7-25-2023

Enhanced SparseEA for large-scale multi-objective feature selection problems

Shu-Chuan Chu

Zhongjie Zhuang

Jeng-Shyang Pan

Ali Wagdy Mohamed

Complex & Intelligent Systems
https://doi.org/10.1007/s40747-023-01177-2

ORIGINAL ARTICLE

Enhanced SparseEA for large-scale multi-objective feature selection problems

Shu-Chuan Chu[1,2] · Zhongjie Zhuang[1] · Jeng-Shyang Pan[1,3] · Ali Wagdy Mohamed[4,5] · Chia-Cheng Hu[6]

Received: 4 September 2022 / Accepted: 22 January 2023

Abstract

Large-scale multi-objective feature selection problems are widely existing in the fields of text classification, image processing, and biological omics. Numerous features usually mean more correlation and redundancy between features, so effective features are usually sparse. SparseEA is an evolutionary algorithm for solving Large-scale Sparse Multi-objective Optimization Problems (i.e., most decision variables of the optimal solutions are zero). It determines feature Scores by calculating the fitness of individual features, which does not reflect the correlation between features well. In this manuscript, ReliefF was used to calculate the weights of features, with unimportant features being removed first. Then combine the weights calculated by ReliefF with Scores of SparseEA to guide the evolution process. Moreover, the Scores of features remain constant throughout all runs in SparseEA. Therefore, the fitness values of excellent and poor individuals in each iteration are used to update the Scores. In addition, difference operators of Differential Evolution are introduced into SparseEA to increase the diversity of solutions and help the algorithm jump out of the local optimal solution. Comparative experiments are performed on large-scale datasets selected from scikit-feature repository. The results show that the proposed algorithm is superior to the original SparseEA and the state-of-the-art algorithms.

Keywords Sparse multi-objective problems · Large-scale feature selection · Difference operator · Feature weights · SparseEA

Introduction

Feature engineering is an important and critical part of machine learning [1]. Essentially, feature engineering is a process of representing data. In practice, feature engineering aims to remove defects and redundancy in the raw data and design more efficient features to describe the relationship between the solved problem and the prediction model. It is generally accepted that data and features determine the upper bound of the performance of machine learning, and the models and algorithms can only approximate this bound as best they can. Thereby, it can be seen that good data and features are the premise for models and algorithms to play an essential role. In detail, feature engineering usually includes feature availability assessment, feature cleaning, feature storage, feature selection, feature extraction, and so on. Among

✉ Jeng-Shyang Pan
jspan@cc.kuas.edu.tw

Shu-Chuan Chu
scchu0803@gmail.com

Zhongjie Zhuang
zhongjiezhuang@126.com

Ali Wagdy Mohamed
aliwagdy@gmail.com

Chia-Cheng Hu
cchu.chiachenghu@gmail.com

1 College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

2 College of Science and Engineering, Flinders University, 1284 South Road, Clovelly Park, SA 5042, Australia

3 Department of Information Management, Chaoyang University of Technology, Taichung 41349, Taiwan

4 Operations Research Department, Faculty of Graduate Studies for Statistical Research, Cairo University, Giza 12613, Egypt

5 Department of Mathematics and Actuarial Science, School of Sciences and Engineering, The American University in Cairo, Cairo, Egypt

6 College of Artificial Intelligence, Yango University, Fuzhou 330015, China

Published online: 25 July 2023

them, feature selection is an important part of feature engineering [2, 3]. The main application fields of feature selection include text classification [4, 5], image recognition [6], bioinformation analysis [7], time series [8], intrusion detection [9], and software defect prediction [10].

The main idea of feature selection is to select the most valuable feature subsets by deleting irrelevant and redundant features from the feature space of the original dataset to improve the prediction accuracy, robustness, and interpretability of the models. The feature selection method was first proposed by Dash and Liu [11]. It can be divided into four steps: generating feature subset, evaluating feature subset, setting stop criterion and judging whether stop is sufficient, and verifying the final result. Suppose there are n features, each of which can be selected or not, then there are 2^n cases of the feature subset. When n is very large, it is obviously not feasible to obtain the optimal subset of features by exhaustive selection due to the time complexity. Therefore, it is an important problem that must be considered and solved to find the optimal feature from the feature space quickly and effectively [12].

Feature selection algorithms can be divided into different categories according to different separability criteria. According to the classification features, it can be divided into supervised and unsupervised feature selection algorithms. In terms of search strategies, feature selection algorithms can be categorized into global search, sequential search, and random search. Depending on the combination form of feature selection and machine learning algorithms, it includes four types [13]: filter, wrapper, embedded, and ensemble. With different evaluation criteria, feature selection algorithms can be divided into several categories: based on distance measurement [14, 15], dependency measurement [16, 17], information measurement [18], and accuracy/error rate measurement [19]. In detail, the core of distance measurement is distance formula, and commonly used distances are Euclidean distance, Hamming distance, Probability distance, and so on. Algorithms based on dependency measures use statistical principles to evaluate the correlation between features and categories, such as T test, Pearson correlation coefficient, and Fisher scores. The information metrics include mutual information, information gain, minimum description length, etc. In particular, the algorithms based on the measurement of accuracy/error rate have the best overall performance. They train the classifier using the selected feature subset and measure the performance of the feature subset by the accuracy/error rate.

Meta-heuristic algorithm is widely used because of its simplicity and generality [20, 21]. In recent years, more and more feature selection algorithms using meta-heuristic algorithms have been proposed, which are based on the measurement of accuracy/error rate [22]. The feature selection algorithms can be divided into single-objective feature selection and multi-objective feature selection according to the number of evaluation criteria. For a long time in the past, feature selection was regarded as a single objective optimization problem, which optimized the weighted sum of the accuracy/error rate and the number of selected features, or only optimized the accuracy/error rate. There are many excellent studies for solving single-objective feature selection problem by meta-heuristic algorithms. In 2020, the improved Binary Grey Wolf Optimizer was proposed and achieved good performance on the single-objective feature selection problem [23]. A surrogate-assisted evolutionary algorithm was proposed in paper [24]. The single-objective feature selection problem is solved by decomposing the large-scale original problem into several small subproblems and establishing a surrogate-assisted model for each subproblem. In paper [25], a hybrid version of Simulated Normal Distribution Optimizer with Simulated Annealing is proposed for feature selection which uses Simulated Annealing as a local search to achieve higher classification accuracy. Whale Optimization Algorithm is used for feature selection of high-dimensional data based on spatial boundary strategy in [26]. The time-varying transfer function was used on Binary Dragonfly Algorithm for feature selection to balance the exploration and exploitation and obtained excellent results [27]. Hybrid feature selection based on Chi-square and binary Particle Swarm Optimization algorithm was designed and applied for Arabic email authorship analysis in 2021 [28].

If the fitness value of the single-objective feature selection algorithms is set to the weighted sum, and the weights are usually predetermined, then the algorithms are not flexible enough. For the algorithms only consider accuracy/error rate, the sparsity of selected features is ignored. As a consequence, like most engineering and scientific problems in practice, feature selection can also be regarded as a multi-objective optimization problem [29, 30]. Multi-objective optimization algorithms are usually to optimize multiple conflicting objectives simultaneously [31, 32]. Evolutionary multi-objective optimization algorithms have gained popularity in the past decade and beyond [33, 34]. The multi-objective feature selection problems mainly optimize two objectives: maximizing the classification accuracy and minimizing the number of features. For the multi-objective feature selection algorithms, it can provide a series of relative optimal solutions for users to choose, instead of a single solution. There are relatively few studies on multi-objective feature selection problem. Two variants using the angle competitive mechanism and Euclidean distance competitive mechanism of differential evolution (DE) algorithm are proposed in paper [35], and are applied to the feature selection problem. In [36], a binary multi-objective grey wolf algorithm was proposed and a wrapper-based Artificial Neural Network is used to assess the classification performance of the selected features for the multi-objective feature selection. Paper [37] studies

a new multi-objective feature selection approach based on the Binary DE with self-learning for solving feature selection and achieves a trade-off between local exploitation and global exploration. A fast multi-objective evolutionary feature selection algorithm is proposed in [38] by embedding an improved Artificial Bee Colony algorithm [39] based on the particle update model. The authors of paper [40] combine binary encoding with real value encoding to utilize the advantages of Genetic Algorithm and Direct Multi-Search to solve multi-objective feature selection of unbalanced production data and obtain significantly good search performance. A multi-objective evolutionary algorithm is proposed for feature selection in learning to rank in paper [41] and get excellent performance.

In large-scale data, because of the large number of features, the efficiency of traditional feature selection algorithms is reduced or even cannot be processed. However, there are many application scenarios of Large-scale Sparse Multi-Objective Feature Selection Problems (LSMFSPs) in real life. For example, in the field of text classification [5], the number of words commonly used in everyday life is about order of magnitude 10^4. In the field of image processing [42], if the image features are pixels, the number of features of a picture with a resolution of 1024×1024 will easily reach the order of 10^6. Biological omics data also usually have large-scale features: DNA microarray chip can detect and obtain thousands of gene expression values at the same time [43]; There are hundreds of protein mass spectrum peaks and related biomarkers in protein mass spectrum data [44]; there are often hundreds of chromatographic peaks in metabolic mass spectrometry data.

Large-scale data usually have a lot of redundancy and require special research. However, there are few studies that are specifically used to deal with LSMFSPs. LSMFSPs is one of Large-scale Multi-Objective Problems (LMOPs). Evolutionary algorithms for solving LMOPs can generally be divided into three categories: the divide-and-conquer, dimensionality reduction, and enhanced search-based approaches. A similar method based on random decomposition is proposed in [45], which improves the MOEA/D framework to enable it to handle LMOPs. Paper [46] proposes a customized evolutionary algorithm based on decision variable clustering method. It uses k-means to divide decision variables into convergence-related variables and diversity-related variables, and optimizes the two variables, respectively. A general, theoretically grounded yet simple approach was proposed in paper [47], which can scale current derivative-free multi-objective algorithms to the high-dimensional non-convex multi-objective functions with low effective dimensions, using random embedding. Based on dimension reduction, it transforms the original decision space into a low-dimensional subspace. In paper [48], an enhanced large-scale multi-objective algorithm based on search is proposed, which incorporates a new solution generator with an external archive, thus forcing the search toward different subregions of the Pareto front using a dual local search mechanism. Paper [49] proposes a novel multi-objective large-scale cooperative co-evolutionary algorithm for three-objective feature selection, and it designs a cooperative searching framework for seeking the optimal feature subset efficiently and effectively.

Paper [50] puts forward the concept of Large-scale Sparse Multi-objective Optimization Problems (LSMOPs) in 2019, which means that most decision variables of these solutions are zero. In this paper, an evolutionary algorithm named SparseEA is designed, which solves the LSMOPs problem by constructing sparse solutions. In particular, LSMFSPs are specific applications of LSMOPs. The experimental results show that SparseEA performs excellent in solving LSMOPs. At present, there are few papers dedicated to dealing with LSMOPs. The authors of paper [51] uses two unsupervised neural networks, a restricted Boltzmann machine and a denoising autoencoder to learn a sparse distribution and a compact representation of the decision variables for LSMOPs. The proposed algorithm in paper [52] suggests an evolutionary pattern mining approach to detect the maximum and minimum candidate sets of the nonzero variables in the Pareto optimal solutions, and uses them to limit the dimensions in generating offspring solutions for LSMOPs. An improved SparseEA was proposed in paper [53] to enhance the connection between real variables and binary variables within the two-layer encoding scheme with the assistance of variable grouping techniques for LSMOPs.

Therefore, this manuscript proposes an enhanced SparseEA algorithm based on ReliefF with difference operators for solving the LSMFSPs. The main contributions of this paper are concluded as follows:

1. It combines a filtering feature selection method with SparseEA. ReliefF was used to calculate the weights of features, with unimportant features being removed first.
2. Combine the weights calculated by ReliefF with Scores of SparseEA to guide the evolution process. Meanwhile, an adaptive score update strategy is designed for solving the *Scores* of decision variables remains constant throughout all iteration.
3. Difference operators of DE are introduced into SparseEA to increase the diversity of solutions and help the algorithm jump out of the local optimal solution.
4. SparseEA with hybrid difference operators is proposed to balance the exploration and exploitation.
5. The proposed algorithm is compared with the excellent algorithms proposed in recent 3 years to solve the LSMFSPs. The experimental results verify the superiority of the proposed algorithm.

The rest of the paper is organized as follows. "SparseEA" shows the original SparseEA algorithm. "SparseEA based on reliefF" depicts the proposed SparseEA based on ReliefF strategy. It describes the details of the SparseEA with binary difference operators in "RA-SparseEA with difference operator". "Experiments" is experimental results and analysis. "Conclusion" depicts the main work of the paper and gives some suggestions for further work.

SparseEA

SparseEA is an evolutionary algorithm for solving large-scale sparse multi-objective optimization problems. In SparseEA, a solution x consists of two components, i.e., a real vector (denoted as Dec) can record the best decision variables found so far, and a binary vector (denoted as $Mask$) can record the decision variables that should be set to zero. For instance, the number of variable is $D = 5$, $Dec = (0.5, 0.3, 0.2, 0.8, 0.1)$, and $Mask = (1, 0, 0, 1, 0)$. Then, x can be obtained by Eq. (1). Therefore, the final solution is $x = (0.5, 0, 0, 0.8, 0)$

$$(x_1, x_2, \ldots, x_D) = (Dec_1 \times Mask_1, Dec_2 \times Mask_2, \ldots, Dec_D \times Mask_D). \quad (1)$$

The framework of SparseEA is very similar to NSGAII which is shown in Algorithm 1. However, the strategies to generate the initial population and offsprings are different from NSGAII, and those can ensure the sparsity of the generated solutions. To begin with, the $Scores$ of each variable are got by the fitness value and the population P with size N is initialized, which is described in Algorithm 2 particularly. After that, fast non-dominated ordering and crowding calculation are performed on P. In the main loop, the binary tournament selection is used to obtain $2N$ parents solutions. Then, N offsprings are generated from $2N$ parents solutions by the new genetic operation which is shown in detail in Algorithm 3. At the last, the environmental selection is executed based on front number and crowding distance.

The initialization process of SparseEA includes calculate the $Scores$ of variables and generate the initial population. In the first step, for real variables, a $D \times D$ random matrix is generated as Dec and a $D \times D$ identity matrix is set to $Mask$. The solutions can be got by Eq. (1). Then, the fitness values of each solution can be calculated and the non-dominated sorting can be executed to obtain the $Scores$ of each variable. However, for the binary problem, the Dec is a $D \times D$ matrix of ones and the $Mask$ is a also $D \times D$ identity matrix. Then, the solutions are also a $D \times D$ identity matrix which is equal to the $Mask$. For the ith solution x_i, all elements are 0 except for the ith element is 1. Therefore, the fitness of x_i can be viewed as the importance of the ith variable. In SparseEA, the Pareto front number of x_i is used as the $Scores$. In the next step, a initial population can be got by a $N \times D$ Dec and a $N \times D$ $Mask$. The Dec is uniformly randomly generated for the real variables, while, it is a matrix of ones for the binary problem. For every solution of $Mask$, $rand() \times D$ times binary tournament selection is performed on the variables

Algorithm 1 Framework of the SparseEA

Require: N(population size)
1: $[P, Scores] \leftarrow$ Initialization(N); \\ Algorithm 2
2: $F = [F_1, F_2, ...] \leftarrow$ Do non-dominated sorting on P;
3: $CrowdDis \leftarrow CrowdingDistance(F)$;
4: **while** $termination\ criterion\ not\ fulfilled$ **do**
5: $P' \leftarrow$ Select $2N$ parents via binary tournament selection according to the non-dominated front number and CrowdDis of solutions in P;
6: $P \leftarrow P \cup GeneticOperator(P', Scores)$;\\ Algorithm 3
7: Execute environmental selection base on front number and crowding distance;
8: **end while**
9: **return** P(final population).

Algorithm 2 Initialization strategy of SparseEA

Require: N(population size)
1: \\ Calculate the scores of variables
2: D $\leftarrow$ Number of decision variables;
3: **if** *the decision variables are real numbers* **then**
4: $Dec \leftarrow D \times D$ random matrix;
5: **else***the decision variables are binary numbers*
6: $Dec \leftarrow D \times D$ matrix of ones;
7: **end if** $Mask \leftarrow D \times D$ identity matrix;
8: $Q \leftarrow$A population whose i-th solution is generated by the i-th rows of Dec and $Mask$;
9: $[F1, F2, ...] \leftarrow$ Do non-dominated sorting on Q;
10: **for** $i = 1$ to D **do**
11: $Scores_i \leftarrow k, s.t.\ Q_i \in F_k$; \\ Q_i denotes the i-th solution in Q
12: **end for**
13: \\ Generate the initial population
14: **if** the decision variables are real numbers **then**
15: $Dec \leftarrow$ Uniformly randomly generate the decision variables of N solutions;
16: **else** the decision variables are binary numbers
17: $Dec \leftarrow N \times D$ matrix of ones;
18: **end if**
19: $Mask \leftarrow N \times D$ matrix of zeros;
20: **for** $i = 1$ to N **do**
21: **for** $j = 1$ to $rand() \times D$ **do**
22: $[m, n] \leftarrow$ Randomly select two decision varaibles;
23: **if** $Scores_m < Scores_n$ **then**
24: Set the m-th element in the i-th binary vector in Mask to 1;
25: **else**
26: Set the n-th element in the i-th binary vector in Mask to 1;
27: **end if**
28: **end for**
29: **end for**
30: $P \leftarrow$A population whose i-th solution is generated by the i-th rows of Dec and $Mask$.
31: **return** P(initial population),$Scores$(scores of decision variables).

and the variable with lower $Scores$ value will be set to 1. Thereby, at most $rand() \times D$ variables are set to 1 for a solution. This strategy ensures the sparsity of the population.

Algorithm 3 Genetic Operator of SparseEA

Require: P'(parent individuals), Scores(scores of decision variables).
1: $O \leftarrow Null$;
2: **while** P' is not empty **do**
3: $[p, q] \leftarrow$ Randomly select two parents from P' and remove them from P'; \\ Generating the mask of offspring o;
4: $o.Mask \leftarrow p.Mask$;
5: \\ Crossover of mask
6: **if** $rand() < 0.5$ **then**
7: Randomly select two decision variables from the nonzero elements in $p.Mask \cap \overline{q.Mask}$;
8: Set the element with bigger fitness in $o.Mask$ to 0;
9: **else**
10: Randomly select two decision variables from the nonzero elements in $\overline{p.Mask} \cap q.Mask$;
11: Set the element with smaller fitness in $o.Mask$ to 1;
12: **end if**
13: \\ Mutation of mask
14: **if** $rand() < 0.5$ **then**
15: Randomly select two decision variables from the nonzero elements in $o.Mask$;
16: Set the element with bigger fitness in $o.Mask$ to 0;
17: **else**
18: Randomly select two decision variables from the nonzero elements in $\overline{o.Mask}$;
19: Set the element with smaller fitness in $o.Mask$ to 1;
20: **end if**
21: \\ Generate the Dec of offspring o
22: **if** the decision variables are real numbers **then**
23: $o.Dec \leftarrow$Perform simulated binary crossover and polynomial mutation based on $p.Dec$ and $q.Dec$;
24: **else**
25: $o.Dec \leftarrow$Vector of ones;
26: **end if**
27: $O \leftarrow O \cup o$;
28: **end while**
29: **return** O(offspring individuals).

The genetic operator is another key component that makes SparseEA different from NSGAII. As shown in Algorithm 3, it is composed of generating the $Mask$ of offsprings and generating the Dec of offsprings. The SparseEA adopts the existing genetic operators for the Dec of offsprings. To be specific, if the decision variables are real numbers, the Dec is got by performing simulated binary crossover and polynomial mutation. And it is simply set to matrix of ones if the decision variables is binary. The main contribution of the genetic operator of SparseEA is the crossover and mutation operator of binary $mask$. Two parents p and q are randomly selected from P' to generate an offspring o each time. Then, the binary vector mask of o is first set to the same to that of p. The crossover of mask is to select one variable which is different in $p.Mask$ and $q.Mask$ to flip. In detail, a random number is used to determine the variable is selected from the zero elements or the nonzero elements in the binary vector $Mask$ with the same probability. If the random number is less than 0.5, two decision variables are randomly selected from $p.Mask \cap \overline{q.Mask}$ and the element with bigger fitness is set to 0. Else, two decision variables are chosen from $\overline{p.Mask} \cap q.Mask$ and the element with bigger fitness is flipped. In the mutation operator, it is also one variable is selected to be flipped. Similarly, randomly select two decision variables from the nonzero elements in $o.Mask$ or $\overline{o.Mask}$, and the element with more contribution is set to 1 or with smaller fitness is set to 0.

SparseEA based on reliefF

It can be observed from line 11 in Algorithm 2 that the $Scores$ in SparseEA is the non-domination Pareto front number of the corresponding solution. For feature selection problem, the $Score$ of the ith feature is the front number of the solution where only the ith element is 1 and the rest are all 0. The fitness of the solution for feature selection problem consists of sparsity and error rate. Since the sparsity of each solution is $1/D$, the Pareto front number of the solution is uniquely determined by the error rate. That is, the $Score$ of the ith feature is only decided by the error rate in SparseEA. What's more, this $Scores$ value remain constant throughout all iteration. Due to the correlation between features, calculating the fitness of a single feature only in the initial stage cannot well reflect the importance of features. However, computing all possible combinations of features is a NP-hard problem. Therefore, in this manuscript, the fitness values of excellent and poor individuals in each iteration are used to update the $Scores$ of features.

In addition, the fitness value of the solution can only reflect the importance of the features from one view. Many traditional feature selection methods evaluate the importance of features based on different criteria. Therefore, in this section, we combine the traditional feature selection method with SparseEA algorithm. Relief is a filtering feature selection algorithm that updates feature weights by looking for the nearest neighbour of each sample. It evaluates the correlation and redundancy of features by calculating adjacent samples of the same and different classes. However, the Relief algorithm was designed to handle only dichotomies, so Kononenko expanded on Relief in 1994 to design ReliefF algorithm that could handle multiple types of data with better performance. The ReliefF algorithm determines the size of feature weights in each sample according to certain weight measures between samples in the original sample set, similar samples, and different samples. Then, according to certain evaluation criteria to distinguish the strong correlation, weak correlation, and no correlation of the sample features. For sparse large-scale feature selection problem, there are a lot

of redundant features. Therefore, in this manuscript, ReliefF algorithm is used first to eliminate unimportant features and build feature subsets. At the same time, the number of features is reduced and the running speed of the algorithm can be accelerated. To some extent, this can offset the time spent in calculating the Relief algorithm. Furthermore, the weights calculated by ReliefF algorithm are combined with $Scores$ of SparseEA to guide the evolution process.

Algorithm 4 Framework of the RA-SparseEA for Feature Selection

Require: N (population size), FE (number of consumed function evaluations),$MaxFE$ (maximum number of function evaluations).
1: $W_{rlf} \leftarrow$ Do ReliefF algorithm to obtain the $Weights$ of each feature;
2: Remove features with low $Weights$; $\leftarrow$ The number of features is reduced from D to D'
3: $[P, Scores = [s_1, ..., s_{D'}]] \leftarrow$ Initialization(N)\\ Algorithm 2, $Scores$ is set to fitness values;
4: $s_i = s_i + \tau \leftarrow$ For the good features in W_{rlf};
5: $s_i = s_i - \tau \leftarrow$ For the bad features in W_{rlf};
6: $[F_1, F_2, ...] \leftarrow$ Do non-dominated sorting on P;
7: $CrowdDis \leftarrow CrowdingDistance(F)$;
8: **while** $termination\ criterion\ not\ fulfilled$ **do**
9: $P' \leftarrow$ Select parents via binary tournament selection according to the non-dominated front number and CrowdDis of solutions in P;
10: $P \leftarrow P \cup GeneticOperator(P', Scores)$; \\ Algorithm 3
11: Delete duplicated solutions from P;
12: $[F_1, F_2, ...] \leftarrow$ Do non-dominated sorting on P;
13: $\varepsilon = t \times \alpha \times (FE/MaxFE)$; t is the number of times a feature is selected in all non-dominated solutions;
14: $s_i = s_i + \varepsilon \leftarrow$ For the features selected in F_1;
15: $s_i = s_i - \varepsilon \leftarrow$ For the features selected in F_{last};
16: $CrowdDis \leftarrow CrowdingDistance(F)$;
17: $k \leftarrow argmin_i \|F_1 \cup ... \cup F_i\| \geq N$;
18: Delete $\|F_1 \cup ... \cup F_k\| - N$ solutions from F_k with the smallest CrowdDis;
19: $P \leftarrow F_1 \cup ... \cup F_k$;
20: **end while**
21: **return** P(final population).

The framework of the RA-SparseEA for feature selection is shown in Algorithm 4. First of all, the ReliefF algorithm is first executed to get the weights of the feature W_{rlf}. Remove features with low W_{rlf} values, and the number of features is reduced from D to D'. In this manuscript, we set $D' = 0.5 * D$ for datasets with more than 1000 features; otherwise, $D' = D$. Then, same as SparseEA, Algorithm 2 is used to initialize population in D' dimension, and the $Scores = [s_1, ..., s_{D'}]$ are obtained. The difference is that the $Scores$ is set to not the Pareto front number but the fitness values. Then, the W_{rlf} is used to guide the updating of the $Scores$. For the good features in W_{rlf}, the scores s_i are add a value τ. And for the poor features, $s_i = s_i - \tau$ is calculated. After that, fast non-dominated ordering and crowding calculation are performed on P. In the main loop, selecting parents' individuals P' and genetic operator is the same as SparseEA. In the subsequent environmental selection stage, delete duplicated solutions and do non-dominated on P first. The features selected in every non-dominated solution are considered to have higher $Scores$, while the features selected in the solutions of last Pareto front should have lower $Scores$. Then, $s_i = s_i + \varepsilon$ and $s_i = s_i - \varepsilon$ is done for the features selected in F_1 and F_{last}, respectively. Where F_{last} is the last Pareto front. To balance the exploration and exploitation at different evolution stages, ε is designed as a linearly increasing function which is shown in Eq. (2), where t is the number of times a feature is selected in all non-dominated solutions, α is the step parameter which usually set to 0.01, FE is the number of consumed function evaluations, and $MaxFE$ is the maximum number of function evaluations

$$\varepsilon = t \times \alpha \times (\text{Iter}/\text{MaxIter}). \tag{2}$$

The rest are the same as the environment selection in SparseEA and will not be repeated.

RA-SparseEA with difference operator

In this section, the SparseEA with difference operator (RA-SparseDO) will be described in detail. As shown in the previous section, the RA-SparseEA reverses only one element per particle in mutation and crossover operations, respectively, in each turn. This limits the diversity of the population and makes the algorithm easily fall into the local optimal solution. What's more, the $Mask$ of offspring is first assigned to that of one parent ($o.Mask = p.Mask$); therefore, the parent p has a great influence on the offspring, while the parent q not. The difference operator proposed in DE can obtain genetic information from multiple parents [54–56].

Feature selection is a binary problem, and there are some important binary variants of the DE. In paper [57], sigmoid transfer function is used to convert the mutation operator into binary form. A new Taper-shaped transfer function was proposed and used to transform the continuous DE algorithm into binary form in [58]. Paper [59] makes use of binary operators such as xor, and, or, and not operators to generate trial solutions. An adaptive quantum-inspired DE was designed in [60] for solving 0–1 Knapsack Problem. Pampara et al. [61] presented angle-modulated DE, which uses angle modulation to evolve the coefficients of the trigonometric function, thus allowing mapping from continuous space to binary space. For multi-objective binary algorithm, scholars also have done some excellent work. A binary differential evolution algorithm with a self-learning strategy for multi-objective feature selection problems was designed in paper [37]. Non-dominated sorting binary differential evolution was proposed for cascading failures protection in complex networks in Ref. [62]. Paper [63] proposed a binary version of generalized

differential evolution for multi-label feature selection based on majority voting of solutions and opposition-based learning. There are not many studies on using binary differential evolution to solve large-scale problems. A new self-adaptive binary variant of a differential evolution algorithm based on measure of dissimilarity was proposed in [64], and used for solving high-dimensional knapsack problems.

Therefore, DE is effective for solving binary problems. Thus, this manuscript attempts to introduce the difference operator into SparseEA to increase the diversity of solutions for solving LSMFSPs. SparseEA selects parents via binary tournament selection and then produces offsprings. Then, we introduce four commonly used difference operators of DE ("DE/rand/1", "DE/rand/2", "DE/best/1", and "DE/beset/2") to SparseEA. Table 1 shows the details of the four difference operators. In the DE algorithm, mutation is done first, and then is crossover, which is different from GA. This manuscript takes "DE/rand/2" and "DE/beset/2" as examples to introduce the difference operator for SparseEA in detail.

Algorithm 5 Difference Operator of SparseDO(Rand/2)

Require: $P'(6N$ parent individuals), Scores(scores of decision variables).
1: $O \leftarrow Null$;
2: **while** P' is not empty **do**
3: \\ Differential mutation of mask
4: $[x_{r1}, x_{r2}, x_{r3}, x_{r4}, x_{r5}] \leftarrow$ Randomly select five particles from P' and remove them from P';
5: $o.Mask = x_{r1}.Mask$; \\ Generating the $Mask$ of offspring o;
6: $R1 = XOR(x_{r2}.Mask, x_{r3}.Mask)$; \\ Do XOR on the mask of x_{r2} and x_{r3};
7: $R2 = XOR(x_{r4}.Mask, x_{r5}.Mask)$; \\ Do XOR on the mask of x_{r4} and x_{r5};
8: $R = R1 + R2$; \\ Do AND on $R1$ and $R2$;
9: **for** each nonzero element in R **do**
10: **if** $rand() < 0.5$ **then**
11: Set the element in $o.Mask$ to 0;
12: **else**
13: Set the element in $o.Mask$ to 1;
14: **end if**
15: **end for**
16: \\ Crossover of mask
17: $[p] \leftarrow$ Randomly select one parent from P' and remove it from P';
18: **if** $rand() < 0.5$ **then**
19: Randomly select two decision variables from the nonzero elements in $o.Mask \cap \overline{p.Mask}$;
20: Set the element with bigger fitness in $o.Mask$ to 0;
21: **else**
22: Randomly select two decision variables from the nonzero elements in $\overline{o.Mask} \cap p.Mask$;
23: Set the element with smaller fitness in $o.Mask$ to 1;
24: **end if**
25: \\ Generate the Dec of offspring o
26: $o.Dec \leftarrow$Vector of ones;
27: $O \leftarrow O \cup \mathbf{o}$;
28: **end while**
29: **return** O(offspring individuals).

Algorithm 5 describes the pseudo-code of "DE/rand/2" difference operator of SparseDO. First of all, randomly select five particles $x_{r1}, x_{r2}, x_{r3}, x_{r4}, x_{r5}$ from P' and remove them from P'. The $Mask$ of the offspring o is first set to that of the x_{r1}. Then, the elements with different values in x_{r2} and x_{r3} are marked in $R1$. Similarly, the elements with different values in x_{r4} and x_{r5} are marked in $R2$. Then, R can be obtained by $R = R1 + R2$. Calculate the marked elements in R, and then, the candidate variables need to change are chosen. In paper [59], the OR operator are used on the produced R and x_1. However, in this manuscript, to increase the sparsity and randomness of the offspring, the nonzero elements in R are directly replace the elements in $o.Mask$ according to a random number. That is, the random number is used to determine the candidate variables of $o.Mask$ is set to 0 or 1 with the same probability for each nonzero element in R. After mutating, crossover operations are performed to determine whether the mutated gene will eventually be passed on to the offspring. In this algorithm, one particle is randomly selected from P' as the parent in the crossover step. Two decision variables are randomly selected from the nonzero elements in $o.Mask \cap \overline{p.Mask}$, and the one with bigger fitness is set to 0; or selected from $\overline{o.Mask} \cap p.Mask$, and the one with smaller fitness is set to 1. The Dec of the offspring are generated in the same way as SparseEA. Due to the feature selection is a binary problem, the Dec of offspring is set to vector of ones.

Similarly, Algorithm 6 describes the "DE/best/2" difference operator of SparseDO. The main difference is the line 5 and line 6 in Algorithm 6. Execute environmental selection based on front number and crowding distance, and then, the mask of the offspring is initialized through a randomly select non-dominant solution. The rest are similar to Algorithm 5 and will not be repeated.

The difference operator "DE/rand/2" is good for exploration and "DE/best/2" is good for exploitation. Then, to balance the exploration and exploitation, SparseEA with hybrid difference operators based on ReliefF for feature selection which named RA-HSparseDO was described in Algorithm 7. It can be seen from Line 11 to 15 in Algorithm 7, do the Algorithm 5 in the early half iteration, while do the Algorithm 6 in the later half iteration.

Experiments

Experiments' settings

In this section, some experiments are designed to test the performance of the proposed algorithms. All the experiments are conducted on the evolutionary multi-objective optimization platform PlatEMO [65].

Table 1 The different mutation schemes of DE

Number	Scheme	Equation
1	DE/rand/1	$v_m = x_{r1} + f * (x_{r2} - x_{r3})$
2	DE/rand/2	$v_m = x_{r1} + f * (x_{r2} - x_{r3}) + f * (x_{r4} - x_{r5})$
3	DE/best/1	$v_m = x_{gbest} + f * (x_{r1} - x_{r2})$
4	DE/best/2	$v_m = x_{gbest} + f * (x_{r1} - x_{r2}) + f * (x_{r3} - x_{r4})$

Algorithm 6 Difference Operator of SparseDO(Best/2)

Require: $P'(5N$ parent individuals), Scores(scores of decision variables).
$O \leftarrow Null$;
while P' is not empty **do**
  \\ Differential mutation of mask
  $[x_{r1}, x_{r2}, x_{r3}, x_{r4}] \leftarrow$ Randomly select three particles from P' and remove them from P';
  Execute environmental selection base on front number and crowding distance;
  $o.Mask = x_{F1}.Mask$; \\ Randomly select a non-dominant solution x_{F1};
  $R1 = XOR(x_{r1}.Mask, x_{r2}.Mask)$; \\ Do XOR on the mask of x_{r1} and x_{r2};
  $R2 = XOR(x_{r3}.Mask, x_{r4}.Mask)$; \\ Do XOR on the mask of x_{r3} and x_{r4};
  $R = R1 + R2$; \\ Do AND on $R1$ and $R2$;
  for each nonzero element in R **do**
    if $rand() < 0.5$ **then**
      Set the element in $o.Mask$ to 0;
    else
      Set the element in $o.Mask$ to 1;
    end if
  end for
  \\ Crossover of mask
  $[p] \leftarrow$ Randomly select one parent from P' and remove it from P';
  if $rand() < 0.5$ **then**
    Randomly select two decision variables from the nonzero elements in $o.Mask \cap \overline{p.Mask}$;
    Set the element with bigger fitness in $o.Mask$ to 0;
  else
    Randomly select two decision variables from the nonzero elements in $\overline{o.Mask} \cap p.Mask$;
    Set the element with smaller fitness in $o.Mask$ to 1;
  end if
  \\ Generate the Dec of offspring o
  $o.Dec \leftarrow$Vector of ones;
  $O \leftarrow O \cup \mathbf{o}$;
end while
return O(offspring individuals).

Algorithm 7 Framework of the RA-HSparseDO(Rand/2-Best/2) for Feature Selection

Require: N(population size), FE(number of consumed function evaluations),$MaxFE$(maximum number of function evaluations).
$W_{rlf} \leftarrow$ Do ReliefF algorithm to obtain the $Weights$ of each feature;
Remove features with low $Weights$; $\leftarrow$ The number of features is reduced from D to D'
$[P, Scores = [s_1, ..., s_{D'}]] \leftarrow$ Initialization(N)\\ Algorithm 2, $Scores$ is set to fitness values;
$s_i = s_i + \tau \leftarrow$ For the good features in W_{rlf};
$s_i = s_i - \tau \leftarrow$ For the bad features in W_{rlf};
$[F_1, F_2, ...] \leftarrow$ Do non-dominated sorting on P;
$CrowdDis \leftarrow CrowdingDistance(F)$;
while $termination\ criterion\ not\ fulfilled$ **do**
  $P' \leftarrow$ Select parents via binary tournament selection according to the non-dominated front number and CrowdDis of solutions in P;
  if $FE < 0.5 \times MaxFE$ **then**
    $P \leftarrow P \cup DifferenceOperator(Rand/2)$; \\ Algorithm 5
  else
    $P \leftarrow P \cup DifferenceOperator(Best/2)$; \\ Algorithm 6
  end if
  Delete duplicated solutions from P;
  $[F_1, F_2, ...] \leftarrow$ Do non-dominated sorting on P;
  $\varepsilon = t \times \alpha \times (FE/MaxFE)$; t is the number of times a feature is selected in all non-dominated solutions;
  $s_i = s_i + \varepsilon \leftarrow$ For the features selected in F_1;
  $s_i = s_i - \varepsilon \leftarrow$ For the features selected in F_{last};
  $CrowdDis \leftarrow CrowdingDistance(F)$;
  $k \leftarrow argmin_i \| F_1 \cup ... \cup F_i \| \geq N$;
  Delete $\| F_1 \cup ... \cup F_k \| - N$ solutions from F_k with the smallest CrowdDis;
  $P \leftarrow F_1 \cup ... \cup F_k$;
end while
return P(final population).

Datasets' description

The scikit-feature repository is selected for LSMFSPs, which is an open-source feature selection repository developed at Arizona State University (https://jundongl.github.io/scikit-feature/index.html). It serves as a platform for facilitating feature selection application, research, and comparative study. The dataset name, number of instances, number of features, number of classes, and the keyword description are shown in Table 2. These datasets include 4 face image data (ORL, warpAR10P, warpPIE10P, and Yale), 6 biological data (lung, lung-discrete, lymphoma, nci9, Prostate-GE, and TOX-171), and 2 text data (BASEHOCK and RELATHE). It can be seen that the dataset with the largest number of features is Prostate-GE, and the number of features is up to 9712. The least number of features is lung-discrete, which still has 325 features.

To verify the effect of the proposed algorithm on small-scale datasets, UCI datasets with less than 500 features (http://archive.ics.uci.edu/ml/index.php) are selected in this manuscript. Table 3 details the dataset name, number of instances, number of features, number of classes, and the

Table 2 The datasets of scikit-feature repository

Dataset name	Instances	Features	Classes	Keywords
BASEHOCK	1993	4862	2	Discrete, binary
RELATHE	1427	4322	2	Discrete, binary
Lung	203	3312	5	Continuous, multi-class
Lung-discrete	73	325	7	Discrete, multi-class
Lymphoma	96	4026	9	Discrete, multi-class
nci9	60	9712	9	Discrete, multi-class
ORL	400	1024	40	Continuous, multi-class
Prostate-GE	102	5966	2	Continuous, binary
TOX-171	171	5748	4	Continuous, multi-class
warpAR10P	130	2400	10	Continuous, multi-class
warpPIE10P	210	2420	10	Continuous, multi-class
Yale	165	1024	15	Continuous, multi-class

keyword description of these datasets, i.e., iris, Lung Cancer, Person-Classification, MUSK1, heart, ionosphere, Parkinson, and COVID-19 Surveillance. It can be found that data types include integer, real, and category.

Stopping condition and performance metrics

For the sake of efficient and fair experiments, the maximum number of function evaluations is adopted as the stopping criteria.

There are two objectives used in the manuscript for multi-objective feature selection problems, that is, the validation error and the ratio of selected features. Since the Pareto fronts of the MOPs in applications are unknown, the hypervolume (HV) is adopted to measure each obtained solution set. The HV index was first proposed by zitzler et al., and it represents the volume of the hypercube surrounded by the individuals and reference points in the solution set in the target space. The reference point for calculating HV in this manuscript is set to (1,1).

Experiment on SMOP test suite

In this section, the performance of SparseDO in generating offspring solutions of real variables will be tested. The sparse multi-objective test suite [50] is adopted in this experiment, which is widely used in assessing the performance of existing multi-objective evolutionary algorithms in obtaining sparse Pareto optimal solutions. The test suite contains eight benchmark problems SMOP1–SMOP8 with scalable number of decision variables. In the experiments, the number of objectives of these problems is set to 2, and the number of decision variables is set to 500, 1000, and 1500.

The algorithms used in this experiment are the original SparseEA, SparseDO with "DE/best/1" which denoted as SparseDO (Best1), SparseDO with "DE/best/2" which denoted as SparseDO (Best2), SparseDO with "DE/rand/1" which denoted as SparseDO (Rand1), SparseDO with "DE/rand/2" which denoted as SparseDO (Rand2), Hybird SparseDO with "DE/best/1" and "DE/rand/1" which denoted as HSparseDO (Rand1Best1), and Hybird SparseDO with "DE/best/2" and "DE/rand/2" which denoted as HSparseDO (Rand2Best2). Each algorithm performs 1000 function evaluations on each function, the population size is set to 10, and the HV index is used to measure the results. The experiments all runs 30 times, and the mean and standard deviation are used to measure the results. The experimental results are shown in Table 4.

The results of the proposed SparseDO (Best1), SparseDO (Best2), SparseDO (Rand1), SparseDO (Rand2), HSparseDO (Rand1Best1), and HSparseDO (Rand2Best2) are compared with those of SparseEA. The number marked in red indicates that the proposed algorithm is better than the original SparseEA. The last row in Table 4 shows the times of the proposed algorithm obtains better results than SparseEA. As can be seen in Table 4, SparseDO (Best1), SparseDO (Best2), SparseDO (Rand1), and SparseDO (Rand2) get better results on 17, 19, 16 and 13 functions than SparseEA, respectively. Hence, the four SparseDO in generating offspring solutions of real variables are effective. The HSparseDO (Rand1Best1) and HSparseDO (Rand2Best2) obtain 19 and 16 better values, respectively. Therefore, the proposed HSparseDO can also improve the performance of the original SparseEA algorithm.

Experiments of diversity

In this section, the effect of binary differential operators on solution diversity will be tested. The most intuitive measure of diversity is the degree of difference between individuals in a population. Since the increase or decrease of individual diversity within a population is caused by the change

Table 3 The datasets of UCI machine learning repository

Dataset name	Instances	Features	Classes	Data types
Iris	150	4	3	Real
Lung cancer	32	56	3	Integer
Person-classification	48	321	15	Real
MUSK1	476	166	2	Integer
Heart	270	13	2	Categorical, real
Ionosphere	351	34	2	Integer, real
Parkinson	240	46	3	Real
COVID-19 Surveillance	14	7	3	Integer

Table 4 The results of the SparseEA and the proposed SparseDO algorithms with different DO on SMOP

Dimension	Datasets	Measure	SparseEA	SparseDO (Best1)	SparseDO (Best2)	SparseDO (Rand1)	SparseDO (Rand2)	HSparseDO (Rand1Best1)	HSparseDO (Rand2Best2)
500	SMOP1	AVG	3.40E-01	3.43E-01	3.59E-01	3.49E-01	3.42E-01	3.52E-01	3.37E-01
		STD	4.10E-02	3.50E-02	3.65E-02	2.94E-02	4.56E-02	3.46E-02	4.13E-02
	SMOP2	AVG	2.63E-01	2.55E-01	2.53E-01	2.47E-01	2.47E-01	2.39E-01	2.76E-01
		STD	3.35E-02	3.59E-02	3.29E-02	4.33E-02	3.67E-02	6.48E-02	3.06E-02
	SMOP3	AVG	3.55E-01	3.40E-01	3.52E-01	3.46E-01	3.42E-01	3.50E-01	3.37E-01
		STD	3.26E-02	3.22E-02	3.97E-02	3.54E-02	6.07E-02	3.14E-02	6.30E-02
	SMOP4	AVG	6.66E-01	6.79E-01	6.97E-01	6.80E-01	6.84E-01	6.81E-01	6.99E-01
		STD	8.27E-02	5.82E-02	3.77E-02	4.80E-02	4.24E-02	6.99E-02	4.44E-02
	SMOP5	AVG	6.33E-01	6.48E-01	6.42E-01	6.37E-01	6.40E-01	6.46E-01	6.15E-01
		STD	5.18E-02	3.96E-02	3.44E-02	5.59E-02	3.08E-02	4.20E-02	5.74E-02
	SMOP6	AVG	6.51E-01	6.52E-01	6.54E-01	6.53E-01	6.39E-01	6.55E-01	6.42E-01
		STD	2.53E-02	3.34E-02	2.77E-02	3.68E-02	3.81E-02	3.19E-02	4.23E-02
	SMOP7	AVG	4.42E-02	4.55E-02	5.01E-02	3.80E-02	4.41E-02	4.54E-02	4.59E-02
		STD	1.46E-02	1.96E-02	1.52E-02	2.22E-02	1.88E-02	1.86E-02	2.06E-02
	SMOP8	AVG	1.87E-05	9.07E-05	5.34E-05	8.25E-05	5.63E-05	4.54E-02	1.57E-04
		STD	8.12E-05	2.13E-04	1.43E-04	1.93E-04	1.75E-04	2.05E-04	2.79E-04
1000	SMOP1	AVG	3.56E-01	3.56E-01	3.61E-01	3.47E-01	3.46E-01	3.48E-01	3.53E-01
		STD	3.47E-02	2.81E-02	2.52E-02	3.21E-02	4.28E-02	3.96E-02	2.40E-02
	SMOP2	AVG	2.53E-01	2.60E-01	2.66E-01	2.63E-01	2.56E-01	2.74E-01	2.67E-01
		STD	4.28E-02	3.48E-02	3.27E-02	2.92E-02	3.82E-02	1.88E-02	2.91E-02
	SMOP3	AVG	3.33E-01	3.41E-01	3.46E-01	3.38E-01	3.43E-01	3.48E-01	3.40E-01
		STD	5.96E-02	4.67E-02	3.81E-02	4.64E-02	4.02E-02	3.83E-02	4.17E-02
	SMOP4	AVG	6.80E-01	6.88E-01	6.98E-01	7.07E-01	7.04E-01	7.04E-01	6.97E-01
		STD	4.36E-02	6.57E-02	3.89E-02	2.31E-02	3.11E-02	3.48E-02	5.84E-02
	SMOP5	AVG	6.26E-01	6.43E-01	6.26E-01	6.04E-01	6.48E-01	6.38E-01	6.50E-01
		STD	6.20E-02	5.61E-02	4.98E-02	7.71E-02	3.84E-02	4.36E-02	4.04E-02
	SMOP6	AVG	6.47E-01	6.33E-01	6.56E-01	6.49E-01	6.39E-01	6.50E-01	6.47E-01
		STD	3.64E-02	4.40E-02	2.59E-02	3.40E-02	3.86E-02	2.78E-02	3.57E-02
	SMOP7	AVG	4.92E-02	4.11E-02	4.90E-02	3.50E-02	4.37E-02	3.75E-02	4.73E-02
		STD	1.82E-02	2.26E-02	1.30E-02	2.40E-02	1.66E-02	2.15E-02	1.69E-02
	SMOP8	AVG	6.48E-05	9.87E-05	1.24E-04	7.11E-05	3.68E-05	8.63E-05	9.11E-05
		STD	1.72E-04	2.42E-04	2.40E-04	1.85E-04	1.47E-04	2.21E-04	2.25E-04
1500	SMOP1	AVG	3.35E-01	3.41E-01	3.64E-01	3.48E-01	3.49E-01	3.49E-01	3.55E-01
		STD	4.25E-02	3.94E-02	2.45E-02	3.21E-02	4.00E-02	3.74E-02	2.86E-02
	SMOP2	AVG	2.49E-01	2.51E-01	2.65E-01	2.62E-01	2.67E-01	2.50E-01	2.61E-01
		STD	7.52E-02	3.97E-02	3.47E-02	3.32E-02	2.05E-02	4.24E-02	3.87E-02
	SMOP3	AVG	3.52E-01	3.53E-01	3.65E-01	3.55E-01	3.47E-01	3.63E-01	3.49E-01
		STD	3.49E-02	3.15E-02	1.99E-02	4.28E-02	4.46E-02	2.53E-02	3.33E-02
	SMOP4	AVG	6.70E-01	6.80E-01	6.98E-01	6.93E-01	6.85E-01	6.97E-01	6.95E-01
		STD	7.35E-02	6.13E-02	3.75E-02	4.24E-02	5.01E-02	2.90E-02	4.59E-02
	SMOP5	AVG	6.36E-01	6.35E-01	6.41E-01	6.31E-01	6.35E-01	6.39E-01	6.44E-01
		STD	5.52E-02	6.33E-02	5.07E-02	4.74E-02	6.29E-02	4.66E-02	5.50E-02
	SMOP6	AVG	6.24E-01	6.27E-01	6.51E-01	1.50E+03	6.35E-01	6.35E-01	6.33E-01
		STD	6.03E-02	5.11E-02	4.14E-02	3.80E-02	5.31E-02	4.25E-02	4.60E-02
	SMOP7	AVG	4.13E-02	3.85E-02	3.62E-02	4.25E-02	4.73E-02	4.58E-02	4.25E-02
		STD	2.17E-02	1.93E-02	2.04E-02	1.53E-02	1.75E-02	2.17E-02	2.01E-02
	SMOP8	AVG	1.50E-04	1.13E-04	1.70E-04	5.98E-05	8.49E-05	1.45E-04	4.97E-05
		STD	2.59E-04	2.15E-04	2.93E-04	1.86E-04	2.07E-04	3.00E-04	1.57E-04
Compare with SparseEA			–	17	19	16	13	19	16

of individual gene loci, the diversity in terms of gene loci should also be considered. In general, population diversity can be considered from both macro- and micro-perspectives. Therefore, both individual diversity and genetic diversity are composed of internal diversity and external diversity [66]. Listed below are four definitions about diversity.

Population P is a set of N individuals which can be denoted as $P = [p_1, p_2, ..., p_N]^{\mathrm{T}}$, where $p_i = [p_i^1, p_i^2, ..., p_i^D]$, is a D dimension vector. P is a $N \times D$ matrix, where each row represents an individual and each column represents a gene. The p_i^j is the jth gene value of the ith individual.

Definition 1 The average of the population P is defined as

$$\bar{P} = \frac{1}{N \times D} \sum_{i=1}^{N} \sum_{j=1}^{D} p_i^j. \tag{3}$$

Definition 2 The overall diversity of the population P is defined as

$$\mathrm{DP} = \frac{1}{N \times D} \sum_{i=1}^{N} \sum_{j=1}^{D} [p_i^j - \bar{P}]^2. \tag{4}$$

Definition 3 Genetic internal diversity is defined as

$$\mathrm{DG}_I = \frac{1}{N \times D} \sum_{i=1}^{N} \sum_{j=1}^{D} [p_i^j - \bar{G^j}]^2, \text{ where } \bar{G^j} = \frac{1}{N} \sum_{i=1}^{N} p_i^j. \tag{5}$$

Genetic external diversity is defined as

$$\mathrm{DG}_E = \frac{1}{D} \sum_{j=1}^{D} [\bar{G^j} - \bar{P}]^2. \tag{6}$$

Genetic diversity is defined as

$$\mathrm{DG} = \mathrm{DG}_I + \mathrm{DG}_E. \tag{7}$$

Definition 4 Individual internal diversity is defined as

$$\mathrm{DI}_I = \frac{1}{N \times D} \sum_{i=1}^{N} \sum_{j=1}^{D} [p_i^j - \bar{P}_i]^2, \text{ where } \bar{P}_i = \frac{1}{D} \sum_{j=1}^{D} p_i^j. \tag{8}$$

Individual external diversity is defined as

$$\mathrm{DI}_E = \frac{1}{N} \sum_{i=1}^{N} [\bar{P}_i - \bar{P}]^2. \tag{9}$$

Individual diversity is defined as

$$\mathrm{DI} = \mathrm{DI}_I + \mathrm{DI}_E. \tag{10}$$

In this manuscript, the overall diversity of the population DP, genetic diversity DG, and individual diversity DI are used as the evaluation indicators. The experiments are performed on five scikit-feature datasets, that is lymphoma, warpPIE10P, ORL, lung-discrete, and warpAR10P. Each algorithm is executed for 5000 function evaluations on each dataset and the HV index is used to measure the results. The experiments all runs 30 times, and the mean and standard deviation are used to assess the results. To test the influence of population size on the algorithm, two experiments are conducted in this section, with 100 and 200 individuals, respectively. The experimental results of population diversity, gene diversity, and individual diversity with population size set to 100 are shown in Table 5, where DP is population diversity, DG is genetic diversity, and DI is individual diversity. The rows of TOTAL under the three diversities is the sum of the values of each diversity in five datasets. The value marked in red indicates that the proposed algorithm performs better than the original SparseEA.

Table 5 Results of population diversity, gene diversity, and individual diversity on scikit-feature repository (with population size set to 100)

Diversity	Dimension	Datasets	SparseEA	SparseDO (Best1)	SparseDO (Best2)	SparseDO (Rand1)	SparseDO (Rand2)	HSparseDO (Rand1Best1)	HSparseDO (Rand2Best2)
DP	4026	lymphoma	6.50E-01	4.04E-01	7.13E-01	6.63E-01	1.13E+00	1.12E+00	1.60E+00
	2420	warpPIE10P	2.00E-01	1.45E-01	2.45E-01	2.11E-01	5.40E-01	4.76E-01	8.53E-01
	1024	ORL	3.55E-01	2.94E-01	6.97E-01	4.20E-01	9.46E-01	5.92E-01	1.20E+00
	325	lung-discrete	3.44E-01	2.98E-01	4.93E-01	3.98E-01	7.87E-01	4.59E-01	1.08E+00
	2400	warpAR10P	2.35E-01	1.58E-01	3.89E-01	2.72E-01	7.17E-01	5.60E-01	1.07E+00
TOTAL			1.78E+00	1.30E+00	2.54E+00	1.96E+00	4.12E+00	3.20E+00	5.81E+00
DG	4026	lymphoma	6.08E-01	3.71E-01	6.60E-01	6.12E-01	1.05E+00	1.03E+00	1.49E+00
	2420	warpPIE10P	1.84E-01	1.30E-01	2.22E-01	1.93E-01	4.92E-01	4.35E-01	7.92E-01
	1024	ORL	3.10E-01	2.45E-01	6.15E-01	3.69E-01	8.48E-01	5.34E-01	1.11E+00
	325	lung-discrete	3.07E-01	2.56E-01	4.38E-01	3.54E-01	7.10E-01	4.14E-01	9.89E-01
	2400	warpAR10P	2.16E-01	1.41E-01	3.47E-01	2.48E-01	6.49E-01	5.10E-01	9.92E-01
TOTAL			1.63E+00	1.14E+00	2.28E+00	1.78E+00	3.75E+00	2.93E+00	5.37E+00
DI	4026	lymphoma	5.72E-01	3.63E-01	6.65E-01	6.08E-01	1.05E+00	9.89E-01	1.40E+00
	2420	warpPIE10P	1.77E-01	1.31E-01	2.29E-01	1.93E-01	5.08E-01	4.22E-01	7.70E-01
	1024	ORL	3.19E-01	2.73E-01	6.54E-01	3.92E-01	8.93E-01	5.46E-01	1.11E+00
	325	lung-discrete	3.14E-01	2.85E-01	4.73E-01	3.79E-01	7.54E-01	4.34E-01	1.03E+00
	2400	warpAR10P	2.06E-01	1.43E-01	3.66E-01	2.49E-01	6.78E-01	5.01E-01	9.75E-01
TOTAL			1.59E+00	1.19E+00	2.39E+00	1.82E+00	3.89E+00	2.89E+00	5.28E+00

It can be found from Table 5, the population diversity values of SparseDO (Best2), SparseDO (Rand1), SparseDO (Rand2), HSparseDO (Rand1Best1), and HSparseDO (Rand2Best2) on the five datasets are better than those of SparseEA, so their Total values are also higher. However, the values of SparseDO (Best1) on all five datasets are less than those of SparseEA. Thus, the population diversity of SparseDO (Best1) is worse than SparseEA at a population size of 100. Meanwhile, compared with the values of SparseDO (Best1), the values of SparseDO (Best2) are higher, and compared with the values of SparseDO (Rand1), the values of SparseDO (Rand2) are higher. Moreover, the values of SparseDO (Rand1) are all higher than those of SparseDO (Best1), and the values of SparseDO (Rand2) are all higher than those of SparseDO (Best2). Thus, random difference operators can increase the diversity of solutions better than best difference operators. In addition, the diversity value of HSparseDO (Rand1Best1) is higher than SparseDO (Rand1) and SparseDO (Best1). Similarly, the values of HSparseDO (Rand2Best2) are higher than SparseDO (Rand2) and SparseDO (Best2). Therefore, the HSparseDO algorithms is effective in population diversity. The same conclusion can be reached for genetic diversity and individual diversity.

Table 6 shows the results of population diversity, gene diversity, and individual diversity on scikit-feature datasets with population size set to 200. It can be seen from Table 6 that SparseDO (Best2), SparseDO (Rand1), SparseDO (Rand2), HSparseDO (Rand1Best1), and HSparseDO (Rand2Best2) are effective for increasing the diversity of SparseEA. SparseDO (Best1) is worse than SparseEA with

population size set to 200. Furthermore, comparing Tables 5 and 6, it is not difficult to find that the values in Table 6 are all greater than the corresponding values in Table 5. Therefore, the diversity of solutions can be increased when the population size is large.

Ablation study

There are two components in the algorithm, the relief-based component and the difference operator-based component. In this subsection, the influence of these two components on the performance of the proposed algorithm for LSMFSPs is tested and analyzed.

In this experiment, five datasets of the scikit-feature repository are selected, which are BASEHOCK, RELATHE, lymphoma, nci9, and warpPIE10P. Similarly, to test the influence of population size on the algorithm, two experiments are conducted in this section, with 10 and 30 individuals, respectively. Each algorithm is executed for 5000 function evaluations on each dataset. The population size and function evaluation times of each algorithm are the same. All algorithms in the experiment run 30 times, and AVG and STD are the mean and standard deviation of the results of 30 times, respectively.

Table 7 shows the experimental results of SparseEA, RA-SparseEA, SparseDO (Best1), SparseDO (Best2), SparseDO (Rand1), SparseDO (Rand2), HSparseDO (Rand1Best1), and HSparseDO (Rand2Best2) on scikit-feature repository with population size set to 10. The value marked in red indicates that the performance of the algorithm is better than that of the original SparseEA. The penultimate row of Table 7 (AVG TOTAL) represents the sum of the algorithm's AVG values over the five datasets. The last row (COUNT) of the table indicates the number of times the algorithm is superior to the SparseEA algorithm in five datasets.

It can be found from the last row of Table 7, RA-SparseEA performs better than SparseEA on all five datasets. In particular, the value of RA-SparseEA is 0.083 higher than that of SparseEA in BASEHOCK, 0.061 higher in RELATHE, and 0.147 higher in nci9. Hence, the SparseEA based on ReliefF is effective for LSMFSPs. SparseDO (Best1), SparseDO (Best2), SparseDO (Rand1), SparseDO (Rand2), HSparseDO (Rand1Best1), and HSparseDO (Rand2Best2) are superior to SparseEA on 4, 4, 4, 5, 4, and 4 datasets, respectively. In particular, the value of SparseDO (Rand1) is at least 0.02 higher than that of SparseEA in data sets BASEHOCK, RELATHE, and lymphoma. Therefore, the difference operator is still valid for feature selection problems. From the perspective of AVG TOTAL, all proposed algorithms are superior to the original algorithm, among which RA-SparseEA has the best performance.

The results of SparseEA, RA-SparseEA, SparseDO (Best1), SparseDO (Best2), SparseDO (Rand1), SparseDO (Rand2), HSparseDO (Rand1Best1), and HSparseDO (Rand2Best2) with population size set to 30 are shown in Table 8. Similarly, the performance of RA-SparseEA on all five datasets is better than that of SparseEA, which is the best among all algorithms. In particular, the value of RA-SparseEA is 0.065 higher than that of SparseEA in BASEHOCK, 0.046 higher in RELATHE, and 0.140 higher in nci9. Compared with SparseEA, SparseDO (Best1), SparseDO (Best2), SparseDO (Rand1), SparseDO (Rand2), HSparseDO (Rand1Best1), and HSparseDO (Rand2Best2) perform better on 4, 3, 5, 5, 4, and 4 datasets, respectively. On nci9 dataset, the values of SparseDO (Best1), SparseDO (Best2), SparseDO (Rand2), HSparseDO (Rand1Best1), and HSparseDO (Rand2Best2) are at least 0.02 higher than that of SparseEA. As can be seen from the penultimate row of Table 8, both the difference operator and ReliefF-based strategy can improve the performance of the original SparseEA.

Table 6 Results of population diversity, gene diversity, and individual diversity on scikit-feature repository (with population size set to 200)

Diversity	Dimension	Datasets	SparseEA	SparseDO (Best1)	SparseDO (Best2)	SparseDO (Rand1)	SparseDO (Rand2)	HSparseDO (Rand1Best1)	HSparseDO (Rand2Best2)
DP	4026	lymphoma	1.43E+00	1.05E+00	1.45E+00	1.45E+00	1.70E+00	1.98E+00	1.96E+00
	2420	warpPIE10P	5.05E-01	3.43E-01	8.30E-01	6.34E-01	1.21E+00	1.19E+00	1.42E+00
	1024	ORL	4.90E-01	4.72E-01	9.27E-01	6.89E-01	1.31E+00	9.78E-01	1.43E+00
	325	lung-discrete	6.33E-01	5.55E-01	1.09E+00	8.95E-01	1.56E+00	1.07E+00	1.62E+00
	2400	warpAR10P	5.16E-01	4.36E-01	9.02E-01	7.56E-01	1.38E+00	1.22E+00	1.53E+00
TOTAL			3.58E+00	2.86E+00	5.20E+00	4.42E+00	7.17E+00	6.43E+00	7.97E+00
DG	4026	lymphoma	1.34E+00	9.72E-01	1.38E+00	1.35E+00	1.59E+00	1.98E+00	1.96E+00
	2420	warpPIE10P	4.70E-01	3.16E-01	7.76E-01	5.91E-01	1.14E+00	1.10E+00	1.33E+00
	1024	ORL	4.48E-01	4.22E-01	8.65E-01	6.36E-01	1.23E+00	9.12E-01	1.34E+00
	325	lung-discrete	5.90E-01	5.07E-01	1.02E+00	8.36E-01	1.47E+00	1.00E+00	1.52E+00
	2400	warpAR10P	4.78E-01	3.97E-01	8.37E-01	7.05E-01	1.29E+00	1.12E+00	1.43E+00
TOTAL			3.32E+00	2.61E+00	4.88E+00	4.12E+00	6.71E+00	6.10E+00	7.59E+00
DI	4026	lymphoma	1.27E+00	9.57E-01	1.28E+00	1.33E+00	1.56E+00	1.98E+00	1.96E+00
	2420	warpPIE10P	4.47E-01	3.11E-01	7.85E-01	5.88E-01	1.14E+00	1.06E+00	1.32E+00
	1024	ORL	4.35E-01	4.28E-01	8.76E-01	6.41E-01	1.22E+00	8.99E-01	1.33E+00
	325	lung-discrete	5.81E-01	5.27E-01	1.04E+00	8.52E-01	1.48E+00	1.01E+00	1.53E+00
	2400	warpAR10P	4.59E-01	4.01E-01	8.55E-01	7.07E-01	1.29E+00	1.09E+00	1.42E+00
TOTAL			3.19E+00	2.62E+00	4.84E+00	4.11E+00	6.70E+00	6.03E+00	7.56E+00

Table 7 Results of ablation study on scikit-feature repository (with population size set to 10)

Dimension	Datasets	Measure	SparseEA	RA-SparseEA	SparseDO (Best1)	SparseDO (Best2)	SparseDO (Rand1)	SparseDO (Rand2)	HSparseDO (Rand1Best1)	HSparseDO (Rand2Best2)
4862	BASEHOCK	AVG	8.37E-01	9.20E-01	8.46E-01	8.44E-01	8.62E-01	8.63E-01	8.45E-01	8.45E-01
		STD	1.52E-02	1.41E-02	1.05E-02	1.25E-02	1.09E-02	1.50E-02	1.37E-02	1.43E-02
4322	RELATHE	AVG	8.46E-01	9.07E-01	8.46E-01	8.49E-01	8.67E-01	8.63E-01	8.47E-01	8.44E-01
		STD	1.28E-02	1.63E-02	1.00E-02	1.71E-02	1.46E-02	1.38E-02	1.33E-02	1.43E-02
4026	lymphoma	AVG	9.36E-01	9.48E-01	9.50E-01	9.67E-01	9.56E-01	9.42E-01	9.61E-01	9.53E-01
		STD	5.69E-02	4.70E-02	4.61E-02	3.59E-02	4.35E-02	6.33E-02	5.24E-02	4.76E-02
9712	nci9	AVG	5.80E-01	7.27E-01	5.88E-01	5.79E-01	5.62E-01	5.88E-01	5.83E-01	5.85E-01
		STD	7.70E-02	9.34E-02	9.65E-02	7.49E-02	8.08E-02	9.86E-02	9.14E-02	7.12E-02
2420	warpPIE10P	AVG	9.80E-01	9.93E-01	9.83E-01	9.84E-01	9.89E-01	9.87E-01	9.75E-01	9.83E-01
		STD	1.93E-02	9.77E-03	1.96E-02	1.81E-02	1.23E-02	1.67E-02	2.83E-02	2.19E-02
	AVG TOTAL		4.18E+00	4.49E+00	4.21E+00	4.22E+00	4.23E+00	4.24E+00	4.21E+00	4.21E+00
	COUNT			5	4	4	4	5	4	4

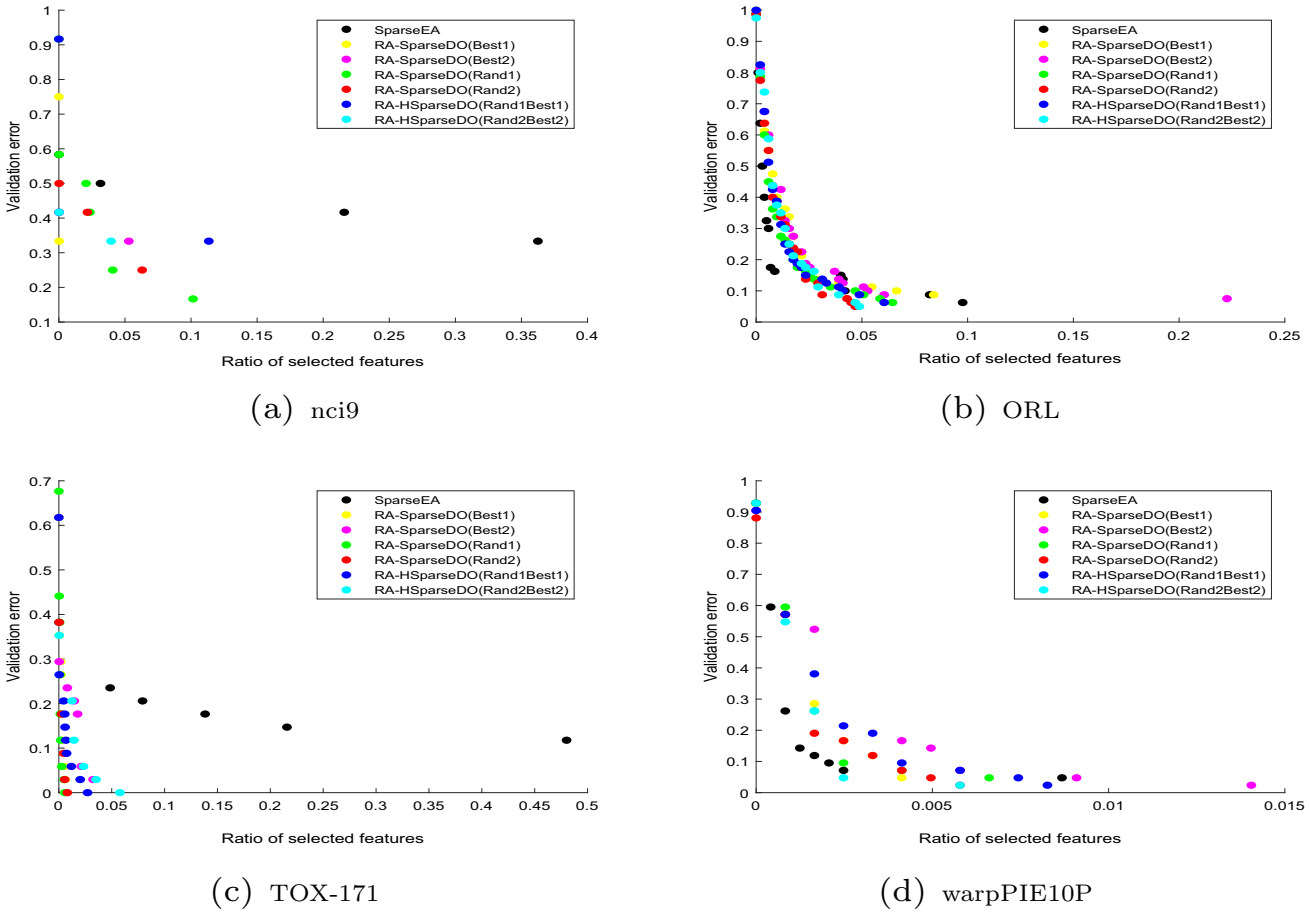

Fig. 1 Obtained Pareto fronts of SparseEA, RA-SparseDO (Best1), RA-SparseDO (Best2), RA-SparseDO (Rand1), RA-SparseDO (Rand2), RA-HSparseDO (Rand1Best1), and RA-HSparseDO (Rand2Best2)

Table 8 Results of ablation study on scikit-feature repository (with population size set to 30)

Dimension	Datasets	Measure	SparseEA	RA-SparseEA	SparseDO (Best1)	SparseDO (Best2)	SparseDO (Rand1)	SparseDO (Rand2)	HSparseDO (Rand1Best1)	HSparseDO (Rand2Best2)
4862	BASEHOCK	AVG	8.43E-01	9.06E-01	8.48E-01	8.47E-01	8.60E-01	8.53E-01	8.52E-01	8.46E-01
		STD	1.55E-02	1.28E-02	1.42E-02	1.44E-02	1.16E-02	8.97E-03	1.21E-02	1.21E-02
4322	RELATHE	AVG	8.48E-01	8.95E-01	8.57E-01	8.45E-01	8.68E-01	8.57E-01	8.48E-01	8.40E-01
		STD	1.44E-02	1.46E-02	1.24E-02	1.54E-02	1.14E-02	1.07E-02	1.30E-02	1.23E-02
4026	lymphoma	AVG	9.41E-01	9.66E-01	9.48E-01	9.51E-01	9.68E-01	9.61E-01	9.62E-01	9.59E-01
		STD	4.82E-02	3.98E-02	3.93E-02	5.09E-02	4.23E-02	3.91E-02	4.66E-02	4.43E-02
9712	nci9	AVG	5.79E-01	7.19E-01	6.06E-01	6.07E-01	5.85E-01	6.05E-01	6.03E-01	6.02E-01
		STD	8.44E-02	1.15E-01	8.91E-02	7.93E-02	8.22E-02	8.51E-02	7.27E-02	9.87E-02
2420	warpPIE10P	AVG	9.87E-01	9.94E-01	9.86E-01	9.78E-01	9.93E-01	9.92E-01	9.91E-01	9.88E-01
		STD	1.76E-02	8.74E-03	1.84E-02	2.35E-02	1.16E-02	1.18E-02	1.06E-02	1.71E-02
	AVG TOTAL		4.20E+00	4.48E+00	4.24E+00	4.23E+00	4.27E+00	4.27E+00	4.26E+00	4.23E+00
	COUNT			5	4	3	5	5	4	4

Comparative experiments for large-scale sparse multi-objective feature selection problems

In this subsection, comparative experiments for LSMFSPs will be verified on scikit-feature repository.

Comparative experiments with 30 individuals

Each algorithm is executed for 5000 function evaluations on each dataset. In these experiments, ten datasets from scikit-feature repository are used to verify the effectiveness of the proposed algorithm for LSMFSPs, that is lung, lung-discrete, lymphoma, nci9, ORL, Prostate-GE, TOX-171, warpAR10P, warpPIE10P, and Yale. The numbers of features on the ten datasets are 3312, 325, 4026, 9712, 1024, 5966, 5748, 2400, 2420, and 1024, respectively. First of all, the SparseEA based on ReliefF with different difference operator will be tested. Then, the proposed algorithm will be compared with the state-of-the-art algorithms.

Table 9 shows the HV values of the SparseEA, RA-SparseDO (Best1), RA-SparseDO (Best2), RA-SparseDO (Rand1), RA-SparseDO (Rand2), RA-HSparseDO (Rand1Best1), and RA-HSparseDO (Rand2Best2) on scikit-feature repository with population size set to 30. Similarly, the values marked in red indicate that the algorithm is better than the original SparseEA. The penultimate row of Table 9 (AVG TOTAL) represents the sum of the algorithm's AVG values

over the ten datasets. The last row (COUNT) of the table indicates the number of times the algorithm is superior to the SparseEA algorithm in ten datasets. As can be seen from the last line of Table 9, RA-SparseDO (Rand2), RA-HSparseDO (Rand1Best1), and RA-HSparseDO (Rand2Best2) all obtain ten better results than SparseEA. Followed by algorithm RA-SparseDO (Rand1), which performs better than SparseEA on 9 dataset. RA-SparseDO (Best1) and RA-SparseDO (Best2) got higher HV values than SparseEA on 6 and 5 datasets, respectively. From the values of AVG TOTAL, we can see that the most effective algorithm is RA-SparseDO (Rand2), followed by RA-SparseDO (Rand1). RA-SparseDO (Best2) has the worst effect of all the proposed algorithms. Specifically, on Prostate-GE, the values of all six proposed algorithms are at least 0.03 higher than that of SparseEA. On nci9, all six proposed algorithms are improved by at least 0.04. On TOX-171, all algorithms are improved by at least 0.09. What's more, the values of RA-SparseDO (Rand2) are at least 0.03 higher than those SparseEA on five datasets.

To show the non-dominated solutions of these algorithms, the Pareto fronts on four datasets (nci9, ORL, TOX-171, and warpPIE10P) that is obtained by SparseEA, RA-SparseDO (Best1), RA-SparseDO (Best2), RA-SparseDO (Rand1), RA-SparseDO (Rand2), RA-HSparseDO (Rand1Best1), and RA-HSparseDO (Rand2Best2) are plotted in Fig. 1. It can be observed from Fig. 1 that the non-dominated solutions of all proposed algorithms are obviously superior to the original SparseEA algorithm on nci9 and TOX-171. For ORL and warpPIE10P, SparseEA can achieve low ratio of selected features, while most proposed algorithms can achieve low validation error.

Then, RA-SparseDO (Rand2), RA-HSparseDO (Rand2Best2), and the original SparseEA are used to compare with other algorithms. Four comparison algorithms were selected in this subsection, i.e., ARMOEA [67], DAEA [68], DEAGNG [69], and MOEAPSL [51]. All four comparison algorithms have been proposed in the past 3 years. ARMOEA algorithm is an adaptive geometry estimation-based many-objective evolutionary algorithm which was proposed in 2019. In 2021, a duplication analysis-based evolutionary algorithm (DAEA) was proposed for solving bi-objective feature selection in classification. DEAGNG is an decomposition-based evolutionary algorithm guided by growing neural gas which was designed in 2020. MOEAPSL is an evolutionary algorithm proposed in 2021 to solve sparse large-scale multi-objective problems by learning Pareto optimal subspace.

The experimental results are shown in Table 10. First, compare SparseDO (Rand2) with ARMOEA, DAEA, DEAGNG, MOEAPSL, and the original SparseEA. Among the six algorithms, the one with the best performance is marked in red, and the penultimate row of Table 10 indicates the number of optimal results obtained by the algorithm. It can be found that SparseDO (Rand2) performs best on seven datasets. Next is DAEA, which performs best on three datasets. ARMOEA, DEAGNG, and SparseEA have not obtained the optimal results. Similarly, RA-HSparseDO (Rand2Best2) is used to compare with five comparison algorithm, and the maximum HV values in all algorithms are marked with bold. The last row of Table 10 represents the number of times the optimal value was obtained. RA-HSparseDO (Rand2Best2) obtained optimal values on 6 datasets, and DAEA obtained optimal values on 4 datasets. Therefore, the proposed algorithm has the best performance, and is significantly better than ARMOEA, DEAGNG, MOEAPSL, and the original SparseEA.

Similarly, the Pareto fronts obtained by ARMOEA, DAEA, DEAGNG, MOEAPSL, SparseEA, RA-SparseDO (Rand2), and RA-HSparseDO (Rand2Best2) are plotted in Fig. 2. As shown in Fig. 2, the solutions obtained by ARMOEA and DEAGNG are not sparse enough and are obviously worse than those obtained by other algorithms. RA-SparseDO (Rand2) has obtained non-dominated solutions on lymphoma, warpAR10P, and Yale with low validation error than other algorithms. The non-dominated solutions of DAEA have lower ratio of selected features than RA-SparseDO (Rand2) and RA-HSparseDO (Rand2Best2) on warpAR10P and Yale.

Comparative experiments with 50 individuals

The next experiment is to verify the effectiveness of the proposed algorithm when the population size is 50. Similarly, the performances of the proposed algorithms under different difference operators are compared with the original SparseEA. The HV values of SparseEA, RA-SparseDO (Best1), RA-SparseDO (Best2), RA-SparseDO (Rand1), RA-SparseDO (Rand2), RA-HSparseDO (Rand1Best1), and RA-HSparseDO (Rand2Best2) are shown in Table 11.

It can be found from the last line of Table 11 that RA-SparseDO (Rand2) performs better than SparseEA on all ten datasets. Next are RA-SparseDO (Best1), RA-SparseDO (Rand1), and RA-HSparseDO (Rand1Best1), all of which have better results on nine datasets. RA-SparseDO (Best2) and RA-HSparseDO (Rand2Best2) obtained 7 and 8 better values, respectively. As can be seen from the penultimate row of Table 11, the six proposed algorithms are all superior to the original SparseEA. The highest AVG TOTAL value is obtained by RA-SparseDO (Rand1), which is 9.41E+00. Followed by RA-HSparseDO (Rand1Best1), which is 9.38E+00. In detail, the values of all six proposed algorithms are at least 0.03 higher than those of SparseEA on lung-discrete, nci9, Prostate-GE, and TOX-171. Specifically, all six proposed algorithms are improved at least 0.09 on TOX-171. On lung-discrete, RA-SparseDO (Rand1), RA-SparseDO (Rand2), RA-HSparseDO (Rand1Best1), and

Table 9 The comparison between SparseEA and RA-SparseDO(Best1), RA-SparseDO(Best2), RA-SparseDO(Rand1), RA-SparseDO(Rand2), RA-HSparseDO(Rand1Best1), and RA-HSparseDO(Rand2Best2) on scikit-feature repository (with population size set to 30)

Datasets	Measure	SparseEA	RA-SparseDO (Best1)	RA-SparseDO (Best2)	RA-SparseDO (Rand1)	RA-SparseDO (Rand2)	RA-HSparseDO (Rand1Best1)	RA-HSparseDO (Rand2Best2)
lung	AVG	9.93E-01	9.98E-01	9.92E-01	9.96E-01	1.00E+00	9.97E-01	9.98E-01
	STD	1.04E-02	5.85E-03	1.38E-02	8.60E-03	1.52E-04	7.81E-03	5.78E-03
lung-discrete	AVG	9.78E-01	9.77E-01	9.67E-01	9.91E-01	9.95E-01	9.84E-01	9.87E-01
	STD	2.88E-02	3.72E-02	4.28E-02	1.64E-02	1.44E-03	2.96E-02	1.95E-02
lymphoma	AVG	9.53E-01	9.52E-01	9.59E-01	9.50E-01	9.79E-01	9.68E-01	9.56E-01
	STD	3.24E-02	4.35E-02	5.89E-02	4.06E-02	3.91E-02	3.40E-02	5.22E-02
nci9	AVG	6.54E-01	7.20E-01	6.99E-01	7.91E-01	7.56E-01	6.98E-01	7.15E-01
	STD	1.03E-01	1.21E-01	9.67E-02	1.15E-01	8.51E-02	9.27E-02	9.47E-02
ORL	AVG	9.31E-01	9.16E-01	9.19E-01	9.37E-01	9.52E-01	9.35E-01	9.46E-01
	STD	2.44E-02	2.71E-02	2.83E-02	1.98E-02	1.87E-02	2.34E-02	2.12E-02
Prostate-GE	AVG	9.59E-01	9.94E-01	9.89E-01	9.94E-01	9.94E-01	9.89E-01	9.89E-01
	STD	2.47E-02	1.57E-02	1.95E-02	1.57E-02	1.57E-02	1.95E-02	1.95E-02
TOX-171	AVG	8.73E-01	9.77E-01	9.66E-01	9.97E-01	9.96E-01	9.80E-01	9.79E-01
	STD	4.52E-02	2.55E-02	2.74E-02	9.77E-03	1.08E-02	2.74E-02	2.40E-02
warpAR10P	AVG	9.14E-01	9.00E-01	8.95E-01	9.27E-01	9.61E-01	9.29E-01	9.35E-01
	STD	4.46E-02	4.20E-02	5.37E-02	5.09E-02	3.54E-02	4.85E-02	3.60E-02
warpPIE10P	AVG	9.90E-01	9.96E-01	9.97E-01	9.96E-01	9.97E-01	9.94E-01	9.98E-01
	STD	1.57E-02	9.30E-03	5.51E-03	9.35E-03	5.43E-03	8.80E-03	3.89E-03
Yale	AVG	7.91E-01	8.11E-01	7.90E-01	8.24E-01	8.42E-01	8.15E-01	8.36E-01
	STD	5.44E-02	5.63E-02	5.70E-02	5.78E-02	4.55E-02	5.80E-02	4.60E-02
AVG TOTAL		9.04E+00	9.24E+00	9.17E+00	9.40E+00	9.47E+00	9.29E+00	9.34E+00
COUNT			6	5	9	10	10	10

Table 10 Comparisons of different algorithms on scikit-feature repository (with population size set to 30)

Datasets	Measure	ARMOEA	DAEA	DEAGNG	MOEAPSL	SparseEA	RA-SparseDO (Rand2)	RA-HSparseDO (Rand2Best2)
lung	AVG	7.01E-01	9.95E-01	7.05E-01	9.68E-01	9.93E-01	1.00E+00	**9.98E-01**
	STD	1.69E-02	9.25E-03	2.27E-02	7.55E-02	1.04E-02	1.52E-04	**5.78E-03**
lung-discrete	AVG	9.49E-01	9.85E-01	9.67E-01	9.85E-01	9.78E-01	9.95E-01	**9.87E-01**
	STD	4.03E-02	2.44E-02	2.42E-02	2.79E-02	2.88E-02	1.44E-03	**1.95E-02**
lymphoma	AVG	6.66E-01	9.55E-01	6.61E-01	9.45E-01	9.53E-01	9.79E-01	**9.56E-01**
	STD	2.99E-02	3.96E-02	4.04E-02	4.56E-02	3.24E-02	3.91E-02	**5.22E-02**
nci9	AVG	3.68E-01	**8.53E-01**	3.42E-01	7.77E-01	6.54E-01	7.56E-01	7.15E-01
	STD	8.86E-02	**1.03E-01**	8.28E-02	2.15E-01	1.03E-01	8.51E-02	9.47E-02
ORL	AVG	7.76E-01	**9.50E-01**	7.98E-01	9.45E-01	9.31E-01	9.52E-01	9.46E-01
	STD	2.29E-02	**1.95E-02**	3.01E-02	3.55E-02	2.44E-02	1.87E-02	2.12E-02
Prostate-GE	AVG	6.13E-01	**9.98E-01**	6.15E-01	9.85E-01	9.59E-01	9.94E-01	9.89E-01
	STD	3.51E-02	**8.29E-03**	3.65E-02	3.78E-02	2.47E-02	1.57E-02	1.95E-02
TOX-171	AVG	5.94E-01	**9.83E-01**	5.93E-01	9.34E-01	8.73E-01	9.96E-01	9.79E-01
	STD	3.07E-02	**2.04E-02**	3.38E-02	9.76E-02	4.52E-02	1.08E-02	2.40E-02
warpAR10P	AVG	4.52E-01	**9.40E-01**	4.79E-01	8.69E-01	9.14E-01	9.61E-01	9.35E-01
	STD	6.03E-02	**3.91E-02**	6.59E-02	1.59E-01	4.46E-02	3.54E-02	3.60E-02
warpPIE10P	AVG	7.20E-01	9.98E-01	7.42E-01	9.79E-01	9.90E-01	9.97E-01	**9.98E-01**
	STD	2.27E-02	3.94E-03	2.19E-02	4.01E-02	1.57E-02	5.43E-03	**3.89E-03**
Yale	AVG	6.35E-01	8.03E-01	6.36E-01	8.06E-01	7.91E-01	8.42E-01	**8.36E-01**
	STD	4.99E-02	5.46E-02	5.53E-02	6.51E-02	5.44E-02	4.55E-02	**4.60E-02**
SparseDO(Rand2)		0	3	0	0	0	7	
RA-HSparseDO(Rand2Best2)		0	4	0	0	0		6

Table 11 The comparison between SparseEA and RA-SparseDO(Best1), RA-SparseDO(Best2), RA-SparseDO(Rand1), RA-SparseDO(Rand2), RA-HSparseDO(Rand1Best1), and RA-HSparseDO(Rand2Best2) on scikit-feature repository (with population size set to 50)

Datasets	Measure	SparseEA	RA-SparseDO (Best1)	RA-SparseDO (Best2)	RA-SparseDO (Rand1)	RA-SparseDO (Rand2)	RA-HSparseDO (Rand1Best1)	RA-HSparseDO (Rand2Best2)
lung	AVG	9.87E-01	9.95E-01	9.94E-01	1.00E+00	9.99E-01	9.99E-01	9.97E-01
	STD	1.27E-02	9.27E-03	9.49E-03	1.46E-04	2.50E-04	4.21E-03	5.89E-03
lung-discrete	AVG	9.29E-01	9.77E-01	9.67E-01	9.91E-01	9.95E-01	9.84E-01	9.87E-01
	STD	1.26E-02	3.72E-02	4.28E-02	1.64E-02	1.44E-03	2.96E-02	1.95E-02
lymphoma	AVG	9.58E-01	9.56E-01	9.38E-01	9.56E-01	9.61E-01	9.69E-01	9.46E-01
	STD	4.01E-02	5.37E-02	7.08E-02	5.23E-02	3.63E-02	2.95E-02	5.23E-02
nci9	AVG	6.33E-01	7.54E-01	7.24E-01	7.48E-01	6.84E-01	7.57E-01	7.19E-01
	STD	8.22E-02	1.02E-01	8.60E-02	1.07E-01	9.28E-02	1.06E-01	9.05E-02
ORL	AVG	9.32E-01	9.36E-01	9.06E-01	9.48E-01	9.37E-01	9.28E-01	9.26E-01
	STD	2.48E-02	2.20E-02	2.49E-02	2.24E-02	2.48E-02	2.64E-02	2.83E-02
Prostate-GE	AVG	9.53E-01	9.83E-01	9.86E-01	9.97E-01	9.98E-01	9.86E-01	9.93E-01
	STD	3.25E-02	2.56E-02	2.42E-02	1.15E-02	8.29E-03	2.11E-02	1.96E-02
TOX-171	AVG	8.76E-01	9.78E-01	9.77E-01	9.97E-01	9.91E-01	9.94E-01	9.71E-01
	STD	6.00E-02	2.41E-02	2.14E-02	6.77E-03	1.70E-02	1.23E-02	3.13E-02
warpAR10P	AVG	9.07E-01	9.10E-01	8.89E-01	9.37E-01	9.43E-01	9.39E-01	9.10E-01
	STD	5.84E-02	6.19E-02	5.46E-02	4.36E-02	3.98E-02	3.69E-02	5.62E-02
warpPIE10P	AVG	9.93E-01	9.96E-01	9.94E-01	9.96E-01	9.98E-01	9.98E-01	9.97E-01
	STD	9.78E-03	9.39E-03	8.69E-03	6.57E-03	3.88E-03	3.94E-03	3.96E-03
Yale	AVG	8.01E-01	8.11E-01	8.01E-01	8.36E-01	8.39E-01	8.29E-01	8.13E-01
	STD	4.68E-02	5.78E-02	4.25E-02	5.22E-02	5.61E-02	5.83E-02	5.78E-02
AVG TOTAL		8.97E+00	9.30E+00	9.17E+00	9.41E+00	9.35E+00	9.38E+00	9.26E+00
COUNT			9	7	9	10	9	8

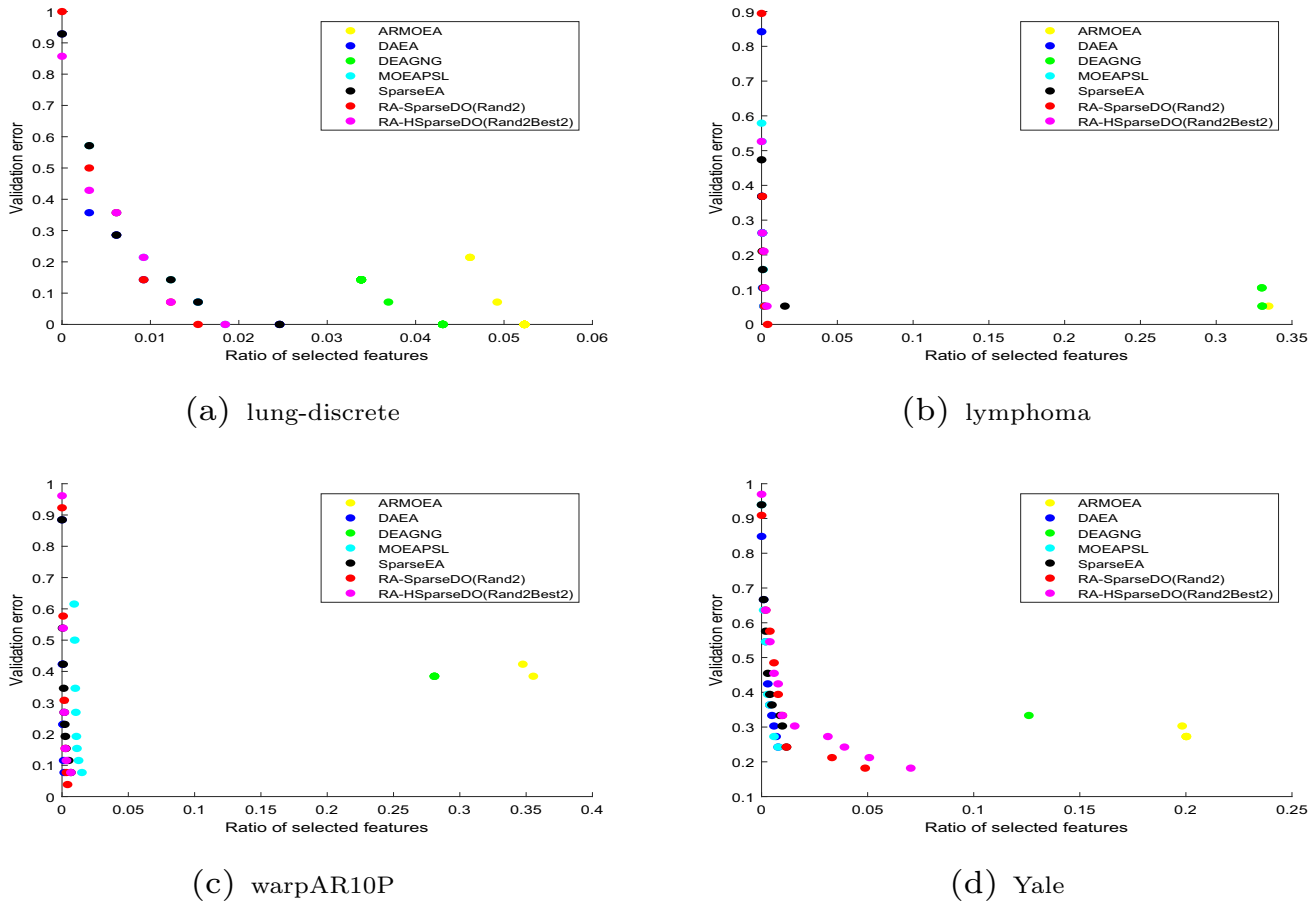

Fig. 2 Obtained Pareto fronts of ARMOEA, DAEA, DEAGNG, MOEAPSL, SparseEA, RA-SparseDO (Rand2), and RA-HSparseDO (Rand2Best2)

RA-HSparseDO (Rand2Best2) are improved at least 0.05. On nci9, five proposed algorithms are improved at least 0.08. What's more, the values of RA-SparseDO (Rand1) are at least 0.03 higher than those SparseEA on six datasets. Similarly, RA-SparseDO (Rand2) and RA-HSparseDO (Rand1Best1) are also at least 0.03 higher than those SparseEA on six datasets.

Next, RA-SparseDO (Rand2) and RA-HSparseDO (Rand 1Best1) are selected to compare with the comparison algorithms. Table 12 shows the results of ARMOEA, DAEA, DEAGNG, MOEAPSL, SparseEA, RA-SparseDO (Rand2), and RA-HSparseDO (Rand1Best1) with population size set to 50. The best results are marked in red. The results of SparseDO (Rand2) compared with ARMOEA, DAEA, DEAGNG, MOEAPSL, and SparseEA are shown in the penultimate row of Table 12. It can be seen that RA-SparseDO (Rand2) got 7 best values, and DAEA got three best results. The last line of Table 12 is the result of the comparison between RAHSparseDO (Rand1Best1) and ARMOEA, DAEA, DEAGNG, MOEAPSL, and SparseEA. The best results are marked in bold. In this comparison, RAHSparseDO (Rand1Best1) won 6 times and DAEA won 4 times. Therefore, the proposed algorithm is slightly better than DAEA, but significantly better than ARMOEA, DEAGG, MOEAPSL and the original SparseEA for solving LSMFSPs.

Comparative experiments with different population size

It is well known that different algorithms usually have different optimal settings of the population size. The third experiment is to verify the effectiveness of the algorithms with different population size. Each algorithm is executed for 5000 function evaluations on each dataset. Table 13 shows the results of RA-SparseDO (Rand2) with different population size set to 10, 30, 50, 70, 90, 110, 130, 150, 170, and 190. Similarly, the best results of different population size are marked in red. The optimal population size is 30, and 6 red values are obtained. Next are 50 and 70, both with 2 best results. When the population size is 10, the optimal solution is obtained on 1 dataset. In particular, the algorithm gets the same value on the lung-discrete dataset when the population size is set to 30 and 50. When the population size is greater than or equal to 90, the algorithm does not get the best result.

Table 12 Comparisons of different algorithms on scikit-feature repository (with population size set to 50)

Datasets	Measure	ARMOEA	DAEA	DEAGNG	MOEAPSL	SparseEA	RA-SparseDO (Rand2)	RA-SparseDO (Rand1Best1)
lung	AVG	7.18E-01	9.98E-01	7.24E-01	9.91E-01	9.87E-01	9.99E-01	**9.99E-01**
	STD	2.58E-02	5.78E-03	1.95E-02	2.26E-02	1.27E-02	2.50E-04	**4.21E-03**
lung-discrete	AVG	9.57E-01	**9.93E-01**	9.59E-01	9.85E-01	9.29E-01	9.95E-01	9.84E-01
	STD	3.46E-02	**1.16E-02**	3.90E-02	2.27E-02	1.26E-02	1.44E-03	2.96E-02
lymphoma	AVG	6.93E-01	9.49E-01	6.87E-01	9.59E-01	9.58E-01	9.61E-01	**9.69E-01**
	STD	2.97E-02	4.85E-02	3.06E-02	4.32E-02	4.01E-02	3.63E-02	**2.95E-02**
nci9	AVG	3.82E-01	**8.08E-01**	3.64E-01	8.39E-01	6.33E-01	6.84E-01	7.57E-01
	STD	8.73E-02	**9.48E-02**	8.81E-02	1.64E-01	8.22E-02	9.28E-02	1.06E-01
ORL	AVG	7.81E-01	**9.52E-01**	7.92E-01	9.42E-01	9.32E-01	9.37E-01	9.28E-01
	STD	1.94E-02	**2.58E-02**	3.36E-02	3.05E-02	2.48E-02	2.48E-02	2.64E-02
Prostate-GE	AVG	6.22E-01	**1.00E+00**	6.42E-01	8.26E-01	9.53E-01	9.98E-01	9.86E-01
	STD	4.87E-02	**1.41E-05**	4.20E-02	3.20E-01	3.25E-02	8.29E-03	2.11E-02
TOX-171	AVG	6.18E-01	9.88E-01	6.25E-01	9.75E-01	8.76E-01	9.91E-01	**9.94E-01**
	STD	2.63E-02	2.18E-02	3.89E-02	6.38E-02	6.00E-02	1.70E-02	**1.23E-02**
warpAR10P	AVG	4.84E-01	9.35E-01	4.93E-01	8.89E-01	9.07E-01	9.43E-01	**9.39E-01**
	STD	6.22E-02	5.12E-02	6.44E-02	1.17E-01	5.84E-02	3.98E-02	**3.69E-02**
warpPIE10P	AVG	7.36E-01	9.98E-01	7.53E-01	9.73E-01	9.93E-01	9.98E-01	**9.98E-01**
	STD	2.12E-02	7.88E-03	2.80E-02	5.16E-02	9.78E-03	3.88E-03	**3.94E-03**
Yale	AVG	6.36E-01	8.25E-01	6.73E-01	8.28E-01	8.01E-01	8.39E-01	**8.29E-01**
	STD	5.36E-02	6.24E-02	5.03E-02	5.98E-02	4.68E-02	5.61E-02	**5.83E-02**
SparseDO(Rand2)		0	3	0	0	0	7	
RA-HSparseDO(Rand1Best1)		0	4	0	0	0		6

Table 13 RA-SparseDO (Rand2) with different population size on scikit-feature repository

Datasets	Measure	10	30	50	70	90	110	130	150	170	190
lung	AVG	9.94E-01	1.00E+00	9.99E-01	9.98E-01	9.99E-01	9.97E-01	9.97E-01	9.92E-01	9.90E-01	9.92E-01
	STD	1.18E-02	1.52E-04	2.50E-04	5.87E-03	6.82E-04	6.19E-03	4.22E-03	1.01E-02	1.10E-02	1.16E-02
lung-discrete	AVG	9.59E-01	9.95E-01	9.95E-01	9.88E-01	9.78E-01	9.75E-01	9.75E-01	9.69E-01	9.66E-01	9.68E-01
	STD	4.69E-02	1.44E-03	1.44E-03	2.79E-03	1.83E-02	2.67E-02	1.40E-02	1.60E-02	1.82E-02	1.37E-02
lymphoma	AVG	9.52E-01	9.79E-01	9.61E-01	9.69E-01	9.37E-01	9.56E-01	9.57E-01	9.56E-01	9.47E-01	9.46E-01
	STD	4.16E-02	3.91E-02	3.63E-02	3.87E-02	6.01E-02	3.81E-02	4.21E-02	4.23E-02	5.40E-02	5.45E-02
nci9	AVG	8.28E-01	7.56E-01	6.84E-01	7.30E-01	7.45E-01	7.35E-01	7.29E-01	7.37E-01	7.35E-01	7.60E-01
	STD	7.85E-02	8.51E-02	9.28E-02	1.21E-01	9.66E-02	8.50E-02	9.38E-02	9.48E-02	8.47E-02	9.98E-02
ORL	AVG	9.21E-01	9.52E-01	9.37E-01	9.29E-01	9.22E-01	9.16E-01	9.21E-01	9.10E-01	9.19E-01	9.07E-01
	STD	3.11E-02	1.87E-02	2.48E-02	2.53E-02	2.61E-02	2.84E-02	2.73E-02	2.38E-02	2.82E-02	2.46E-02
Prostate-GE	AVG	9.85E-01	9.94E-01	9.98E-01	9.88E-01	9.87E-01	9.91E-01	9.88E-01	9.78E-01	9.80E-01	9.76E-01
	STD	2.76E-02	1.57E-02	8.29E-03	2.04E-02	2.04E-02	1.74E-02	2.29E-02	2.81E-02	2.19E-02	3.03E-02
TOX-171	AVG	9.86E-01	9.96E-01	9.91E-01	9.97E-01	9.84E-01	9.76E-01	9.70E-01	9.76E-01	9.62E-01	9.63E-01
	STD	1.94E-02	1.08E-02	1.70E-02	8.84E-03	1.98E-02	2.75E-02	2.95E-02	2.90E-02	3.23E-02	3.10E-02
warpAR10P	AVG	8.86E-01	9.61E-01	9.43E-01	8.98E-01	8.92E-01	8.88E-01	8.55E-01	8.50E-01	8.41E-01	8.27E-01
	STD	5.23E-02	3.54E-02	3.98E-02	5.30E-02	6.42E-02	5.88E-02	5.77E-02	6.13E-02	6.70E-02	6.66E-02
warpPIE10P	AVG	9.91E-01	9.97E-01	9.98E-01	9.98E-01	9.96E-01	9.92E-01	9.93E-01	9.91E-01	9.92E-01	9.90E-01
	STD	1.18E-02	5.43E-03	3.88E-03	3.11E-03	5.53E-03	9.88E-03	7.31E-03	8.45E-03	7.09E-03	9.04E-03
Yale	AVG	7.96E-01	8.42E-01	8.39E-01	8.08E-01	8.00E-01	7.82E-01	7.83E-01	7.96E-01	7.89E-01	7.64E-01
	STD	4.89E-02	4.55E-02	5.61E-02	4.61E-02	6.30E-02	5.06E-02	4.63E-02	4.99E-02	4.93E-02	5.86E-02

Table 14 shows the results of SparseEA with different population size. It can be seen that SparseEA has the best results when the population size is set to 30, 50, 70, 90, 110. The optimal population number is 70, and 3 red values are obtained. Next are 90 and 110, both with 2 best results. When the number of individuals is 30 and 50, only one optimal solution can be obtained.

Similarly, Tables 15, 16, 17 and 18 are the HV results of ARMOEA, DAEA, DEAGNG, and MOEAPSL with different population size. The optimal population number of ARMOEA is 70, and 5 red values are obtained. This is followed by 10 and 50, with 3 and 2 best values, respectively. For DAEA, the optimal population number is 50, and four red values are obtained. It also gets red values at 30, 70, and 150. DEAGNG gets red values at 10, 70, 90, and 110. When the population number is 70 and 90, three red values are obtained, while when the population number is 10 and 110, two red values are obtained. MOEAPSL has obtained the optimal value in seven population sizes, so the population size has relatively little influence on it. MOEAPSL get two red results with population size set to 50, 70, and 150. It obtains one best value on 30, 90, 110, and 170 individuals.

Table 19 is the best results with different population size of ARMOEA, DAEA, DEAGNG, MOEAPSL, SparseEA, and RA-SparseDO (Rand2) on scikit-feature repository That is, the red values of Tables 13, 14, 15, 16, 17 and 18 are shown in Table 19. Among the six algorithms, the one with the best performance is marked in red. It can be found form Table 19, the proposed RA-SparseDO (Rand2) performs best on 7 datasets. Next is DAEA, which performs best on three datasets. ARMOEA, DEAGNG, MOEAPSL, and SparseEA have not obtained the red results.

Comparative experiments for small-scale multi-objective feature selection problems

In this subsection, the experiments will be done to further verify the effectiveness of the proposed algorithm for small-scale

Table 14 SparseEA with different population size on scikit-feature repository

Datasets	Measure	10	30	50	70	90	110	130	150	170	190
lung	AVG	9.90E-01	9.93E-01	9.87E-01	9.94E-01	9.88E-01	9.88E-01	9.83E-01	9.84E-01	9.85E-01	9.85E-01
	STD	1.32E-02	1.04E-02	1.27E-02	1.33E-02	1.19E-02	1.38E-02	1.53E-02	1.87E-02	1.56E-02	1.70E-02
lung-discrete	AVG	9.79E-01	9.78E-01	9.29E-01	9.79E-01	9.85E-01	9.95E-01	9.78E-01	9.81E-01	9.85E-01	9.75E-01
	STD	2.86E-02	2.88E-02	1.26E-02	4.09E-02	2.40E-02	1.52E-03	2.95E-02	2.68E-02	2.32E-02	3.00E-02
lymphoma	AVG	9.57E-01	9.53E-01	9.58E-01	9.52E-01	9.54E-01	9.66E-01	9.33E-01	9.47E-01	9.65E-01	9.34E-01
	STD	3.71E-02	3.24E-02	4.01E-02	4.51E-02	4.26E-02	3.50E-02	5.81E-02	4.21E-02	3.12E-02	3.89E-02
nci9	AVG	5.74E-01	6.54E-01	6.33E-01	5.82E-01	5.84E-01	6.58E-01	6.24E-01	6.10E-01	6.07E-01	6.53E-01
	STD	5.27E-02	1.03E-01	8.22E-02	9.95E-02	6.96E-02	1.09E-01	8.15E-02	8.60E-02	7.39E-02	9.66E-02
ORL	AVG	9.31E-01	9.31E-01	9.32E-01	9.28E-01	9.39E-01	9.34E-01	9.34E-01	9.25E-01	9.28E-01	9.18E-01
	STD	2.83E-02	2.44E-02	2.48E-02	3.02E-02	2.06E-02	2.24E-02	2.23E-02	2.76E-02	2.51E-02	2.96E-02
Prostate-GE	AVG	9.57E-01	9.59E-01	9.53E-01	9.52E-01	9.76E-01	9.49E-01	9.58E-01	9.70E-01	9.54E-01	9.68E-01
	STD	3.24E-02	2.47E-02	3.25E-02	3.69E-02	2.72E-02	3.49E-02	2.38E-02	2.17E-02	3.27E-02	2.59E-02
TOX-171	AVG	8.45E-01	8.73E-01	8.76E-01	9.15E-01	8.95E-01	9.01E-01	8.99E-01	8.90E-01	8.91E-01	8.95E-01
	STD	4.47E-02	4.52E-02	6.00E-02	5.30E-02	3.75E-02	4.99E-02	4.36E-02	4.76E-02	5.64E-02	5.55E-02
warpAR10P	AVG	9.00E-01	9.14E-01	9.07E-01	9.09E-01	8.60E-01	8.84E-01	8.61E-01	8.51E-01	8.53E-01	8.61E-01
	STD	7.10E-02	4.46E-02	5.84E-02	6.45E-02	5.89E-02	4.26E-02	5.49E-02	7.75E-02	6.06E-02	5.72E-02
warpPIE10P	AVG	9.84E-01	9.90E-01	9.93E-01	9.93E-01	9.87E-01	9.92E-01	9.87E-01	9.85E-01	9.88E-01	9.91E-01
	STD	1.68E-02	1.57E-02	9.78E-03	1.01E-02	1.48E-02	1.27E-02	1.91E-02	1.60E-02	1.48E-02	1.44E-02
Yale	AVG	7.76E-01	7.91E-01	8.01E-01	8.05E-01	7.87E-01	7.98E-01	7.92E-01	7.96E-01	7.45E-01	7.85E-01
	STD	6.27E-02	5.44E-02	4.68E-02	5.98E-02	4.72E-02	5.17E-02	6.68E-02	4.44E-02	5.66E-02	5.00E-02

Table 15 ARMOEA with different population size on scikit-feature repository

Datasets	Measure	10	30	50	70	90	110	130	150	170	190
lung	AVG	6.98E-01	7.01E-01	7.18E-01	6.99E-01	6.98E-01	6.74E-01	6.66E-01	6.56E-01	6.44E-01	6.32E-01
	STD	1.68E-02	1.69E-02	2.58E-02	2.04E-02	1.71E-02	1.28E-02	1.75E-02	1.39E-02	9.56E-03	1.29E-02
lung-discrete	AVG	9.75E-01	9.49E-01	9.57E-01	9.55E-01	9.42E-01	9.33E-01	9.09E-01	8.78E-01	8.47E-01	8.51E-01
	STD	2.48E-02	4.03E-02	3.46E-02	2.83E-02	3.37E-02	1.97E-02	2.69E-02	2.13E-02	3.56E-02	3.80E-02
lymphoma	AVG	6.63E-01	6.66E-01	6.93E-01	6.76E-01	6.64E-01	6.48E-01	6.46E-01	6.18E-01	6.06E-01	6.10E-01
	STD	3.19E-02	2.99E-02	2.97E-02	3.39E-02	3.20E-02	3.74E-02	2.40E-02	4.78E-02	2.23E-02	2.38E-02
nci9	AVG	3.39E-01	3.68E-01	3.82E-01	3.93E-01	3.63E-01	3.77E-01	3.71E-01	3.50E-01	3.39E-01	3.28E-01
	STD	9.93E-02	8.86E-02	8.73E-02	8.41E-02	8.50E-02	7.05E-02	9.53E-02	7.77E-02	6.81E-02	5.97E-02
ORL	AVG	8.18E-01	7.76E-01	7.81E-01	7.83E-01	7.80E-01	7.65E-01	7.41E-01	7.24E-01	7.11E-01	7.05E-01
	STD	3.45E-02	2.29E-02	1.94E-02	2.72E-02	3.17E-02	2.38E-02	1.93E-02	2.23E-02	1.86E-02	2.08E-02
Prostate-GE	AVG	5.87E-01	6.13E-01	6.22E-01	6.33E-01	6.26E-01	6.15E-01	6.08E-01	5.91E-01	5.96E-01	5.81E-01
	STD	3.65E-02	3.51E-02	4.87E-02	2.59E-02	3.29E-02	3.56E-02	3.49E-02	2.54E-02	3.20E-02	3.46E-02
TOX-171	AVG	5.78E-01	5.94E-01	6.18E-01	6.22E-01	6.20E-01	6.13E-01	6.05E-01	5.98E-01	5.70E-01	5.74E-01
	STD	4.17E-02	3.07E-02	2.63E-02	3.33E-02	2.68E-02	3.24E-02	3.16E-02	2.28E-02	3.67E-02	3.18E-02
warpAR10P	AVG	4.32E-01	4.52E-01	4.84E-01	4.87E-01	4.80E-01	4.71E-01	4.72E-01	4.52E-01	4.56E-01	4.29E-01
	STD	7.23E-02	6.03E-02	6.22E-02	5.75E-02	4.51E-02	7.31E-02	4.42E-02	5.86E-02	6.47E-02	5.90E-02
warpPIE10P	AVG	7.42E-01	7.20E-01	7.36E-01	7.18E-01	7.16E-01	7.04E-01	6.90E-01	6.76E-01	6.54E-01	6.51E-01
	STD	2.09E-02	2.27E-02	2.12E-02	2.74E-02	2.20E-02	1.87E-02	1.79E-02	2.69E-02	2.24E-02	2.19E-02
Yale	AVG	6.48E-01	6.35E-01	6.36E-01	6.49E-01	6.37E-01	6.40E-01	6.21E-01	5.92E-01	5.75E-01	5.64E-01
	STD	5.05E-02	4.99E-02	5.36E-02	3.43E-02	5.38E-02	5.18E-02	4.39E-02	4.87E-02	4.48E-02	3.85E-02

Table 16 DAEA with different population size on scikit-feature repository

Datasets	Measure	10	30	50	70	90	110	130	150	170	190
lung	AVG	9.93E-01	9.95E-01	9.98E-01	9.98E-01	9.97E-01	9.98E-01	9.98E-01	9.98E-01	9.96E-01	9.97E-01
	STD	1.36E-02	9.25E-03	5.78E-03	6.16E-03	7.86E-03	5.86E-03	6.94E-03	5.87E-03	8.61E-03	6.89E-03
lung-discrete	AVG	9.89E-01	9.85E-01	9.93E-01	9.78E-01	9.84E-01	8.34E-01	8.50E-01	8.28E-01	8.18E-01	8.01E-01
	STD	1.96E-02	2.44E-02	1.16E-02	2.91E-02	2.99E-02	6.39E-02	4.86E-02	4.42E-02	4.03E-02	3.65E-02
lymphoma	AVG	9.46E-01	9.55E-01	9.49E-01	9.34E-01	9.57E-01	9.58E-01	9.55E-01	9.68E-01	9.53E-01	9.60E-01
	STD	5.86E-02	3.96E-02	4.85E-02	6.22E-02	3.84E-02	3.71E-02	5.32E-02	3.83E-02	4.44E-02	3.35E-02
nci9	AVG	8.05E-01	8.53E-01	8.08E-01	8.28E-01	8.13E-01	7.95E-01	7.95E-01	7.85E-01	7.93E-01	8.08E-01
	STD	7.36E-02	1.03E-01	9.48E-02	8.88E-02	9.47E-02	1.02E-01	9.75E-02	8.90E-02	8.17E-02	9.25E-02
ORL	AVG	9.04E-01	9.50E-01	9.52E-01	9.56E-01	9.46E-01	9.52E-01	9.37E-01	9.42E-01	9.46E-01	9.37E-01
	STD	3.74E-02	1.95E-02	2.58E-02	2.45E-02	1.96E-02	1.78E-02	3.03E-02	2.86E-02	1.99E-02	2.61E-02
Prostate-GE	AVG	9.98E-01	9.98E-01	1.00E+00	9.98E-01	9.94E-01	9.98E-01	9.97E-01	9.94E-01	9.97E-01	9.94E-01
	STD	8.30E-03	8.29E-03	1.41E-05	8.30E-03	1.57E-02	8.30E-03	1.15E-02	1.57E-02	1.15E-02	1.57E-02
TOX-171	AVG	9.52E-01	9.83E-01	9.88E-01	9.90E-01	9.89E-01	9.83E-01	9.85E-01	9.82E-01	9.82E-01	9.82E-01
	STD	3.75E-02	2.04E-02	2.18E-02	1.91E-02	1.93E-02	1.91E-02	2.40E-02	1.91E-02	2.47E-02	2.03E-02
warpAR10P	AVG	9.27E-01	9.40E-01	9.35E-01	9.33E-01	9.32E-01	9.05E-01	9.00E-01	8.91E-01	8.61E-01	8.59E-01
	STD	4.40E-02	3.91E-02	5.12E-02	4.70E-02	3.66E-02	4.78E-02	5.43E-02	4.33E-02	6.39E-02	5.72E-02
warpPIE10P	AVG	9.92E-01	9.98E-01	9.98E-01	9.98E-01	9.98E-01	9.98E-01	9.98E-01	9.98E-01	9.97E-01	9.95E-01
	STD	1.54E-02	3.94E-03	7.88E-03	5.48E-03	5.46E-03	5.50E-03	5.54E-03	3.95E-03	6.59E-03	1.05E-02
Yale	AVG	8.06E-01	8.03E-01	8.25E-01	8.22E-01	8.23E-01	7.95E-01	8.13E-01	8.05E-01	7.93E-01	7.72E-01
	STD	6.61E-02	5.46E-02	6.24E-02	4.53E-02	4.51E-02	3.98E-02	4.93E-02	6.46E-02	5.41E-02	5.24E-02

Table 17 DEAGNG with different population size on scikit-feature repository

Datasets	Measure	10	30	50	70	90	110	130	150	170	190
lung	AVG	7.18E-01	7.05E-01	7.24E-01	7.44E-01	7.37E-01	7.20E-01	7.08E-01	6.90E-01	6.81E-01	6.68E-01
	STD	2.26E-02	2.27E-02	1.95E-02	1.44E-02	1.43E-02	1.01E-02	1.42E-02	1.68E-02	1.13E-02	1.25E-02
lung-discrete	AVG	9.66E-01	9.67E-01	9.59E-01	9.63E-01	9.65E-01	9.68E-01	9.58E-01	9.12E-01	9.06E-01	8.91E-01
	STD	3.29E-02	2.42E-02	3.90E-02	3.62E-02	3.51E-02	3.40E-02	1.52E-02	5.01E-02	3.32E-02	2.77E-02
lymphoma	AVG	6.75E-01	6.61E-01	6.87E-01	6.95E-01	6.97E-01	6.88E-01	6.75E-01	6.56E-01	6.54E-01	6.45E-01
	STD	3.41E-02	4.04E-02	3.06E-02	4.94E-02	3.23E-02	3.99E-02	4.13E-02	3.71E-02	1.72E-02	2.12E-02
nci9	AVG	3.10E-01	3.42E-01	3.64E-01	3.50E-01	3.81E-01	4.01E-01	3.42E-01	3.72E-01	3.65E-01	3.55E-01
	STD	8.53E-02	8.28E-02	8.81E-02	7.42E-02	8.58E-02	7.95E-02	8.76E-02	8.32E-02	7.25E-02	9.04E-02
ORL	AVG	8.25E-01	7.98E-01	7.92E-01	8.11E-01	8.12E-01	7.88E-01	7.66E-01	7.49E-01	7.27E-01	7.15E-01
	STD	3.34E-02	3.01E-02	3.36E-02	2.87E-02	2.09E-02	2.64E-02	2.64E-02	2.30E-02	2.53E-02	2.54E-02
Prostate-GE	AVG	5.79E-01	6.15E-01	6.42E-01	6.50E-01	6.41E-01	6.36E-01	6.30E-01	6.09E-01	6.04E-01	5.94E-01
	STD	3.98E-02	3.65E-02	4.20E-02	3.05E-02	3.60E-02	2.99E-02	3.64E-02	2.45E-02	3.91E-02	3.36E-02
TOX-171	AVG	5.71E-01	5.93E-01	6.25E-01	6.34E-01	6.41E-01	6.27E-01	6.13E-01	5.94E-01	5.96E-01	5.89E-01
	STD	3.02E-02	3.38E-02	3.89E-02	3.70E-02	3.10E-02	3.77E-02	2.91E-02	3.68E-02	2.93E-02	3.01E-02
warpAR10P	AVG	4.73E-01	4.79E-01	4.93E-01	5.15E-01	5.31E-01	4.87E-01	5.07E-01	4.83E-01	4.73E-01	4.64E-01
	STD	7.60E-02	6.59E-02	6.44E-02	6.77E-02	6.59E-02	7.48E-02	5.40E-02	4.45E-02	5.13E-02	5.61E-02
warpPIE10P	AVG	7.54E-01	7.42E-01	7.53E-01	7.65E-01	7.50E-01	7.39E-01	7.28E-01	7.02E-01	6.99E-01	6.82E-01
	STD	2.65E-02	2.19E-02	2.80E-02	2.32E-02	2.41E-02	2.27E-02	1.62E-02	2.57E-02	1.60E-02	1.75E-02
Yale	AVG	6.91E-01	6.36E-01	6.73E-01	6.58E-01	6.80E-01	6.43E-01	6.39E-01	6.01E-01	6.04E-01	5.86E-01
	STD	5.01E-02	5.53E-02	5.03E-02	4.84E-02	5.22E-02	5.13E-02	5.13E-02	5.15E-02	5.74E-02	5.24E-02

Table 18 MOEAPSL with different population size on scikit-feature repository

Datasets	Measure	10	30	50	70	90	110	130	150	170	190
lung	AVG	9.23E-01	9.68E-01	9.91E-01	9.75E-01	9.75E-01	9.71E-01	9.86E-01	9.76E-01	9.91E-01	9.83E-01
	STD	1.28E-01	7.55E-02	2.26E-02	6.11E-02	4.41E-02	5.16E-02	4.40E-02	6.24E-02	2.60E-02	4.34E-02
lung-discrete	AVG	9.75E-01	9.85E-01	9.85E-01	9.89E-01	9.87E-01	9.79E-01	9.73E-01	9.80E-01	9.84E-01	9.76E-01
	STD	3.34E-02	2.79E-02	2.27E-02	1.67E-02	2.23E-02	3.26E-02	2.93E-02	3.54E-02	2.25E-02	2.80E-02
lymphoma	AVG	9.09E-01	9.45E-01	9.59E-01	9.46E-01	9.38E-01	9.47E-01	9.56E-01	9.62E-01	9.49E-01	9.31E-01
	STD	1.07E-01	4.56E-02	4.32E-02	5.87E-02	5.35E-02	3.93E-02	5.42E-02	3.47E-02	5.31E-02	5.34E-02
nci9	AVG	7.19E-01	7.77E-01	8.39E-01	7.85E-01	8.43E-01	8.10E-01	8.17E-01	7.92E-01	7.75E-01	8.04E-01
	STD	2.25E-01	2.15E-01	1.64E-01	1.54E-01	1.29E-01	1.36E-01	1.26E-01	1.28E-01	1.62E-01	1.04E-01
ORL	AVG	9.24E-01	9.45E-01	9.42E-01	9.41E-01	9.36E-01	9.33E-01	9.31E-01	9.26E-01	9.29E-01	9.34E-01
	STD	4.10E-02	3.55E-02	3.05E-02	2.69E-02	3.79E-02	3.64E-02	5.22E-02	4.62E-02	3.51E-02	3.55E-02
Prostate-GE	AVG	9.21E-01	9.85E-01	8.26E-01	9.72E-01	9.85E-01	9.79E-01	9.82E-01	9.86E-01	9.78E-01	9.80E-01
	STD	1.12E-01	3.78E-02	3.20E-01	5.65E-02	4.07E-02	5.38E-02	4.61E-02	2.78E-02	5.55E-02	2.98E-02
TOX-171	AVG	9.09E-01	9.34E-01	9.75E-01	9.70E-01	9.74E-01	9.90E-01	9.80E-01	9.60E-01	9.76E-01	9.42E-01
	STD	1.35E-01	9.76E-02	6.38E-02	5.42E-02	4.92E-02	2.34E-02	3.21E-02	6.98E-02	4.07E-02	8.59E-02
warpAR10P	AVG	7.47E-01	8.69E-01	8.89E-01	9.04E-01	8.79E-01	8.70E-01	8.52E-01	8.54E-01	8.31E-01	8.20E-01
	STD	2.15E-01	1.59E-01	1.17E-01	8.79E-02	1.01E-01	9.16E-02	9.84E-02	8.86E-02	9.48E-02	7.71E-02
warpPIE10P	AVG	9.56E-01	9.79E-01	9.73E-01	9.91E-01	9.77E-01	9.84E-01	9.79E-01	9.86E-01	9.94E-01	9.85E-01
	STD	7.12E-02	4.01E-02	5.16E-02	3.67E-02	5.21E-02	3.93E-02	6.38E-02	2.77E-02	1.45E-02	2.52E-02
Yale	AVG	7.88E-01	8.06E-01	8.28E-01	8.25E-01	8.05E-01	7.88E-01	8.05E-01	7.87E-01	7.92E-01	7.82E-01
	STD	7.39E-02	6.51E-02	5.98E-02	6.83E-02	7.02E-02	6.73E-02	6.57E-02	7.58E-02	6.69E-02	6.96E-02

Table 19 The best results with different population size of ARMOEA, DAEA, DEAGNG, MOEAPSL, SparseEA, and RA-SparseDO (Rand2) on scikit-feature repository

Datasets	Measure	ARMOEA	DAEA	DEAGNG	MOEAPSL	SparseEA	RA-SparseDO (Rand2)
lung	AVG	7.18E-01	9.98E-01	7.44E-01	9.91E-01	9.94E-01	1.00E+00
	STD	2.58E-02	5.78E-03	1.44E-02	2.26E-02	1.33E-02	1.52E-04
lung-discrete	AVG	9.75E-01	9.93E-01	9.68E-01	9.89E-01	9.95E-01	9.95E-01
	STD	2.48E-02	1.16E-02	3.40E-02	1.67E-02	1.52E-03	1.44E-03
lymphoma	AVG	6.93E-01	9.68E-01	6.97E-01	9.62E-01	9.66E-01	9.79E-01
	STD	2.97E-02	3.83E-02	3.23E-02	3.47E-02	3.50E-02	3.91E-02
nci9	AVG	3.93E-01	8.53E-01	4.01E-01	8.43E-01	6.58E-01	8.28E-01
	STD	8.41E-02	1.03E-01	7.95E-02	1.29E-01	1.09E-01	7.85E-02
ORL	AVG	8.18E-01	9.56E-01	8.25E-01	9.45E-01	9.39E-01	9.52E-01
	STD	3.45E-02	2.45E-02	3.34E-02	3.55E-02	2.06E-02	1.87E-02
Prostate-GE	AVG	6.33E-01	1.00E+00	6.50E-01	9.86E-01	9.76E-01	9.98E-01
	STD	2.59E-02	1.41E-05	3.05E-02	2.78E-02	2.72E-02	8.29E-03
TOX-171	AVG	6.22E-01	9.90E-01	6.41E-01	9.90E-01	9.15E-01	9.97E-01
	STD	3.33E-02	1.91E-02	3.10E-02	2.34E-02	5.30E-02	8.84E-03
warpAR10P	AVG	4.87E-01	9.40E-01	5.31E-01	9.04E-01	9.14E-01	9.61E-01
	STD	5.75E-02	3.91E-02	6.59E-02	8.79E-02	4.46E-02	3.54E-02
warpPIE10P	AVG	7.42E-01	9.98E-01	7.65E-01	9.94E-01	9.93E-01	9.98E-01
	STD	2.09E-02	3.94E-03	2.32E-02	1.45E-02	9.78E-03	3.11E-03
Yale	AVG	6.49E-01	8.25E-01	6.91E-01	8.28E-01	8.05E-01	8.42E-01
	STD	3.43E-02	6.24E-02	5.01E-02	5.98E-02	5.98E-02	4.55E-02

Table 20 Comparisons of different algorithms on UCI machine learning repository (with population size set to 100)

Datasets	Measure	NSGAII	DEAGNG	ARMOEA	SparseEA	RA-SparseDO (Rand1)	RA-HSparseDO (Rand1Best1)
iris	AVG	**8.43E-01**	8.42E-01	8.38E-01	8.42E-01	8.45E-01	8.38E-01
	STD	**2.66E-02**	2.11E-02	2.27E-02	2.54E-02	2.92E-02	2.35E-02
lungcancer	AVG	9.83E-01	**9.85E-01**	9.78E-01	9.76E-01	9.82E-01	9.83E-01
	STD	5.72E-03	**4.62E-03**	2.58E-02	2.73E-02	5.45E-03	4.87E-03
Person-Classification	AVG	9.34E-01	9.31E-01	9.22E-01	8.92E-01	9.58E-01	**9.41E-01**
	STD	1.17E-01	1.03E-01	1.21E-01	1.13E-01	8.10E-02	**9.41E-02**
MUSK1	AVG	9.73E-01	9.70E-01	9.71E-01	9.72E-01	9.77E-01	**9.77E-01**
	STD	6.88E-03	1.02E-02	9.14E-03	1.24E-02	1.33E-02	**8.98E-03**
heart	AVG	8.69E-01	8.66E-01	8.68E-01	8.65E-01	8.74E-01	**8.73E-01**
	STD	2.54E-02	2.45E-02	2.62E-02	2.39E-02	2.69E-02	**2.27E-02**
ionosphere	AVG	9.64E-01	9.67E-01	**9.69E-01**	9.67E-01	9.70E-01	9.64E-01
	STD	1.41E-02	1.41E-02	**1.46E-02**	1.57E-02	1.43E-02	1.28E-02
Parkinson	AVG	5.94E-01	6.00E-01	6.08E-01	6.03E-01	6.20E-01	**6.11E-01**
	STD	3.45E-02	2.80E-02	3.37E-02	3.19E-02	3.21E-02	**3.05E-02**
COVID-19 Surveillance	AVG	8.38E-01	7.81E-01	**8.61E-01**	7.57E-01	8.01E-01	8.48E-01
	STD	1.86E-01	2.17E-01	**1.53E-01**	2.10E-01	2.57E-01	1.69E-01
RA-SparseDO(Rand1)		0	1	1	0	6	
RA-HSparseDO(Rand1Best1)		1	1	2	0		4

multi-objective feature selection problems. The comparison algorithms used in this experiment are NSGAII [70], DEAGNG, ARMOEA, and the original SparseEA. Specifically, the NSGAII is one of the most classical and popular multi-objective optimization algorithms, which is based on genetic algorithm. The datasets chosen in this subsection are shown in Table 3. In this experiment, RA-SparseDO (Rand1) and RA-HSparseDO (Rand1Best1) are selected to compare with the comparison algorithms. The number of function evaluations for each algorithm is 10,000, and the population size of each algorithm is set to 100.

Table 20 shows the HV values obtained by NSGAII, DEAGNG, ARMOEA, SparseEA, and RA-SparseDO (Rand1). Similarly, the best results of the five algorithms are marked in red. From the penultimate row of Table 20, we can see that both DEAGNG and ARMOEA have the best performance on the one dataset. RA-SparseDO (Rand1) has the best performance, and obtains the best value on six datasets. The best results of RA-HSparseDO (Rand1Best1) compared with NSGAII, DEAGNG, ARMOEA, SparseEA are marked in bold. It can be found from the last line of Table 20 that RA-HSparseDO (Rand1Best1) got the best results on four datasets. Next is ARMOE, which obtains two best values. Both NSGAII and DEAGN perform best on only one dataset. Therefore, the proposed algorithm also has certain advantages for small-scale datasets.

Running time

Finally, the computational efficiency of the five algorithms is compared in this subsection. The experiment is performed on scikit-feature repository and the results are shown in Table 21. The last line refers to the total times on ten datasets. It can be found that the original SparseEA algorithm has the longest running time and MOEAPSL has the shortest running time. DAEA takes slightly more time to run than MOEAPSL. ARMOEA and DEAGNG take about the same time. Since the ReliefF algorithm has reduced the dimensions first for LSMFSPs, the running times of the proposed RA-SparseDO (Best2), RA-SparseDO (Best2), and RA-HSparseDO (Rand2Best2) are all less than that of the original SparseEA; especially on nci9, Prostate-GE and TOX-171 datasets, because the number of features in these datasets exceeds 5000.

Conclusion

There are many application scenarios of LSMFSPs in real life, but there is little research dedicated to solving this problem. SparseEA is an excellent algorithm for solving LSMOPs, and LSMFSPs are specific applications of LSMOPs. Therefore, this manuscript proposes an enhanced SparseEA algorithm based on ReliefF with difference operators to specifically solve the LSMFSPs. SparseEA determines feature Scores by calculating the fitness of individual features, which does not reflect the correlation between features well. Therefore, combining the filter feature selection algorithm ReliefF and SparseEA, a Filter-Wrapper feature selection method is proposed in this manuscript for LSMFSPs. Furthermore, to improve the performance of SparseEA, difference operators and adaptive scores strategy are used in this manuscript.

Experiments on the SMOP test suite show that SparseDO is effective in generating offspring solutions of real variables. To verify the effect of binary differential operators on solution diversity, we conducted experiments on the scikit-feature repository. It can be seen from the three diversity indicators (the overall diversity of the population DP, genetic diversity DG, and individual diversity DI) that

Table 21 The time consumption of NSGAII, DEAGNG, ARMOEA, SparseEA, and RA-SparseDO on scikit-feature repository (all results were performed on 10 independent runs)

Dataset	ARMOEA	DAEA	DEAGNG	MOEAPSL	SparseEA	RA-SparseDO (Best2)	RA-SparseDO (Rand2)	RA-HSparseDO (Rand2Best2)
Lung	0.7405	0.2316	0.7040	0.2381	0.3227	0.5748	0.3944	0.4800
Lung-discrete	0.1640	0.1323	0.1644	0.1304	0.1497	0.2695	0.1931	0.2587
Lymphoma	0.6274	0.2657	0.6218	0.1996	0.3194	0.7045	0.5252	0.6076
nci9	1.3552	0.4281	1.3336	0.3173	8.9549	4.0904	2.8096	3.0510
ORL	0.3730	0.2014	0.3541	0.2169	0.1945	0.5093	0.2585	0.4144
Prostate-GE	0.9550	0.3467	0.9333	0.2602	4.1442	1.3462	1.2510	1.1689
TOX-171	1.1298	0.3050	1.1074	0.2968	4.6080	1.4550	1.2578	1.1473
warpAR10P	0.4440	0.2163	0.4266	0.1783	0.2150	0.3935	0.2926	0.3386
warpPIE10P	0.5168	0.2263	0.4825	0.1792	0.2966	0.4119	0.2917	0.3414
Yale	0.2913	0.1928	0.2850	0.1918	0.1643	0.3053	0.1792	0.2284
	6.5970	2.5462	6.4127	2.2085	19.3693	10.0604	7.4530	8.0364

SparseDO (Best2), SparseDO (Rand1), SparseDO (Rand2), HSparseDO (Rand1Best1), and HSparseDO (Rand2Best2) are effective for increasing the diversity of SparseEA. The ablation experiments show that both the ReliefF-based component and the difference operator-based component are effective for solving LSMFSPs. Comparative experiments for LSMFSPs are verified on scikit-feature repository. The experimental results show that the proposed algorithm is significantly better than ARMOEA, DEAGG, MOEAPSL and the original SparseEA for solving LSMFSPs. Meanwhile, experiment on UCI repository shows that the proposed algorithm also has certain advantages for small-scale datasets. In addition, since the ReliefF algorithm has reduced the dimensions first for LSMFSPs, the running times of the proposed algorithms are less than that of the original SparseEA.

Later work will study the other ways of combining difference operators with SparseEA. The combination of other traditional feature selection algorithms with meta-heuristic algorithms is also one of the key points in the future work.

Data availability Data sharing not applicable.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Rivera-López R, Mezura-Montes E, Canul-Reich J, Cruz-Chávez MA (2020) A permutational-based differential evolution algorithm for feature subset selection. Pattern Recognit Lett 133:86–93
2. Wang X-D, Chen R-C, Yan F (2019) High-dimensional data clustering using k-means subspace feature selection. J Netw Intell 4(3):80–87
3. Ibrahim RA, Abd Elaziz M, Ewees AA, El-Abd M, Lu S (2021) New feature selection paradigm based on hyper-heuristic technique. Appl Math Model 98:14–37
4. Forman G et al (2003) An extensive empirical study of feature selection metrics for text classification. J Mach Learn Res 3:1289–1305
5. Deng X, Li Y, Weng J, Zhang J (2019) Feature selection for text classification: a review. Multimed Tools Appl 78(3):3797–3816

6. Chen Y, Tao J, Wang J, Liao Z, Xiong J, Wang L (2019) The image annotation method by convolutional features from intermediate layer of deep learning based on internet of things. In: 2019 15th international conference on mobile ad-hoc and sensor networks (MSN). IEEE, pp 315–320
7. Wang Z, Dong J, Zhen J, Zhu F (2019) Template protection based on chaotic map and DNA encoding for multimodal biometrics at feature level fusion. J Inf Hiding Multimed Signal Process 10(1):1–10
8. Chaudhary V, Deshbhratar A, Kumar V, Paul D (2018) Time series based LSTM model to predict air pollutant's concentration for prominent cities in India, UDM
9. Lin W, Yang C, Zhang Z, Xue X, Haga R (2021) A quantitative assessment method of network information security vulnerability detection risk based on the meta feature system of network security data. KSII Trans Internet Inf Syst (TIIS) 15(12):4531–4544
10. Jung J, Park J, Cho S-J, Han S, Park M, Cho H-H (2021) Feature engineering and evaluation for android malware detection scheme. J Internet Technol 22(2):423–440
11. Molina LC, Belanche L, Nebot À (2002) Feature selection algorithms: a survey and experimental evaluation. In: 2002 IEEE international conference on data mining, proceedings. IEEE, pp 306–313
12. Pan J-S, Liu N, Chu S-C, Lai T (2021) An efficient surrogate-assisted hybrid optimization algorithm for expensive optimization problems. Inf Sci 561:304–325
13. Fister D, Fister I, Jagrič T, Brest J (2019) Wrapper-based feature selection using self-adaptive differential evolution. In: Zamuda A, Das S, Suganthan PN, Panigrahi BK (eds) Zamuda A, Das S, Suganthan PN, Panigrahi BK (eds) Swarm, evolutionary, and memetic computing and fuzzy and neural computing. Springer, pp 135–154
14. Banka H, Dara S (2015) A hamming distance based binary particle swarm optimization (hdbpso) algorithm for high dimensional feature selection, classification and validation. Pattern Recognit Lett 52:94–100
15. Ramírez-Gallego S, García S, Xiong N, Herrera F (2018) Belief: a distance-based redundancy-proof feature selection method for big data. arXiv preprint arXiv:1804.05774
16. Chaves R, Ramírez J, Górriz J, López M, Salas-Gonzalez D, Alvarez I, Segovia F (2009) Svm-based computer-aided diagnosis of the Alzheimer's disease using t-test nmse feature selection with feature correlation weighting. Neurosci Lett 461(3):293–297
17. Sun L, Zhang J, Ding W, Xu J (2022) Mixed measure-based feature selection using the fisher score and neighborhood rough sets. Appl Intell 15:1–25
18. Azhagusundari B, Thanamani AS et al (2013) Feature selection based on information gain. Int J Innov Technol Explor Eng (IJITEE) 2(2):18–21
19. Janecek A, Gansterer W, Demel M, Ecker G (2008) On the relationship between feature selection and classification accuracy. In: New challenges for feature selection in data mining and knowledge discovery. PMLR, pp 90–105
20. Chu SC, Xu XW, Yang SY, Pan JS (2022) Parallel fish migration optimization with compact technology based on memory principle for wireless sensor networks. Knowl Based Syst 241:108124
21. Pan J-S, Song P-C, Pan C-A, Abraham A (2021) The phasmatodea population evolution algorithm and its application in 5g heterogeneous network downlink power allocation problem. J Internet Technol 22(6):1199–1213
22. Agrawal P, Abutarboush HF, Ganesh T, Mohamed AW (2021) Metaheuristic algorithms on feature selection: a survey of one decade of research (2009–2019). IEEE Access 9:26766–26791
23. Hu P, Pan J-S, Chu S-C (2020) Improved binary grey wolf optimizer and its application for feature selection. Knowl Based Syst 195:105746
24. Fu G, Sun C, Tan Y, Zhang G, Jin Y (2020) A surrogate-assisted evolutionary algorithm with random feature selection for large-scale expensive problems. In: International conference on parallel problem solving from nature. Springer, pp 125–139
25. Ahmed S, Sheikh KH, Mirjalili S, Sarkar R (2022) Binary simulated normal distribution optimizer for feature selection: theory and application in COVID-19 datasets. Expert Syst Appl 200:116834
26. Too J, Mafarja M, Mirjalili S (2021) Spatial bound whale optimization algorithm: an efficient high-dimensional feature selection approach. Neural Comput Appl 33(23):16229–16250
27. Mafarja M, Aljarah I, Heidari AA, Faris H, Fournier-Viger P, Li X, Mirjalili S (2018) Binary dragonfly optimization for feature selection using time-varying transfer functions. Knowl Based Syst 161:185–204
28. BinSaeedan W, Alramlawi S (2021) Cs-bpso: hybrid feature selection based on chi-square and binary PSO algorithm for Arabic email authorship analysis. Knowl Based Syst 227:107224
29. Wang H, Wang W, Cui L, Sun H, Zhao J, Wang Y, Xue Y (2018) A hybrid multi-objective firefly algorithm for big data optimization. Appl Soft Comput 69:806–815
30. Li G, Wang G-G, Dong J, Yeh W-C, Li K (2021) Dlea: a dynamic learning evolution algorithm for many-objective optimization. Inf Sci 574:567–589
31. Tian Y, He C, Cheng R, Zhang X (2019) A multistage evolutionary algorithm for better diversity preservation in multiobjective optimization. IEEE Trans Syst Man Cybern Syst 51(9):5880–5894
32. Wang C, Wang Z, Tian Y, Zhang X, Xiao J (2021) A dual-population based evolutionary algorithm for multi-objective location problem under uncertainty of facilities. IEEE Trans Intell Transp Syst 23:7692–7707
33. Said LB, Bechikh S, Ghédira K (2010) The r-dominance: a new dominance relation for interactive evolutionary multicriteria decision making. IEEE Trans Evol Comput 14(5):801–818
34. Fan Z, Fang Y, Li W, Cai X, Wei C, Goodman E (2019) Moea/d with angle-based constrained dominance principle for constrained multi-objective optimization problems. Appl Soft Comput 74:621–633
35. Pan J-S, Liu N, Chu S-C (2022) A competitive mechanism based multi-objective differential evolution algorithm and its application in feature selection. Knowl Based Syst 245:108582
36. Al-Tashi Q, Abdulkadir SJ, Rais HM, Mirjalili S, Alhussian H, Ragab MG, Alqushaibi A (2020) Binary multi-objective grey wolf optimizer for feature selection in classification. IEEE Access 8:106247–106263
37. Zhang Y, Gong D-W, Gao X-Z, Tian T, Sun X-Y (2020) Binary differential evolution with self-learning for multi-objective feature selection. Inf Sci 507:67–85
38. Wang X-H, Zhang Y, Sun X-Y, Wang Y-L, Du C-H (2020) Multi-objective feature selection based on artificial bee colony: an acceleration approach with variable sample size. Appl Soft Comput 88:106041
39. Gao KZ, Suganthan PN, Chua TJ, Chong CS, Cai TX, Pan QK (2015) A two-stage artificial bee colony algorithm scheduling flexible job-shop scheduling problem with new job insertion. Expert Syst Appl 42(21):7652–7663
40. Li A-D, Xue B, Zhang M (2020) Multi-objective feature selection using hybridization of a genetic algorithm and direct multisearch for key quality characteristic selection. Inf Sci 523:245–265
41. Cheng F, Guo W, Zhang X (2018) Mofsrank: a multiobjective evolutionary algorithm for feature selection in learning to rank. Complexity 2018:1–14
42. Huang K, Aviyente S (2008) Wavelet feature selection for image classification. IEEE Trans Image Process 17(9):1709–1720
43. Pok G, Liu J-CS, Ryu KH (2010) Effective feature selection framework for cluster analysis of microarray data. Bioinformation 4(8):385

44. Sahni G, Mewara B, Lalwani S, Kumar R (2022) CF-PPI: centroid based new feature extraction approach for protein–protein interaction prediction. J Exp Theor Artif Intell 1–21
45. Zhang X, Tian Y, Cheng R, Jin Y (2016) A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization. IEEE Trans Evol Comput 22(1):97–112
46. Miguel Antonio L, Coello Coello CA (2016) Decomposition-based approach for solving large scale multi-objective problems. In: International conference on parallel problem solving from nature. Springer, pp 525–534
47. Qian H, Yu Y (2017) Solving high-dimensional multi-objective optimization problems with low effective dimensions. In: Proceedings of the AAAI conference on artificial intelligence, vol 31
48. Hong W, Tang K, Zhou A, Ishibuchi H, Yao X (2018) A scalable indicator-based evolutionary algorithm for large-scale multiobjective optimization. IEEE Trans Evol Comput 23(3):525–537
49. Li H, He F, Chen Y, Pan Y (2021) Mlfs-ccde: multi-objective large-scale feature selection by cooperative coevolutionary differential evolution. Memet Comput 13(1):1–18
50. Tian Y, Zhang X, Wang C, Jin Y (2019) An evolutionary algorithm for large-scale sparse multiobjective optimization problems. IEEE Trans Evol Comput 24(2):380–393
51. Tian Y, Lu C, Zhang X, Tan KC, Jin Y (2020) Solving large-scale multiobjective optimization problems with sparse optimal solutions via unsupervised neural networks. IEEE Trans Cybern 51(6):3115–3128
52. Tian Y, Lu C, Zhang X, Cheng F, Jin Y (2020) A pattern mining-based evolutionary algorithm for large-scale sparse multiobjective optimization problems. IEEE Trans Cybern 52:6784–6797
53. Zhang Y, Tian Y, Zhang X (2021) Improved sparseea for sparse large-scale multi-objective optimization problems. Complex Intell Syst 1–16
54. Elsayed SM, Sarker RA, Essam DL (2012) An improved self-adaptive differential evolution algorithm for optimization problems. IEEE Trans Ind Inform 9(1):89–99
55. Viktorin A, Senkerik R, Pluhacek M, Kadavy T, Zamuda A (2019) Distance based parameter adaptation for success-history based differential evolution. Swarm Evol Comput 50:100462
56. Brest J, Zamuda A, Boskovic B, Maucec MS, Zumer V (2009) Dynamic optimization using self-adaptive differential evolution. In: IEEE congress on evolutionary computation. IEEE, pp 415–422
57. Hou GP, Ma X (2010) A novel binary differential evolution for discrete optimization. In: Key engineering materials, vol 439. Trans Tech Publ, pp 1493–1498
58. He Y, Zhang F, Mirjalili S, Zhang T (2022) Novel binary differential evolution algorithm based on taper-shaped transfer functions for binary optimization problems. Swarm Evol Comput 69:101022
59. Deng C, Zhao B, Yang Y, Peng H, Wei Q (2011) Novel binary encoding differential evolution algorithm. In: International conference in swarm intelligence. Springer, pp 416–423
60. Hota AR, Pat A (2010) An adaptive quantum-inspired differential evolution algorithm for 0–1 knapsack problem. In: Second world congress on nature and biologically inspired computing (NaBIC). IEEE, pp 703–708
61. Pampara G, Engelbrecht AP, Franken N (2006) Binary differential evolution. In: IEEE international conference on evolutionary computation. IEEE, pp 1873–1879
62. Li Y-F, Sansavini G, Zio E (2013) Non-dominated sorting binary differential evolution for the multi-objective optimization of cascading failures protection in complex networks. Reliab Eng Syst Saf 111:195–205
63. Bidgoli AA, Rahnamayan S, Ebrahimpour-Komleh H (2019) Opposition-based multi-objective binary differential evolution for multi-label feature selection. In: International conference on evolutionary multi-criterion optimization. Springer, pp 553–564
64. Banitalebi A, Abd Aziz MI, Aziz ZA (2016) A self-adaptive binary differential evolution algorithm for large scale binary optimization problems. Inf Sci 367:487–511
65. Tian Y, Cheng R, Zhang X, Jin Y (2017) Platemo: a matlab platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput Intell Mag 12(4):73–87
66. Lin HE, Wang K, Guo-Bin LI, Jin H (1999) The analysis and research of genetic algorithms' population diversity. J Harbin Eng Univ 20:27–33
67. Tian Y, Cheng R, Zhang X, Cheng F, Jin Y (2017) An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility. IEEE Trans Evol Comput 22(4):609–622
68. Xu H, Xue B, Zhang M (2020) A duplication analysis-based evolutionary algorithm for biobjective feature selection. IEEE Trans Evol Comput 25(2):205–218
69. Liu Y, Ishibuchi H, Masuyama N, Nojima Y (2019) Adapting reference vectors and scalarizing functions by growing neural gas to handle irregular pareto fronts. IEEE Trans Evol Comput 24(3):439–453
70. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: Nsga-II. IEEE Trans Evol Comput 6(2):182–197