, our binary differential evolution algorithm with the self-learning strategy, MOFS-BDE, is described. First, we present the binary mutation on the basis of the probability difference. The self-learning strategy, i.e., one-bit purifying search, is also introduced to enhance the performance of MOFS-BDE. The improved selection operator combining the non-dominated sorting and crowding strategy is next presented. Finally, the steps of the proposed MOFS-BDE are elaborated, and its computational complexity is further discussed.

5.1. The binary mutation with probability difference

This section proposes a new binary mutation on the basis of the probability difference, which can improve the algorithm's convergence without sacrificing its global exploration ability. Different from the DE/rand/1/bin strategy used in the existing DE-based feature selection algorithms [11], the proposed mutation operator selects the best one among the three random vectors as the base vector, and employs the difference between the remaining two vectors as a mutation probability to be used on the base vector to generate a mutation vector for the next crossover operator.

Consider the ith target vector in the population $X_i(t)$, the new mutation is given as follows:

$$\begin{cases} V_i(t) = (v_{i,1}(t), v_{i,2}(t), \ldots, v_{i,D}(t)) \\ v_{i,j}(t) = \begin{cases} x_{best,j}(t), & c_{i,j} < rand \\ 1 - x_{best,j}(t), & \text{otherwise} \end{cases} \\ C_i = \begin{cases} \sigma, & \text{if} X_{best}(t) \prec X_i(t) \\ \min(1,\ F \cdot (X_{r1}(t) \oplus X_{r2}(t)) + \sigma), & \text{otherwise} \end{cases} \end{cases} \tag{5}$$

where $X_{best}(t)$ refers to the best one among the three vectors randomly selected. $X_{r1}(t)$ and $X_{r2}(t)$ are the remaining two among these three vectors. $X_{best}(t) \prec X_i(t)$ indicates that $X_{best}(t)$ dominates $X_i(t)$. The symbol "$\oplus$" is an XOR operator, and $C_i = (c_{i,1}, c_{i,2}, ..., c_{i,D})$ is a probability vector produced by performing XOR on $X_{r1}(t)$ and $X_{r2}(t)$. The parameter σ is a small turbulence coefficient. $F \in (0, 1]$ is a scale parameter for controlling the learning rate of an individual from $X_{best}(t)$. The item $\min(1,\ F \cdot (X_{r1}(t) \oplus X_{r2}(t)) + \sigma)$ ensures that the value of $F \cdot (X_{r1}(t) \oplus X_{r2}(t)) + \sigma$ is less than 1.

Compared with the conventional mutation in Eq. (3), our mutation operator has the following characteristics:

(1) It can generate mutation vectors with a good diversity. When $X_{best}(t)$ is inferior to $X_i(t)$, the probability difference, i.e., $\min(1,\ F \cdot (X_{r1}(t) \oplus X_{r2}(t)) + \sigma)$, is used to produce a mutation vector $V_i(t)$. Since both $X_{r1}(t)$ and $X_{r2}(t)$are randomly selected, adding their difference to $X_{best}(t)$ is beneficial for maintaining the diversity of the mutation vectors.

(2) It can improve the convergence of the population. When $X_{best}(t)$ dominates $X_i(t)$, i.e., $X_{best}(t) \prec X_i(t)$, we directly set $X_{best}(t)$ as the mutation vector of the ith target vector. Since the base vector $X_{best}(t)$ is superior to the target vector $X_i(t)$, combining them together can make the newly generated solutions inherit good information from $X_{best}(t)$.

(3) The turbulence coefficient σ is employed to guarantee that the mutation probability on the base vector is always larger than 0. This will prevent new solutions from falling into local optima. As a matter of fact, a large value of σ can improve the diversity of new solutions, but may destroy the useful information inherited from $X_{best}(t)$. On the other hand, if the value of σ is set to be too small, it is difficult to enable individuals to escape from the local optima. Our experimental results indicate that a small value within [0.001, 0.01] is appropriate.

5.2. The one-bit purifying search

To improve the self-learning ability of the elite individuals in the population, we design a new self-learning strategy by utilizing the importance degree of features, namely the One-bit Purifying Search (OPS). In order to compare the importance degree between any two features, the relative importance is first defined by comparing their influences on enhancing the performance of a specified optimal solution.

Relative importance: Let u_1 and u_2 be two feature bits randomly selected from a non-dominated solution, $X = (x_1, x_2, \ldots, x_D)$, which satisfy $x_{u_1} = 1$ and $x_{u_2} = 0$. $X_{X(u_1)=0,X(u_2)=1}$ is a new solution generated by setting the u_1 and u_2 values of X to be 0 and 1, respectively. In addition,$X_{X(u_1)=0}$ is another solution generated by setting the u_1 value of X to be 0. We define that the u_1-th feature is more important than the u_2-th feature with respect to X, denoted as $u_1 \succ_{imp} u_2$, if the degradation degree of the classification accuracy satisfies:

$$\left|Cerror(X_{X(u_1)=0}) - Cerror(X)\right| > \left|Cerror(X_{X(u_1)=0}) - Cerror(X_{X(u_1)=0,X(u_2)=1})\right|,$$

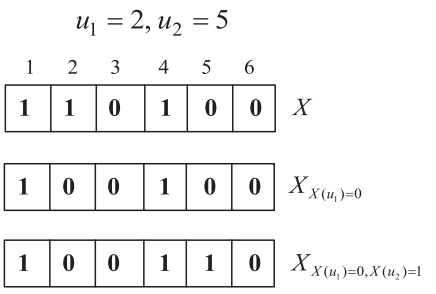

Fig. 1. An example of calculating the relative importance.

Algorithm 1
One-bit purifying search (OPS).

Input: The populationP_t, and the set of non-dominated solutions S_t;
Output: The new populationP_t.
Step 1: Randomly select a solution fromS_t, and set it as the reference solution,$X_{ref} = (x_{ref.1}, x_{ref.2}, ..., x_{ref.D})$;
Step 2: Randomly select two feature bits, u_1 and u_2, fromX_{ref}, satisfying $x_{ref.u_1} = 1$ and $x_{ref.u_2} = 0$;
Step 3: Judge the relative importance between the two feature bits u_1 and u_2;
Step 4: For an optimal solution,$X_h \in S_t$, **do**
  Step 4.1: Generate a new individualX'_h by checking the following four cases: (Without loss of generality, we suppose $u_1 \succ_{imp} u_2$)
  Initialize $X'_h = X_h$
  (Case 1) If $x_{h,u_1} = x_{h,u_2} = 1$, set $x'_{h,u_2} = 0$
  (Case 2) else if $x_{h,u_1} = x_{h,u_2} = 0$, set $x'_{h,u_1} = 1$
  (Case 3) else if $x_{h,u_1} = 1, x_{h,u_2} = 0$, set$x'_{h,u_1} = 0$
  (Case 4) else set $x'_{h,u_1} = 1, x'_{h,u_2} = 0$.
  End if
  Step 4.2: If X'_h dominates X_h, populationP_t saves X'_h to replace X_h; if X'_h is dominated byX_h, the population keeps X_h unchanged; otherwise, it saves both X'_h and X_h into P_t.
  End for
Step 5: If the size of P_t is larger than N, remove$|P_t| - N$individuals with high ranks, and reduce the crowding distances from P_t using the method in Section 5.3;
Step 6: Output population P_t

where $Cerror()$ represents the classification error rate. $|Cerror(X_{X(u_1)=0}) - Cerror(X)|$ describes the change degree of the classification performance, when the u_1th feature is added into the subset $X_{X(u_1)=0}$, and $|Cerror(X_{X(u_1)=0}) - Cerror(X_{X(u_1)=0,X(u_2)=1})|$ describes the change degree of the classification performance, when the u_2th feature is added into the subset $X_{X(u_1)=0}$. The method of calculating the error rate of classification is given in Section 6.1.

Fig. 1 provides an example of calculating the relative importance. Assume that $X = (110100)$, and the two selected feature bits are $u_1 = 2$ and $u_2 = 5$, respectively, we have $X_{X(u_1)=0,X(u_2)=1} = (100110)$ and $X_{X(u_1)=0} = (100100)$.

Algorithm 1 shows the calculation procedure of the OPS. Firstly, we randomly select a reference solution X_{ref} from the non-dominated set. Secondly, two feature bits amongX_{ref}, i.e.,u_1 and u_2, are randomly selected in Step 2, and Step 3 compares their relative importance degrees. Next, for a non-dominated individual,$X_h \in S_t$, $h = 1, 2, ..., |S_t|$, the OPS operator is used to generate a new individual X'_h by checking the following four cases in Step 4.1. Suppose that $u_1 \succ_{imp} u_2$, and the initial value $X'_h = X_h$, the detailed explanations of these four cases are given as follows.

Case 1: if the values of X'_h on both u_1 and u_2 are equal to 1, the value of X'_h on u_2 is set to be 0, i.e., removing the u_2-thfeature with less importance degree from X'_h.

Case 2: if the values of X'_h on both u_1 and u_2 are equal to 0, the value of X'_h on u_1 is set to be 1, i.e., adding the u_1-th feature with more importance degree into X'_h.

Case 3: if the values of X'_h on u_1 and u_2 are equal to 0 and 1, respectively, the value of X'_h on u_2 is set to be 0 and that of X'_h on u_1 to be 1, i.e., adding the u_1th feature with more importance degree into X'_h, and removing the u_2th feature with less importance degree from X'_h.

Case 4: if the values of X'_h on u_1 and u_2 are equal to 1 and 0, respectively, the value of X'_h on u_1 is set to be 0, i.e., removing both these two features from X'_h.

Fig. 2 shows an example of the one-bit purifying search, where u_1 and u_2 are two random feature bits, and "?" is a wild card character. $X_i (i = 1, 2, 3, 4)$ is a candidate solution from the population, and $X'_i (i = 1, 2, 3, 4)$ is a new individual generated by modifying the value of the candidate solution on u_1 and u_2. Fig. 2(a) and (b) illustrate the process of performing the one-bit purifying search, when $u_1 \succ_{imp} u_2$ and $u_2 \succ_{imp} u_1$, respectively.

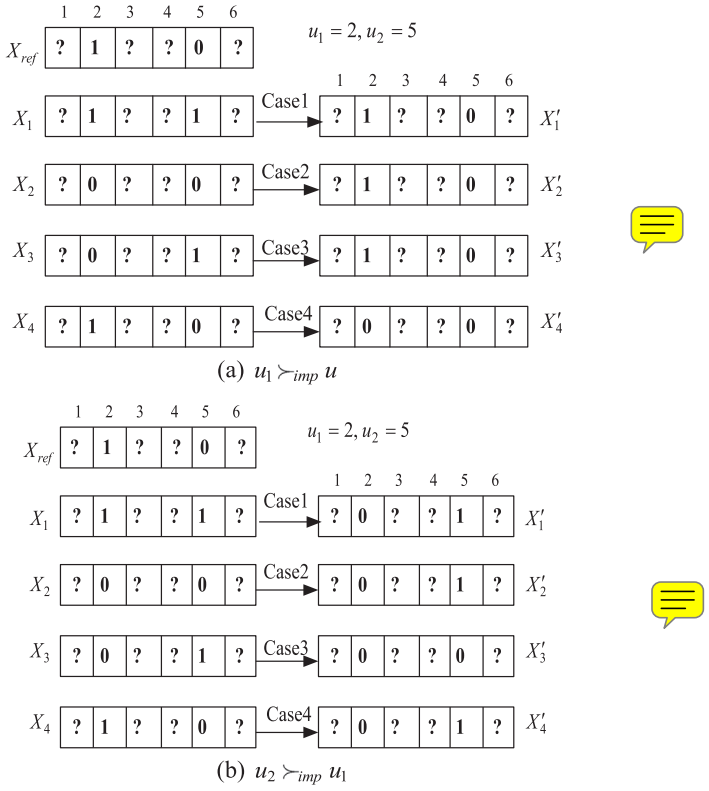

Fig. 2. An example of the OPS.

Repeating the above procedure on the elitist individuals, the one-bit purifying search can generate some new individuals. The new individuals will be saved into the population P_t in Step 4.2, and the method in Section 5.3 is employed to prune the new population in case its size is larger than N.

Actually, the OPS operator is a small-scale refinement process with the purpose of improving the quality of non-dominated solutions (elite individuals among the population). Nevertheless, the DE also performs a large-scale search in the variable space. By incorporating the OPS operator into the modified DE, our new algorithm can achieve an appropriate trade-off between the exploitation and exploration.

5.3. The selection based on an efficient non-dominated sorting

The Fast Non-dominated Sorting (FNS) [9], as an effective selection method, has been applied to rank the optimal individuals in DEMOFS [37]. However, this technique is computationally intensive, and its time complexity is $O(M \times N^2)$, where N is the population size, and M is the number of the objectives. The Efficient Non-dominated Sorting (ENS) [41] is a new comparison technique, which has a lower complexity of $O(M \times N \times \log N)$. In this section, we introduce an improved selection operator by combining the ENS with the crowding distance.

Let the set of the target vectors (or parent individuals) at the tth generation be P_t, the set of the trial vectors (or offspring individuals) produced by the crossover and mutation be Q_t, and the new population be $P_{t+1} = \emptyset$. The selection process is explained as follows. Firstly, the individuals in P_t are examined. For an individual $X_i(t)$ in P_t, if it is dominated by a trial vector, $U_i(t) \in Q_t$, $U_i(t)$ will be saved into P_{t+1}; if it dominates $U_i(t)$, $X_i(t)$ will be saved into P_{t+1}; otherwise, both $U_i(t)$ and $X_i(t)$ will be saved into P_{t+1}. If the size of P_{t+1} is larger than the population size N, these solutions in the population P_{t+1} will be sorted in an ascending order according to the first objective function value. If two solutions have the same values for the first objective function, they are sorted using the second objective function. Secondly, the ENS is applied by assigning each individual with a rank and classifying each into a rank set. Finally, the new population is formed by selecting the individuals based on the order of their ranks. If two solutions have the same rank, the one with a higher crowding distance wins. For the calculation of the crowding distance of a solution, refer to [9].

5.4. Procedure and computational complexity of MOFS-BDE

The proposed MOFS-BDE is explained in details in Algorithm 2. In this algorithm, when determining the base vector for an individual, if the three random vectors include more than one optimal solution, the one with the largest crowding

Algorithm 2
MOFS-BDE.

Parameters: The maximal iteration times $T_{\max}$, the population size N, the frequency of implementing OPS T_{loc}, the scale F, and the crossover probability CR.
Input: The dataset for classification.
Output: The Pareto-optimal solutions with each corresponding to a feature subset.
Step 1: Initialize a number of individuals, $P_0 = \{X_1, X_2, ..., X_N\}^T$
Step 2: Let $t = 0$. //Iteration steps
Step 3: Iteration
Set the set $P_{t+1} = \emptyset$;
for $i = 1, 2, ..., N$, **do**
Step 3.1: Randomly select three vectors from the population P_t, denoted as $X_{r1}(t)$,$X_{r2}(t)$, and $X_{r3}(t)$, $r1 \neq r2 \neq r3 \neq i$;
Step 3.2: Select the best one from the three vectors as the base vector,$X_{best}(t)$;
Step 3.3: Generate a new mutation vector $V_i(t)$ for the ith individual according to Eq. (5);
Step 3.4: Generate a trial vector $U_i(t)$ for the ith individual according to Eq. (4);
Step 3.5: Evaluate the fitness of the trial vector $U_i(t)$;
Step 3.6: Compare the ith individual $X_i(t)$ with $U_i(t)$.If $X_i(t)$ dominates $U_i(t)$, save $X_i(t)$ into P_{t+1};if $U_i(t)$ dominates$X_i(t)$, save $U_i(t)$ into P_{t+1};otherwise, save both $U_i(t)$ and $X_i(t)$ into P_{t+1}.
Endfor
Step 3.7: If the size of $P_{t+1}$is larger than N, remove $|P_{t+1}| - N$ individuals with higher ranks and shorter crowding distances from P_{t+1}using the method in Section 5.3;
Step 3.8: If $t/T_{loc} = \lceil t/T_{loc} \rceil$,run the problem-specific local search (refer to Algorithm 1 for details);
Step 4: If $t < T_{\max}$, let t ++, and return back to Step 3; otherwise, terminate the algorithm, and output the Pareto-optimal solutions.

distance will be selected as the base vector $X_{best}(t)$. Additionally, a parameter T_{loc} is employed to control the frequency of implementing the OPS operator in *Algorithm 1*.

The scale factor F, the basic crossover probability CR, and the parameter T_{loc} play key roles in our MOFS-BDE. The parameter F is a scale factor determining the difference between the generated offspring and $X_{best}(t)$. The higher the value of F, the more the diversity of the generated offspring. Like in the standard DE [8], $0 < F < 1$ is usually a good and reliable choice. The parameter CR is a probability within [0,1]. Similar to the mutation probability in the genetic algorithms, it determines how many feature bits under expectation are inherited from the target vector. A low value of CR indicates that a small number of feature bits are changed and the exploration capability of the population is weak. On the contrary, increasing the value of CR can enhance the exploration ability of the population. In the MOFS-BDE, the OPS operator is designed to improve the self-learning ability of the elite individuals. In other words, it is used to further exploit the potential areas, and the value of T_{loc} is capable of controlling the frequency of exploitation. Therefore, a small value of T_{loc} can lead to a better exploitation capability of the population, although frequently applying this operator may increase the computational cost of the MOFS-BDE.

The complexity of Algorithm 2 mainly depends on Step 3, because Steps 1, 2, and 4 can be finished in linear time scale. Step 3 includes the following four parts: the mutation operator from Steps 3.1 to 3.3, the crossover operator in Step 3.4, the selection operator in Steps 3.6 and 3.7, and the OPS operator in Step 3.8. In fact, the mutation operator executes $O(N)$ basic operations, and the crossover operator does $O(N)$ basic operations. Therefore, their computational complexities are $O(N)$. Since the ENS has the complexity of $O(M \times N \times logN)$, and that of the crowding distance is $O(M \times N \times logN)$, the selection operator has the complexity of $O(M \times N \times logN)$. For the self-learning OPS operator in *Algorithm 1*, its complexity is in Steps 4 and 5, because Steps 1, 2, 3, and 6 can be implemented with linear time scale. Step 4 executes $O(|S_t|)$ basic operations to generate new individuals, and when the number of the individuals in P_t is larger than N, Step 5 needs $O(M \times N \times logN)$ basic operations to prune the population. Given the fact that $|S_t| \leq N$, the computational complexity of the OPS operator is $O(M \times N \times logN)$.

Fig. 3 shows the complexities in the MOFS-BDE compared with that in the DEMOFS. The framework of a multi-objective DE is divided into six parts: initialization (Step 1 in Algorithm 2), mutation (Steps 3.1, 3.2, and 3.3 in Algorithm 2), crossover (Step 3.4 in Algorithm 2), self-learning operator (Step 3.8 in Algorithm 2), evaluation of the individuals (Step 3.5 in Algorithm 2), and selection (Steps 3.6, 3.7 in Algorithm 2). Note that for evaluating the individuals, i.e., Step 3.5 in Algorithm 2, the above analysis only considers the number of the individuals evaluated in each iteration. It can be found out that the overall complexity of our MOFS-BDE is $O(M \times N \times logN)$.

As aforementioned, the multi-objective DE-based feature selection algorithm (DEMOFS) [37], the NSGA-II-based feature selection algorithm (NSGAFS) [15], the particle swarm optimization-based feature selection algorithm (MOPSOFS) [38], the binary multi-objective DE-based feature selection algorithm (B-DEMOFS) [32], and the Pareto front feature selection algorithm based on artificial bee colony optimization (B-MOABCFS) [17] are five representative algorithms for feature selection. Since the B-MOABCFS, DEMOFS, and NSGAFS all use the non-dominated sorting strategy to update their populations, they have the same computational complexity, $O(M \times N^2)$. The B-DEMOFS and MOPSOFS have the same computational complexity of $O(M \times N \times logN)$, because they adopt the crowding distance to prune the archive. Therefore, the computational complexity of the proposed MOFS-BDE is competitive compared with the above five evolutionary FS algorithms.

However, as we know that these five EA-based algorithms all need a classifier to calculate the objective values of the individuals, i.e., the classification performance of the feature subsets. Additionally, calculating the classification performance

	MOFS-BDE	DEMOFS
Initialization	$O(N)$	$O(N)$
Mutation	$O(N)$	$O(N)$
Crossover	$O(N)$	$O(N)$
Self-learning operator OPS	$O(M \times N \times logN)$	0
Evaluating the individuals	$O(N)$	$O(N)$
Selection	$O(M \times N \times logN)$	$O(M \times N^2)$
The whole complexity	$O(M \times N \times logN)$	$O(M \times N^2)$

Fig. 3. The computational complexities in MOFS-BDE and DEMOFS.

of a feature subset often results in a high computational cost. More importantly, different feature subsets or individuals usually cause different computation costs for the same problem, because these subsets include different numbers of features. Like the four EA-based algorithms, i.e., DEMOFS, NSGAFS, MOPSOFS, and B-MOABCFS, in case of high-dimensional datasets, the real running time of the MOFS-BDE mainly depends on the evaluation of the individuals in Fig. 3, which is the process of calculating the objective values of the individuals. This is the reason why the real running time of the six algorithms in Table 9 does not accord with the computation cost from the theoretical analysis in Section 5.4.

6. Experimental study

This section evaluates our new feature selection algorithm on a total of 20 standard datasets. These datasets are from the website of the UCI repository [26]. They have been processed by the providers in advance. The proposed algorithm and other methods in comparison are implemented using Matlab language. All the experiments are conducted on an Intel Core (TM) i5-3470 CPU with 4 GB of RAM.

6.1. Algorithms and performance metric

To validate the performance of the proposed MOFS-BDE algorithm, four popular feature selection algorithms, DEMOFS [37], NSGAFS [15], MOPSOFS [38], and B-MOABCFS [17], are used in comparison. To our best knowledge, the DEMOFS is the first attempt to employ the multi-objective DE for feature selection. Since the NSGA-II is one of the most widely employed multi-objective evolutionary algorithms, the NSGAFS has been considered as a benchmark method here. The MOPSOFS is a well-known multi-objective particle swarm optimization-based feature selection algorithm. The B-MOABCFS is a relatively new multi-objective feature selection approach.

Moreover, a novel multi-objective evolutionary algorithm, namely multi-objective evolutionary algorithm based on the $l2$-norm Tchebycheff decomposition and maximal fitness improvement (MOEA/D-2TMFI) [23], is also selected for effectiveness comparison. Proposed by Zhang and Li [40], the multi-objective evolutionary algorithm based on decomposition (MOEA/D) is one of the classic multi-objective evolutionary methods. The MOEA/D-2TMFI is an enhanced version of the MOEA/D. In order to apply MOEA/D-2TMFI in coping with binary feature selection problems, the following strategy is used to transform a real individual to the binary version: if an element of a new individual is more than or equal to 0.5, its value is set to be 1; otherwise, this element is set to be 0. Note that employing MOEA/D-2TMFI to the binary feature selection problem may fail to demonstrate its real advantages, because it is originally proposed for continuous optimization problems.

Since all the algorithms belong to the wrapper approaches, they need some learning algorithms to calculate the classification error ratio of a feature subset (solution). It is worth pointing out that many classifiers can be used including artificial

neural networks and SVM. Since the type of the classifier selected influences the feature selection results, the same classifier, *K*-Nearest Neighbor (*K*-NN), is utilized for all the above algorithms. As one of popular learning algorithms, *K*-NN has been applied in numerous EA-based FS approaches [43]. In this paper, we use the Leave-One-Out Cross Validation (LOOCV) of *K*-NN to calculate the classification error ratio of a feature subset (solution). In the LOOCV, a sample from the original dataset is chosen as the testing datum, and the remaining samples are considered as the training data. The *K*-NN classifier can predict the class label of this testing datum. If the prediction result is incorrect, the number of the incorrectly predicted samples is increased by one. Each sample in the dataset is used only once. The error rate of a feature subset is the proportion of those incorrectly predicted samples to all the samples.

For the *K*-NN classifier, the optimal value of *K* has a close relation with the dataset under consideration, and is often estimated based on the training samples available [14]. A large value of *K* is beneficial to decrease the influence of noises on the classification accuracy, whereas the boundary between the classes becomes less distinct [6]. Moreover, we target at validating the capability of MOFS-BDE of seeking the optimal feature subset instead of investigating the impact of *K* on its performance. Therefore, for the sake of simplicity, the 1-NN classifier is used in our simulations.

In the NSGAFS [38], the bit-flip mutation and single-point crossover are employed with the mutation rate being 1/D and the probability of crossover being 0.9. Based on the suggestions given in [37], in the DEMOFS, we set the crossover rate and scaling factor to be 0.3 and 0.5, respectively. The inertia weight w is set to be a random value within [0.1, 0.5], the acceleration constants, c_1 and c_2, are two random values within [1.5, 2.0], and the mutation rate is 1/D in the MOPSOFS [38]. In the B-MOABCFS [17], the number of the colony size is set to be 50, the number of food sources equals to the half of the colony size, and the limit trial parameter is set to be 5. For the MOEA/D-2TMFI [23], the DE and polynomial mutation are used to generate new individuals, where the mutation rate, the crossover rate, and the selection probability of the neighborhoods are set to be 1/D, 0.3, and 0.9, respectively. The scale factor is randomly set to be within [0, 1]. In our MOFS-BDE, we choose the scale factor F to be 0.5*rand, the crossover probability to be 0.3, and T_{loc} to be 5. For all these six algorithms, both the population size and archive size are set to be 50, and the maximal number of evaluations is set to be 5000 for the datasets with less than 100 features and 15,000 for the other datasets. Moreover, Table 2 gives the parameter configurations of the above algorithms.

In order to evaluate the performance of a multi-objective feature selection algorithm, the Hyper-Volume (HV) metric [20] is used in our experiments. Take a set of Pareto optimal solutions, Z, as an example, HV is applied to calculate the hyper-volume of the region enclosed by Z and a reference point, where the reference point is a vector dominated by all the solutions obtained. Since the HV metric can simultaneously evaluate the distribution and convergence of solutions, it has been widely employed in evaluating the multi-objective evolutionary algorithms [5,22,29]. Moreover, the Set Coverage (SC) [49] is utilized to compare the convergence of Pareto optimal solutions obtained by two algorithms. Let A1 and A2 be two sets of the Pareto optimal solutions, the value C(A1,A2)=1 means that all the solutions of A2 are dominated by or equal to some solutions of A1 and the convergence *of* A2 is no better than that of A1. The number of feature subsets (FN) is used to compare the diversity of solutions obtained by an algorithm. The value of FN is the average of the number of the feature subsets found by an algorithm in 30 runs. Therefore, a higher FN value implies that there are more diversified feature subsets for the decision makers to choose from.

6.2. Analyses of the proposed operators

As previously discussed, our paper employs two new operators, i.e., modified binary mutation and OPS operator, to improve the performance of MOFS-BDE. This section performs an extensive analysis on these two key operators. Table 3 shows the average HV values of MOFS-BDE, MOFS-BDE without OPS, and DEMOFS for the eight typical datasets taken from Table 1.

We first investigate the effectiveness of the modified binary mutation in improving the performance of MOFS-BDE. The main difference between DEMOFS and MOFS-BDE is the mutation and selection. As aforementioned, DEMOFS and MOFS-BDE adopt the FNS- and ENS-based selection, respectively. Although ENS has a lower computational complexity than that of FNS, both these two strategies get the same results for each input. Therefore, it is reasonable to validate the modified binary mutation operator by comparing MOFS-BDE without OPS with DEMOFS. It can be discovered from Table 3 that MOFS-BDE without OPS achieves larger HV values than that of DEMOFS for six out of the eight datasets, which suggests that the proposed binary mutation is more effective in improving the proposed algorithm.

Next, we explore the OPS operator by comparing MOFS-BDE with MOFS-BDE without OPS. Table 3 demonstrates that the two algorithms can obtain similar HV values for the datasets with small features, i.e., Vowel, Vehicle, and Ionosphere. However, for those with more features, MOFS-BDE with OPS yields better performances than the one without OPS with respect to the HV metric. That is, the OPS operator plays an important role in the MOFS-BDE.

Moreover, an optimal learning strategy on the best base vector has been designed to improve the convergence of the population. We can modify Eq. (5) as follows:

$$\begin{cases} V_i(t) = (v_{i,1}(t), v_{i,2}(t), \infty \ldots, v_{i,D}(t)) \\ v_{i,j}(t) = \begin{cases} x_{r3,j}(t), & c_{i,j} < rand \\ 1 - x_{r3,j}(t), & \text{otherwise} \end{cases} \\ C_i = \min\left(1,\ F \cdot (X_{r1}(t) \oplus X_{r2}(t)) + \sigma\right) \end{cases} \tag{6}$$

Table 1
Basics of test databases.

Datasets	The number of samples	The number of classes	The number of features
Vowel	990	11	10
Wine	178	3	14
Zoo	101	7	17
Vehicle	846	4	18
WDBC	569	2	32
Ionosphere	351	2	34
Satellite	856	7	37
SPECTF	160	2	44
Parkinson	240	2	46
Sonar	208	2	60
Libras Movement	360	15	90
Hill-valley	606	2	100
Urban land cover(ULC)	507	9	148
Musk	476	2	167
SCADI	70	7	206
LSVT	126	2	309
CNAE-9	540	9	857
Yale_64	165	15	1024
SRBCT	83	4	2308
DLBCL	45	2	5469

Table 2
Parameters of comparison algorithms.

Algorithms	Values of related parameters
NSGAFS	The mutation rate $= 1/D$, and the crossover probability $y = 0.9$.
DEMOFS	The crossover rate $CR = 0.3$, and the scaling factor $F = 0.5$.
MOPSOFS	The inertia weight $w = 0.1 + 0.4 \times$ rand, the acceleration constants, $c_1 = c_2 = 1.5 + 0.5 \times$ rand, the mutation rate $1/D$, and the archive size $= 50$.
B-MOABCFS	The colony size $N = 50$, the number of food sources $= N/2$, and the limitation trial parameter $= 5$.
MOEA/D-2TMFI	The mutation rate $= 1/D$, the crossover rate $CR = 0.3$, the selection probability of neighborhoods $= 0.9$, and the scale factor $F = rand$.
MOFS-BDE	The scale factor $F = 0.5 \times$ rand, the basic crossover probability $CR = 0.3$, the parameter $T_{loc} = 5$, and the turbulence coefficient $\sigma = 0.01$.

Table 3
The average HV values of MOFS-BDE, MOFS-BDE without OPS, and DEMOFS.

Algorithms	Vowel	Vehicle	Ionosphere	Sonar	Hill-valley	ULC	LSVT	CNAE-9
MOFS-BDE	0.8262	0.6964	0.9300	0.9175	0.6681	0.8873	0.8163	0.8381
DEMOFS	0.8260	0.6894	0.9296	0.9038	0.6517	0.8788	0.7510	0.8113
MOFS-BDE without OPS	0.8264	0.6936	0.9286	0.9105	0.6590	0.8833	0.7975	0.8209

Where, $X_{r1}(t)$, $X_{r2}(t)$, and $X_{r3}(t)$ are randomly selected from the population, and $r1 \neq r2 \neq r3 \neq i$. We denote MOFS-BDE with Eq. (6) as MOFS-BDE/random. Take the data sets Vowel, Ionosphere, Hill-valley, and LSVT as examples, Fig. 4 shows the HV curves obtained by MOFS-BDE and MOFS-BDE/random. We can find out that for Vowel and Ionosphere, the optimal learning strategy obviously accelerates the convergence speed of the population at the early stage of MOFS-BDE. For the datasets with more features, Hill-valley and LSVT, with the help of this optimal learning strategy, MOFS-BDE shows a better convergence than that of MOFS-BDE/random. Hence, the proposed optimal learning strategy is regarded as another strength of our MOFS-BDE.

6.3. Sensitivity analysis on key parameters

In our algorithm, the two new parameters, i.e., the frequency of implementing OPS T_{loc}, and the turbulence coefficient σ, both play important roles. This section aims at analyzing the sensitivity of the algorithm to these two parameters. We change the value of T_{loc} in the range of {1, 3, 5, 7, 9}, and alter the value of σ in range of {0.001, 0.005, 0.01, 0.02, 0.05, 0.1}. A total of four datasets, Vowel, Ionosphere, Hill-valley, and LSVT, are used as test beds. Table 4 shows the HV values obtained by MOFS-BDE with different T_{loc} values. Clearly, for the two simple datasets, Vowel and Ionosphere, the proposed algorithm provides good HV values, when the value of T_{loc} varies from 1 to 9. For the other two datasets, Hill-valley and LSVT, it gives satisfactory HV values, only when the value of T_{loc} changes from 1 to 5. Since frequently implementing the OPS operator can increase the algorithm's computational cost, we set T_{loc} to be only 5 in the simulations.

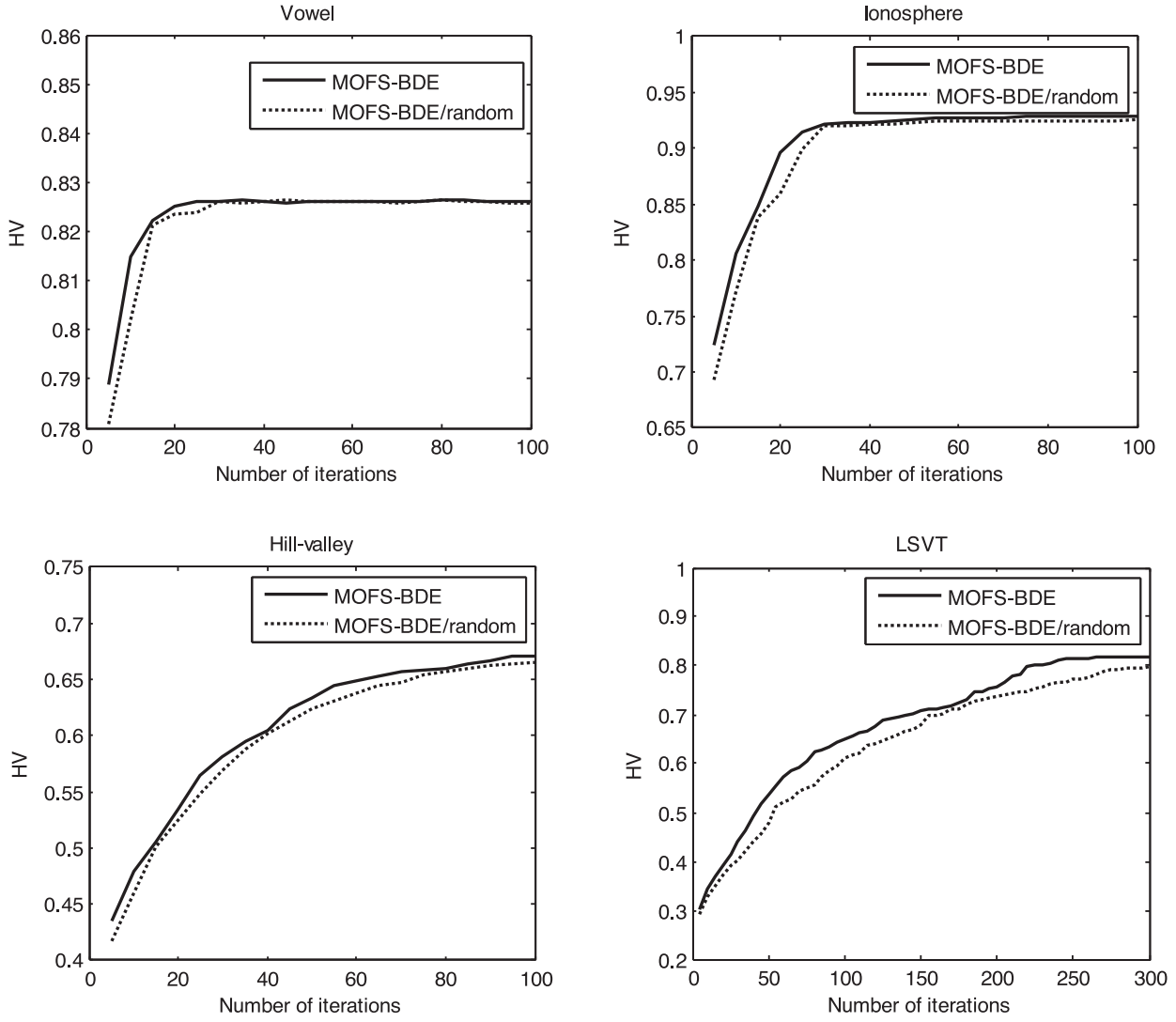

Fig. 4. The HV curves obtained by MOFS-BDE and MOFS-BDE/random.

Table 4
The HV values obtained by the proposed algorithm with different T_{loc} (Average/Std.).

Datasets	$T_{loc}=1$	$T_{loc}=3$	$T_{loc}=5$	$T_{loc}=7$	$T_{loc}=9$
Vowel	0.8264/0.0004	0.8262/0.0005	0.8262/0.0004	0.8260/0.0003	0.8261/0.0005
Ionosphere	0.9293/ 0.0020	0.9303/ 0.0016	0.9300/0.0030	0.9298/ 0.0034	0.9297/ 0.0021
Hill-valley	0.6662/0.0024	0.6692/ 0.0035	0.6681/0.0029	0.6612/0.0039	0.6601/0.0040
LSVT	0.8134/0.0272	0.8197/ 0.0228	0.8163/0.0275	0.8104/ 0.0487	0.8084/ 0.0487

Table 5
The HV values obtained by the proposed algorithm with different σ (Average/Std.).

Datasets	$\sigma=0.001$	$\sigma=0.005$	$\sigma=0.01$	$\sigma=0.02$	$\sigma=0.05$	$\sigma=0.1$
Vowel	0.8260/0.0003	0.8262/0.0004	0.8262/0.0004	0.8260/0.0005	0.8253/0.0006	0.8228/0.0007
Ionosphere	0.9294/0.0028	0.9296/0.0028	0.9300/0.0030	0.9301/0.0032	0.9290/0.0035	0.9275/0.0043
Hill-valley	0.6680/0.0057	0.6689/0.0027	0.6681/0.0029	0.6614/0.0031	0.6607/0.0039	0.6531/0.0068
LSVT	0.8155/0.0241	0.8167/0.0228	0.8163/0.0275	0.8101/0.0291	0.7797/0.0321	0.7049/0.0376

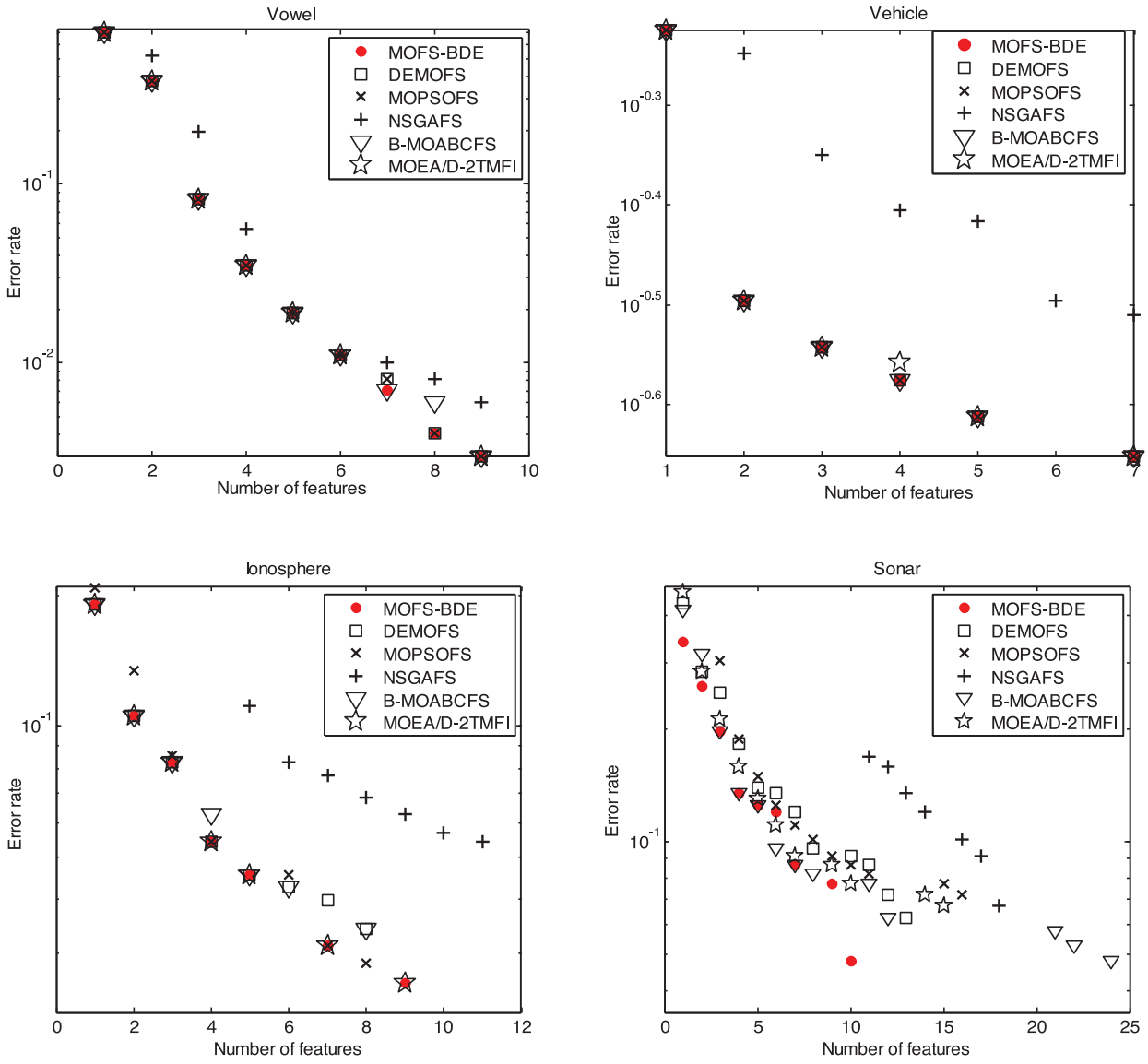

Fig. 5. Pareto optimal sets found by the six algorithms for Vowel, Vehicle, Ionosphere, and Sonar.

Table 5 shows the HV values acquired by MOFS-BDE with different σ values. For the two datasets, Vowel and Ionosphere, this algorithm yields satisfactory HV values, when the value of σ varies from 0.001 to 0.05. However, for the two datasets with more features, Hill-valley and LSVT, when the value of σ is higher than 0.01, its performance significantly deteriorates. Therefore, setting the value of σ within [0.001, 0.01] is a good choice.

6.4. Analysis of the best Pareto set

For alleviating the randomness in the feature selection results, we run each of these algorithms for 30 independent times for every dataset. From the 30 results, the best one (i.e., the solution set with the largest HV value) is selected. Figs. 5 and 6 demonstrate the results obtained by the six algorithms for the eight datasets. In each figure, the horizontal axis means the number of selected features, and the vertical axis represents the classification error rate. As shown in Fig. 5, for the datasets with small features, Vowel and Vehicle, MOFS-BDE, DEMOFS, MOPSOFS, and B-MOABCFS have the same results, which are clearly better than that of NSGAFS. MOEA/D-2TMFI also obtains results slightly worse than MOFS-BDE, DEMOFS, MOPSOFS, and B-MOABCFS. Especially, for the dataset, Vehicle, all the five algorithms have smaller error rates than that of NSGAFS, when more than two features are selected. Thus, the two datasets in fact are not challenging for MOFS-BDE, DEMOFS, MOPSOFS, B-MOABCFS, and MOEA/D-2TMFI.

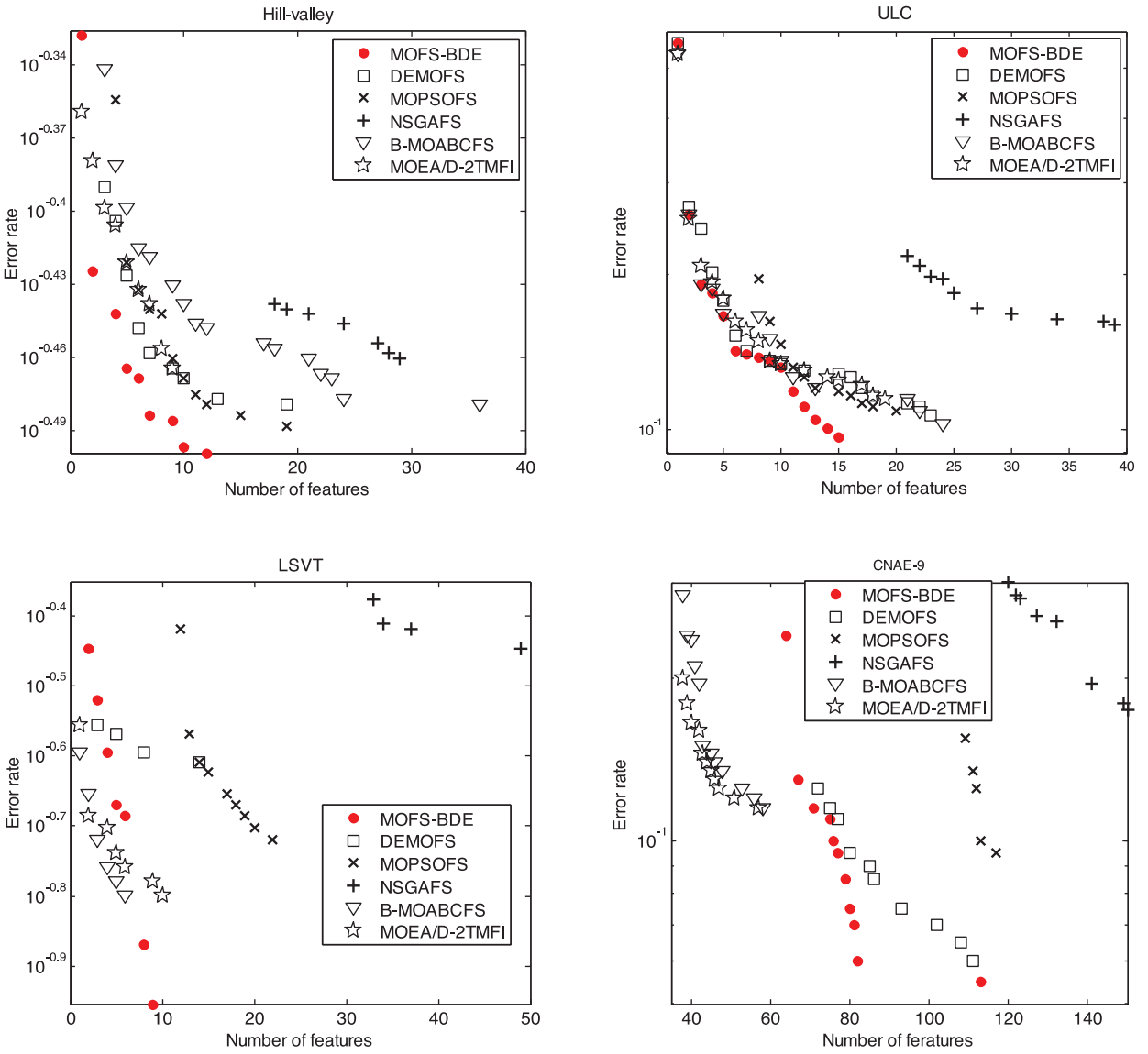

Fig. 6. Pareto optimal sets found by six algorithms for Hill-valley, ULC, LSVT, and CNAE-9.

Regarding the dataset Ionosphere with 34 features, MOFS-BDE and MOEA/D-2TMFI have the same results, which are clearly better than that of NSGAFS and MOPSOFS. For MOFS-BDE, DEMOFS, B-MOABCFS, and MOEA/D-2TMFI, their classification error rates all drop from 20% to 8%, when the size of selected features varies from 1 to 3. However, when the size of selected features is greater than 3, B-MOABCFS and DEMOFS still have 2/4 and 2/5 solutions, respectively, which are dominated by both MOFS-BDE and MOEA/D-2TMFI. MOFS-BDE and MOEA/D-2TMFI yield the smallest error rate, 2.56%, in the case of nine features.

With respect to the dataset with 60 features, Sonar, MOFS-BDE achieves better results than that of DEMOFS, MOPSOFS, and NSGAFS regarding both the error rate and the size of selected features. Although B-MOABCFS and MOEA/D-2TMFI can find a better solution than that of MOFS-BDE, when the size of selected features is equal to six, most of their remaining solutions are dominated by that of MOFS-BDE. When 13, 16, and 18 features are selected, DEMOFS, MOPSOFS, and NSGAFS achieve their smallest error rates, 6.25%, 7.21% and 6.73%, respectively. Among all the six comparison algorithms, MOFS-BDE has the smallest error rate, 4.81%, when 10 features are selected. B-MOABCFS also finds the smallest error rate, 4.81%, but it needs 24 features. Moreover, NSGAFS fails to obtain any solutions, when the number of features is less than 10.

Fig. 6 shows that for the dataset with 100 features, Hill-valley, MOFS-BDE provides better results than that of DEMOFS, MOPSOFS, NSGAFS, B-MOABCFS, and MOEA/D-2TMFI with respect to both the error rate and the number of selected features. When 19 features are selected, DEMOFS yields its smallest error rate, 33.3%, whereas this value is larger than that of

Table 6
The set coverage values between MOFS-BDE and the five comparison algorithms (Average/Std.).

Datasets	Algorithms	SC(MOFS-BDE,*)	SC(*, MOFS-BDE)	Datasets	Algorithms	SC(MOFS-BDE,*)	SC(*, MOFS-BDE)
Vowel	DEMOFS	0.0444/0.0304	0/0	Libras	DEMOFS	0.5432/0.2095	0.3484/0.1664
	MOPSOFS	0/0	0/0	Movement	MOPSOFS	0.7840/0.2346	0.1558/ 0.1812
	NSGAFS	0.8851/0.1117	0/0		NSGAFS	1/0	0/1
	B-MOABCFS	0.0222/0.0248	0/0		B-MOABCFS	0.3654/0.2174	0.2765/0.2013
	MOEA/D-2TMFI	0.0571/0.0782	0/0		MOEA/D-2TMFI	0.4921/0.2199	0.3776/0.2007
Wine	DEMOFS	0/0	0.0520/0.1096	Hill-valley	DEMOFS	0.9492/0.0705	0.0677/0.0395
	MOPSOFS	0.0286/0.0904	0/0		MOPSOFS	0.8492/0.0930	0.1556/0.1117
	NSGAFS	0.9124/0.1545	0/0		NSGAFS	1/0	0/0
	B-MOABCFS	0.0917/0.1682	0/0		B-MOABCFS	0.9429/0.0531	0.0647/0.0609
	MOEA/D-2TMFI	0.0667/0.1405	0/0		MOEA/D-2TMFI	0.9244/0.0735	0.0472/0.0648
Zoo	DEMOFS	0/0	0/0	ULC	DEMOFS	0.5482/0.1365	0.2824/0.0736
	MOPSOFS	0.0567/0.0917	0/0		MOPSOFS	0.9818/0.0406	0/0
	NSGAFS	1/0	0/0		NSGAFS	1/0	0/0
	B-MOABCFS	0.0567/0.0917	0/0		B-MOABCFS	0.6613/0.1415	0.2027/0.0976
	MOEA/D-2TMFI	0.0200/0.0632	0/0		MOEA/D-2TMFI	0.4983/0.1199	0.2484/0.0745
Vehicle	DEMOFS	0.0333/0.0372	0/0	Musk	DEMOFS	0.7235/ 0.2265	0.1918/0.1758
	MOPSOFS	0.0667/0.0745	0/0		MOPSOFS	1/0	0/0
	NSGAFS	0.9333/0.0745	0/0		NSGAFS	1/0	0/0
	B-MOABCFS	0.0533/0.0712	0/0		B-MOABCFS	0.5251/ 0.2861	0.2621/0.1689
	MOEA/D-2TMFI	0/0	0/0		MOEA/D-2TMFI	0.8041/ 0.1777	0.1426/0.1503
WDBC	DEMOFS	0.1932/0.1458	0.1405/0.1378	SCADI	DEMOFS	0.5299/ 0.2728	0.2449/ 0.2150
	MOPSOFS	0.4477/0.2523	0.1847/0.1475		MOPSOFS	0.7950/0.2370	0.0596/ 0.1234
	NSGAFS	1/0	0/0		NSGAFS	1/0	0/0
	B-MOABCFS	0.5006/0.1832	0.1329/0.1452		B-MOABCFS	0.5051/0.2046	0.2294/0.2040
	MOEA/D-2TMFI	0.4619/0.1380	0.0536/0.0694		MOEA/D-2TMFI	0.7253/0.2151	0.1136/0.1583
Ionosphere	DEMOFS	0.3557/0.0834	0.2007/0.0844	LSVT	DEMOFS	0.7200/0.1981	0.1714/0.1833
	MOPSOFS	0.3924/0.1021	0.1536/0.0712		MOPSOFS	1/0	0/0
	NSGAFS	1/0	0/0		NSGAFS	1/0	0/0
	B-MOABCFS	0.5631/0.1370	0.1536/0.0725		B-MOABCFS	0.6333/0.2801	0.2333/0.1528
	MOEA/D-2TMFI	0.4707/0.1576	0.2071/ 0.0865		MOEA/D-2TMFI	0.5167/0.2580	0.3093/0.2249
Satellite	DEMOFS	0.3253/0.2052	0.2511/0.1546	CNAE-9	DEMOFS	0.6349/0.1409	0.1800/0.0957
	MOPSOFS	0.6416/0.1386	0.0999/0.1093		MOPSOFS	1/0	0/0
	N**SG**AFS	1/0	0/1		NSGAFS	1/0	0/0
	B-MOABCFS	0.5182/0.1404	0.1515/0.1164		B-MOABCFS	0/0	0.2448/0.1024
	MOEA/D-2TMFI	0.4705/0.1862	0.1648/0.1479		MOEA/D-2TMFI	0/0	0.3139/0.1723
SPECTF	DEMOFS	0.1155/0.1115	0.0734/0.0896	Yale_64	DEMOFS	1/0	0/0
	MOPSOFS	0.2570/0.1626	0.0588/0.0809		MOPSOFS	1/0	0/0
	NSGAFS	1/0	0/0		NSGAFS	1/0	0/0
	B-MOABCFS	0.1868/0.1359	0.0563/0.0762		B-MOABCFS	0/0	0.8538/0.1397
	MOEA/D-2TMFI	0.3366/0.1722	0.0426/0.0645		MOEA/D-2TMFI	1/0	0/0
Parkinson	DEMOFS	0.4686/0.2205	0.2540/ 0.1972	SRBCT	DEMOFS	0.9375/ 0.1398	0/0
	MOPSOFS	0.6578/0.2247	0.1394/0.0985		MOPSOFS	1/0	0/0
	NSGAFS	1/0	0/1		NSGAFS	1/0	0/0
	B-MOABCFS	0.5083/0.2157	0.1883/0.1376		B-MOABCFS	1/0	0/0
	MOEA/D-2TMFI	0.4602/0.1934	0.2084/0.1741		MOEA/D-2TMFI	1/0	0/0
Sonar	DEMOFS	0.6270/0.0852	0.2795/0.1125	DLBCL	DEMOFS	0.9491/0.1058	0/0
	MOPSOFS	0.8302/0.0726	0.0308/0.0344		MOPSOFS	1/0	0/0
	NSGAFS	1/0	0/0		NSGAFS	1/0	0/0
	B-MOABCFS	0.6171/0.2152	0.2436/0.1032		B-MOABCFS	1/0	0/0
	MOEA/D-2TMFI	0.4879/0.2194	0.2766/0.1115		MOEA/D-2TMFI	1/0	0/0

MOFS-BDE, 31.7%, with only 12 features. In addition, B-MOABCFS achieves its smallest error rate, 33.2%, with 36 features, MOEA/D-2TMFI gets the smallest error rate, 34.3%, with 9 features, and DEMOFS obtains its smallest error rate, 33.3%, with 19 features.

For the dataset with 148 features, ULC, MOFS-BDE provides better results than that of DEMOFS, MOPSOFS, and NSGAFS with respect to both the error rate and the number of selected features. Although B-MOABCFS and MOEA/D-2TMFI can find better solutions than that of MOFS-BDE, when the size of selected features is equal to one or two, most of their remaining solutions are still dominated by that of MOFS-BDE. In particular, MOPSOFS and NSGAFS cannot classify the data within seven features. For the 15 features selected, MOFS-BDE reduces the error rate to 9.66%, which is 0.98% lower than the minimal error rate obtained by DEMOFS with 23 features, 1.16% lower than that of MOPSOFS with 20 features, 6.32% lower than that of NSGAFS with 39 features, 0.60% lower than that of B-MOABCFS with 24 features, and 1.78% lower than that of B-MOEA/D-2TMFI with 19 features.

Regarding the dataset with 309 features, LSVT, MOFS-BDE provides better results than that of MOPSOFS and NSGAFS with respect to both the error rate and the number of selected features. Although DEMOFS finds a better solution than that of MOFS-BDE, when the size of selected features is equal to two, its remaining solutions are all dominated by that of MOFS-

Table 7
The FN values found by the six algorithms for 20 datasets (Average/Std.).

FN	MOFS-BDE	DEMOFS	MOPSOFS	NSGAFS	B-MOABCFS	MOEA/D-2TMFI
Vowel	**9.00/0**	**9.00/0**	8.60/0.55	8.20/0.13	9.00/0	8.00/0
Wine	**5.85/0.36**	5.45/ 0.51	5.80/0.63	5.80/0.78	4.50/1.43	5.20/0.42
Zoo	**5.80/ 0.41**	5.75/0.44	5.60/0.51	5.10/1.56	5.50/0.52	5.40/0.51
Vehicle	**5.80/0.45**	5.40/0.55	5.60/0.55	3.80/0.16	5.20/0.45	5.10/0.32
WDBC	7.60/0.67	7.30/0.48	7.**80/1.14**	5.00/1.70	7.30/1.34	6.70/0.48
Ionosphere	**7.56/0.53**	**7.56/0.73**	6.40/0.89	5.20/1.31	7.45/1.05	6.65/0.87
Satellite	**13.00/1.49**	12.70/1.41	12.10/1.52	8.20/2.30	11.22/0.97	10.22/1.48
SPECTF	10.05/0.68	9.00/0.72	**10.30/0.48**	8.20/1.75	10.00/0.56	10.30/0.67
Parkinson	9.58/1.08	9.40/1.17	**9.90/1.37**	5.80/1.98	8.60/1.17	8.70/1.15
Sonar	**11.20/2.28**	10.78/1.99	11.00/2.73	5.80/1.30	10.70/1.75	10.00/1.83
Libras Movement	**12.10/0.99**	10.75/1.48	10.90/1.79	5.67/2.29	10.15/1.29	9.70/1.05
Hill-valley	8.80/0.45	**9.60/1.67**	8.00/2.64	6.80/2.17	9.75/2.63	8.30/1.820
ULC	**14.20/2.15**	12.20/2.39	11.00/1.20	9.20/2.17	12.05/2.37	10.95/1.633
Musk	**15.0/2.50**	12.80/1.55	9.16/1.17	7.88/3.26	10.83/2.48	13.67/1.86
SCADI	**6.50/0.97**	6.40/1.07	4.10/0.73	3.20/1.81	5.60/1.17	5.90/0.87
LSVT	**6.15/1.22**	4.60/0.89	5.15/3.17	2.80/0.84	4.15/1.14	4.25/1.80
CNAE-9	**9.80/2.17**	7.65/1.95	5.20/0.84	7.40/2.70	9.70/3.11	8.15/1.84
Yale_64	7.33/1.52	5.37/1.59	3.14/1.46	5.33/1.52	**15.50/2.64**	4.25/1.50
SRBCT	**4.12/1.07**	3.20/0.91	2.57/0.78	3.70/1.88	1.58/0.72	1.86/1.05
DLBCL	2.07/0.57	2.67/0.57	1.16/0.41	2.69/0.53	**4.16/0.75**	1.42/0.55

BDE. B-MOABCFS and MOEA/D-2TMFI can identify better feature subsets than that of MOFS-BDE, when the size of selected features is less than 7. However, they cannot achieve an error rate lower than 15.87%, when more features are selected. When the number of features is nine, MOFS-BDE is capable of achieving the lowest error rate, 11.11%, which is 4.76% lower than that obtained by MOABCFS and MOEA/D-2TMFI. Note that most solutions acquired by NSGAFS concentrate on the top left corner of the figure.

Regarding the dataset with 857 features, CNAE-9, MOFS-BDE finds a good Pareto front, which dominates all the solutions obtained by DEMOFS, MOPSOFS, and NSGAFS. When the size of selected features is less than 60, both B-MOABCFS and MOEA/D-2TMFI can locate the non-dominated solutions (feature subsets), and achieve the lowest error rate, 11.50%, when a total of 58 features are selected. Although MOFS-BDE identifies the first non-dominated solution until 64 features are selected, more features can help it reduce the error rate to 5.50% (when 113 features are selected), which is 6.0% lower than the minimal error rates obtained by B-MOABCFS and MOEA/D-2TMFI with 58 features, 4.0% lower than that obtained by MOPSOFS with 117 features, and 12% lower than that obtained by NSGAFS with 150 features. DEMOFS can achieve its lowest error rate, 6.0%, when 111 features are selected, whereas MOFS-BDE has the same error rate with only 82 features.

Generally, the experimental results in Figs. 5 and 6 suggest that MOFS-BDE, DEMOFS, MOPSOFS, B-MOABCFS, and MOEA/D-2TMFI are competitive with each other, when dealing with the datasets with less than 40 features, i.e., Vowel, Vehicle, and Ionosphere. However, for the datasets, Sonar, Hill-valley, and ULC, MOFS-BDE can achieve moderately better performances than the other five algorithms with respect to both the error rate and the number of selected features. For the datasets with more features, LSVT and CNAE-9, MOFS-BDE shows a good capability of reducing the classification error rate, and the smallest error rates obtained by MOFS-BDE are apparently lower than that obtained by both B-MOABCFS and MOEA/D-2TMFI.

6.5. Analysis of the average performances

We here compare the average performances of MOFS-BDE, DEMOFS, MOPSOFS, NSGAFS, B-MOABCFS, and MOEA/D-2TMFI by analyzing their experimental results over 30 independent runs. Tables 6 and 7 give the average performances of these six algorithms with respect to the SC and FN, respectively.

For the datasets with less than 20 features, Vowel, Wine, Zoo, and Vehicle, the optimal solution sets obtained by MOFS-BDE, DEMOFS, MOPSOFS, B-MOABCFS, and MOEA/D-2TMFI show similar convergence characteristics. Take the dataset Wine as an example, in the worst case, only 5.20% of the solutions of MOFS-BDE are dominated by that of DEMOFS, and 9.17% of the solutions of B-MOABCFS are dominated by that of MOFS-BDE. The optimal solution set acquired by NSGAFS shows the worst convergence. More than 88.5% of the solutions of NSGAFS are dominated by that of MOFS-BDE for all the four datasets. Moreover, MOFS-BDE yields a better performance than that of MOPSOFS, NSGAFS, and MOEA/D-2TMFI regarding the diversity of solutions, where it produces a larger number of optimal solutions than that of MOPSOFS, NSGAFS, and MOEA/D-2TMFI, as given in Table 7.

For all the 12 datasets with 30–500 features, MOFS-BDE yields a better performance than that of DEMOFS, MOPSOFS, NSGAFS, B-MOABCFS, and MOEA/D-2TMFI with respect to the convergence. The proportions that MOFS-BDE dominates the five comparison algorithms are obviously higher than those dominated by the latter. Take the dataset LSVT as example, 72.0% of the solutions of DEMOFS, 100% of the solutions of MOPSOFS and NSGAFS, 63.33% of the solutions of B-MOABCFS, and 51.67% of the solutions of MOEA/D-2TMFI are dominated by MOFS-BDE. Only 30.93% of the solutions of MOFS-BDE are

Table 8
The HV values found by the six algorithms for 20 datasets.

HV	MOFS-BDE	DEMOFS		MOPSOFS		NSGAFS		B-MOABCFS		MOEA/D-2TMFI	
	Average/Std.	Average/Std.	*t*-test	Average/Std.	*t*-test	Average/Std.	*t*-test	Average/Std.	*t*-test	Average/Std.	*t*-test
Vowel	0.8262/0.0004	0.8260/0.0007	N	**0.8267/0.0006**	N	0.8099/0.0372	N	0.8259/0.0004	N	0.8252/0.0005	N
Wine	0.8835/0.0022	0.8849/0.0019	N	**0.8845/0.0021**	N	0.8060/0.0484	Y+	0.8841/0.0076	N	0.8830/0.0020	N
Zoo	**0.8870/0.0021**	0.8868/0.0022	N	0.8856/0.0026	N	0.7822/0.0420	Y+	0.8852/0.0029	N	0.8850/0.0026	N
Vehicle	**0.6964/0.0058**	0.6894/0.0062	N	0.6926/0.0063	N	0.5961/0.0367	Y+	0.6846/0.0067	N	0.6871/0.0042	N
WDBC	**0.9433/0.0008**	0.9425/0.0006	N	0.9415/0.0016	Y+	0.7537/0.0610	Y+	0.9397/0.0030	Y+	0.9382/0.0027	Y+
Ionosphere	**0.9300/0.0030**	0.9296/0.0034	N	0.9270/0.0037	N	0.8374/0.0981	Y+	0.9218/0.0034	N	0.9238/0.0051	N
Satellite	**0.8704/0.0022**	0.8696/0.0027	N	0.8645/0.0035	Y+	0.6885/0.0381	Y+	0.8653/0.0041	Y+	0.8618/0.005	Y+
SPECTF	**0.6887/0.0009**	0.6642/0.0194	Y+	0.6875/0.0026	N	0.5858/0.0306	Y+	0.6878/0.0021	N	0.6868/0.0023	Y+
Parkinson	**0.8928/0.002**2	0.8865/0.0083	Y+	0.8829/0.0064	Y+	0.7149/0.0354	Y+	0.8846/0.0047	Y+	0.8836/0.0077	Y+
Sonar	**0.9175/0.0043**	0.9038/0.0055	Y+	0.8851/0.0161	Y+	0.6693/0.0218	Y+	0.9049/0.0045	Y+	0.9031/0.1010	Y+
Libras Movement	**0.8709/0.0033**	0.8655/0.0032	N	0.8547/0.0123	Y+	0.6123/0.0267	Y+	0.8612/0.0050	Y+	0.8596/0.0058	Y+
Hill-valley	**0.6681/0.0029**	0.6517/0.0035	Y+	0.6501/0.0062	Y+	0.4407/0.0159	Y+	0.6384/0.0184	Y+	0.6392/0.0048	Y+
ULC	**0.8873/0.0040**	0.8788/0.0058	Y+	0.8306/0.0101	Y+	0.5771/0.0254	Y+	0.8679/0.0088	Y+	0.8701/0.0034	Y+
Musk	**0.9531/0.0033**	0.9407/0.0045	Y+	0.8799/0.0201	Y+	0.6019/0.0155	Y+	0.9373/0.0081	Y+	0.9401/0.0038	Y+
SCADI	**0.9117/0.0094**	0.9097/0.0088	N	0.8811/0.0126	Y+	0.5559/0.0133	Y+	0.8881/0.0174	Y+	0.8961/0.0128	Y+
LSVT	**0.8663/0.0275**	0.7510/0.0414	Y+	0.7421/0.0042	Y+	0.3967/0.0064	Y+	0.8159/0.0809	Y+	0.7981/0.0937	Y+
CNAE-9	0.8381/0.0076	0.8113/0.0106	Y+	0.7461/0.0081	Y+	0.4639/0.0186	Y+	0.8361/0.0119	N	**0.8405/0.0159**	N
Yale_64	0.7489/0.0035	0.6719/0.0093	Y+	0.6238/0.0055	Y+	0.4221/0.0111	Y+	**0.7535/0.0038**	Y-	0.7133/0.0127	Y+
SRBCT	**0.8250/0.0095**	0.7480/0.0109	Y+	0.7148/0.0149	Y+	0.4799/0.0101	Y+	0.7715/0.0130	Y+	0.7269/0.0089	Y+
DLBCL	**0.6970/0.0154**	0.6556/0.0082	Y+	0.6250/0.0063	Y+	0.4667/0.0137	Y+	0.5944/0.0225	Y+	0.5080/0.0079	Y+

dominated by MOEA/D-2TMFI in the worst case. Moreover, for eight out of the 12 datasets, i.e., Ionosphere, Satellite, Sonar, Libras Movement, ULC, Musk, SCADI, and LSVT, MOFS-BDE also shows better performances than that of DEMOFS, MOPSOFS, NSGAFS, B-MOABCFS, and MOEA/D-2TMFI regarding the diversity of solutions, as shown in Table 7.

For the dataset with 857 features, CNAE-9, MOFS-BDE cannot find a solution dominating the solutions obtained by B-MOABCFS and MOEA/D-2TMFI. Their SC values show, on the contrary, that 24.48% and 31.39% of the solutions of MOFS-BDE are dominated by that of B-MOABCFS and MOEA/D-2TMFI, respectively. The excellent SC performances of B-MOABCFS and MOEA/D-2TMFI are mainly due to their capabilities of rapidly reducing features. As demonstrated in Fig. 6, the number of features in each solution of B-MOABCFS and MOEA/D-2TMFI is obviously less than that of MOFS-BDE. However, reducing the features too fast may sacrifice the classification accuracy of B-MOABCFS and MOEA/D-2TMFI. We can observe from Fig. 6 that the minimal error rate obtained by MOFS-BDE is 6.0% lower than that of B-MOABCFS and MOEA/D-2TMFI. In addition, MOFS-BDE has a larger FN value, 9.80, while the FN values of B-MOABCFS and MOEA/D-2TMFI are 9.70 and 8.15, respectively.

For the dataset with 1024 features, Yale_64, MOFS-BDE has the second best performance with respect to both the convergence and diversity of solutions, but its value is better than that of DEMOFS, MOPSOFS, NSGAFS, and MOEA/D-2TMFI. B-MOABCFS is well capable of achieving the best convergence as well as the diversity of solutions, because of the remarkable mutation performance of B-MOABCFS in the scout bee phase.

For the dataset with more than 2308 features, SRBCT, B-MOABCFS shows the best convergence, which is fairly better than that of the five comparison algorithms. All the solutions obtained by NSGAFS, MOPSOFS, B-MOABCFS, and MOEA/D-2TMFI are dominated by that of MOFS-BDE, and 93.75% of the solutions of DEMOFS are dominated by MOFS-BDE. In addition, MOFS-BDE yields the best performance with respect to the diversity of solutions, as given in Table 7.

For the dataset with more than 5000 features, DLBCL, although B-MOABCFS obtains more solutions than MOFS-BDE, their SC values show that those solutions' convergence is fairly better than that of DEMOFS. Note that all the solutions of DEMOFS are dominated by that of MOFS-BDE.

Moreover, the *t*-test is used to investigate the statistical robustness of MOFS-BDE. It is a two-sample location test for testing the hypothesis that two populations have the equal means. We set the significance level to be 0.05, and the HV metric is employed here. Table 8 gives the results of the six algorithms with respect to HV. We emphasize that '*Y*+' indicates that MOFS-BDE is significantly better than the selected one by the two-tailed test at the 0.05 level, 'Y-' indicates that MOFS-BDE is significantly worse than the selected one by the two-tailed test at the 0.05 level, and 'N' indicates that the difference between these two algorithms is not significant at the 0.05 level. We can find out that, for the four datasets with less than 20 features, i.e., Vowel, Wine, Zoo, and Vehicle, MOFS-BDE can acquire similar results to that of DEMOFS, MOPSOFS, B-MOABCFS, and MOEA/D-2TMFI. However, for eight out of the remaining 16 datasets, i.e., Parkinson, Sonar, Hill-valley, ULC, Musk, LSVT, SRBCT, and DLBCL, the HV performance of MOFS-BDE is considerably better than that of DEMOFS, MOPSOFS, NSGAFS, B-MOABCFS, and MOEA/D-2TMFI at the 0.05 level. For four out of the 16 datasets, i.e., WDBC, Satellite, Libras Movement, and SCADI, MOFS-BDE can get the HV values comparable with that of DEMOFS, and its HV values are significantly better than that of MOPSOFS, NSGAFS, B-MOABCFS, and MOEA/D-2TMFI at the 0.05 level. For two out of the 16 datasets, i.e., Ionosphere and SPECTF, MOFS-BDE also obtains competitive HV values compared with the other algorithms. There is only

Table 9
Running time consumed by six algorithms for 20 datasets (unit: minute).

Algorithms	MOFS-BDE	DEMOFS	MOPSOFS	NSGAFS	B-MOABCFS	MOEA/D-2TMFI
Vowel	7.91/0.27	8.34/0.35	7.32/0.41	8.29/0.57	7.65/0.39	**6.83/0.26**
Wine	0.35/0.007	**0.19/ 0.003**	0.25/0.007	0.28/0.016	0.36/0.009	0.32/0.003
Zoo	0.17/0.002	0.12/0.002	**0.08/0.004**	0.13/0.007	0.19/0.010	0.23/0.005
Vehicle	0.16/0.012	0.15/0.010	**0.07/0.009**	0.10/0.009	0.09/0.014	0.18/0.002
WDBC	3.14/0.12	3.29/0.08	2.99/0.14	4.27/0.42	5.23/0.49	**2.33/0.07**
Ionosphere	1.13/0.04	1.29/0.05	**1.02/0.04**	1.51/0.07	1.12/0.10	1.08/0.05
Satellite	9.72/0.33	9.31/0.22	8.18/0.63	11.70/0.36	14.42/0.89	**7.04/0.25**
SPECTF	0.38/0.006	0.31/ 0.005	**0.25/0.006**	0.33/0.009	0.51/0.012	0.38/0.004
Parkinson	0.84/0.01	0.77/0.04	**0.67/0.04**	1.08/0.12	1.16/0.06	0.71/0.02
Sonar	0.69/0.05	0.77/0.06	**0.65/0.07**	1.03/0.04	0.67/0.09	0.67/0.03
Libras Movement	3.00/ 0.10	2.87/0.15	2.47/0.27	4.62/0.16	3.87/0.45	**1.92/0.08**
Hill-valley	8.82/0.83	9.48/0.78	6.69/1.10	12.06/0.48	9.25/1.45	**4.86/0.39**
ULC	17.18/2.11	17.44/1.95	18.17/2.42	31.15/3.25	16.80/2.87	**14.60/1.02**
Musk	18.01/1.71	17.38/0.90	20.07/2.95	37.11/4.29	25.72/2.88	**14.14/0.47**
SCADI	0. 79/0.02	0.63/0.02	**0.42/0.05**	1.41/0.07	0.65/0.02	0.76/0.03
LSVT	2.10/0.11	2.63/0.12	1.95/0.17	4.00/0.13	1.82/0.09	**1.73/0.06**
CNAE-9	20.30/0.46	20.82/0.71	18.93/1.08	33.59/1.17	19.21/1.89	**15.34/0.98**
Yale_64	16.54/0.36	23.67/0.63	20.34/0.97	36.41/0.96	34.17/1.51	**13.99/1.33**
SRBCT	16.58/0.32	17.37/0.27	**13.91/0.50**	22.10/0.39	30.18/ 0.53	20.61/0.31
DLBCL	29.45/2.69	21.30/2.65	**14.58/0.69**	17.80/0.29	27.97/0.45	17.45/0.85

one among the 20 datasets, i.e., Yale_64, for which MOFS-BDE has a much worse HV value than that of B-MOABCFS at the 0.05 level.

A non-parametric statistical test, Friedman test, is utilized to verify whether different algorithms have similar classification accuracies for all the test problems. Based on the results in Table 8, we can obtain the p-value of the Friedman test $p = 1.6099\text{e-}13$. A small value of p indicates that all the algorithms have the same HV performance rejected at 0.05 significance level. Considering the HV values in Table 8 and two statistical test values, we can conclude that MOFS-BDE is a competitive alternate for the feature selection problems.

6.6. Running time

The running time of MOFS-BDE, DEMOFS, MOPSOFS, NSGAFS, B-MOABCFS, and MOEA/D-2TMFI is also compared in our simulations. All the six algorithms have the same evaluation time as in Section 6.1. Table 9 shows their average running time.

It is clearly visible that our MOFS-BDE consumes a longer running time than that of MOPSOFS or MOEA/D-2TMFI for most datasets. There are two main reasons: (1) In order to increase the classification accuracy, MOFS-BDE selects more features than that of MOPSOFS and MOEA/D-2TMFI. As we know that the larger the number of selected features, the larger the size of the dataset used to train classifiers and the longer the running time the algorithm spends on individual evaluation. In other words, MOPSOFS and MOEA/D-2TMFI have a shorter running time by sacrificing the classification accuracy; (2) MOFS-BDE uses the non-dominated sorting strategy with a higher computational complexity, i.e., FNS, which, to some extent, can prolong its running time. Given the fact that ENS has a lower time complexity than that of FNS, MOFS-BDE consumes a shorter time than that of DEMOFS for simple datasets. Although B-MOABCFS also uses the non-dominated sorting strategy, it consumes a shorter time than that of MOFS-BDE and DEMOFS for some datasets, due to the less features selected.

In general, with more features to improve the classification accuracy, MOFS-BDE consumes a relatively long running time for most datasets compared with MOPSOFS and MOEA/D-2TMFI. However, the growth in the running time is acceptable for the improvement of the classification performance. In summary, the above experimental results have shown that the proposed algorithm is indeed a competitive solution to feature selection problems, especially with high-dimension data.

7. Conclusions

In this paper, we propose a new multi-objective differential evolution algorithm, namely MOFS-BDE, for the feature selection problems. Several new operators including the probability difference-based binary mutation, the one-bit purifying search, and the one-bit purifying search, have been developed and embedded into MOFS-BDE. By comparing MOFS-BDE with four popular feature selection approaches (DEMOFS, NSGAFS, MOPSOFS, and B-MOABCFS) and a new MOEA/D method (MOEA/D-2TMFI), we demonstrate that our MOFS-BDE is efficient in dealing with feature selection problems. The main findings of the paper can be summarized as follows.

(1) On the basis of the optimal learning strategy, the proposed probability difference-based binary mutation can effectively improve the convergence of our algorithm. For most of the test datasets, MOFS-BDE can obtain satisfactory solutions dominating those solutions acquired from the five comparison algorithms.

(2) The novel one-bit purifying search makes full use of the characteristics of feature selection problems. With the self-learning ability, elite individuals among the population can effectively guide the search of other individuals.
(3) The one-bit purifying search is designed as a small-scale refinement process, while the new mutation and crossover in the DE can perform a large-scale search in the parameter space. By combining these operators together, our algorithm is well capable of achieving an appropriate trade-off between exploitation and exploration.

Compared with conventional approaches, such as the mRMR and sequential forward selection, evolutionary feature selection methods usually have high computational complexities, because they need to repeatedly evaluate the individuals. As the number of samples/features grows, the running time may also go up. How to reduce their computational complexities by using the approximate evaluation and sample reduction strategies is one of our future research topics. Moreover, we are going to apply meta-heuristic approaches for more complex data mining problems, such as multi-objective multi-label and stream feature selection.

Declaration of Competing Interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgements

This work was jointly supported by the National Natural Science Foundation of China (No. 61876185, 51875113, and 61573361), and Six Talent Peaks Project of Jiangsu Province (No. DZXX-053).

References

[1] A. Al-Ani, A. Alsukker, R. Khushaba, Feature subset selection using differential evolution and a wheel based search strategy, Swarm Evolut. Comput. 9 (2013) 15–26.
[2] M.Z. Baig, N. Aslam, H.P.H. Shum, L. Zhang, Differential evolution algorithm as a tool for optimal feature subset selection in motor imagery EEG, Expert Syst. Appl. 90 (2017) 184–195.
[3] F. Barani, M. Mirhosseini, H. Nezamabadi-pour, Application of binary quantum-inspired gravitational search algorithm in feature subset selection, Appl. Intell. 47 (2) (2017) 304–318.
[4] K. Chen, F.Y. Zhou, X.F. Yuan, Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection, Expert Syst. Appl. 128 (2019) 140–156.
[5] F. Cheraghchi, I. Abualhaol, R. Falcon, R. Abielmona, B. Raahemi, E. Petriu, Modeling the speed-based vessel schedule recovery problem using evolutionary multiobjective optimization, Inf. Sci. 448 (2018) 53–74.
[6] L.Y. Chuang, C.H. Yang, K.C. Wu, C.H. Yang, A hybrid feature selection method for DNA microarray data, Comput. Biol. Med. 41 (2011) 228–237.
[7] A.K. Das, S. Das, A. Ghosh, Ensemble feature selection using bi-objective genetic algorithm, Knowl. Based Syst. 123 (2017) 116–127.
[8] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2011) 4–31.
[9] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[10] D.Y. Eroglu, K. Kilic, A novel hybrid genetic local search algorithm for feature selection and weighting with an application in strategic decision making in innovation management, Inf. Sci. 405 (2017) 18–32.
[11] Z. Ezgi, O. Selma Ayse, A hybrid approach of differential evolution and artificial bee colony for feature selection, Expert Syst. Appl. 62 (2016) 91–103.
[12] H. Faris, M. Mafarja, A.A. Heidari, I. Aljarah, A.M. Al-Zoubi, S. Mirjalili, H. Fujita, An efficient binary Salp swarm algorithm with crossover scheme for feature selection problems, Knowl. Based Syst. 154 (2018) 43–67.
[13] I.A. Gheyas, L.S. Smith, Feature subset selection in large dimensionality domains, Pattern Recognit. 43 (1) (2010) 5–13.
[14] A.K. Ghosh, On optimum choice of k in nearest neighbor classification, Comput. Stat. Data Anal. 50 (2006) 3113–3123.
[15] T.M. Hamdani, J.M. Won, A.M. Alimi, F. Karray, Multi-objective feature selection with NSGA-II, in: Proceeding of the Eighth ICANNGA Part I, Springer, 2007, pp. 240–247.
[16] E. Hancer, B. Xue, M.J. Zhang, Differential evolution for filter feature selection based on information theory and feature ranking, Knowl. Based Syst. 140 (2018) 103–119.
[17] E. Hancer, B. Xue, M.J. Zhang, D. Karaboga, B. Akay, Pareto front feature selection based on artificial bee colony optimization, Inf. Sci. 422 (2018) 462–479.
[18] F. Jimenez, G. Sanchez, J.M. Garcia, G. Sciavicco, L. Miralles, Multi-objective evolutionary feature selection for online sales forecasting, Neurocomputing 234 (2017) 75–92.
[19] R.N. Khushaba, A. Al-Ani, A. Al-Jumaily, Feature subset selection using differential evolution and a statistical repair mechanism, Expert Syst. Appl. 38 (9) (2011) 11515–11526.
[20] J.D. Knowles, D.W. Corne, On metrics for comparing non-dominated sets, in: Proceedings of the Congress on Evolutionary Computation Conference, 2002, pp. 711–716.
[21] S.U. Kumar, E. Asif, S. Sriparna, U. Olga, P. Massimo, Differential evolution-based feature selection technique for anaphora resolution, Soft Comput. 19 (8) (2015) 2149–2161.
[22] Q.Z. Lin, Z.W. Liu, Q. Yan, et al., Adaptive composite operator selection and parameter control for multiobjective evolutionary algorithm, Inf. Sci. 339 (2016) 332–352.
[23] X.L. Ma, Q.F. Zhang, G.D. Tian, J.S. Yang, Z.X. Zhu, On tchebycheff decomposition approaches for multiobjective evolutionary optimization, IEEE Trans. Evol. Comput. 22 (2) (2018) 226–244.
[24] M. Mafarja, I. Aljarah, A.A. Heidari, A.I. Hammouri, H. Faris, A.M. Al-Zoubi, S. Mirjalili, Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems, Knowl. Based Syst. 145 (2018) 25–45.
[25] U. Mlakar, I. Fister, J. Brest, B. Potocnik, Multi-objective differential evolution for feature selection in facial expression recognition systems, Expert Syst. Appl. 89 (2017) 129–137.
[26] P.M. Murphy, D.W. Aha, UCI Repository of Machine Learning Databases, Technical report, Department of Information and Computer Science, University of California, CA, DOI: <http://www.ics.uci.edu/~mlearn/MLRepository.html>
[27] G.P. Nicolás, H.G. Aida, P.R. Javier, A scalable memetic algorithm for simultaneous instance and feature selection, Evol. Comput. 22 (1) (2014) 1–45.
[28] L.S. Oliveira, M. Morita, R. Sabourin, Feature selection for ensembles using the multi-objective optimization approach, in: Multi-Objective Machine Learning, Springer, Berlin Heidelberg, 2006, pp. 49–74.

[29] A.Q. Pan, L. Wang, W.A. Guo, Q.D. Wu, A diversity enhanced multiobjective particle swarm optimization, Inf. Sci. 436 (2018) 441–465.
[30] H.Y. Peng, Y. Fan, Feature selection by optimizing a lower bound of conditional mutual information, Inf. Sci. 418 (2017) 652–667.
[31] M.S. Raza, U. Qamar, An incremental dependency calculation technique for feature selection using rough sets, Inf. Sci. 343 (2016) 41–65.
[32] U.K. Sikdar, A. Ekbal, S. Saha, MODE: multiobjective differential evolution for feature selection and classifier ensemble, Soft Comput. 19 (2015) 3529–3549.
[33] T. Sina, M. Parham, Relevance-redundancy feature selection based on ant colony optimization, Pattern Recognit. 48 (9) (2015) 2798–2811.
[34] M.K. Sohrabi, A. Tajik, Multi-objective feature selection for warfarin dose prediction, Comput. Biol. Chem. 69 (2017) 126–133.
[35] M. Taradeh, M. Mafarja, A.A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, H. Fujita, An evolutionary gravitational search-based feature selection, Inf. Sci. 497 (2019) 219–239.
[36] G.H. Wu, X. Shen, H.F. Li, H.K. Chen, A.P. Lin, P.N. Suganthan, Ensemble of differential evolution variants, Inf. Sci. 423 (2018) 172–186.
[37] B. Xue, W.L. Fu, M.J. Zhang, Differential evolution (DE) for multi-objective feature selection in classification, in: Proceedings of the Sixteenth Genetic and Evolutionary Computation Conference, Springer, 2014, pp. 83–84.
[38] B. Xue, M.J. Zhang, W.N. Browne, Particle swarm optimization for feature selection in classification: a multi-objective approach, IEEE Trans. Cybern. 43 (6) (2013) 1656–1671.
[39] B. Xue, M.J. Zhang, W.N. Browne, X. Yao, A survey on evolutionary computation approaches to feature selection, IEEE Trans. Evol. Comput. 20 (4) (2016) 606–626.
[40] Q.F. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.
[41] X. Zhang, Y. Tian, R. Cheng, Y.C. Jin, An efficient approach to non-dominated sorting for evolutionary multi-objective optimization, IEEE Trans. Evol. Comput. 19 (2) (2014) 201–213.
[42] Y. Zhang, D.W. Gong, J. Cheng, Multi-objective particle swarm optimization approach for cost-based feature selection in classification, IEEE/ACM Trans. Comput. Biol. Bioinf. 22 (99) (2017) 64–75.
[43] Y. Zhang, D.W. Gong, Y. Hu, Feature selection algorithm based on bare bones particle swarm optimization, Neurocomputing 148 (2014) 150–157.
[44] Y. Zhang, D.W. Gong, M. Rong, Multi-objective differential evolution algorithm for multi-label feature selection in classification, in: Y. Tan, Y. Shi, F. Buarque, A. Gelbukh, S. Das, A. Engelbrecht (Eds.), Advances in Swarm and Computational Intelligence. ICSI 2015, LNCS 9140, 2015, pp. 339–345.
[45] Y. Zhang, D.W. Gong, Y.H. Shi, X.C. Zhao, Cost-sensitive feature selection using two-archive multi-objective artificial bee colony algorithm, Expert Syst. Appl. (2019), doi:10.1016/j.eswa.2019.06.044.
[46] Y. Zhang, X.F. Song, D.W. Gong, A return-cost-based binary firefly algorithm for feature selection, Inf. Sci. 418 (2017) 561–574.
[47] Z. Zhu, S. Jia, Z. Ji, Towards a memetic feature selection paradigm, IEEE Comput. Intell. Mag. 5 (2) (2010) 41–53.
[48] Z. Zhu, Y. Ong, M. Dash, Wrapper-filter feature selection algorithm using a memetic framework, IEEE Trans. Syst. Man Cybern. Part B-Cybern. 37 (1) (2007) 70–76.
[49] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE Trans. Evol. Comput. 3 (4) (1999) 257–271.