Knowledge-Based Systems 227 (2021) 107218

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Adaptive crossover operator based multi-objective binary genetic algorithm for feature selection in classification

Yu Xue [a,*], Haokai Zhu [a], Jiayu Liang [b], Adam Słowik [c]

[a] School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China
[b] Tianjin Key Laboratory of Autonomous Intelligent Technology and System, Tiangong University, Tianjin 300387, China
[c] Department of Electronics and Computer Science, Koszalin University of Technology, Koszalin 75-453, Poland

ARTICLE INFO

Article history:
Received 6 December 2020
Received in revised form 12 May 2021
Accepted 9 June 2021
Available online 11 June 2021

Keywords:
Feature selection
Binary genetic algorithm
Multi-objective optimization
Adaptive operator selection
Classification

ABSTRACT

Feature selection is a key pre-processing technique for classification which aims at removing irrelevant or redundant features from a given dataset. Generally speaking, feature selection can be considered as a multi-objective optimization problem, i.e, removing number of features and improving the classification accuracy. Genetic algorithms (GAs) have been widely used for feature selection problems. The crossover operator, as an important technique to search for new solutions in GAs, has a strong impact on the final optimization results. However, many crossover operators are problem-dependent and have different search abilities. Thus, it is a challenge to select the most efficient one to solve different feature selection problems, especially when the nature of feature selection problems is unknown in advance. In order to overcome this challenge, in this paper, a multi-objective binary genetic algorithm integrating an adaptive operator selection mechanism (MOBGA-AOS) is proposed. In MOBGA-AOS, five crossover operators with different search characteristics are used. Each of them is assigned a probability based on the performance in the evolution process. In different phases of evolution, the proper crossover operator is selected by roulette wheel selection according to the probabilities to produce new solutions for the next generation. The proposed algorithm is compared with five well-known evolutionary multi-objective algorithms on ten datasets. The experimental results reveal that MOBGA-AOS is capable of removing a large amount of features while ensuring a small classification error. Moreover, it obtains prominent advantages on large-scale datasets, which demonstrates that MOBGA-AOS is competent to solve high-dimensional feature selection problems.

1. Introduction

Classification is an important issue in machine learning and data mining. In classification problems, it is a hard task to collect and make a dataset with most relevant features. In fact, many candidate features are collected. Unfortunately, it has been observed that many of collected features are irrelevant or redundant to classifiers [1]. Therefore, it is necessary to select the most relevant features to obtain a more compact dataset. One of the most popular used techniques to solve this problem is feature selection. By eliminating irrelevant or redundant features, it not only ensures higher classification accuracy, reduces over-fitting, and improves the generalization ability of the classifiers, but also enables the classifiers to obtain better interpretability [2].

Feature selection can be regarded as a combination optimization problem, but there may be interactions among features, which makes feature selection differ from standard combination optimization problem, i.e., an individually relevant (redundant) feature could be redundant (relevant) when co-working with the others [3]. As we all know, when the scale of a problem is particularly large, it consumes a large amount of computing resources to gain the optimal solution by cause of the huge and complex search space. Suppose the number of features in a structured dataset is n. For all the features to be combined, there is a total of 2^n different ways to combine them, resulting in "curse of dimensionality" for feature selection approaches [4]. As a result, most of the existing algorithms end up finding the sub-optimal solutions.

Recently, evolutionary computation (EC) techniques have attracted a lot of attention for their powerful search ability in the whole search space. Hence, they have been widely applied to many real-world applications, such as unmanned aerial vehicle

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author.
E-mail addresses: xueyu@nuist.edu.cn (Y. Xue), zhuhaokai@nuist.edu.cn (H. Zhu), liangjiayu@tiangong.edu.cn (J. Liang), adam.slowik@tu.koszalin.pl (A. Słowik).

https://doi.org/10.1016/j.knosys.2021.107218
0950-7051/

(UAV) path planning [5], flow shop scheduling problems [6], wireless sensor networks [7]. With the population-based and stochastic search manner, they are suitable to solve feature selection problems, which are difficult to tackle for the aforementioned reasons. Genetic algorithms (GAs) [8] and particle swarm optimization (PSO) [9] are two of the most popular EC techniques. In EC techniques, a solution of feature selection problems is represented as an individual. A threshold is often set to convert continuous encoding of an individual to binary encoding in PSO-based feature selection approaches [10]. However, the threshold is hard to determine and is usually set based on experience. Not only that, but since the process of searching feature subsets is first done in a continuous space, it can be a burden to search for satisfying solutions because of the huge search space. GAs, by virtue of their binary coding scheme, can be easily applied to feature selection problems without tuning the value of the threshold so that the search space is much smaller than the PSO-based ones. Therefore, in this paper, GA is used to solve the feature selection problems.

As one of the most important parts of GAs, genetic operators, i.e., crossover and mutation, play a role in exploring and exploiting the search space to search for new solutions. The crossover operators are able to search for new solutions using information already available in the population while the mutation operators are responsible for producing new information by modifying some of it. Generally, in GAs, the use of crossover operators to find new solutions is much more frequent than the use of mutation operators to do so. Although the mutation operators help to escape from the local optima during the search process, the number of times it acts in the evolutionary process is limited. That is to say, crossover operators are the major force in the search for new solutions. Moreover, different crossover operators have different search properties. According to No-Free Lunch (NFL) theorem [11], one crossover operator may achieve stunning results on some certain problems while failing to achieve good performance on the others. Consequently, determining the proper one to a specific dataset or even to different phases of the evolution process is quite difficult and requires experience.

Generally, some features are time-consuming to obtain and some need to be obtained with special equipment in real-world applications. Meanwhile, many features are irrelevant or redundant, which may deteriorate the classification accuracy. As a result, reducing the number of features should be considered as an objective besides improving the classification accuracy. In other words, feature selection is treated as a multi-objective problem in this paper where the classification accuracy and the number of features should be optimized at the same time. However, a large amount of GA-based feature selection approaches are mainly designed for only optimizing the classification accuracy, and there exists just a few studies concerning multi-objective feature selection based on GAs in the literature [3,12].

This paper proposes a multi-objective binary genetic algorithm integrating an adaptive operator selection mechanism (MOBGA-AOS) for feature selection. The proposed MOBGA-AOS is an improvement of NSGA-II. In MOBGA-AOS, five different crossover operators are adopted, and it is able to perform crossover operation using the currently preferred crossover operator based on their previous performance, followed by the uniform mutation operation, while NSGA-II uses only one crossover operator and the uniform mutation operator throughout evolution process. The target is to make the best use of the search ability of different crossover operators in GAs to produce a set of optimal feature subsets characterized by small size and high classification accuracy. To the best of our knowledge, the use of GAs for multi-objective feature selection with adaptive operator selection (AOS) [13] has not been investigated so far. To evaluate our approach, we conducted experiments on ten datasets. Meanwhile, five existing feature selection algorithms were also used for comparison.

The rest of the paper is organized as follows. In Section 2, background information and a brief literature review on solving feature selection with GAs are provided. Section 3 describes the proposed MOBGA-AOS in detail. The experimental design is presented in Section 4. Section 5 presents the discussion of results. In Section 6, conclusions and future research work are provided.

2. Related work

2.1. Multi-objective optimization

In most practical applications, multiple targets need to be met at the same time. These targets can be regarded as sub-problems in the multi-objective optimization. By simultaneously optimizing several objective functions, a set of tradeoffs can be obtained for decision makers to choose [14]. The general mathematical narration of multi-objective optimization of a minimization problem with no constraint or box constraint is as follows:

$$\begin{aligned} &min\, F(X) = (f_1(X), f_2(X), \ldots, f_m(X)) \\ &s.t.\ X \in \Omega \subseteq R^n \end{aligned} \tag{1}$$

where $X = (x_1, x_2, \ldots, x_n)$ is a decision vector in the search space Ω (n is the number of decision variables), and $F(X)$ defines m objective functions.

2.2. Genetic algorithms

GAs follow the principle of survival of the fittest and are kinds of stochastic search algorithms that are inspired by the natural selection and natural genetic mechanism of the biological world. GAs simulate the natural evolution process. Through the mechanisms of selection, crossover, and mutation, a set of candidate individuals are maintained in each iteration. This procedure is repeated until the termination criterion is satisfied [15]. The selection, crossover, and mutation operators are summarized as follows:

2.2.1. Selection

The selection operator aims to select some individuals (parents) to produce the next generation. The most commonly used methods are roulette wheel selection [16] and binary tournament selection.

2.2.2. Crossover

Crossover operation refers to the exchange of some genes between two paired parents in a certain way to form two children. Crossover is the main method for generating new individuals.

2.2.3. Mutation

The mutation operator replaces the gene value at some locus in the chromosome with other alleles of the locus to form a new individual. The mutation operator is competent to maintain diversity of the population and increase the possibility of achieving global optimization.

2.3. GAs for feature selection

GAs are the most popular evolutionary algorithms applied to feature selection problems. There are roughly three research directions on enhancing the performance of GAs to solve feature selection problems, i.e., representation, fitness function, and search mechanisms.

In terms of representation, Vignolo et al. [12] proposed two different ways to represent the solutions for face recognition tasks. Hong and Cho [17] redesigned the representation of a solution for high-dimensional feature selection. Jeony et al. [18] proposed a new representation that the length of a chromosome is equivalent to the number of desired features. However, the encoding length of individuals needs to be predefined in the aforementioned literatures. In order to represent the chromosome in a dynamic way, Yahya et al. [19] came up with a novel representation that only the selected features are encoded in the chromosome. Meanwhile, novel genetic operators were designed for reproduction of the chromosome with variable length.

When it comes to the fitness function, most of studies deal with feature selection as a single-objective problem [20–22]. However, feature selection should also be viewed as a multi-objective optimization problem. For example, in Ref. [23], the fault classification error is minimized, and the accuracy of dissolved gas analysis and diagnosis of power transformer is improved by selecting the optimal feature subset and the optimal feature number. Karasu and Saraç [24] used NSGA-II to find the optimal solutions for two different fitness functions, i.e., the number of features and classification accuracy. Das et al. [25] considered the boundary region analysis of rough set theory and the multivariate mutual information of information theory as objective functions to select accurate and informative data from the dataset.

From the perspective of search mechanisms, Gheyas and Smith [26] combined a simulated annealing, a GA, greedy algorithms and the generalized regression neural networks to form a new algorithm. Results reveal that the proposed algorithm is competent to keep generating splendid solutions. Tan et al. [27] proposed a framework based on GA for feature subset selection that combines a variety of existing feature selection approaches. Oh et al. [28] added local search technique into original GA to strengthen the local search ability, which leads to a notable improvement of the overall performance.

3. The multi-objective binary genetic algorithm

3.1. Framework of MOBGA-AOS

The framework of MOBGA-AOS is presented in Algorithm 1. Its evolutionary process is similar to that of NSGA-II. Firstly, N individuals are randomly initialized in the decision space Ω, and they compose the population P. Meanwhile, Operator Selection Probability (OSP) and its related parameters used to calculate it are initialized. The probability that each crossover operator being selected is initialized to $1/Q$, where Q is the number of all candidate crossover operators. According to OSP, one crossover operator is selected by roulette wheel selection, and then two individuals are randomly selected as parents from P. By applying the selected crossover operator together with the uniform mutation operator to the parents, two children are produced and added to the offspring population P_{new}. By comparing the dominance relationship between two parents and two children, the evolution information of whether it is successful to use the selected crossover operator to generate promising solutions is recorded to *nReward* and *nPenalty*. After $N/2$ times of evolution, an offspring population P_{new}, which consists of N child solutions, can be obtained. The fast non-dominated sorting and crowded distance calculation approaches are introduced to construct the new population for the next population by selecting N solutions from R (the union of P and P_{new}). Note that this is the evolution of one generation, and the reward and penalty information of accordingly operators in one single generation is recorded by matrices $RD_{LP\times Q}$ and $PN_{LP\times Q}$, respectively. If the number of the repeated generations is equal to LP (a pre-set value), the OSP will be updated according to $RD_{LP\times Q}$ and $PN_{LP\times Q}$. The above procedures keep repeating until the number of fitness evaluations meets *maxFEs*.

Algorithm 1 MOBGA-AOS

Input:
maxFEs: Maximum number of fitness evaluations
N: Population size
D: Number of dimensionality (original features)
M: Number of objectives
Q: Number of crossover operators
LP: Number of repeated generations for OSP
Output:
Optimal feature subsets

$P \leftarrow initPop(N)$
Initialize *nReward*, *nPenalty*, $RD_{LP\times Q}$, $PN_{LP\times Q}$ according to Eqs. (4)–(5)
$\vec{P} = \{p_1, p_2, ..., p_Q\} \leftarrow initOSP(Q)$
$k \leftarrow 0$
$nFE \leftarrow 0$
$P_{new} \leftarrow \emptyset$
while $nFE < maxFEs$ **do**
  for i = 1 to $N/2$ **do**
    $operator\_idx \leftarrow rouletteWheelSelection(\vec{P})$
    Randomly select two individuals as parents: P_p
    $P_c \leftarrow crossover(P_p, operator\_idx)$
    $P_c \leftarrow uniformMutation(P_c)$
    $nFE \leftarrow nFE + 2$
    $\{nReward, nPenalty\} \leftarrow creditAssignment(P_p, P_c)$
    Add P_c to P_{new}
  end for
  $k \leftarrow k + 1$
  Append *nReward* to the k^{th} row of $RD_{LP\times Q}$
  Append *nPenalty* to the k^{th} row of $PN_{LP\times Q}$
  if $k = LP$ **then**
    $\vec{P} \leftarrow updateOSP(RD_{LP\times Q}, PN_{LP\times Q})$
    $k = 0$
  end if
  $R \leftarrow P \cup P_{new}$
  $P \leftarrow environmentalSelection(R)$
  Select non-dominated solutions in P as PF
end while
Optimal feature subsets $\leftarrow PF$
return Optimal feature subsets

In the next five subsections, we will further specify the details of the proposed MOBGA-AOS.

3.2. Population initialization

Different from most studies [3,10], this paper introduces a straightforward method to represent the feature selection problem, i.e., 0–1 encoding schema. To be more specific, if the encoding of an individual is a vector of 0101100, it means that the number of the original features is seven, and the second feature, the fourth feature and the fifth feature is selected. After encoding, the initial population can be generated by a random approach.

3.3. Fitness evaluation

The purpose of the multi-objective feature selection problem is to find a set of optimal features with high classification accuracy and small solution size (number of features). For easy tackling the multi-objective problem, we consider minimizing classification error as the first fitness function instead of maximizing classification accuracy and the solution size are used as the second fitness function. In addition, k nearest neighbors (k-NN, $k = 3$) is used as the classifier to evaluate solutions, and n fold cross validation ($n = 3$) is used for the k-NN. The classification error used as the first fitness function is calculated

as follows:

$$minf_1(X) = (\frac{1}{n}\sum_{l=1}^{n}\frac{N_{Error}}{N_{All}}) \times 100\% \quad (2)$$

where X is a solution, N_{Error} and N_{All} are the number of wrongly predicted instances and all instances in datasets, respectively.

In addition, the solution size which is considered as the second fitness function can be calculated as follows:

$$minf_2(X) = \sum_{i=1}^{D} x_i \quad (3)$$

where x_i is the value of the ith value in individual X, D is the number of original features.

3.4. Crossover operator pool

Many different binary genetic crossover operators have been designed. Five of them that have unique search ability are elected for the generation of new solutions in MOBGA-AOS.

3.4.1. Single-point crossover operator

Firstly, one crossover point $k \in [1, 2, \ldots, L]$, where L is the length of a chromosome, is randomly selected at the same position on two parents. Then, two children are generated by swapping the genes of two parents beyond or after the crossover point [29].

3.4.2. Two-point crossover operator

Similar to single-point crossover, two points need to be selected at random. Genes of two parents between the points are exchanged to form two children [30].

3.4.3. Uniform crossover operator

Single and two-point crossover need to determine the crossover point(s) before performing crossover operations. Uniform crossover extends the scheme to each locus. In other words, every gene locus may become a crossover point. To achieve this, a binary string with the same length as the chromosome is randomly built at first. If there exists bits on this binary string that have a value of 1, the genes of the two parents need to be exchanged at the same position. After the exchanging process, two children are created [31].

3.4.4. Shuffle crossover operator

When shuffle crossover is applied, the genes of two chromosomes are randomly shuffled. Then, it applies the one-point crossover to the shuffled chromosomes after selecting a crossover point at random and two children are generated. After the recombination, the genes of the children are then unshuffled in the same way as they have been shuffled [32].

3.4.5. Reduced surrogate crossover operator

Reduced surrogate crossover constrains crossover operations in case of the parents having same genes, that is to say, the crossover points can only be the ones where the genes of the both parents are different. One position is randomly selected after all these points are found. Finally, the single-point crossover is performed at this point to create the children [29].

3.5. Credit assignment

We suppose there are $Q > 1$ operators in the operator pool, and the qth operator is selected for reproduction. Our intention lies in rewarding operators that can produce promising children that are comparative to their parents. To attain this, two vectors, i.e., *nReward* and *nPenalty*, are constructed to record the evolution information of the used operators in each generation and they are implemented as follows:

$$\begin{aligned} nReward &= \begin{pmatrix} 0 & 0 & \cdots & 0 \end{pmatrix}_{1\times Q} \\ nPenalty &= \begin{pmatrix} 0 & 0 & \cdots & 0 \end{pmatrix}_{1\times Q} \end{aligned} \quad (4)$$

In GAs, two parents can generate two children through reproduction process. Hence, the Pareto dominance relationship between the parents and the children are analyzed to update *nReward* and *nPenalty*. According to the Pareto dominance relationship between the two parents, there are two cases:

3.5.1. One parent dominates the other

Let us suppose that parent i is dominated by parent j, each child is used to compare the Pareto dominance relationship with the parent j, respectively. If the child do not dominated by parent j, then $nReward_q + 1$, otherwise $nPenalty_q + 1$.

3.5.2. The two parents are non-dominated each other

Each child is used to compare the Pareto dominance relationship with the two parents, respectively. If the child is not dominated by the two parents at the same time, $nReward_q + 1$, otherwise $nPenalty_q + 1$. The detailed procedure is given in Algorithm 2.

Algorithm 2 *creditAssignment*(P, R, q)

Input: Set of two parents P, set of two children R, the selected crossover operator q
Output: *nReward*, *nPenalty*

```
{P_nd, P_d} ← dominanceComparison(P)   //P_nd and P_d refer to the sets of non-dominated and dominated solutions, respectively.
if P_d ≠ ∅ then
    // One parent dominates the other, and suppose P_1 ≺ P_2.
    for i = 1 to 2 do
        if P_1 ≺ R_i then
            nPenalty_q ← nPenalty_q + 1
        else
            nReward_q ← nReward_q + 1
        end if
    end for
else
    // The two parents are non-dominated each other.
    for i = 1 to 2 do
        if P_1 ⊀ R_i && P_2 ⊀ R_i then
            nReward_q ← nReward_q + 1
        else
            nPenalty_q ← nPenalty_q + 1
        end if
    end for
end if
```

3.6. Update of OSP

In order to employ the appropriate operator for reproduction at different phases according to the previous performance of all operators, we update OSP every *LP* generations, which is the same as in [10]. As mentioned in the previous subsection, the evolution information of each generation is recorded in *nReward* and *nPenalty*. Therefore, two matrices, *RD* and *PN*, are constructed

as follows to store them:

$$RD=\begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}_{LP\times Q} \quad \text{and} \quad PN=\begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}_{LP\times Q} \tag{5}$$

nReward and *nPenalty* obtained in each generation are appended to *RD* and *PN* respectively by rows until the *LP* generations have passed. With the evolution information of the last *LP* generations stored in *RD* and *PN*, the OSP of the operators are recalculated. To calculate the probability for the qth ($q = 1, 2, \ldots, Q$) operator, we first sum the qth column of *RD* and *PN*, respectively:

$$S_q^1 = \sum_{k=1}^{LP} RD_{k,q} \tag{6}$$

$$S_q^2 = \sum_{k=1}^{LP} PN_{k,q} \tag{7}$$

where S_q^1 is the numbers of promising solutions generated by the qth operator in the previous *LP* generations, while S_q^2 refers to the unsatisfactory solutions.

Then, the OSP for the qth operator can be calculated as follows:

$$S_q^3=\begin{cases} \delta, & if\ S_q^1=0 \\ S_q^1, & otherwise \end{cases} \tag{8}$$

$$S_q^4 = S_q^1/(S_q^3 + S_q^2) \tag{9}$$

where δ is very small number and its value is 0.0001. It is employed to prevent being divided by zero in the case of the qth operator being never selected in the last *LP* generations. In addition, S_q^4 refers to the probability assigned to qth operator.

According to the above process, we can calculate the probability of each strategy being selected during *LP* generations. To make the sum of probabilities equal to 1, we finally normalize the probabilities as follows:

$$p_q = S_q^4/\sum_{q=1}^{Q} S_q^4 \tag{10}$$

where p_q is the normalized probability for the qth operator.

4. Experimental design

4.1. Datasets and classifiers

In order to validate the effectiveness of MOBGA-AOS, ten datasets with different characteristics are chosen from the UCI Machine Learning Repository [33]. The details of them can be seen from Table 1. These datasets are composed of different numbers of features (*NoF* in Table 1), labels (*NoL* in Table 1) and instances (*NoI* in Table 1). All datasets are then split into training sets (70% of the original datasets) and test sets (30% of the original datasets) at random. The k-NN was used as the classifier to evaluate feature subsets ($k = 3$), and n fold cross validation ($n = 3$) was introduced to assess the classification accuracy for k-NN [10].

4.2. Benchmark algorithms and parameter settings

To validate the effectiveness of MOBGA-AOS, NSGA-II [34], NSPSOFS [3], CMDPSOFS [3], SPEA2 [35] and MOEA/D [36] were employed for comparison. Each algorithm was run 30 times on the datasets. For fair comparison, the number of fitness evaluation (*nFE*) was used as the termination criterion, i.e., the same maximum fitness function evaluations (*maxFEs*) was used for all algorithms. The following parameter values were used in MOBGA-AOS. *maxFEs* = 300,000, $N = 100$, D equals the number of original features of each dataset, $M = 2$, $Q = 5$, $LP = 5$ [10]. The crossover rate P_c for all crossover operators is set to 0.9 and the mutation rate P_m is set to $1/D$ for the uniform mutation operator. The parameter values of five benchmark algorithms are set as in their referred papers.

Table 1
Details about used datasets.

ID	Datasets	NoF	NoL	NoI
DS01	Wdbc	30	2	596
DS02	LungCancer	56	3	32
DS03	ConnectionistBenchData	60	2	208
DS04	OpticalRecognitionofHandwritten	64	10	1000
DS05	MadelonValid	500	2	600
DS06	UJIIndoorLoc	522	3	900
DS07	Har	561	6	900
DS08	HAPT	561	12	1200
DS09	Isolet5	617	26	1040
DS10	MultipleFeaturesDigit	649	10	1000

4.3. Performance metrics for algorithms

In this paper, inverted generational distance (IGD) [37] and hypervolume (HV) [38] are employed to assess convergence and distribution performance of different multi-objective algorithms. In addition, a statistical significance test (T-test) [39] with a confidence interval of 95% is used on the IGD and HV metrics.

5. Results and discussions

In this section, we discuss the performance of MOBGA-AOS comparing with five benchmark algorithms from the aspects of IGD and HV metrics. Also, results of training Pareto fronts obtained by all six algorithms are given as well. Note that the training Pareto front of each algorithm is composed of all non-dominated solutions of that in 30 independent runs. The IGD and HV are introduced to evaluate the convergence and diversity of the results achieved by these six algorithms on both training and test sets. When comparing metrics obtained by two different algorithms on the same dataset, the algorithm that obtains the smaller IGD value or the larger HV value is superior to the other. As the true Pareto front is not available in the datasets used in this paper, we first merge the training (or testing) Pareto front calculated by all the comparison algorithms and MOBGA-AOS into a set, *PF-Union*. Then, we select the non-dominated solutions by non-dominated sorting mechanism from *PF-Union* as the "true Pareto front" for the calculation of IGD and HV.

5.1. Results of IGD on the training and test sets

Tables 2 and 3 present IGD with respect to the mean values and standard deviations achieved by NSGA-II, NSPSOFS, CMDPSOFS, SPEA2, MOEA/D and MOBGA-AOS on ten training and test sets, respectively. Best mean values are typed in bold. In addition, T-Sig indicates whether there is a significant difference between MOBGA-AOS and other algorithms on the IGD and HV metrics. "+" or "-" means that MOBGA-AOS achieves significantly better or worse Pareto front than other algorithms, "=" means there is no significant difference between two compared algorithms.

Table 2
Mean values and standard deviations of IGD values obtained by the six algorithms on the training sets.

Datasets		NSGA-II		NSPSOFS		CMDPSOFS		SPEA2		MOEA/D		MOBGA-AOS	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
DS01	IGD	4.45E−01	2.79E−01	1.93E−01	1.75E−01	1.88E−01	1.90E−01	2.44E−01	3.07E−01	9.17E−01	5.06E−01	**1.38E−01**	1.84E−01
	T-Sig	+		=		=		=		+			
DS02	IGD	1.90E+00	1.47E+00	2.43E+00	1.06E+00	**2.65E−01**	2.02E−01	5.21E−01	2.68E−01	1.11E+00	7.84E−01	8.30E−01	5.39E−01
	T-Sig	+		+		−		−		=			
DS03	IGD	1.79E+00	1.24E+00	2.86E+00	1.09E+00	2.04E−01	1.42E−01	3.98E−01	1.91E−01	1.34E+00	9.33E−01	**1.03E−01**	9.66E−02
	T-Sig	+		+		+		+		+			
DS04	IGD	1.15E+00	6.42E−01	1.70E+00	3.48E−01	5.06E−01	1.77E−01	6.97E−01	3.23E−01	2.56E+00	3.90E−01	**3.62E−01**	2.02E−01
	T-Sig	+		+		+		+		+			
DS05	IGD	1.34E+02	7.68E+00	1.24E+02	2.01E+01	7.55E+00	3.84E+00	1.88E+01	4.10E+00	5.21E+01	1.31E+01	**6.80E−01**	3.72E−01
	T-Sig	+		+		+		+		+			
DS06	IGD	1.39E+02	9.44E+00	1.31E+02	1.77E+01	5.27E+00	3.69E+00	2.70E+01	6.32E+00	5.58E+01	1.11E+01	**0.00E+00**	0.00E+00
	T-Sig	+		+		+		+		+			
DS07	IGD	1.49E+02	8.86E+00	1.32E+02	2.39E+01	2.30E+01	5.10E+00	2.25E+01	7.94E+00	4.86E+01	1.16E+01	**4.16E−01**	3.20E−01
	T-Sig	+		+		+		+		+			
DS08	IGD	1.44E+02	9.66E+00	1.32E+02	1.90E+01	1.92E+01	5.08E+00	1.85E+01	5.57E+00	4.75E+01	1.58E+01	**6.95E−01**	3.78E−01
	T-Sig	+		+		+		+		+			
DS09	IGD	1.54E+02	1.09E+01	1.41E+02	2.64E+01	1.49E+01	3.75E+00	1.52E+01	6.10E+00	4.50E+01	1.73E+01	**1.06E+00**	6.56E−01
	T-Sig	+		+		+		+		+			
DS10	IGD	1.72E+02	9.40E+00	1.55E+02	2.32E+01	2.76E+01	5.29E+00	2.32E+01	6.25E+00	5.48E+01	1.52E+01	**1.61E+00**	1.05E+00
	T-Sig	+		+		+		+		+			

Table 3
Mean and Standard Deviation of IGD Values Obtained by the Six Algorithms on the Test Sets.

Datasets		NSGA-II		NSPSOFS		CMDPSOFS		SPEA2		MOEA/D		MOBGA-AOS	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
DS01	IGD	6.67E−01	4.33E−01	**3.89E−01**	2.68E−01	7.15E−01	3.37E−01	9.33E−01	1.73E−01	1.12E+00	4.69E−01	8.39E−01	2.68E−01
	T-Sig	=		−		=		=		+			
DS02	IGD	2.48E+00	1.67E+00	3.01E+00	1.30E+00	**8.95E−01**	3.56E−01	1.11E+00	3.95E−01	1.69E+00	9.35E−01	1.01E+00	4.63E−01
	T-Sig	+		+		=		=		+			
DS03	IGD	2.37E+00	1.32E+00	3.43E+00	1.17E+00	6.66E−01	2.71E−01	8.94E−01	4.10E−01	1.99E+00	9.66E−01	**6.06E−01**	2.53E−01
	T-Sig	+		+		=		+		+			
DS04	IGD	1.44E+00	8.03E−01	2.29E+00	5.32E−01	5.62E−01	1.91E−01	**3.87E−01**	1.50E−01	1.43E+00	2.77E−01	4.21E−01	2.15E−01
	T-Sig	+		+		+		=		+			
DS05	IGD	1.34E+02	7.68E+00	1.24E+02	2.01E+01	7.59E+00	3.61E+00	1.85E+01	4.09E+00	5.18E+01	1.31E+01	**1.78E+00**	4.00E−01
	T-Sig	+		+		+		+		+			
DS06	IGD	1.39E+02	9.44E+00	1.31E+02	1.77E+01	5.27E+00	3.69E+00	2.70E+01	6.32E+00	5.58E+01	1.11E+01	**1.07E−03**	2.51E−03
	T-Sig	+		+		+		+		+			
DS07	IGD	1.42E+02	8.86E+00	1.25E+02	2.39E+01	2.13E+01	4.04E+00	2.17E+01	6.08E+00	4.27E+01	1.05E+01	**7.47E+00**	1.51E+00
	T-Sig	+		+		+		+		+			
DS08	IGD	1.48E+02	9.66E+00	1.36E+02	1.90E+01	2.47E+01	4.70E+00	2.42E+01	5.13E+00	5.17E+01	1.55E+01	**3.08E+00**	1.02E+00
	T-Sig	+		+		+		+		+			
DS09	IGD	1.63E+02	1.09E+01	1.50E+02	2.64E+01	2.21E+01	4.53E+00	2.22E+01	7.30E+00	5.35E+01	1.73E+01	**1.85E+00**	7.98E−01
	T-Sig	+		+		+		+		+			
DS10	IGD	1.74E+02	9.40E+00	1.56E+02	2.32E+01	3.20E+01	5.47E+00	2.84E+01	6.58E+00	5.83E+01	1.40E+01	**4.26E+00**	1.46E+00
	T-Sig	+		+		+		+		+			

Table 4
Mean Values and Standard Deviations of HV Values Obtained by the Six Algorithms on the Training Sets.

Datasets		NSGA-II		NSPSOFS		CMDPSOFS		SPEA2		MOEA/D		MOBGA-AOS	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
DS01	HV	3.72E−02	1.70E−02	8.34E−03	4.46E−03	4.59E−02	3.44E−03	**6.07E−02**	6.01E−03	2.01E−02	1.39E−02	6.01E−02	5.11E−03
	T-Sig	+		+		+		=		+			
DS02	HV	1.87E−01	2.39E−01	3.81E−03	1.50E−02	1.01E+00	4.14E−02	**1.05E+00**	8.55E−02	3.45E−01	2.38E−01	7.33E−01	3.48E−01
	T-Sig	+		+		−		−		+			
DS03	HV	1.24E−01	1.13E−01	2.85E−03	7.38E−03	4.81E−01	3.87E−02	6.03E−01	4.82E−02	1.64E−01	1.25E−01	**6.20E−01**	3.87E−02
	T-Sig	+		+		+		=		+			
DS04	HV	1.77E+01	1.24E+00	1.57E+01	5.69E−01	2.00E+01	2.33E−01	2.06E+01	6.03E−02	1.79E+01	7.96E−01	**2.07E+01**	1.72E−02
	T-Sig	+		+		+		+		+			
DS05	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	5.40E−01	6.11E−01	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**3.27E+00**	3.98E−02
	T-Sig	+		+		+		+		+			
DS06	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**1.00E−25**	5.84E−41
	T-Sig	+		+		+		+		+			
DS07	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E−02	5.49E−02	4.75E−02	1.58E−01	0.00E+00	0.00E+00	**7.66E+00**	2.31E−01
	T-Sig	+		+		+		+		+			
DS08	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.13E+00	2.27E+00	4.67E+00	2.76E+00	1.87E−01	6.08E−01	**1.91E+01**	5.37E−01
	T-Sig	+		+		+		+		+			
DS09	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.96E+01	3.50E+00	2.04E+01	6.13E+00	2.25E+00	3.61E+00	**4.25E+01**	1.15E+00
	T-Sig	+		+		+		+		+			
DS10	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.35E+00	3.26E+00	1.27E+01	4.18E+00	8.01E−01	1.82E+00	**3.12E+01**	5.79E−01
	T-Sig	+		+		+		+		+			

Table 5
Mean Values and Standard Deviations of HV Values Obtained by the Six Algorithms on the Test Sets.

Datasets		NSGA-II		NSPSOFS		CMDPSOFS		SPEA2		MOEA/D		MOBGA-AOS	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
DS01	HV	4.84E−02	6.12E−02	4.17E−02	3.70E−02	1.25E−01	3.04E−02	**1.41E−01**	3.49E−02	3.23E−02	4.10E−02	1.15E−01	3.53E−02
	T-Sig	+		+		=		−		+			
DS02	HV	4.98E−02	1.63E−01	1.19E−02	2.57E−02	1.25E−01	1.46E−01	**1.59E−01**	2.47E−01	1.38E−01	1.53E−01	1.45E−01	1.93E−01
	T-Sig	+		+		=		=		=			
DS03	HV	1.53E−02	3.81E−02	3.02E−03	1.25E−02	**1.41E−01**	9.18E−02	1.22E−01	9.23E−02	6.67E−02	1.12E−01	1.01E−01	8.62E−02
	T-Sig	+		+		=		=		=			
DS04	HV	9.69E+00	1.00E+00	8.20E+00	5.35E−01	1.20E+01	2.18E−01	1.21E+01	1.85E−01	1.01E+01	7.04E−01	**1.22E+01**	1.00E−01
	T-Sig	+		+		+		+		+			
DS05	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.16E−01	5.98E−01	1.91E−02	8.92E−02	0.00E+00	0.00E+00	**2.93E+00**	2.94E−01
	T-Sig	+		+		+		+		+			
DS06	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**8.00E−26**	4.07E−26
	T-Sig	+		+		+		+		+			
DS07	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.38E+01	2.03E+00	1.34E+01	3.09E+00	3.37E+00	2.63E+00	**2.62E+01**	7.37E−01
	T-Sig	+		+		+		+		+			
DS08	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.81E+00	2.12E+00	8.06E+00	2.38E+00	7.47E−01	1.64E+00	**2.07E+01**	7.02E−01
	T-Sig	+		+		+		+		+			
DS09	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.56E+01	3.56E+00	1.59E+01	5.91E+00	1.07E+00	2.46E+00	**3.71E+01**	1.30E+00
	T-Sig	+		+		+		+		+			
DS10	HV	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.16E+01	3.38E+00	3.44E+01	4.03E+00	1.37E+01	7.20E+00	**5.28E+01**	9.07E−01
	T-Sig	+		+		+		+		+			

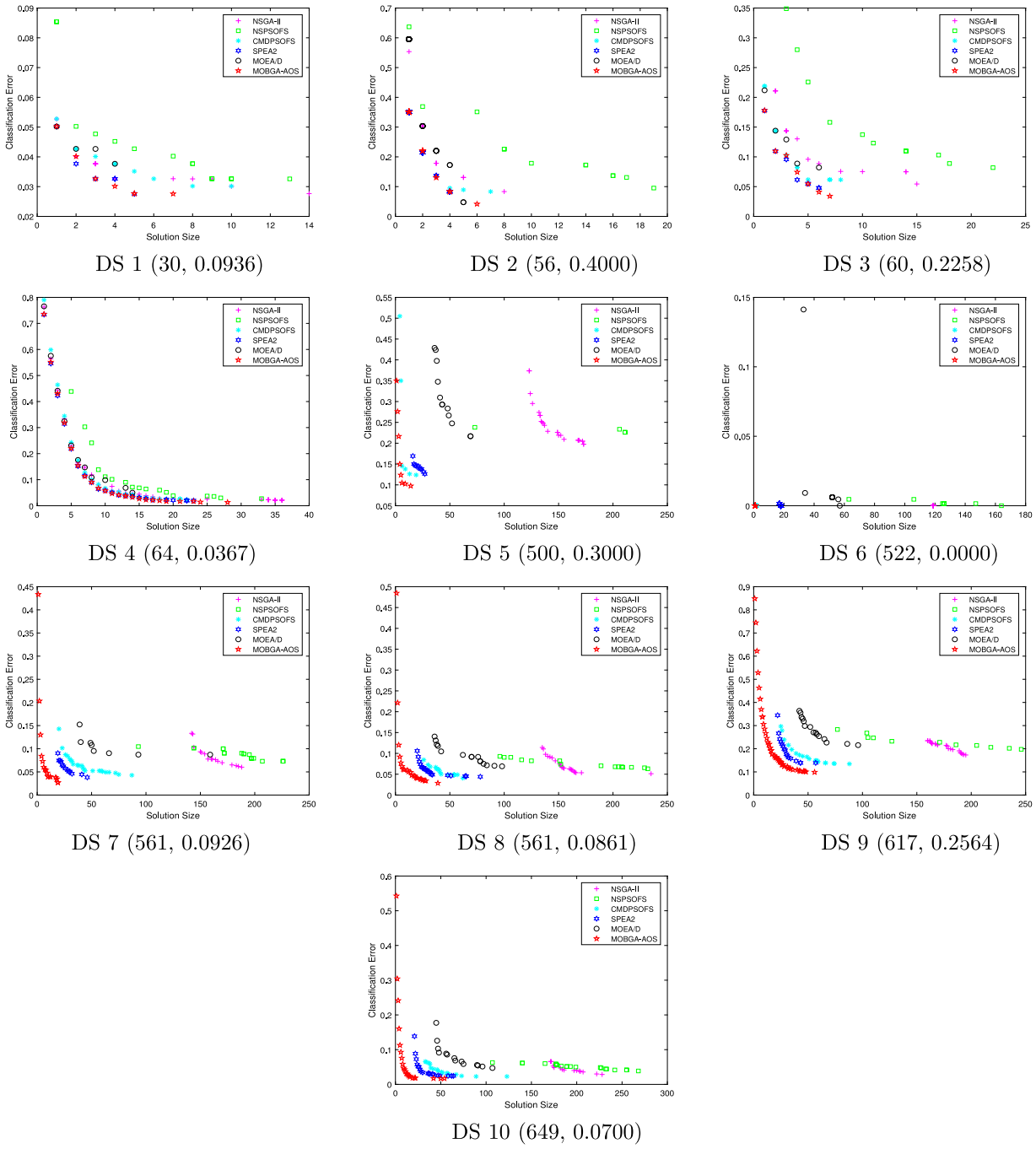

Fig. 1. Pareto fronts of different algorithms on the training sets.

As can be seen in Table 2 that MOBGA-AOS obtains the smallest mean IGD on nine out of ten training sets. Besides, MOBGA-AOS is significantly better than the other five benchmark algorithms on eight training sets, which are the top eight in terms of the number of features. They are all datasets except DS 1 and DS 2. On DS 1, which is characterized by a smallest number of features compared to the other datasets used in this paper, MOBGA-AOS can achieve better results than NSGA-II and MOEA/D and similar results to NSPSOFS, CMDPSOFS and SPEA2. On DS 2, whose number of instance is relatively small, MOBGA-AOS can achieve better results than NSGA-II and NSPSOFS and similar results to MOEA/D. However, when compared to CMDPSOFS and SPEA2, results obtained by MOBGA-AOS are slightly inferior to them.

Table 3 illustrates that MOBGA-AOS outperforms all benchmark algorithms on six out of ten test sets, all of which have a number of features equal to or greater than 500. Meanwhile, although MOBGA-AOS is significantly worse than NSPSOFS on DS 1, it achieves better or similar results compared to NSPSOFS, CMDPSOFS, SPEA2, MOEA/D, and it also achieves better or similar results compared to all benchmark algorithms on DS 2-DS 4.

5.2. Results of HV on the training and test sets

Tables 4 and 5 present HV with respect to the mean values and standard deviations achieved by NSGA-II, NSPSOFS, CMDPSOFS, SPEA2, MOEA/D and MOBGA-AOS on ten training and test sets, respectively. It can be observed from Table 4 that MOBGA is

superior to NSGA-II, NSPSOFS, and MOEA/D on all training sets. Besides, it can also achieve better or similar results than CMDPSOFS and SPEA2 on nine out of ten training sets except for DS 2.

According to Table 5, MOBGA-AOS achieves better or similar results than NSGA-II, NSPSOFS, CMDPSOFS and MOEA/D on all test sets. When it comes to the comparison between MOBGA-AOS and SPEA2, the former beats the latter on nine out of ten test sets while the latter is much superior to the former on DS 1. Moreover, it can be also concluded from Tables 4 and 5 that MOBGA-AOS performs well on DS 4-DS 10, all but DS 4 of which have a great number of features while the number of labels in DS 4 is relatively large.

5.3. Results of training pareto fronts

To further evaluate the performance of MOBGA-AOS on searching for feature subsets, non-dominated solutions for each algorithm in the results of 30 independent runs on each training sets, which are called training Pareto fronts in this paper, are plotted in Fig. 1. The horizontal axis of each subgraph represents the size of the feature subset and the vertical axis represents the classification error. In addition, for each train set, the original number of features and the classification error using all features for classification are also marked at the bottom of each subgraph.

It can be easily seen from Fig. 1 that MOBGA-AOS is competitive with all the benchmark algorithms on almost all training sets. When the dimensionality of the training sets is high, such as DS 5-DS 10, MOBGA-AOS tends to achieve the smaller solution size while ensuring a lower classification error. It is ahead of other algorithms and is ranked in the first place. For example, on DS 10, which has 649 original features, the solution size is eliminated to around 30 after the optimization of SPEA2 and around 50 after the optimization of CMDPSOFS. Although SPEA2 and CMDPSOFS ranking the second and third place can achieve satisfied results, MOBGA-AOS is able to reduce about 97% original features to around 15 which is half that of SPEA2 and is one thirds that of CMDPSOFS. Meanwhile, MOBGA-AOS remarkably reduce 522 original features to 1 while guaranteeing that all categories are correctly classified. Another point worth noting is that on these datasets with a large number of features, MOBGA-AOS can always find some solutions with very few features and extremely large classification error. Although the classification error of these solutions is much greater than that of using all features, we can argue that thanks to these solutions, the population is guaranteed to have excellent diversity enabling MOBGA-AOS to avoid falling into the local optima and premature convergence during the searching process.

In terms of training sets with relatively small number of features, such as DS 1-DS 4, MOBGA-AOS's performance is not much different from the benchmark algorithms, but still at a good level. It can reduce the features from 30 to around 4 on DS 1, from 56 to around 3 on DS 2 and from 60 to around 4 on DS 3. In addition to that, the classification error of all obtained feature subsets is lower than that of using all features on DS 1-DS 3. On DS 4, MOBGA-AOS is on par with NSGA-II, CMDPSOFS, SPEA2, MOEA/D and is slightly better than NSPSOFS.

Overall, the results demonstrate that the adaptive operator selection mechanism allows MOBGA-AOS to make full use of the characteristics of every operator to maintain convergence and diversity simultaneously.

6. Conclusion and future work

In this paper, we have proposed a multi-objective GA-based algorithm, called MOBGA-AOS, to obtain a set of feature subsets with less feature number and lower classification error. By adopting the adaptive operator selection mechanism, MOBGA-AOS takes advantage of five crossover operators with distinct capability to effectively search in the search space. To validate the competitiveness of MOBGA-AOS, we adopt five well-known algorithms for comparison. Ten benchmark datasets are used to assess the performance of MOBGA-AOS and its comparison algorithms. It can be observed from the experimental results that MOBGA-AOS performs well on most datasets especially on large-scale ones, and is capable of finding a set of well-converged and diverse optimal feature subsets. In the future work, we are about to add the different mutation operators to constitute a mutation operator pool so that appropriate combinations of crossover and mutation operators can be selected at different stages of evolution to further enhance the performance of the algorithm. Meanwhile, we also intend to adjust the parameters associated with these operators in an automatic way while adaptively selecting the operators in the evolution process.

CRediT authorship contribution statement

Yu Xue: Methodology. **Haokai Zhu:** Writing - original draft. **Jiayu Liang:** Writing - review & editing. **Adam Słowik:** Writing - review & editing.

Declaration of competing interest

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

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61876089, 61876185, 61902281), the Opening Project of Jiangsu Key Laboratory of Data Science and Smart Software (No. 2019DS301), the Natural Science Foundation of Jiangsu Province (BK20141005), the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (14KJB520025).

References

[1] P. Hu, J.-S. Pan, S.-C. Chu, Improved binary grey wolf optimizer and its application for feature selection, Knowl.-Based Syst. 195 (2020) 105746.
[2] M. Dash, H. Liu, Feature selection for classification, Intell. Data Anal. 1 (3) (1997) 131–156.
[3] B. Xue, M. Zhang, W.N. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, IEEE Trans. Cybern. 43 (6) (2012) 1656–1671.
[4] H. Dong, J. Sun, X. Sun, R. Ding, A many-objective feature selection for multi-label classification, Knowl.-Based Syst. 208 (2020) 106456.
[5] C. Qu, W. Gai, J. Zhang, M. Zhong, A novel hybrid grey wolf optimizer algorithm for unmanned aerial vehicle (UAV) path planning, Knowl.-Based Syst. (2020) 105530.
[6] D. Gong, Y. Han, J. Sun, A novel hybrid multi-objective artificial bee colony algorithm for blocking lot-streaming flow shop scheduling problems, Knowl.-Based Syst. 148 (2018) 115–130.
[7] Y. Xu, O. Ding, R. Qu, K. Li, Hybrid multi-objective evolutionary algorithms based on decomposition for wireless sensor network coverage optimization, Appl. Soft Comput. 68 (2018) 268–282.
[8] H. Dong, T. Li, R. Ding, J. Sun, A novel hybrid genetic algorithm with granular information for feature selection and optimization, Appl. Soft Comput. 65 (2018) 33–46.
[9] B. Tran, B. Xue, M. Zhang, Variable-length particle swarm optimization for feature selection on high-dimensional classification, IEEE Trans. Evol. Comput. 23 (3) (2018) 473–487.

[10] Y. Xue, B. Xue, M. Zhang, Self-adaptive particle swarm optimization for large-scale feature selection in classification, ACM Trans. Knowl. Discov. Data 13 (5) (2019) 1–27.
[11] J.-W. Kang, H.-J. Park, J.-S. Ro, H.-K. Jung, A strategy-selecting hybrid optimization algorithm to overcome the problems of the no free lunch theorem, IEEE Trans. Magn. 54 (3) (2018) 1–4.
[12] L.D. Vignolo, D.H. Milone, J. Scharcanski, Feature selection for face recognition based on multi-objective evolutionary wrappers, Expert Syst. Appl. 40 (13) (2013) 5077–5084.
[13] K. Li, A. Fialho, S. Kwong, Q. Zhang, Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 18 (1) (2013) 114–130.
[14] E. Hancer, B. Xue, M. Zhang, Differential evolution for filter feature selection based on information theory and feature ranking, Knowl.-Based Syst. 140 (2018) 103–119.
[15] A.K. Das, S. Sengupta, S. Bhattacharyya, A group incremental feature selection for classification using rough set theory based genetic algorithm, Appl. Soft Comput. 65 (2018) 400–411.
[16] W. Qian, J. Chai, Z. Xu, Z. Zhang, Differential evolution algorithm with multiple mutation strategies based on roulette wheel selection, Appl. Intell. 48 (10) (2018) 3612–3629.
[17] J.H. Hong, S.B. Cho, Efficient huge-scale feature selection with speciated genetic algorithm, Pattern Recognit. Lett. 27 (2) (2006) 143–150.
[18] Y.-S. Jeong, K.S. Shin, M.K. Jeong, An evolutionary algorithm with the partial sequential forward floating search mutation for large-scale feature selection problems, J. Oper. Res. Soc. 66 (4) (2015) 529–538.
[19] A.A. Yahya, A. Osman, A.R. Ramli, A. Balola, Feature selection for high dimensional data: An evolutionary filter approach, J. Comput. Sci. 7 (5) (2011) 800–820.
[20] S.F. Da Silva, M.X. Ribeiro, J.d.E.B. Neto, C. Traina-Jr, A.J. Traina, Improving the ranking quality of medical image retrieval using a genetic feature selection method, Decis. Support Syst. 51 (4) (2011) 810–820.
[21] S.M. Winkler, M. Affenzeller, W. Jacak, H. Stekel, Identification of cancer diagnosis estimation models using evolutionary algorithms: a case study for breast cancer, melanoma, and cancer in the respiratory system, in: Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation, 2011, pp. 503–510.
[22] A.M. Canuto, D.S. Nascimento, A genetic-based approach to features selection for ensembles using a hybrid and adaptive fitness function, in: The 2012 International Joint Conference on Neural Networks, IJCNN, IEEE, 2012, pp. 1–8.
[23] A. Peimankar, S.J. Weddell, T. Jalal, A.C. Lapthorn, Evolutionary multi-objective fault diagnosis of power transformers, Swarm Evol. Comput. 36 (2017) 62–75.
[24] S. Karasu, Z. Saraç, Investigation of power quality disturbances by using 2D discrete orthonormal S-transform, machine learning and multi-objective evolutionary algorithms, Swarm Evol. Comput. 44 (2019) 1060–1072.
[25] A.K. Das, S. Das, A. Ghosh, Ensemble feature selection using bi-objective genetic algorithm, Knowl.-Based Syst. 123 (2017) 116–127.
[26] I.A. Gheyas, L.S. Smith, Feature subset selection in large dimensionality domains, Pattern Recognit. 43 (1) (2010) 5–13.
[27] F. Tan, X. Fu, Y. Zhang, A.G. Bourgeois, A genetic algorithm-based method for feature subset selection, Soft Comput. 12 (2) (2008) 111–120.
[28] I.-S. Oh, J.-S. Lee, B.-R. Moon, Hybrid genetic algorithms for feature selection, IEEE Trans. Pattern Anal. Mach. Intell. 26 (11) (2004) 1424–1437.
[29] A.J. Umbarkar, P.D. Sheth, Crossover operators in genetic algorithms: a review, ICTACT J. Soft Comput. 6 (1) (2015).
[30] W.M. Spears, K.A.D. Jong, An analysis of multi-point crossover, Found. Genet. Algorithms 1 (1991) 301–315.
[31] W.M. Spears, K.A. Dejong, On the virtues of parameterised uniform crossover, in: International Conference on Genetic Algorithms, 1991.
[32] R.A. Caruana, L.J. Eshelman, J.D. Schaffer, Representation and hidden bias II: Eliminating defining length bias in genetic search via shuffle crossover, 1989.
[33] K. Bache, M. Lichman, UCI machine learning repository, 2013.
[34] 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.
[35] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the strength Pareto evolutionary algorithm, TIK-Report 103 (2001).
[36] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.
[37] Y. Sun, G.G. Yen, Z. Yi, IGD indicator-based evolutionary algorithm for many-objective optimization problems, IEEE Trans. Evol. Comput. 23 (2) (2018) 173–187.
[38] H. Ishibuchi, R. Imada, Y. Setoguchi, Y. Nojima, How to specify a reference point in hypervolume calculation for fair performance comparison, Evol. Comput. 26 (3) (2018) 411–440.
[39] T.K. Kim, T test as a parametric statistic, Korean J. Anesthesiol. 68 (6) (2015) 540.