This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

Sparsity Preserved Pareto Optimization for Subset Selection

Lei Zhang, *Member, IEEE*, Xiang Sun, Haipeng Yang, and Fan Cheng

***Abstract*—Subset selection that selects a limited number of variables optimizing many given criteria, is a fundamental problem with various applications such as sparse regression and unsupervised feature selection. Among the existing algorithms for subset selection, evolutionary algorithms (EAs) have achieved great performance in obtaining the subsets with high quality. However, as the selected subsets are sparse, i.e., most variables of these solutions are zero, most existing EAs for subset selection problems pose great challenges to find the optimal solutions effectively and efficiently, especially when the size of original set is large. To this end, we propose the SPESS approach which is a sparsity preserved evolutionary algorithm for subset selection. To be specific, we design two sparsity preserved operators (i.e., sparsity preserved crossover operator and sparsity preserved mutation operator), which can be used in SPESS to ensure the sparsity of the generated solutions in the process of Pareto optimization. The main advantages of the proposed sparsity preserved operators are that they can obtain high quality solutions and improve the search efficiency of the algorithm. According to the experimental results on 12 sparse regression problems and 12 unsupervised feature selection problems, the proposed SPESS is superior over the state-of-the-arts in solving subset selection tasks and the effectiveness of the proposed two sparsity preserved operators is also verified.**

***Index Terms*—Subset selection, sparsity preserved Pareto optimization, multi-objective evolutionary algorithms, sparse regression, unsupervised feature selection.**

I. Introduction

In machine learning and data mining, subset selection, as an important problem, is to select a subset with the size at most k from a total set of n variables for maximizing (or minimizing) some given objective function f. This problem is generally NP-hard [1], [2], and occurs in many real applications, such as sparse regression [3], unsupervised feature selection [4], maximum coverage [5], influence maximization [6], sensor placement [7], document summarization [8], critical node detection [9], and so on.

To tackle the above fundamental issues, a number of subset selection approaches have been proposed [10], [11]. A representative work is convex relaxation method, which relaxes the original problem by replacing the set size constraint (i.e., the ℓ_0-norm constraint) with convex constraints (e.g., the ℓ_1-norm constraint [12] and the elastic net penalty [13]). However, the optimal solution of the relaxed problem could be distant to the optimal solution of the original problem [11]. Another representative work is the standard greedy algorithm, which iteratively selects one item with the largest marginal gain [10], [14]–[18]. Despite the greedy algorithms can perform well in many cases, they could be trapped in local optimum [19].

Recent works for escaping the local optimum in subset selection problem mainly concentrate on evolutionary algorithms (EAs) which have global search capabilities. The simple evolutionary multi-objective optimizer (SEMO) algorithm is an EA for maximizing multiple pseudo-Boolean functions simultaneously [20]–[22]. The SEMO algorithm is the preliminary multi-objective EA (MOEA) with theoretical analysis results due to its simplicity. By replacing the one-bit mutation operator in SEMO with the bit-wise mutation operator which searches globally, the global SEMO (GSEMO) algorithm is obtained [20]. In particular, the GSEMO is widely applied to multi-objective combinatorial optimization problems with monotone submodular functions and uniform cardinality constraints [23]–[25]. Based on GSEMO, a new method Pareto Optimization for Subset Selection (POSS) [11] by using bit-wise mutation operator can achieve the better performance than the greedy algorithms by maximizing the given objective and minimizing the subset size simultaneously. POSS can escape from local optima to achieve good-quality solutions compared with the greedy algorithms and the relaxation methods on the application of sparse regression. Inspired by POSS, more EAs are proposed to solve different subset selection problems. For example, for large-scale subset selection, the distributed setting of POSS has exhibited a good approximation ratio [26], [27]. For subset selection with noise [28] and dynamic cost constraints [29]–[32], they have also shown good approximation capabilities. Recently, Subset Selection by Pareto Optimization with Recombination (PORSS) is suggested to further enhance the performance of POSS, and can also attain optimal approximate solutions in practice for most subset selection instances [33].

Although these methods above based on Pareto optimization for subset selection reveal the superior performance compared to greedy and relaxation methods, they still encounter main challenges, i.e., the search space grows exponentially with the number of decision variables increases, especially on the high dimensional situations such as gene analysis [34] and image recognition [35]. On the contrary, one fact is that the final solution is usually very sparse in these large-scale subset selection

This work was supported by the National Natural Science Foundation of China (61976001, 62076001, 61876184), the Key Projects of University Excellent Talents Support Plan of Anhui Provincial Department of Education (gxyqZD2021089), the University Synergy Innovation Program of Anhui Province(GXXT-2020-050), and also supported by the Natural Science Foundation of Anhui Province (2008085QF309). *(Corresponding author: Fan Cheng.)*

L. Zhang, X. Sun, H. Yang and F. Cheng are with the Information Materials and Intelligent Sensing Laboratory of Anhui Province, the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, Hefei 230039, China (email: zl@ahu.edu.cn, sunx1997@163.com, haipengyang@126.com, chengfan@mail.ustc.edu.cn).

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

problems [36], [37]. For example, for the feature selection problem, the proportion of finally selected features ranges from 0.1% to 1% [38], which means the optimal solutions of the subset selection problem could be very sparse. In such cases, it is also indicated that the larger the number of decision variables, the more sparse the selected subset [39]. Due to the neglect of the sparsity of the generated solutions in the process of Pareto optimization, it is very challenging for the existing EAs to find higher quality solutions as well as have the good search efficiency. For example, the bit-wise mutation operator in many EAs without considering cardinality constraints is likely to pull the subset away from solutions of the original problem, i.e., approximately half the variables will be nonzero after a sufficient number of generations [40]. Therefore, the search efficiency of the algorithm is degraded by generating many infeasible solutions.

To tackle the issues above, we propose the SPESS algorithm, which employs Pareto optimization with the suggested sparsity preserved operators to solve subset selection problems effectively and efficiently. Specifically, the two sparsity preserved operators, i.e., sparsity preserved mutation and sparsity preserved crossover, are designed to ensure the sparsity of the generated solutions in the process of Pareto optimization. The proposed two sparsity preserved operators can not only obtain high quality solutions but also improve the search efficiency of the algorithm. Experimental study on two popular subset tasks (i.e., sparse regression and unsupervised feature selection) clearly reveals the superior performance of SPESS, especially on the large-scale subset selection problems. In addition, the effectiveness of the proposed two sparsity preserved operators is also verified in the experiments.

We start the next section of the paper by introducing some preliminaries on the studied problem and presenting main Pareto optimization techniques for solving subset selection. Then we present the proposed SPESS method and the empirical studies in the following sections. The final section concludes this paper and also gives the future work.

II. Preliminaries and Related Work

In this section, we first give some preliminaries about the subset selection problem, and then present the related work about Pareto optimization for the problem of subset selection.

A. Subset Selection

The notion of subset selection problem as presented in Definition 1 is to select a subset S of V such that a given objective f is optimized with the constraint $|S| \leq k$, where $|\cdot|$ denotes the size of a set. In addition, we only consider maximization since maximizing f is equivalent to minimizing $-f$.

Definition 1 (Subset selection). *Given all items $V = \{v_1, v_2, \cdots, v_n\}$, an objective function f and a budget k, the task of subset selection is to find a subset of at most k items maximizing f, i.e.,*

$$\arg\max_{S \subseteq V} f(S) \quad s.t. \quad |S| \leq k. \tag{1}$$

There are two typical applications of subset selection with monotone (i.e., sparse regression and unsupervised feature selection), but not necessarily submodular, objective functions, which will be investigated in this paper.

Sparse regression (SR) as presented in Application 1 is to find a sparse approximation solution to the linear regression problem, where the optimal solution vector can only have a few non-zero elements. Note that S and its index set $\{i | v_i \in S\}$ are not distinguished for notational convenience. Without loss of generality, we assume that all random variables are normalized to have the expectation of 0 and the variance of 1. Thus, $R^2_{z,S}$ is simplified to be $1 - \mathrm{MSE}_{z,S}$.

Application 1 (Sparse regression [41]). *Given all observation variables $V = \{v_1, v_2, \cdots, v_n\}$, a predictor variable z and a budget k, the task of sparse regression is to find at most k variables from V maximizing the squared multiple correlation, i.e.,*

$$\arg\max_{S \subseteq V} R^2_{z,S} = 1 - \mathrm{MSE}_{z,S} \quad s.t. \quad |S| \leq k,$$

where $\mathrm{MSE}_{z,S}$ denotes the mean squared error, i.e.,

$$\mathrm{MSE}_{z,S} = \min_{\alpha \in \mathbb{R}^{|S|}} \mathbb{E}[(z - \sum_{i \in S} \alpha_i v_i)^2].$$

As another typical application of subset selection, unsupervised feature selection (UFS) as presented in Application 2 is to select at most k columns from all the n columns of a matrix A to best approximate A. The goodness of approximation is measured by the sum of squared errors between the original matrix A and the approximation SS^+A based on the selected columns of S.

Application 2 (Unsupervised feature selection [42]). *Given a matrix $A \in \mathbb{R}^{m \times n}$ and a budget k, the task of unsupervised feature selection is to find a submatrix S of A with at most k columns minimizing $\|A - SS^+A\|_F^2$, i.e.,*

$$\arg\min_{S: a\ submatrix\ of\ A} \|A - SS^+A\|_F^2 \quad s.t. \quad |S| \leq k.$$

where SS^+ is the projection matrix onto the space spanned by the columns of S.

For the ease of evaluating a submatrix S in Section IV, the UFS problem is equivalently reformulated as *ErrorRatio* [33]:

$$ErrorRatio = \frac{\|A - SS^+A\|_F^2}{\|A - A_k\|_F^2},$$

where A_k denotes the best rank-k approximation to A via SVD.

B. Related Works of Pareto Optimization for Subset Selection

Subset selection can be formulated as maximizing a monotone submodular function problem under a cardinality constraint. In recent years, many Pareto optimization algorithms have been proposed to approximate the optimal solution of the subset selection problem.

For example, one representative algorithm named POSS [11] is developed for subset selection with theoretical guarantee, using the idea of multi-objective evolutionary optimization. The main idea of POSS is to transform the original subset

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

TABLE I
AN OVERVIEW OF THE EXISTING PARETO OPTIMIZATION ALGORITHMS FOR SUBSET SELECTION.

Category	Method	Strategies
Cardinality constraints, $\|S\| \leq k$	POSS [11]	Bit-wise mutation
	PONSS [28]	Bit-wise mutation, Noise-aware comparison
	PPOSS [43]	Bit-wise mutation, Parallel iteration
	DPOSS [26]	Bit-wise mutation, Two-round divide
	DPONSS [27]	Bit-wise mutation, Noise-aware comparison, Two-round divide
	PORSS [33]	Bit-wise mutation, One-point crossover, Uniform crossover
	EPORSS [44]	Bit-wise mutation, Optimizing robust objective
Monotone cost constraints, $c(S) \leq k$	POMC [29], [30]	Bit-wise mutation
	EAMC [31]	Bit-wise mutation, Optimizing surrogate objective
	FPOMC [32]	Bit-wise mutation, Greedy selection

selection problem as a bi-objective optimization problem, i.e., maximizing the objective function f as well as minimizing the subset size $|S|$, then solve the problem by an EA, and finally output the best feasible solution from the generated non-dominated solution set. In each iteration, the new solution is generated by the bit-wise mutation operator.

In experiments, POSS can achieve significantly better performance than the greedy algorithms and relaxation methods on the application of sparse regression. For robust subset selection (which aims to select a subset that performs best on the worst objective function) [44], POSS can be more efficient on the application of robust influence maximization.

Although the above POSS is a powerful approximation solver in finding optimal solutions, it is still inefficient in real applications of subset selection with a noisy evaluation. The PONSS algorithm [28], which introduces a noise-aware comparison strategy into POSS, addresses the subset selection problem with monotone objective functions under multiplicative and additive noises. For the two noise models, the two types of θ-domination are used to compare two solutions. θ-domination intuitively means that a solution better than the other requires its objective value to be greater by at least a threshold $\frac{1+\theta}{1-\theta}$ or 2θ ($\theta \in [0, 1]$). Since PONSS can simultaneously keep two solutions with close noisy objective values, it clearly outperforms POSS on influence maximization and sparse regression problems.

To solve large-scale real-world problems, a parallel version of POSS (PPOSS) [43] is presented by generating as many solutions as the number of processors N instead of generating one solution at a time. The running time of PPOSS decreases linearly as the number of processors increases, eventually reaching a constant. That is, PPOSS can theoretically achieve linear speedup in the number of iterations without sacrificing the solution quality. Experimental results suggest that the asynchronous implementation of PPOSS is more efficient, while there is almost no loss in solution quality.

However, the large-scale data set cannot be stored on one single machine, and must be distributed among a set of machines. Naturally, a distributed version of POSS (DPOSS) [27] based on the divide and conquer idea apply to large-scale subset selection tasks. DPOSS is implemented in the following two rounds: the ground set V is partitioned into $\{V_1, V_2, \cdots, V_m\}$ where $(V = \cup_{i=1}^{m} V_i) \wedge (\forall i \neq j : V_i \cap V_j = \emptyset)$, and these m sets are distributed over m machines while running POSS in parallel on each machine; then POSS is run on the union of these m subsets generated in the first round, and DPOSS finally returns the best one among above $(m + 1)$ subsets. Extensive experiments using Spark on real-world data sets with size ranging from thousands to millions show that DPOSS can achieve competitive performance to the centralized POSS.

Since recombination (or called crossover) is a core feature of EAs, the PORSS algorithm [33] for subset selection by introducing recombination (including one-point recombination and uniform recombination) into POSS can achieve the optimal approximation solution. Meanwhile, it discloses theoretically insight on the effect of recombination, and thus leading to better exploration.

In addition, the resources often vary in real-world subset selection scenarios, i.e., the budget often changes over time, requiring the algorithms to adapt the solutions quickly. However, when the budget changes dynamically (i.e., when k increases to k'), all these algorithms either achieve arbitrarily bad approximation guarantees, or require a long running time. To this end, novel Pareto optimization algorithms (e.g., POMC [29], [30], EAMC [31], FPOMC [32]) for subset selection with dynamic cost constraints can maintain the polynomial-time approximation guarantee. For instance, FPOMC, which combines the merits of the generalized greedy algorithm and POMC (which simultaneously maximizes the original objective function and minimizes the approximate cost function), can maintain the best approximation capability efficiently.

Table I provides an overview of the main algorithms discussed above. In this table, the column $Category$ represents the subset selection problem under different constraints used in the corresponding algorithm. The last column shows the operators and strategies involved in the corresponding algorithms for subset selection. As shown in Table I, we can find that all of the above existing Pareto optimization approaches are used to generate solutions by means of the bit-wise mutation, where the sparsity preservation of solutions is rarely considered in the process of Pareto optimization.

Despite these Pareto optimization methods have shown their effectiveness in tackling the different kinds of subset selection problems. However, most of them neglect the fact that the final solution is usually very sparse, which diminishes the algorithm's approximation capability. Hence, in this paper we propose a sparsity preserved Pareto optimization for subset

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

selection, where the two proposed sparsity preserved operators are suggested to ensure that the sparsity of the solution is maintained. Moreover, we claim that the proposal method can significantly outperform other state-of-the-art methods on the applications of sparse regression and unsupervised feature selection.

III. The Proposed SPESS Algorithm

In this section, we first introduce the proposed Sparsity Preserved Evolutionary Algorithm for Subset Selection, called SPESS. After that, two main components of SPESS are explained one by one.

A. General Framework of SPESS

Based on Pareto optimization, the subset selection problem is usually transformed into a bi-objective optimization problem. That is, a subset S of V can be represented by a binary vector $\boldsymbol{x} \in \{0,1\}^n$, where $x_i = 1$ if and only if the item $v_i \in S$, and $x_i = 0$ otherwise. Note that $\boldsymbol{x} \in \{0,1\}^n$ and its corresponding subset will not be distinguished for notational convenience. The SPESS method reformulates the original problem Eq.(1) as a bi-objective maximization problem:

$$\arg\max\nolimits_{\boldsymbol{x}\in\{0,1\}^n} (f_1(\boldsymbol{x}), f_2(\boldsymbol{x})), \tag{2}$$

where

$$f_1(\boldsymbol{x}) = \begin{cases} -\infty, & |\boldsymbol{x}| > k, \\ f(\boldsymbol{x}), & \text{otherwise} \end{cases}, \qquad f_2(\boldsymbol{x}) = -|\boldsymbol{x}|.$$

It means that SPESS maximizes the original objective function f and minimizes the subset size $|\boldsymbol{x}|$ simultaneously. Note that setting $f_1 = -\infty$ for $|\boldsymbol{x}| > k$ is to exclude infeasible solutions whose cardinality is larger than k.

To compare two solutions in bi-objective optimization, SPESS uses the Pareto dominance relationship. For two solutions $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, $\boldsymbol{x}_1$ *weakly dominates* $\boldsymbol{x}_2$, denoted as $\boldsymbol{x}_1 \succeq \boldsymbol{x}_2$, if $(f_1(\boldsymbol{x}_1) \geq f_1(\boldsymbol{x}_2)) \wedge (f_2(\boldsymbol{x}_1) \geq f_2(\boldsymbol{x}_2))$; $\boldsymbol{x}_1$ *dominates* $\boldsymbol{x}_2$, denoted as $\boldsymbol{x}_1 \succ \boldsymbol{x}_2$, if $\boldsymbol{x}_1 \succeq \boldsymbol{x}_2$ and either $(f_1(\boldsymbol{x}_1) > f_1(\boldsymbol{x}_2))$ or $(f_2(\boldsymbol{x}_1) > f_2(\boldsymbol{x}_2))$; they are *incomparable*, if neither $\boldsymbol{x}_1 \succeq \boldsymbol{x}_2$ nor $\boldsymbol{x}_2 \succeq \boldsymbol{x}_1$.

As described in Algorithm 1, SPESS uses a randomized iterative procedure to solve the subset selection problem. It starts from the solution $\{0\}^n$ representing an empty set (line 1) and then iteratively tries to improve the solutions in the population P (lines 2-16). In each iteration, two solutions are selected randomly from P with replacement (line 3). If the condition in line 4 is satisfied (i.e., $R(P) < k$, where R is the rank of a matrix), we apply bit-wise mutation [11] with probability $1/n$ (line 5) to generate solutions. If the size of each individual in the population corresponds to $[k] = \{1, 2, \cdots, k\}$ one-to-one (i.e., $R(P) = k$), the proposed sparsity preserved operators are presented to generate two new solutions sequentially in lines 7 and 8, which are the key components of the proposed method with the aim to improve the search efficiency and produce higher quality solutions. The Pareto dominance relationship is used to update the population P (lines 10-14), i.e., P always contains incomparable solutions. Note that this scheme of selecting and updating one solution in P for each evolution is very different from the traditional multi-objective evolutionary algorithms [45], which select and update all solutions in P for each evolution. After running T iterations, the best solution w.r.t. the original problem Eq.(1) is selected from P.

Algorithm 1 SPESS algorithm

```
Input: universal set $V = \{v_1, v_2, \cdots, v_n\}$, objective $f : 2^V \to \mathbb{R}$, budget $k$
Parameter: the number $T$ of iterations
Output: a solution $\boldsymbol{x} \in \{0,1\}^n$ of a ground set $V$ with $|\boldsymbol{x}| \leq k$
 1: Let $\boldsymbol{x} = \{0\}^n$, $P = \{\boldsymbol{x}\}$ and $t = 0$.
 2: while $t < T$ do
 3:     Select $\boldsymbol{x}$, $\boldsymbol{y}$ from $P$ randomly with replacement.
 4:     if $R(P) < k$ then
 5:         Apply bit-wise mutation on $\boldsymbol{x}$, $\boldsymbol{y}$ to generate $\boldsymbol{x}''$, $\boldsymbol{y}''$.
 6:     else
 7:         Apply $\underline{\text{the proposed sparsity preserved crossover}}$ on $\boldsymbol{x}$, $\boldsymbol{y}$ to generate $\boldsymbol{x}'$, $\boldsymbol{y}'$.
 8:         Apply $\underline{\text{the proposed sparsity preserved mutation}}$ on $\boldsymbol{x}'$, $\boldsymbol{y}'$ to generate $\boldsymbol{x}''$, $\boldsymbol{y}''$.
 9:     end if
10:     for $\forall \boldsymbol{q} \in \{\boldsymbol{x}'', \boldsymbol{y}''\}$ do
11:         if $\nexists \boldsymbol{z} \in P$ such that $\boldsymbol{z} \succ \boldsymbol{q}$ then
12:             $P = (P \setminus \{\boldsymbol{z} \in P \mid \boldsymbol{q} \succeq \boldsymbol{z}\}) \cup \{\boldsymbol{q}\}$
13:         end if
14:     end for
15:     $t = t + 1$
16: end while
17: return $\arg\max_{\boldsymbol{x}\in P, |\boldsymbol{x}|\leq k} f(\boldsymbol{x})$
```

As shown in the process of SPESS, the solutions maintained by the population P are not comparable to each other. Therefore, each value of an objective corresponds to at most one solution in P. By the procedure of updating the population P in lines 10-14, $R(P)$ is always unchanging or increasing. Moreover, once $R(P)$ reaches k, it does not change anymore in the subsequent iterations. In other words, there will be no effect of the bit-wise mutation as soon as the proposed sparsity preserved operators are applied.

From the above, the two proposed sparsity preserved operators (i.e., sparsity preserved crossover and sparsity preserved mutation) determine whether SPESS can produce solutions with sparsity. In the following, we will theoretically introduce the notion and analyze the influence of the sparsity preserved operators of SPESS.

B. Sparsity Preserved Crossover Operator

To make that the crossover of two individuals $\boldsymbol{x}$ and $\boldsymbol{y}$ produces two sparse solutions $\boldsymbol{x}'$ and $\boldsymbol{y}'$, a kind of *sparsity preserved crossover* (SPC) based on expectation is designed as follows:

$$(\boldsymbol{x}'_i, \boldsymbol{y}'_i) = \begin{cases} (1,0), & (\boldsymbol{x}_i = 0) \wedge (\boldsymbol{y}_i = 1) \wedge (rand < p_{01}) \\ (0,1), & (\boldsymbol{x}_i = 1) \wedge (\boldsymbol{y}_i = 0) \wedge (rand < p_{10}) \\ (\boldsymbol{x}_i, \boldsymbol{y}_i), & \text{otherwise} \end{cases}, \tag{3}$$

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

where p_{01} denotes the probability to exchange 0-bit in $\boldsymbol{x}$ and 1-bit in $\boldsymbol{y}$, and p_{10} denotes the probability to exchange 1-bit in $\boldsymbol{x}$ and 0-bit in $\boldsymbol{y}$. p_{01} and p_{10} are designed to control the sparsity of $\boldsymbol{x}'_i$ and $\boldsymbol{y}'_i$, respectively.

It should be noted that for one-point and uniform crossover, the sparsity of the two offspring solutions may be far from the sparsity of the two parent solutions. For example, for solutions 0^n and 1^n, two new solutions $1^\rho 0^{n-\rho}$ and $0^\rho 1^{n-\rho}$ must be generated by one-point crossover, where the ρ-th bit indicates the position of a randomly selected crosspoint. And two new solutions $1^\beta 0^{n-\beta}$ and $0^\beta 1^{n-\beta}$ must be generated by uniform crossover, where β indicates the number of 0 and 1 exchanged. It is obvious that the two new solutions cannot maintain the same sparsity as the two parent solutions by one-point and uniform crossover.

To this end, for ensuring the sparsity of offsprings, we make the expectation of exchanged 0-bit of $\boldsymbol{x}$ and the corresponding bit of $\boldsymbol{y}$ equal to the expectation of exchanged 1-bit of $\boldsymbol{x}$ and the corresponding bit of $\boldsymbol{y}$. At the same time, the sum of the above two exchange expectations should be the same as the exchange expectation of $\boldsymbol{x}$ and $\boldsymbol{y}$. In other words, we have

$$\begin{cases} p_{01} \cdot a^* = p_{10} \cdot b^* \\ p_{01} \cdot a^* + p_{10} \cdot b^* = p_c \cdot h \end{cases},$$

where a^* is the number of 0-bit in $\boldsymbol{x}$ and 1-bit in $\boldsymbol{y}$ exchanged, b^* is the number of 1-bit in $\boldsymbol{x}$ and 0-bit in $\boldsymbol{y}$ exchanged, and h denotes the total number of bits to be exchanged between $\boldsymbol{x}$ and $\boldsymbol{y}$. Note that, $h = |\boldsymbol{x} \oplus \boldsymbol{y}| = a^* + b^*$, and $a^* = |\boldsymbol{y} \otimes (\boldsymbol{x} \oplus \boldsymbol{y})|$, where $\oplus$ is an xor operator, and $\otimes$ is a bit-wise multiplying operator. So we have

$$\begin{cases} p_{01} = \dfrac{p_c \cdot h}{2a^*} \\ p_{10} = \dfrac{p_c \cdot h}{2b^*} \end{cases}.$$

Hence, to ensure the sparsity of offsprings $\boldsymbol{x}'$ and $\boldsymbol{y}'$, the crossover probability p_c should be set to at least $\frac{1}{k}$ since $h \leq 2k$. An example to illustrate the SPC operator is shown in Fig. 1, where the cardinality of the two parents $\boldsymbol{x}$ and $\boldsymbol{y}$ are 5 and 3, respectively. In this example, we set the value of p_c to $\frac{1}{2}$. Then, we can calculate $p_{01} = \frac{3}{4}$ and $p_{10} = \frac{3}{8}$. Thus, two offsprings $\boldsymbol{x}'$ and $\boldsymbol{y}'$ with cardinality of 5 and 3 respectively can be generated by using SPC.

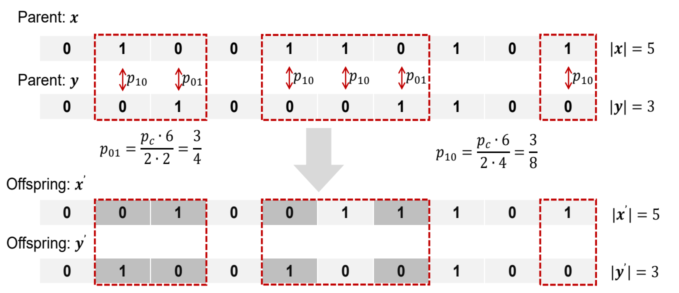


Fig. 1. An example for illustrating SPC, where two offspring solutions $\boldsymbol{x}'$ and $\boldsymbol{y}'$ with cardinality of 5 and 3 are generated by the parent solutions $\boldsymbol{x}$ and $\boldsymbol{y}$.

C. Sparsity Preserved Mutation Operator

In order to ensure that offspring solutions are as sparse as possible during the evolutionary process, the *sparsity preserved mutation* (SPM) operator is proposed to generate new mutation solutions in the algorithm. Considering the i-th bit in a parent solution $\boldsymbol{x}_i$ and an offspring solution $\boldsymbol{x}'_i$, SPM is defined based on the following equations:

$$\boldsymbol{x}'_i = \begin{cases} 1, & (\boldsymbol{x}_i = 0) \wedge (rand < p_0) \\ 0, & (\boldsymbol{x}_i = 1) \wedge (rand < p_1) \\ \boldsymbol{x}_i, & \text{otherwise} \end{cases}, \tag{4}$$

where p_0 and p_1 denote the probabilities to flip each zero variable and each nonzero variable to control the sparsity of $\boldsymbol{x}'_i$, respectively.

For bit-wise mutation, mostly existing algorithms are bound to generate many infeasible solutions under a cardinality constraint problem after a sufficient number of generations, i.e., the ℓ_0-norm of the offspring solution is in contradiction with the cardinality constraint k. For example, for a solution $\boldsymbol{x} = 1^\varphi 0^{n-\varphi}$ ($1 \leq \varphi \leq k$, where φ denotes the number of nonzero elements in $\boldsymbol{x}$) and the mutation probability p, the expectation of the number of nonzero variables will be changed to

$$\begin{aligned} &\mathbb{E}^t_{nonzero} = \mathbb{E}^{t-1}_{nonzero} + p \cdot (n - \mathbb{E}^{t-1}_{nonzero}) - p \cdot \mathbb{E}^{t-1}_{nonzero} \\ &= (1-2p)\mathbb{E}^{t-1}_{nonzero} + pn \\ &= (1-2p)^t \varphi + [(1-2p)^{t-1} pn + \cdots + (1-2p)pn + pn] \\ &= (1-2p)^t \varphi + [1 - (1-2p)^t] \frac{n}{2}, \end{aligned}$$

where the mutation is performed for t times. When $\varphi = k$ and $t \to +\infty$, $\lim \mathbb{E}^t_{nonzero} = n/2 \gg k$. That is, since k is a small constant, the population cannot be updated after a moderate number of iterations by the addition of offspring solutions. According to the above conclusion, there exists one critical problem that the cardinality of offspring solutions increases with the iteration proceeds so as to existing operators can hardly search for optimal solutions. Note that the above conclusion is applied to EAs by using bit-wise mutation without considering cardinality constraints. Similarly, the above problem can also exist in some EAs by using bit-wise mutation considering cardinality constraints. For example, POSS and its variants are more likely to produce infeasible solutions (whose sizes are larger than k) when individuals that satisfy cardinality k to $2k$ are selected as parents for evolution, leading to the decrease of the search ability. In a word, for the subset selection problem, the bit-wise mutation will produce many infeasible solutions as the evolution continues, i.e., the sparsity of offsprings is hardly preserved, thus reducing the search efficiency of the algorithm.

To this end, for ensuring the sparsity of offsprings, we make the expectation of the number of flipped nonzero variables equal to the expectation of the number of flipped zero variables. In other words, for a solution $\boldsymbol{x} = 0^a 1^b$, we make $p_0 \cdot a = p_1 \cdot b$, where a and b denote the number of 0-bit and 1-bit in $\boldsymbol{x}$

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

respectively. In addition, $p_0 \cdot a + p_1 \cdot b = p_m \cdot n$ should be met, where p_m denotes the mutation probability. So we have

$$\begin{cases} p_0 = \dfrac{p_m \cdot n}{2a} \\ p_1 = \dfrac{p_m \cdot n}{2b} \end{cases}.$$

Since a very sparse solution (i.e., $b \ll n$) may make $p_1 > 1$ and $b < a$, to ensure the sparsity of the offspring $\boldsymbol{x}'$, the value of mutation probability p_m should be not greater than $\frac{2b}{n}$. An example to illustrate the SPM operator is shown in Fig. 2, where the cardinality of the parent $\boldsymbol{x}$ is 3. In this example, we set the value of p_m to $\frac{1}{10}$. Then, we can calculate $p_0 = \frac{1}{14}$ and $p_1 = \frac{1}{6}$. Thus, the offspring $\boldsymbol{x}'$ with cardinality of 3 can be generated by using SPM.

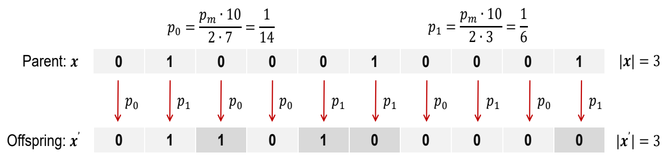

$p_0 = \frac{p_m \cdot 10}{2 \cdot 7} = \frac{1}{14}$
$p_1 = \frac{p_m \cdot 10}{2 \cdot 3} = \frac{1}{6}$
Parent: x
0 1 0 0 0 1 0 0 0 1
$|x| = 3$
p_0 p_1 p_0 p_0 p_1 p_1 p_0 p_1 p_0 p_1
Offspring: x'
0 1 1 0 1 0 0 0 0 0
$|x'| = 3$

Fig. 2. An example for illustrating SPM, where the offspring solution $\boldsymbol{x}'$ with cardinality of 3 is generated by the parent solution $\boldsymbol{x}$.

In summary, the sparsity preserved operators can maintain the size of the offspring solutions, leading to decreasing the size of population comparing to POSS and it's variants. Intuitively, the proposed sparsity preserved operators can shrink the search space, however, they can be used to guarantee the algorithm producing more feasible solutions (whose sizes are not larger than k) in the process of evolution. It is worth noting that the proposed operators could produce a few infeasible solutions with a small probability. To this end, the proposed algorithm with the sparsity preserved operators can speed up the approximation of the optimal solution, thus can increase the search ability to some extent. The experimental results in Section IV can further verify the good search ability of the proposed algorithm.

TABLE II
PARAMETER SETTINGS OF DIFFERENT ALGORITHMS.

Algorithm	Parameter values
POSS [11]	$T = \lfloor 2ek^2 n \rfloor$, $p_m = 1/n$
PORSSo [33]	$T = \lfloor ek^2 n \rfloor$, $p_m = 1/n$, $p_c = 1/n$
PORSSu [33]	$T = \lfloor ek^2 n \rfloor$, $p_m = 1/n$, $p_c = 1/2$
SPESS	$T = \lfloor ek^2 n \rfloor$, $p_m = 1/n$, $p_c = 1/2$

IV. Experimental Study

In this section, we empirically investigate the actual performance of SPESS on the two typical applications of sparse regression (SR) and unsupervised feature selection (UFS) with various real-world data sets[1] with 1,000 to 30,000 decision variables (i.e., the third column of DataSet in Table III and Table IV). In the following, we first introduce the experimental settings, including representative comparison algorithms and the adopted parameter settings, and then compare the experimental results with baseline algorithms, finally discuss the effectiveness of the suggested strategies and the generalization performance of the proposed SPESS.

[1]The data sets are downloaded from https://jundongl.github.io/scikit-feature/datasets.html and http://archive.ics.uci.edu/ml/datasets.php

A. Experimental Settings

1) Comparison Algorithms: The performance of the SPESS is compared with two state-of-the-art Pareto optimization algorithms with theoretical guarantee for subset selection, namely, POSS [11] and PORSS [33]. POSS is a representative algorithm by using the mutation operator, whereas PORSS is another one with recombination and mutation operators. Note that PORSSo and PORSSu denote PORSS with one-point and uniform recombination, respectively. The state-of-the-art POSS and PORSS are chosen as the comparison algorithms in our experiments, since all of them have the ability of tackling the general subset selection problem with theoretical guarantee. In addition, it has been verified that POSS and PORSS have better performance compared with that of the convex relaxation methods (e.g., LASSO [12], SCAND [46], MCP [47]) and the greedy algorithms (e.g., forward regression [3], orthogonal matching pursuit [48], recursive feature elimination [49]).

2) Parameter Settings: For the two baseline algorithms, to make a fair comparison, we adopt the recommended parameter values suggested in their original papers. Table II lists the details of the parameters settings for different algorithms. Since all comparison algorithms are evolutionary ones, for the fairness of comparison, the number T of iterations of SPESS is set to $\lfloor ek^2 n \rfloor$ as also suggested in POSS (which requires only one objective evaluation in each iteration) and PORSS (which requires two objective evaluations in each iteration), where n denotes the size of decision variables in the data set. In order to guarantee the sparsity of offspring solutions during the evolution of SPESS, we set $p_m = 1/n < 2b/n$ and $p_c = 1/2 > 1/k$ according to Section III, where b denotes the cardinality of an arbitrary solution and $1 \leq b \leq k$. Note that p_{01}, p_{10}, p_0 and p_1 can be directly obtained according to p_m and p_c in Section III-B and III-C. The number of variables in the selected subset k is set to 8 as suggested by PORSS [33] in our experiments. For these comparison algorithms, we run them 20 times independently and report the average values of evaluation metrics.

B. Comparison between SPESS and Baselines

In this subsection, we perform extensive experiments of comparison algorithms on two applications, namely Sparse Regression and Unsupervised Feature Selection, for evaluation.

For Sparse Regression, we use $R^2_{z,S}$ (the calculation of this metric can be found in Application 1) to evaluate a subset S of variables, and the larger the better. The results are summarized in Table III. The rank of each algorithm on each data set reflects that SPESS is significantly better than other algorithms, according to the average of Table III, where the rank is computed as in [50]. Besides, we provide the Friedman test [51] with Posthoc-analysis, due to the fact that it is a good

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

TABLE III
SPARSE REGRESSION: THE R^2 VALUE (THE LARGER, THE BETTER) OF THE COMPARED ALGORITHMS ON 12 DATA SETS FOR $k = 8$. THE MEAN±STD. IS REPORTED FOR RANDOMIZED ALGORITHMS. IN EACH DATA SET, THE LARGEST VALUES ARE BOLDED. THE COUNT OF DIRECT WIN DENOTES THE NUMBER OF DATA SETS ON WHICH POSS HAS A LARGER R^2 VALUE THAN THE CORRESPONDING ALGORITHM (1 TIE IS COUNTED AS 0.5 WIN), WHERE SIGNIFICANT CELLS BY THE *sign-test* WITH CONFIDENCE LEVEL 0.05 ARE BOLDED. THEN, THE BEST RANK IS BOLDED AND THE R^2 OBTAINED BY DIFFERENT ALGORITHMS WITH FRIEDMAN TEST. IN THE LAST ROW, THE COHEN'S D IS APPLIED TO CALCULATE THE DIFFERENCE BETWEEN DIFFERENT ALGORITHMS.

Dataset	#Inst	#Feat	POSS	PORSSo	PORSSu	SPESS
coil20	1,440	1,024	0.664±0.003	0.666±0.003	**0.667±0.002**	0.666±0.001
ORL2048	400	2,048	0.480±0.004	0.483±0.004	**0.485±0.003**	**0.485±0.002**
lung	203	3,312	0.873±0.004	0.874±0.006	**0.875±0.006**	**0.875±0.004**
lymphoma	96	4,026	0.846±0.009	0.845±0.010	0.845±0.011	**0.851±0.008**
DLBCL	77	5,469	0.903±0.009	0.902±0.009	0.902±0.008	**0.904±0.010**
DriveFace	606	6,400	0.671±0.009	0.681±0.007	0.682±0.006	**0.683±0.006**
nci9	60	9,712	0.896±0.014	0.896±0.007	0.893±0.008	**0.903±0.009**
arcene	200	10,000	0.607±0.008	0.609±0.010	**0.617±0.017**	**0.617±0.010**
11Tumor	174	12,533	0.818±0.008	0.819±0.010	0.818±0.010	**0.825±0.008**
smallNORB600	600	18,432	0.622±0.012	0.627±0.005	0.629±0.007	**0.632±0.007**
SMKCAN187	187	19,993	0.610±0.010	0.609±0.014	0.615±0.012	**0.617±0.008**
GLI_85	85	22,283	0.915±0.007	0.918±0.011	0.915±0.009	**0.921±0.008**
POSS: Count of direct win			—	**3.5**	**3.5**	**0**
Average rank			3.38	2.92	2.46	**1.25**
p-value			0.0005	0.0009	0.0196	—
Average Cohen's d			0.786	0.470	0.367	—

TABLE IV
UNSUPERVISED FEATURE SELECTION: THE $ErrorRatio$ VALUE (THE SMALLER, THE BETTER) OF THE COMPARED ALGORITHMS ON 12 DATA SETS FOR $k = 8$. THE MEAN±STD. IS REPORTED FOR RANDOMIZED ALGORITHMS. IN EACH DATA SET, THE SMALLEST VALUES ARE BOLDED. THE COUNT OF DIRECT WIN DENOTES THE NUMBER OF DATA SETS ON WHICH POSS HAS A SMALLER $ErrorRatio$ VALUE THAN THE CORRESPONDING ALGORITHM (1 TIE IS COUNTED AS 0.5 WIN), WHERE SIGNIFICANT CELLS BY THE *sign-test* WITH CONFIDENCE LEVEL 0.05 ARE BOLDED. THEN, THE BEST RANK IS BOLDED AND THE $ErrorRatio$ OBTAINED BY DIFFERENT ALGORITHMS WITH FRIEDMAN TEST. IN THE LAST ROW, THE COHEN'S D IS APPLIED TO CALCULATE THE DIFFERENCE BETWEEN DIFFERENT ALGORITHMS.

Dataset	#Inst	#Feat	POSS	PORSSo	PORSSu	SPESS
Yale	165	1,024	1.303±0.010	1.291±0.005	1.286±0.003	**1.282±0.003**
colon-cancer	62	2,000	1.288±0.008	1.274±0.007	1.273±0.010	**1.269±0.008**
warpAR10P	130	2,400	1.359±0.014	1.323±0.012	1.315±0.009	**1.313±0.008**
warpPIE10P	210	2,420	1.313±0.012	1.288±0.009	1.278±0.009	**1.270±0.012**
cifar10-600	600	3,072	1.397±0.007	1.388±0.006	1.383±0.007	**1.377±0.008**
GLIOMA	50	4,434	1.187±0.010	1.176±0.009	1.171±0.007	**1.166±0.006**
Brain1	90	5,920	1.271±0.014	1.264±0.012	1.263±0.014	**1.256±0.009**
Prostate_GE	102	5,966	1.103±0.009	1.098±0.007	1.095±0.008	**1.094±0.003**
ALLAML	72	7,129	1.101±0.005	1.095±0.004	1.094±0.003	**1.092±0.003**
orlraws10P	100	10,304	1.250±0.009	1.240±0.008	1.234±0.004	**1.231±0.003**
CLL_SUB_111	111	11,340	1.256±0.016	1.247±0.010	1.237±0.005	**1.232±0.005**
gse19804	120	21,000	1.171±0.011	1.161±0.009	1.158±0.009	**1.152±0.009**
POSS: Count of direct win			—	**0**	**0**	**0**
Average rank			4.00	3.00	2.00	**1.00**
p-value			0.0005	0.0005	0.0005	—
Average Cohen's d			2.352	1.177	0.666	—

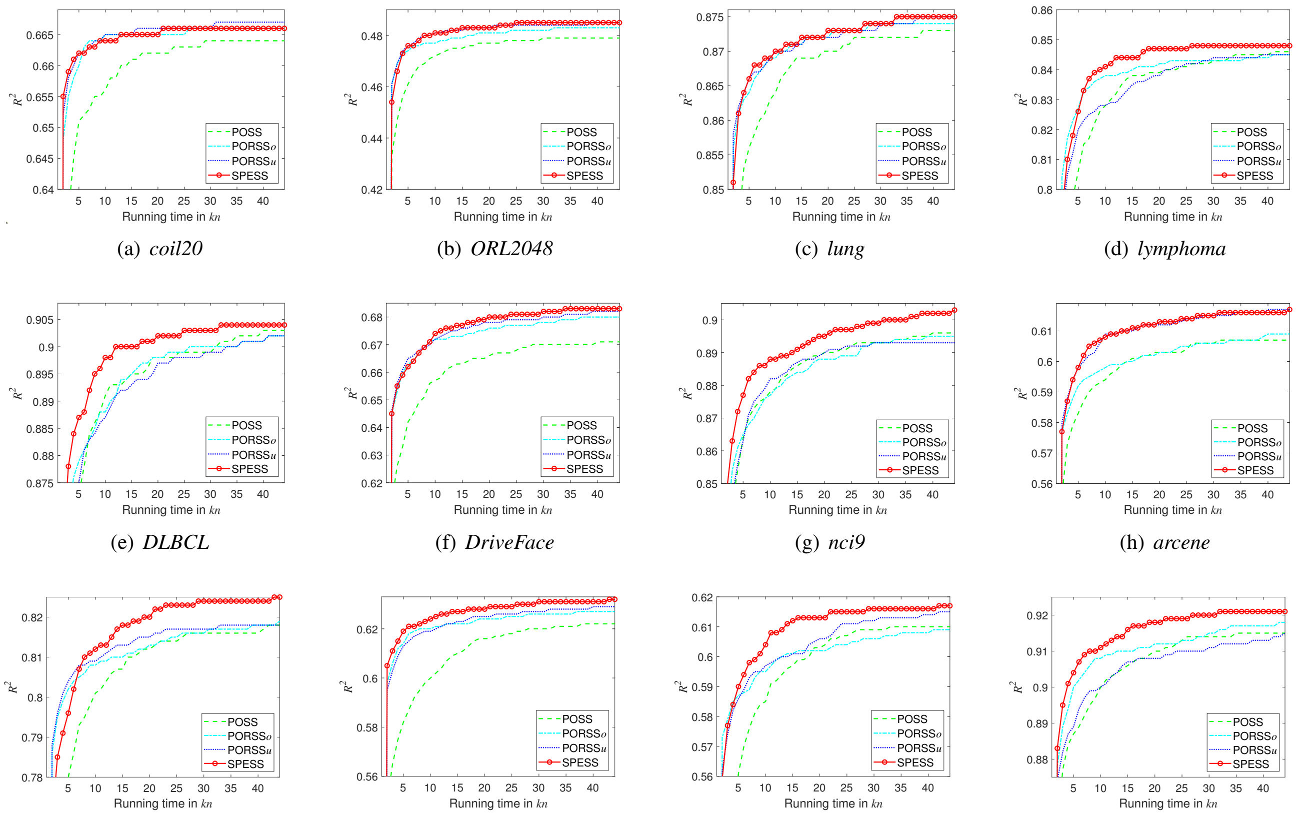
statistical analysis for comparing more than one algorithm. And in the last row, the Cohen's d [52], as a common effect size metric, indicates the standard difference between the two means to quantify the performance improvements. According to the last two rows of Table III, we can find that the proposed SPESS also has a strong effect size (Cohen's d > 0.2) when the significance test p-value is significant (p-value < 0.05). It implies that SPESS is more effective in practice. Note that PORSS and SPESS obtain the same performance on the small-scale datasets since they can achieve the nearly optimal solution. In other words, all algorithms have reached their maximum approximation capability when the size of datasets is relatively small, while the proposed SPESS can show better advantages as the size of dataset increases (i.e., the sparsity of the optimal solution increases).

For Unsupervised Feature Selection, we use $ErrorRatio$ (the calculation of this metric can be found in Application 2) to evaluate a submatrix S, and the smaller the better. The results are summarized in Table IV. We can see from Table IV that SPESS always achieves the best performance. By the average rank, p-value and average Cohen's d in Table IV, SPESS is significantly better than the other three algorithms. From the two tables, we find that when compared with other state-of-the-arts the proposed SPESS achieve the best performance on most of datasets, which has demonstrated the effectiveness of SPESS.

The above experiments have demonstrated that SPESS has excellent approximation capability, in order to further show

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456



(a) *coil20* (b) *ORL2048* (c) *lung* (d) *lymphoma*

(e) *DLBCL* (f) *DriveFace* (g) *nci9* (h) *arcene*

(i) *11Tumor* (j) *smallNORB* (k) *SMKCAN187* (l) *GLI_85*

Fig. 3. $R^2_{z,S}$ (the larger, the better) vs. runing time of POSS, PORSSo, PORSSu, and SPESS on SR.

the convergence capability of different comparison algorithms, we plot the curves of $R^2_{z,S}$ and $ErrorRatio$ over the running time for different algorithms on the 24 datasets as shown in Fig. 3 and Fig. 4. Note that the running time is considered in the number of objective function f evaluations, and one unit on the x-axis corresponds to kn evaluations. It can be clearly observed that the curves of SPESS are almost always better that of POSS and PORSS in Fig. 3 and Fig. 4, implying that SPESS has faster convergency than POSS and PORSS. The reason for the better convergence of our proposed algorithm is that the proposed crossover and mutation operators used in SPESS can guarantee producing more feasible solutions (whose sizes are not larger than k) in the process of evolution, results in speeding up the approximation of the optimal solution.

To investigate the performance of the proposed SPESS that can generate more feasible solutions, the comparison algorithms are evaluated on SR and UFS as shown in Fig. 5(a) and (b), respectively. Note that the Rs on the y-axis is considered as the ratio of the number of feasible solutions generated by parent individuals to the number of iterations in different algorithms, and one unit on the x-axis corresponds to one comparision algorithm. Since similar results can be found on different data sets, we only show the results on the two data sets (i.e., $nci9$ for SR and $warpPIE10P$ for UFS). It can be found from Fig. 5 that the proposed SPESS can produce more number of feasible solutions than that of POSS and PORSS in optimization process, which is consistent with the discussion in Section III.

From the above experimental results, we can make a conclusion that SPESS can achieve the optimal approximation performance with minimum running time, especially on high-dimensional datasets (see Fig. 3(k, l) and Fig. 4(k, l)). In summary, SPESS exhibits an excellent performance in terms of running time and solution quality. The better performance of SPESS is attributed to the proposed crossover and mutation operators (i.e. SPM and SPC), which can ensure the sparsity of the generated solutions in the process of Pareto optimization. In the following, we will demonstrate the effectiveness of the proposed operators in SPESS.

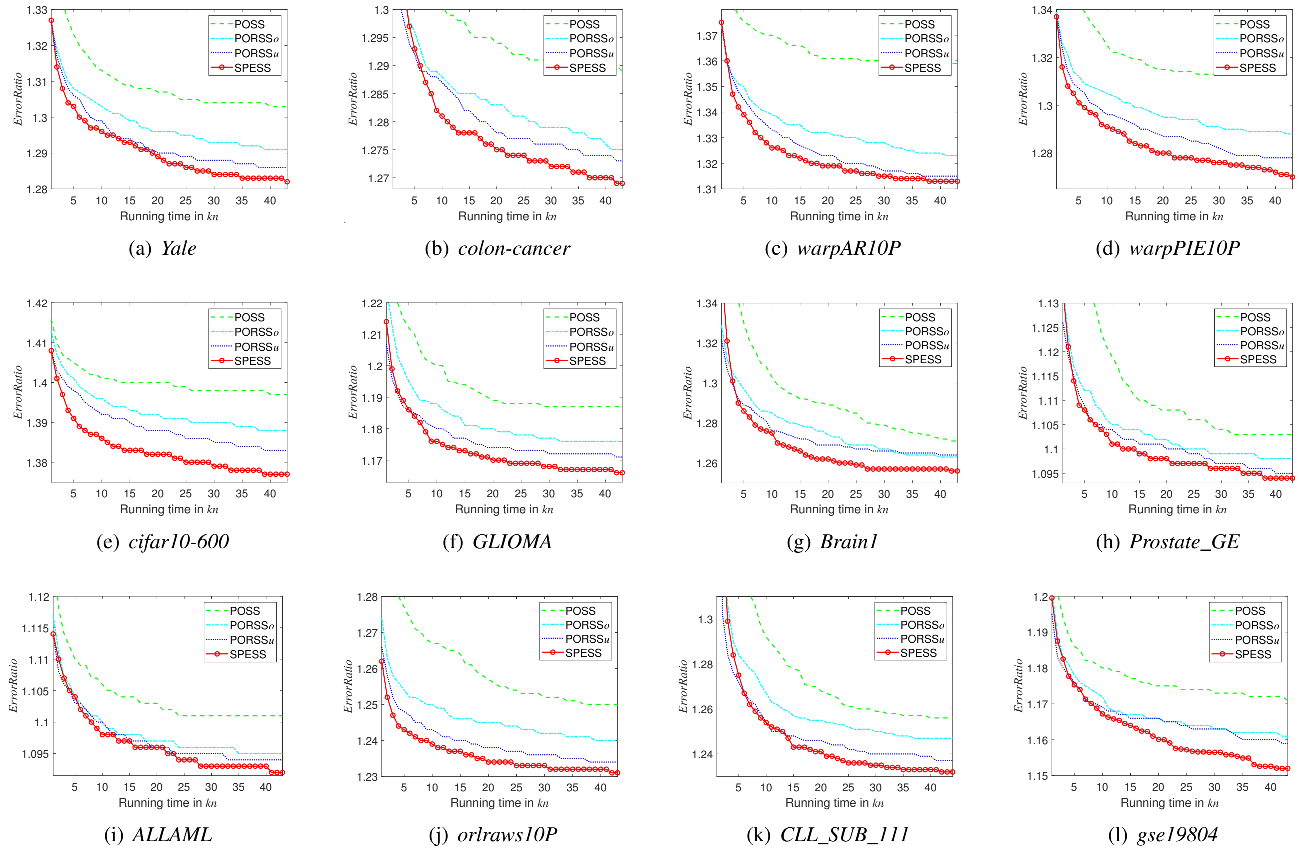
C. Effectiveness of the Proposed Operators

In this subsection, we will further validate the effectiveness of the proposed operators used in SPESS, namely, the sparsity preserved crossover operator (SPC) and the sparsity preserved mutation operator (SPM). Note that SPESS using only SPM and SPC are denoted as SPESSm and SPESSc, respectively. Table V and Table VI show the comparison of the proposed SPESS and its two variants (i.e., SPESSm and SPESSc), on the 12 datasets for SR and UFS respectively. In addition, in the two tables we also give significance tests and effect sizes.

As indicated by the results in the above two tables, we can find that the proposed SPESS outperforms the other two

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456



(a) *Yale* (b) *colon-cancer* (c) *warpAR10P* (d) *warpPIE10P*

(e) *cifar10-600* (f) *GLIOMA* (g) *Brain1* (h) *Prostate_GE*

(i) *ALLAML* (j) *orlraws10P* (k) *CLL_SUB_111* (l) *gse19804*

Fig. 4. $ErrorRatio$ (the smaller, the better) vs. runing time of POSS, PORSSo, PORSSu and SPESS on UFS.

variants on most of datasets no matter for Sparse Regression or Unsupervised Feature Selection tasks. The average rank investigates that SPESS exhibits the best performance. Moreover, according to the values in Table V and Table VI, the corresponding p-value of Friedman test are 2.2797×10^{-5} and 1.8185×10^{-5}, respectively. This means that at 0.05, there is evidence to reject the null hypothesis and the above three algorithms are significantly different. Furthermore, compared with SPESS, the p-value of SPESSm and SPESSc are all below 0.05, which also means SPESS outperforms than the two baselines statistically in terms of R^2 and $ErrorRatio$. At the bottom of the two tables, the effect size analysis also strengthens the reliability of hypothesis test. To be specific, due to the abandon of SPM, SPESSc has lower performance than SPESS on all datasets. Besides, SPESSm is also worse than SPESS according to the three metrics (i.e., the p-value, the average rank and Cohen's d), due to the fact that SPC used in SPESS by recombining diversity of solutions is more likely to generate various variations of the solution than SPM. This means that SPESS can bring further improvements by introducing the SPC operator, which verifies the effectiveness of SPC. In short, by combining SPM and SPC, SPESS can get the best results. Thus, the effectiveness of the proposed operators used in SPESS is verified.

D. Generalization Performance of the Proposed SPESS

As discussed in Section III, parameter values in SPC and SPM are related to the budget k, further affecting the efficiency of the algorithm. In practical applications, the budget k is usually not large, and different k corresponds to different practical problems. Hence, we finally examine the influence of budget k on the approximation performance of different algorithms. That is, we study the performance of different comparison algorithms for generalization in this subsection.

The curves of R^2 (the larger, the better) and $ErrorRatio$ (the smaller, the better) over the budget k for different algorithms are presented in Fig. 6. Since the results are similar to each other data sets, we also show the results on the two representative datasets, i.e., $nci9$ for SR and $warpPIE10P$ for UFS when $k \in \{6, 7, 8, 9, 10\}$. As can be seen from Fig. 6, the performance of SPESS tends to increase monotonically with the budget k. It can be clearly observed that the curve of SPESS is almost always above (below) that of POSS and PORSS in Fig. 6(a) ((b)), implying that the proposed SPESS always manifests obvious advantages than the baseline algorithms. In summary, the proposed SPESS has great generalization performance.

V. Conclusion

In this paper, we proposed a novel sparsity preserved evolutionary algorithm named SPESS for subset selection. Specifically, two sparsity preserved operators named SPC and

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

TABLE V

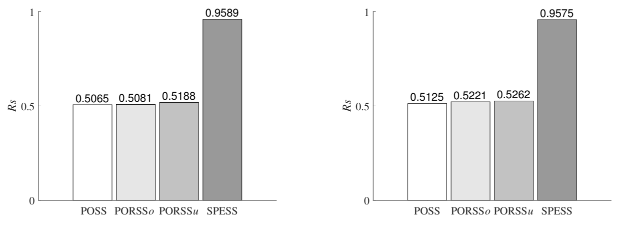
COMPARISON OF SPESS AND ITS TWO VARIANTS FOR THE R^2 VALUE (THE LARGER, THE BETTER). IN EACH DATA SET, THE BEST VALUE IS BOLDED. THE THIRD-TO-LAST ROW INDICATE THE AVERAGE RANKING OF THE THREE ALGORITHMS, AND THE BEST RANK IS BOLDED. THE p-VALUE AND COHEN'S D ARE APPLIED TO STATISTICAL TESTS AND TO QUANTIFY PERFORMANCE IMPROVEMENTS, RESPECTIVELY.

Sparse Regression: R^2			
Dataset	SPESS*m*	SPESS*c*	SPESS
coil20	**0.666±0.002**	0.457±0.107	**0.666±0.001**
ORL2048	**0.485±0.003**	0.276±0.029	**0.485±0.002**
lung	0.874±0.005	0.589±0.056	**0.875±0.004**
lymphoma	0.848±0.010	0.353±0.059	**0.851±0.008**
DLBCL	**0.906±0.006**	0.465±0.109	0.904±0.010
DriveFace	0.678±0.006	0.322±0.058	**0.683±0.006**
nci9	0.900±0.008	0.329±0.059	**0.903±0.009**
arcene	**0.617±0.014**	0.269±0.048	**0.617±0.010**
11Tumor	0.820±0.008	0.402±0.073	**0.825±0.008**
smallNORB	0.629±0.007	0.228±0.053	**0.632±0.007**
SMKCAN187	0.608±0.010	0.212±0.043	**0.617±0.008**
GLI_85	0.918±0.009	0.477±0.061	**0.921±0.008**
Average rank	1.79	3.00	**1.21**
p-value	0.0196	0.0005	—
Average Cohen's d	0.339	8.774	—

TABLE VI

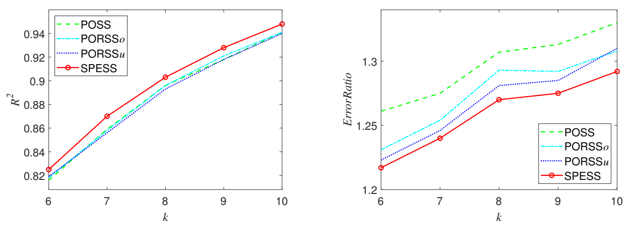
COMPARISON OF SPESS AND ITS TWO VARIANTS FOR THE $ErrorRatio$ VALUE (THE SMALLER, THE BETTER). IN EACH DATA SET, THE BEST VALUE IS BOLDED. THE THIRD-TO-LAST ROW INDICATE THE AVERAGE RANKING OF THE THREE ALGORITHMS, AND THE BEST RANK IS BOLDED. THE p-VALUE AND COHEN'S D ARE APPLIED TO STATISTICAL TESTS AND TO QUANTIFY PERFORMANCE IMPROVEMENTS, RESPECTIVELY.

Unsupervised Feature Selection: $ErrorRatio$			
Dataset	SPESS*m*	SPESS*c*	SPESS
Yale	1.283±0.004	1.677±0.082	**1.282±0.003**
warpAR10P	**1.308±0.009**	1.800±0.116	1.313±0.008
colon-cancer	1.269±0.008	1.567±0.042	**1.269±0.008**
warpPIE10P	1.271±0.012	1.740±0.105	**1.270±0.012**
cifar10-600	1.378±0.006	1.569±0.040	**1.377±0.008**
GLIOMA	**1.166±0.005**	1.547±0.069	**1.166±0.006**
Brain1	1.257±0.009	2.409±0.222	**1.256±0.009**
Prostate_GE	1.095±0.005	1.494±0.073	**1.094±0.003**
ALLAML	1.094±0.004	1.279±0.030	**1.092±0.003**
orlraws10P	1.232±0.004	1.636±0.065	**1.231±0.003**
CLL_SUB_111	1.234±0.007	4.316±1.505	**1.232±0.005**
gse19804	1.153±0.007	1.596±0.086	**1.152±0.009**
Average rank	1.88	3.00	**1.13**
p-value	0.0067	0.0005	—
Average Cohen's d	0.219	6.356	—



(a) SR on *nci9* data set (b) UFS on *warpPIE10P* data set

Fig. 5. The ratio of the number of feasible solutions generated by parent individuals to the number of iterations (Rs) in POSS, PORSSo, PORSSu, and SPESS on dataset $nci9$ for SR and $warpPIE10P$ for UFS, respectively.



(a) SR on *nci9* data set (b) UFS on *warpPIE10P* data set

Fig. 6. The $R^2_{z,S}$ (the larger, the better) and $ErrorRatio$ (the smaller, the better) value of the compared algorithms on data set $nci9$ for SR and $warpPIE10P$ for UFS at each $k \in \{6, 7, 8, 9, 10\}$, respectively.

SPM were suggested in SPESS to ensure the sparsity of the generated solutions in the process of Pareto optimization, which can be used to improve the search efficiency and produce high-quality solutions simultaneously. In the experiments, the proposed SPESS has been compared with the state-of-the-art Pareto optimization algorithms on 12 sparse regression problems and 12 unsupervised feature selection problems. The experimental results have demonstrated that the proposed SPESS is more effective than the state-of-the-arts in solving subset selection tasks.

The work in this paper has shown the effectiveness of the suggested sparsity preserved operators in the proposed SPESS. However, like the most of EAs, the cost of evaluating an individual in SPESS for subset selection is usually expensive, especially when the number of instances is large. In the future, we plan to exploit surrogate-assisted methods [53] to reduce the calculation cost of evaluation functions, thus further improving the efficiency of SPESS for solving large-scale subset selection with a large number of instances. In addition, it is very interesting to theoretically investigate the influence of our proposed operators on the convergence of EAs, as well as extend SPESS for solving subset selection problems under the noisy or dynamic situations.

REFERENCES

[1] B. Natarajan, "Sparse approximate solutions to linear systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, 1995.

[2] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations," *Constructive Approximation*, vol. 13, no. 1, pp. 57–98, 1997.

[3] A. Miller, *Subset selection in regression*. chapman and hall/CRC, 2002.

[4] A. Farahat, A. Ghodsi, and M. Kamel, "An efficient greedy method for unsupervised feature selection," in *Proceedings of the 11th International Conference on Data Mining*, 2011, pp. 161–170.

[5] U. Feige, "A threshold of ln *n* for approximating set cover," *Journal of the ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[6] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the 9th International Conference on Knowledge Discovery and Data Mining*, 2003, pp. 137–146.

[7] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, vol. 9, pp. 235–284, 2008.

[8] H. Lin and J. Bilmes, "A class of submodular functions for document summarization," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 510–520.

[9] L. Zhang, J. Xia, F. Cheng, J. Qiu, and X. Zhang, "Multi-objective optimization of critical node detection based on cascade model in complex networks," *IEEE Transactions on Network Science and Engeering*, vol. 7, no. 3, pp. 2052–2066, 2020.

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

[10] A. Das and D. Kempe, "Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection," in *Proceedings of the 28th International Conference on Machine Learning*, 2011, pp. 1057–1064.

[11] C. Qian, Y. Yu, and Z. Zhou, "Subset selection by pareto optimization," in *Advances in Neural Information Processing Systems 28 (NIPS'15)*, 2015, pp. 1765–1773.

[12] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[13] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 67, no. 2, pp. 301–320, 2005.

[14] J. A. Tropp, "Greed is good: algorithmic results for sparse approximation," *IEEE Transactions Information Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[15] K. Shang, H. Ishibuchi, and W. Chen, "Greedy approximated hypervolume subset selection for many-objective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2021, pp. 448–456.

[16] G. Nemhauser and L. Wolsey, "Best algorithms for approximating the maximum of a submodular set function," *Mathematic of Operations Research*, vol. 3, no. 3, pp. 177–188, 1978.

[17] C. Harshaw, M. Feldman, J. Ward, and A. Karbasi, "Submodular maximization beyond non-negativity: Guarantees, fast algorithms, and applications," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 2634–2643.

[18] J. Altschuler, A. Bhaskara, G. Fu, V. Mirrokni, A. Rostamizadeh, and M. Zadimoghaddam, "Greedy column subset selection: New bounds and distributed algorithms," in *Proceedings of the 33nd International Conference on Machine Learning*, 2016, pp. 2539–2548.

[19] M. C. de Oliveira, M. R. Delgado, and A. Britto, "A hybrid greedy indicator- and pareto-based many-objective evolutionary algorithm," *Applied Intelligence*, vol. 51, no. 7, pp. 4330–4352, 2021.

[20] O. Giel, "Expected runtimes of a simple multi-objective evolutionary algorithm," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2003, pp. 1918–1925.

[21] M. Laumanns, L. Thiele, and E. Zitzler, "Running time analysis of multiobjective evolutionary algorithms on pseudo-boolean functions," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 2, pp. 170–182, 2004.

[22] O. Giel and P. K. Lehre, "On the effect of populations in evolutionary multi-objective optimisation," *Evolutionary Computation*, vol. 18, no. 3, pp. 335–356, 2010.

[23] T. Friedrich and F. Neumann, "Maximizing submodular functions under matroid constraints by evolutionary algorithms," *Evolutionary Computation*, vol. 23, no. 4, pp. 543–558, 2015.

[24] A. Neumann and F. Neumann, "Optimising monotone chance-constrained submodular functions using evolutionary multi-objective algorithms," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2020, pp. 404–417.

[25] F. Neumann, "Expected runtimes of a simple evolutionary algorithm for the multi-objective minimum spanning tree problem," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1620–1629, 2007.

[26] C. Qian, G. Li, C. Feng, and K. Tang, "Distributed pareto optimization for subset selection," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 1492–1498.

[27] C. Qian, "Distributed pareto optimization for large-scale noisy subset selection," *IEEE Transactions Evolutionary Computation*, vol. 24, no. 4, pp. 694–707, 2020.

[28] C. Qian, J. Shi, Y. Yu, K. Tang, and Z. Zhou, "Subset selection under noise," in *Advances in Neural Information Processing Systems*, 2017, pp. 3560–3570.

[29] C. Qian, J.-C. Shi, Y. Yu, and K. Tang, "On subset selection with general cost constraints." in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, vol. 17, 2017, pp. 2613–2619.

[30] V. Roostapour, A. Neumann, F. Neumann, and T. Friedrich, "Pareto optimization for subset selection with dynamic cost constraints," *Artifical Intelligence*, vol. 302, p. 103597, 2022.

[31] C. Bian, C. Feng, C. Qian, and Y. Yu, "An efficient evolutionary algorithm for subset selection with general cost constraints," in *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 3267–3274.

[32] C. Bian, C. Qian, F. Neumann, and Y. Yu, "Fast pareto optimization for subset selection with dynamic cost constraints," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021, pp. 2191–2197.

[33] C. Qian, C. Bian, and C. Feng, "Subset selection by pareto optimization with recombination," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, 2020, pp. 2408–2415.

[34] L. Sun, X. Zhang, Y. Qian, J. Xu, and S. Zhang, "Feature selection using neighborhood entropy-based uncertainty measures for gene expression data classification," *Information Sciences*, vol. 502, pp. 18–41, 2019.

[35] V. Bolón-Canedo and B. Remeseiro, "Feature selection in image analysis: a survey," *Artificial Intelligence Review*, vol. 53, no. 4, pp. 2905–2931, 2020.

[36] Y. Tian, X. Zhang, C. Wang, and Y. Jin, "An evolutionary algorithm for large-scale sparse multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 380–393, 2019.

[37] Z. Tan, H. Wang, and S. Liu, "Multi-stage dimension reduction for expensive sparse multi-objective optimization problems," *Neurocomputing*, vol. 440, pp. 159–174, 2021.

[38] F. Cheng, F. Chu, Y. Xu, and L. Zhang, "A steering-matrix-based multiobjective evolutionary algorithm for high-dimensional feature selection," *IEEE Transactions on Cybernetics*, 2021.

[39] J. Zhou and D. Liu, "A redundancy based unsupervised feature selection method for high-dimensional data," in *Proceedings of the 13th International Conference on Machine Learning and Computing*, 2021, pp. 285–289.

[40] Y. Tian, C. Lu, X. Zhang, F. Cheng, and Y. Jin, "A pattern mining-based evolutionary algorithm for large-scale sparse multiobjective optimization problems," *IEEE Transactions on Cybernetics*, 2020.

[41] G. Diekhoff, *Statistics for the social and behavioral sciences: Univariate, bivariate, multivariate*. William C Brown Pub, 1992.

[42] A. Civril, "Column subset selection problem is ug-hard," *Journal of Computer and System Sciences*, vol. 80, no. 4, pp. 849–859, 2014.

[43] C. Qian, J. Shi, Y. Yu, K. Tang, and Z. Zhou, "Parallel pareto optimization for subset selection," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, 2016, pp. 1939–1945.

[44] C. Bian, Y. Zhou, and C. Qian, "Robust subset selection by greedy and evolutionary pareto optimization," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, 2022, pp. 4726–4732.

[45] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[46] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1348–1360, 2001.

[47] C.-H. Zhang, "Nearly unbiased variable selection under minimax concave penalty," *The Annals of Statistics*, vol. 38, no. 2, pp. 894–942, 2010.

[48] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[49] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, no. 1, pp. 389–422, 2002.

[50] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[51] R. L. Wasserstein and N. A. Lazar, "The ASA statement on p-values: Context, process, and purpose," *The American Statistician*, vol. 70, no. 2, pp. 129–133, 2016.

[52] S. Nakagawa and I. C. Cuthill, "Effect size, confidence interval and statistical significance: a practical guide for biologists," *Biological reviews*, vol. 82, no. 4, pp. 591–605, 2007.

[53] Q. Zhou, J. Wu, T. Xue, and P. Jin, "A two-stage adaptive multi-fidelity surrogate model-assisted multi-objective genetic algorithm for computationally expensive problems," *Engineering with Computers*, vol. 37, no. 1, pp. 623–639, 2021.

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2023.3281456

Lei Zhang received the B.Sc. from Anhui Agriculture University in 2007, and the Ph.D. in 2014 from University of Science and Technology of China. Currently, he is an associate professor in the School of Computer Science and Technology, Anhui University, China. His main research interests include multi-objective optimization and their applications, data mining, machine learning, social network analysis and recommendation.

He has published more than 80 papers in refereed journals and conferences, such as *IEEE TEVC*, *IEEE TCYB*, *IEEE TBD*, *IEEE TNSE*, *ACM TKDD*, *IEEE CIM*, *IEEE TCSS*, *AAAI*, *IJCAI*, *ACM SIGKDD*. He is the recipient of the ACM CIKM'12 Best Student Paper Award. He is a member of ACM and IEEE.

Xiang Sun received the B.Sc. degree from the School of Mathematical Sciences, Anhui University, China in 2019.

He is currently working toward the master degree from the School of Computer Science and Technology, Anhui University, China. His current research interests include multi-objective optimization and subset selection.

Haipeng Yang received the B.Sc. degree from the School of Computer Science and Technology, Anhui University, China in 2019.

He is currently working toward the PhD degree from the School of Computer Science and Technology, Anhui University, China. His research interests include multi-objective optimization and social network analysis.

Fan Cheng received the B.Sc. in 2000 and M.Sc. in 2003 both from HeFei University of Technology, China. He received the Ph.D. in 2012 from University of Science and Technology of China, China. His main research interests include machine learning, imbalanced classification, multi-objective optimization, and complex network.

He is currently a full professor of School of Computer Science and Technology at Anhui University, China. He has published over 50 papers in refereed conferences and journals, such as *IEEE TEVC*, *IEEE TCYB*, *IEEE TNSE*, *IEEE TBD*, *IEEE CIM*, *Applied Soft Computing*, and *Information Sciences*.

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on June 27,2023 at 07:55:29 UTC from IEEE Xplore. Restrictions apply.
© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.