Particle Swarm Optimization for Feature Selection in Classification: A Multi-Objective Approach

Bing Xue, *Member, IEEE*, Mengjie Zhang, *Senior Member, IEEE*, and Will N. Browne

***Abstract*—Classification problems often have a large number of features in the data sets, but not all of them are useful for classification. Irrelevant and redundant features may even reduce the performance. Feature selection aims to choose a small number of relevant features to achieve similar or even better classification performance than using all features. It has two main conflicting objectives of maximizing the classification performance and minimizing the number of features. However, most existing feature selection algorithms treat the task as a single objective problem. This paper presents the first study on multi-objective particle swarm optimization (PSO) for feature selection. The task is to generate a Pareto front of nondominated solutions (feature subsets). We investigate two PSO-based multi-objective feature selection algorithms. The first algorithm introduces the idea of nondominated sorting into PSO to address feature selection problems. The second algorithm applies the ideas of crowding, mutation, and dominance to PSO to search for the Pareto front solutions. The two multi-objective algorithms are compared with two conventional feature selection methods, a single objective feature selection method, a two-stage feature selection algorithm, and three well-known evolutionary multi-objective algorithms on 12 benchmark data sets. The experimental results show that the two PSO-based multi-objective algorithms can automatically evolve a set of nondominated solutions. The first algorithm outperforms the two conventional methods, the single objective method, and the two-stage algorithm. It achieves comparable results with the existing three well-known multi-objective algorithms in most cases. The second algorithm achieves better results than the first algorithm and all other methods mentioned previously.**

***Index Terms*—Feature selection, multi-objective optimization, particle swarm optimization (PSO).**

I. Introduction

CLASSIFICATION is an important task in machine learning and data mining, which aims to classify each instance in the data set into different groups based on the information described by its features. Without prior knowledge, it is difficult to determine which features are useful. As a result, a large number of features are usually introduced to the data set, which include relevant, irrelevant, and redundant features. However, irrelevant and redundant features are not useful for classification, and they may even reduce the classification performance due to the large search space known as "the curse of dimensionality" [1]. Feature selection can address this problem by selecting only relevant features for classification. By eliminating/reducing irrelevant and redundant features, feature selection could reduce the number of features, shorten the training time, simplify the learned classifiers, and/or improve the classification performance [2], [3].

Feature selection is a difficult task because there can be complex interaction among features. An individually relevant (redundant or irrelevant) feature may become redundant (relevant) when working together with other features. Therefore, an optimal feature subset should be a group of complementary features that span over the diverse properties of the classes to properly discriminate them. The feature selection task is challenging also because of the large search space. The size of the search space increases exponentially with respect to the number of available features in the data set [4]. Therefore, an exhaustive search is practically impossible in most situations. In order to solve this problem, a variety of search methods have been applied to feature selection, such as greedy search based sequential forward selection (SFS) [5] and sequential backward selection (SBS) [6]. However, these feature selection approaches still suffer from a variety of problems, such as stagnation in local optima and high computational cost.

In order to better address feature selection problems, an efficient global search technique is needed. Evolutionary computation (EC) techniques are well known for their global search ability. Particle swarm optimization (PSO) [7], [8] is a relatively recent EC technique based on swarm intelligence. Compared with other EC algorithms such as genetic algorithms (GAs) and genetic programming (GP), PSO is computationally less expensive and can converge more quickly. Therefore, PSO has been used as an effective technique in many fields, including feature selection [3], [9], [10].

Generally, feature selection is a multi-objective problem. It has two main objectives, which are to maximize the classification performance (minimize the classification error rate) and to minimize the number of features. These two objectives are usually conflicting, and the optimal decision needs to be made in the presence of a tradeoff between them. Treating feature selection as a multi-objective problem can obtain a set of nondominated feature subsets to meet different requirements in real-world applications. Although PSO, multi-objective optimization, and feature selection have been individually investigated frequently, there are very few studies on multi-objective feature selection. Meanwhile, existing feature selection

Manuscript received March 19, 2012; revised August 11, 2012; accepted October 21, 2012. Date of publication December 13, 2012; date of current version November 18, 2013. This work was supported in part by the Marsden Fund of the New Zealand Government (VUW0806 and 12-VUW-134), administrated by the Royal Society of New Zealand, and the University Research Fund (200457/3230) at Victoria University of Wellington. This paper was recommended by Editor H. Ishibuchi.

The authors are with the Evolutionary Computation Research Group, Victoria University of Wellington, Wellington 6140, New Zealand (e-mail: bing.xue@ecs.vuw.ac.nz; mengjie.zhang@ecs.vuw.ac.nz; Will.Browne@ecs.vuw.ac.nz).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2012.2227469

algorithms suffer from the problems of high computational cost, and PSO is argued computationally less expensive than other EC techniques. In addition, the use of PSO for multi-objective feature selection has not been investigated.

This paper represents the first time that PSO has been applied to multi-objective feature selection. This will require novel methods to be introduced as there is no longer a single basis global solution but a set of solutions to meet different requirements.

A. Goals

The overall goal of this paper is to develop a PSO-based multi-objective feature selection approach to classification with the expectation of achieving a Pareto front of nondominated solutions, which include a small number of features and achieve a lower classification error rate than using all available features. In order to achieve this goal, we investigate two Pareto front feature selection algorithms based on multi-objective PSO, which are *NSPSOFS* using the idea of nondominated sorting and *CMDPSOFS* using the ideas of crowding, mutation, and dominance. The two feature selection algorithms will be examined and compared with two traditional methods, a single objective algorithm, a two-stage training algorithm, and three well-known evolutionary multi-objective algorithms on 12 benchmark data sets with different numbers of features, classes, and instances (details are shown in Section IV). Specifically, we will investigate the following:

1) whether using a standard single objective PSO with the overall classification error rate as the fitness function can select a good feature subset and achieve similar or even better classification performance than using all features, and can outperform the two traditional methods;
2) whether the PSO-based two-stage training algorithm can further improve the feature subset evolved by the aforementioned PSO-based single objective algorithm;
3) whether NSPSOFS can evolve a Pareto front of nondominated solutions, which can outperform the two conventional methods, the single objective algorithm, the two-stage algorithm, and three well-known multi-objective algorithms;
4) whether CMDPSOFS can evolve a better Pareto front than NSPSOFS and outperform all other methods mentioned previously.

The scope of this paper is continuous (standard) PSO rather than binary PSO [11]. Although both versions of PSO have been successfully applied to single objective feature selection [3], [9], [10], [12], [13], binary PSO has potential limitations, such as the position of a particle in binary PSO is updated solely based on the velocity while the position in standard PSO is updated based on both the velocity and current position [14]. Therefore, as the first work on multi-objective PSO for feature selection, we will start with developing a multi-objective feature selection approach using continuous PSO.

B. Organization

The remainder of this paper is organized as follows. Section II provides background information. Section III describes the PSO-based multi-objective feature selection algorithms. Section IV describes the experimental design, and Section V presents the experimental results with discussions. Section VI provides the conclusion and future work.

II. Background

This section provides background about PSO and multi-objective optimization and also reviews typical related work on feature selection.

A. PSO

PSO is an EC technique proposed by Kennedy and Eberhart in 1995 [7], [8]. PSO is motivated by social behaviors such as bird flocking and fish schooling. The underlying phenomenon of PSO is that knowledge is optimized by social interaction in the population where thinking is not only personal but also social.

PSO is based on the principle that each solution can be represented as a particle in the swarm. Each particle has a position in the search space, which is represented by a vector $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, where D is the dimensionality of the search space. Particles move in the search space to search for the optimal solutions. Therefore, each particle has a velocity, which is represented as $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$. During the movement, each particle updates its position and velocity according to its own experience and that of its neighbors. The best previous position of the particle is recorded as the personal best $pbest$, and the best position obtained by the population thus far is called $gbest$. Based on $pbest$ and $gbest$, PSO searches for the optimal solutions by updating the velocity and the position of each particle according to the following equations:

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \tag{1}$$
$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * r_1 * \left(p_{id} - x_{id}^{t}\right) + c_2 * r_2 * \left(p_{gd} - x_{id}^{t}\right) \tag{2}$$

where t represents the tth iteration in the evolutionary process. $d \in D$ represents the dth dimension in the search space. w is inertia weight, which is to control the impact of the previous velocities on the current velocity. c_1 and c_2 are acceleration constants. r_1 and r_2 are random values uniformly distributed in [0, 1]. p_{id} and p_{gd} denote the elements of $pbest$ and $gbest$ in the dth dimension. The velocity is limited by a predefined maximum velocity, $v_{\max}$, and $v_{id}^{t+1} \in [-v_{\max}, v_{\max}]$. The algorithm stops when a predefined criterion is met, which could be a good fitness value or a predefined maximum number of iterations.

B. Multi-Objective Optimization

Multi-objective problems happen wherever optimal decisions need to be taken in the presence of tradeoffs between two or more conflicting objectives. Multi-objective optimization involves minimizing or maximizing multiple conflicting objective functions. In mathematical terms, the formulas of a minimization problem with multiple objective functions can be written as follows:

$$minimize \quad F(x) = [f_1(x), f_2(x), \ldots, f_k(x)] \tag{3}$$

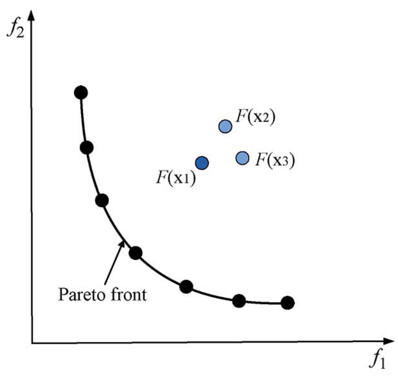

Fig. 1. Minimization problem with two objective functions.

subject to

$$g_i(x) \leq 0,\ i = 1, 2, \ldots m \tag{4}$$

$$h_i(x) = 0,\ i = 1, 2, \ldots l \tag{5}$$

where x is the vector of decision variables, $f_i(x)$ is a function of x, k is the number of objective functions to be minimized, and $g_i(x)$ and $h_i(x)$ are the constraint functions of the problem.

In multi-objective optimization, the quality of a solution is explained in terms of tradeoffs between conflicting objectives. Let y and z be two solutions of the aforementioned k-objective minimization problem. If the following conditions are met, one can say that y dominates z or y is better than z:

$$\forall i : f_i(y) \leq f_i(z) \quad and \quad \exists j : f_j(y) < f_j(z) \tag{6}$$

where $i, j \in \{1, 2, 3, \ldots k\}$.

Take a two-objective minimization problem (shown in Fig. 1) as an example; x_1 dominates both x_2 and x_3. For the case that neither x_2 dominates x_3 nor x_3 dominates x_2, x_2 and x_3 are called nondominated solutions or tradeoff solutions of each other. When a solution is not dominated by any other solutions, it is referred to as a Pareto-optimal solution. The set of all Pareto-optimal solutions forms the tradeoff surface in the search space, the *Pareto front*. A multi-objective algorithm is designed to search for a set of nondominated solutions.

Feature selection has two main conflicting objectives, which are minimizing both the number of features and the classification error rate. Therefore, feature selection can be expressed as a two-objective minimization problem.

C. Related Work on Feature Selection

Existing feature selection algorithms can be broadly classified into two categories: wrapper approaches and filter approaches. Wrapper approaches include a learning/classification algorithm in the evaluation procedure, while filter approaches do not. Filter approaches are argued to be computationally less expensive and more general, while wrapper approaches can usually achieve better results [15]. A number of feature selection algorithms have been proposed in recent years [2]. Typical feature selection algorithms are reviewed in this section.

1) Traditional Feature Selection Approaches: Two commonly used wrapper feature selection methods are SFS [5] and SBS [6]. SFS (SBS) starts with no features (all features); then, candidate features are sequentially added to (removed from) the initial feature subset until the further addition (removal) does not increase the classification performance. The limitation of these two methods is that, once a feature is selected (eliminated), it cannot be eliminated (selected) later, which is the so-called nesting effect [16]. This limitation can be overcome by combining both SFS and SBS into one algorithm. Therefore, the "plus-l-take away-r" method is proposed by Stearns [17]. "plus-l-take away-r" performs l times forward selection, followed by r times backward elimination. The challenge is to determine the optimal values of (l, r). To address this challenge, two floating feature selection algorithms are proposed by Pudil *et al.* [18], namely, sequential forward floating selection (SFFS) and sequential backward floating selection (SBFS). SFFS and SBFS are developed to automatically determine the values for (l, r). These two floating methods are regarded to be at least as good as the best sequential method, but they also suffer from the problem of stagnation in local optima [16].

The Relief algorithm [19] is a classical filter feature selection algorithm. Relief assigns a weight to each feature to denote the relevance of the feature to the target concept. However, Relief does not deal with redundant features because it attempts to find all relevant features regardless of the redundancy between them. Decision trees (DTs) use only relevant features that are required to completely classify the training set and remove all other features. Cardie [20] proposes a filter feature selection algorithm that uses a DT algorithm to select a subset of features for a nearest neighbor algorithm. The FOCUS algorithm [21], a filter algorithm, exhaustively examines all possible feature subsets and then selects the smallest feature subset. However, the FOCUS algorithm is computationally inefficient because of the exhaustive search.

2) EC Algorithms (Non-PSO) for Feature Selection: Recently, EC techniques have been applied to address feature selection problems, such as GAs, GP, and ant colony optimization (ACO). This section briefly reviews some typical work in the literature.

Based on GAs, Chakraborty [22] proposes a feature selection algorithm using a fuzzy set based fitness function. However, PSO with the same fitness function in [23] achieves better performance than this GA-based algorithm. Hamdani *et al.* [24] develop a multi-objective feature selection algorithm using nondominated sorting-based multi-objective GA II (NSGAII), but the performance of the proposed algorithm has not been compared with any other feature selection algorithms. Based on NSGAII and rough set theory, Banerjee *et al.* [25] propose a feature selection method for microarray gene expression data. An initial redundancy reduction of the features is conducted to enable faster convergence and reduce the computational complexity. Although experiments show the effectiveness of the proposed algorithm, only using three data sets is not convincing enough, and the results are achieved in a very long evolutionary process (around 15 000 generations with 100 individuals).

Zhu *et al.* [26] propose a hybrid wrapper and filter feature selection algorithm (WFFSA) based on a memetic algorithm, i.e., a combination of GA and local search. In WFFSA, GA adds or deletes a feature based on the ranked individual features.

Three different local search strategies, namely, improvement first strategy, greedy strategy, and sequential strategy, are investigated in WFFSA. Experiments show that WFFSA outperforms GA and other methods. This paper also shows that a good balance between local search and genetic search can improve the search quality and efficiency of WFFSA.

Muni *et al.* [27] develop a multitree GP algorithm for feature selection (GPmtfs) to simultaneously select a feature subset and design a classifier using the selected features. For a c-class problem, each classifier in GPmtfs has c trees. Two new crossover operators, namely, homogeneous crossover and heterogeneous crossover, are introduced into GPmtfs. Comparisons suggest that GPmtfs achieves better results than SFS, SBS, and other methods. However, the number of features selected increases when there are (synthetically added) noisy features. Kourosh and Zhang [28] propose a GP relevance measure (GPRM) to evaluate and rank subsets of features in binary classification tasks, and GPRM is also efficient in terms of feature selection. There are also some other GP-related works, which can be seen in [29]–[33].

Gao *et al.* [34] propose an ACO-based wrapper feature selection algorithm to network intrusion detection. Fisher discrimination rate is adopted as the heuristic information for ACO. Ming [35] proposes a feature selection method based on ACO and rough set theory. The proposed algorithm starts with the features included in the core of the rough set. Forward selection is adopted into the proposed method to search for the best feature subset. Experimental results show that the proposed algorithm achieves better classification performance with fewer features than a C4.5-based feature selection algorithm. However, experiments do not compare the proposed method with other commonly used feature selection algorithms.

3) PSO-Based Feature Selection Approaches: As an EC technique, PSO has recently gained more attention for solving feature selection problems.

Since rough set can handle imprecision, uncertainty, and vagueness, Wang *et al.* [36] propose a filter feature selection algorithm based on an improved binary PSO and rough set theory. The goodness of a particle is assigned as the dependence degree between class labels and selected features, which is measured by rough set. This paper also shows that the computation of the rough set consumes most of the running time, which is a drawback when using rough set theory in feature selection problems. Fuzzy set can also show the dependence between features and class labels. Chakraborty [23] compares the performance of PSO with that of a GA in a filter feature selection algorithm with a fuzzy set based fitness function. The results show that PSO performs better than GA in terms of the classification performance.

Azevedo *et al.* [37] propose a wrapper feature selection algorithm using PSO and a support vector machine (SVM) for personal identification in keystroke dynamics systems. However, the proposed algorithm obtains a relatively high false acceptance rate, which should be low in most identification systems. Later, Lin *et al.* [38] propose a wrapper feature selection algorithm (PSO+SVM) using PSO and SVM. The difference from the method in [37] is that PSO+SVM could optimize the parameters in SVM and search for the best feature subset simultaneously. Huang and Dun [13] also propose a similar feature selection method but using two versions of PSO. Binary PSO is used to search for the optimal feature subset, and continuous PSO is used to simultaneously optimize the parameters in the kernel function of SVM. Mohemmed *et al.* [10] propose a hybrid method (PSOAdaBoost) that incorporates PSO with an AdaBoost framework for face detection. PSOAdaBoost aims to search for the best feature subset and determine the decision thresholds of AdaBoost simultaneously, which speeds up the training process and increases the accuracy of weak classifiers in AdaBoost.

The performance of PSO could be improved by properly setting the value of inertia weight to balance its local search and global search. Yang *et al.* [39] propose two feature selection algorithms, which are based on two inertia weight strategies to properly balance the local search and global search of PSO. The two proposed algorithms outperform other algorithms, such as SFS, "plus-l-take away-r," SFFS, a sequential GA, and different hybrid GAs.

In standard PSO, $gbest$ is updated only when a better solution is found. Chuang *et al.* [12] develop a strategy for $gbest$ in PSO for feature selection in which $gbest$ will be reset to zero if it maintains the same value after several iterations. Chuang *et al.* [40] apply the so-called catfish effect to PSO for feature selection, which is to introduce new particles into the swarm by initializing the worst particles when $gbest$ has not improved for a number of iterations. The introduced catfish particles could help PSO avoid premature convergence and lead to better results than sequential GA, SFS, SFFS, and other methods.

Liu *et al.* [9] introduce a multiswarm PSO algorithm to search for the optimal feature subset and optimize the parameters of SVM simultaneously. Experiments show that the proposed feature selection method could achieve higher classification accuracy than grid search, standard PSO, and GA. However, the proposed algorithm is computationally more expensive than the other three methods because of the large population size and complicated communication rules between different subswarms. Hamed *et al.* [41] propose a dynamic quantum-inspired PSO algorithm (DQiPSO) for single objective feature selection and parameter optimization in neural networks for classification. DQiPSO has two parts, where, in the first part, the quantum information principle is embedded in PSO as a mechanism for feature probability calculation and, in the second part, the standard PSO with real-number encoding is used to optimize the parameters. Experimental results show that the proposed DQiPSO can simultaneously select a good feature subset and optimize the parameters in an artificial neural network. Compared with other two methods (PSO with quantum information principle and standard PSO), the proposed DQiPSO is faster and achieves better classification performance.

Based on PSO, Unler and Murat [3] propose a feature selection algorithm with an adaptive selection strategy, where a feature is chosen not only according to the likelihood calculated by PSO but also to its contribution to the features already selected. Experiments suggest that the proposed method outperforms the tabu search and scatter search algorithms. Esseghir *et al.* [42] propose a filter–wrapper feature selection method based on PSO, which aims to integrate the strengths of both filters and wrappers. The proposed filter–wrapper scheme encodes the position of each particle with filter scores of features to reflect feature-class dependence levels. The fitness of each particle is the classification accuracy achieved by the selected features. Experimental results show that the proposed method could achieve slightly better performance than a binary PSO-based filter algorithm. However, the performance of the proposed algorithm has not been compared with that of a wrapper approach, which usually can achieve better results than a filter approach. Our recent work on PSO for feature selection and dimension reduction can be seen in [43]–[46].

A variety of feature selection approaches have been proposed, but most of them aim to minimize the classification error rate only, and not much work has been conducted for solving a feature selection task as a multi-objective problem. Although Hamdani *et al.* [24] develop a NSGAII-based multi-objective algorithm, there are no comparisons to test its performance. Many studies have shown that PSO is an efficient search technique for feature selection, but the use of PSO for multi-objective feature selection has not been investigated. Therefore, the development of multi-objective PSO as the basis for feature selection is still an open issue.

III. Multi-Objective Approaches

In this section, a commonly used PSO-based single objective feature selection algorithm and a PSO-based two-stage training feature selection algorithm [46] are firstly described, which are used as the baseline to test the performance of multi-objective algorithms. Then, we develop two multi-objective feature selection algorithms using PSO, with the goals of selecting a smaller number of features and achieving a lower classification error rate.

A. Two Existing PSO-Based Feature Selection Algorithms

1) Commonly Used PSO Algorithm (ErFS): PSO has been used as a single objective technique to minimize the classification error rate only [12], [38], [40]. This algorithm is considered in this paper to see whether PSO can function well for feature selection. The fitness function is shown in (7), which is to minimize the classification error rate obtained by the selected features during the evolutionary training process

$$F_1 = ErrorRate = \frac{FP + FN}{TP + TN + FP + FN} \tag{7}$$

where TP, TN, FP, and FN stand for true positives, true negatives, false positives, and false negatives, respectively.

The representation of a particle in PSO is a vector of n real numbers, where n is the number of available features in the data set and also the dimensionality of the search space. A threshold θ is needed to compare with the value (x_{id}) in the position vector. If $x_{id} > \theta$, feature d is selected; otherwise, feature d is not selected. This representation is used in all of the PSO-based algorithms in this paper.

2) PSO With a Two-Stage Fitness Function (2SFS): The basic fitness function (7) only considers the classification performance and does not intend to minimize the number of features. Therefore, the feature subset evolved by PSO may still have redundant features, and the same classification performance can be achieved by a smaller number of features. In order to address this problem, a two-stage feature selection approach (2SFS) was proposed [46].

In 2SFS, the whole evolutionary process is divided into two stages. In the first stage, the fitness function is to minimize the classification error rate only. In the second stage, the number of features is taken into account in the fitness function. The second stage starts with the solutions achieved in the first stage, which is expected to ensure that the minimization of the number of features is based on the achieved feature subsets with high classification performance. The fitness function used in 2SFS is shown in

$$F_2 = \begin{cases} ErrorRate, \text{ Stage 1} \\ \alpha * \frac{\#Features}{\#All\ Features} + (1-\alpha) * \frac{ErrorRate}{ER}, \text{ Stage 2} \end{cases} \tag{8}$$

where α is a constant value and $\alpha \in [0, 1]$. $\#Features$ stands for the number of features selected. $\#All\ Features$ represents the total number of available features. $ErrorRate$ is the classification error rate obtained by the selected feature subset. ER is the error rate obtained by using all available features for classification on the training set.

In the second stage, both the number of features and the classification performance are considered in the two-stage fitness function (8), where α shows the relative importance of the number of features and $(1-\alpha)$ shows the relative importance of the classification performance. As the classification performance is assumed to be more important than the number of features, α is set to be smaller than $(1-\alpha)$. For fair comparisons, the standard (continuous) PSO is used in 2SFS in this paper while binary PSO is used in [46].

B. PSO-Based Multi-Objective Feature Selection Algorithm 1: NSPSOFS

In this section, we investigate a new approach to feature selection using multi-objective PSO, with the two main objectives to explore the Pareto front of feature subsets. However, standard PSO was originally proposed for single objective optimization and could not directly be used to address multi-objective problems. In order to investigate a PSO-based multi-objective (Pareto front) feature selection algorithm, one of the most important tasks is to determine a good leader $(gbest)$ for each particle from a set of potential nondominated solutions. NSGAII is one of the most popular evolutionary multi-objective techniques [47]. Li [48] introduces the idea of NSGAII into PSO to develop a multi-objective PSO algorithm and achieves promising results on benchmark function optimization. However, this algorithm has never been applied to feature selection problems. In this paper, we investigate a PSO-based multi-objective feature selection algorithm (NSPSOFS) based on the idea of nondominated sorting in NSGAII to see whether a relatively simple multi-objective PSO can achieve good performance for feature selection problems.

Algorithm 1: Pseudo-Code of NSPSOFS

```
1  begin
2    divide $Dataset$ into a Training set and a Test set;
3    initialize the position and velocity of each particle in the swarm ($Swarm$);
4    while Maximum Iterations is not reached do
5      evaluate two objective values of each particle; /* number of features and the classification error rate on the Training set */
6      identify the particles ($nonDomS$) that have nondominated solutions in $Swarm$;
7      calculate crowding distance of each particle in $nonDomS$;
8      Sort particles in $nonDomS$ based on the crowding distance;
9      copy all the particles in $Swarm$ to a union ($Union$);
10     for $i = 1$ to Population Size ($P$) do
11       update the $pbest$ of particle $i$;
12       randomly selecting a $gbest$ for particle $i$ from the highest ranked (least crowded) solutions in $nonDomS$;
13       update the velocity and postion of particle $i$
14       add the updated particle $i$ to $Union$;
15     end
16     identify different levels of nondominated fronts $F = (F_1, F_2, F_3, \ldots)$ in $Union$;
17     empty the current $Swarm$ for the next iteration;
18     $i = 1$;
19     while $|Swarm| < P$ do
20       if ($|Swarm| + |F_i| \leq P$) then
21         add $F_i$ to $Swarm$;
22         $i = i + 1$;
23       end
24       if ($|Swarm| + |F_i| > P$) then
25         calculate crowding distance in $F_i$;
26         sort particles in $F_i$;
27         add the ($P - |Swarm|$) least crowded particles to $Swarm$;
28       end
29     end
30   end
31   calculate the classification error rate of the solutions (feature subsets) in the $F_1$ on the test set; /* $F_1$ is the achieved Pareto front */
32   return the positions of particles in $F_1$;
33   return the training and test classification error rates of the solutions in $F_1$;
34 end
```

Algorithm 1 shows the pseudocode of NSPSOFS. The two most important ideas in NSPSOFS are to select a good $gbest$ for each particle and to update the swarm during the evolutionary process. As shown in Algorithm 1, in each iteration, the fitness values (the number of features and the classification error rate) of each particle are calculated. Then, the algorithm identifies the nondominated solutions in the swarm according to the fitness values. The crowding distance of each nondominated solution is calculated, and all of the nondominated solutions are sorted according to the crowding distance. When updating the swarm to a new iteration, a $gbest$ for each particle is randomly selected from the highest ranked nondominated solutions, which are the least crowded solutions. For each particle, $pbest$ is replaced with the new position only if the new position dominates the current $pbest$. After determining the $gbest$ and $pbest$, the new velocity and the new position of each particle are calculated according to (1) and (2). The old positions (solutions) and the new positions of all particles are first combined into one union. The nondominated solutions in the union are called the first nondominated front, which are excluded from the union. Then, the nondominated solutions in the new union are called the second nondominated front. The following levels of nondominated fronts are identified by repeating this procedure. For the next iteration, solutions (particles) are selected from the top levels of the nondominated fronts, starting from the first front (from line 17 to line 29). If the number of solutions needed is larger than the number of solutions in the current nondominated front, all the solutions are added into the next iteration. Otherwise, the solutions in the current nondominated front are ranked according to the crowding distance, and the highest ranked solutions are added into the next iteration.

C. PSO-Based Multi-Objective Feature Selection Algorithm 2: CMDPSOFS

NSPSOFS can extend PSO to tackle multi-objective problems. However, NSPSOFS has a potential limitation of losing the diversity of the swarm quickly during the evolutionary process. Specifically, when using the idea of NSGAII to update the population, many of the particles in the new iteration may be identical. Because new particles are selected from the combination of current particles and the updated particles, all nondominated particles that share the same position will be added into the next iteration. Therefore, the diversity of the swarm might be lost fast during the evolutionary process. In order to better address feature selection problems, we use another multi-objective PSO to develop a multi-objective feature selection algorithm, CMDPSOFS, which is based on the ideas of crowding, mutation, and dominance [49]. CMDPSO has never been applied to feature selection problems.

Algorithm 2 shows the pseudocode of CMDPSOFS. In order to address the main issue of determining a good leader ($gbest$), CMDPSOFS employs a leader set to store the nondominated solutions as the potential leaders for each particle. A $gbest$ is selected from the leader set according to their crowding distances and a binary tournament selection. Specifically, a crowding factor is employed to decide which nondominated solutions should be added into the leader set and kept during the evolutionary process. The binary tournament selection is used to select two solutions from the leader set, and the less crowded solution is chosen as the $gbest$. The maximum size of the leader set is usually set as the number of particles in the swarm. Mutation operators are adopted to keep the diversity of the swarm and to improve the search ability of the algorithm.

A dominance factor is adopted to determine the size of the archive, which is the number of nondominated solutions that the algorithm reports. The solutions (feature subsets) in the final archive are used for classification on the test set in each data set.

Algorithm 2: Pseudo-Code of CMDPSOFS

```
1  begin
2    divide $Dataset$ into a Training set and a Test set; initialize the swarm;
3    initialize the set of leaders $LeaderSet$ and $Archive$
4    calculate the crowding distance of each member in $LeaderSet$;
5    while Maximum Iterations is not reached do
6      for each particle do
7        select a leader ($gbest$) from $LeaderSet$ for each particle by using a binary tournament selection based on the crowding distance;
8        update the velocity and position of particle $i$
9        apply mutation operators;
10       evaluate two objective values for each particle; /* number of features and the classification error rate on the Training set */
11       update the $pbest$ of each particle;
12     end
13     identify the nondominated solutions (particles) to update $LeaderSet$;
14     send leaders to $Archive$;
15     calculate the crowding distance of each member in $LeaderSet$;
16   end
17   calculate the classification error rate of the solutions in $Archive$ on the test set;
18   return the solutions in $Archive$ and their training and test classification error rates;
19 end
```

Note that CMDPSOFS employs two different mutation operators, uniform mutation in which the variability range allowed for each decision variable is kept constant over generations and nonuniform mutation in which the variability range allowed for each decision variable decreases over time. The two mutation operators are used to maintain the diversity of the swarm and to improve the search ability of the algorithm. In order to achieve this, CMDPSOFS randomly divides the whole swarm into three different groups in the initialization procedure. The first group does not have any mutation. The second group employs uniform mutation to keep the global search ability, and the third group employs nonuniform mutation to keep the local search ability. Furthermore, the three groups have the same leader set, which allows them to share their success to take advantage of the different behaviors to improve the abilities to search for the Pareto nondominated solutions. Meanwhile, in CMDPSOFS, w is a random value in [0.1, 0.5], and c_1 and c_2 are random values in [1.5, 2.0], which are different from most of other PSO-based algorithms in which these values are constants. This is used as a convenient way in addressing the problem of tuning these parameters for different test problems.

Both NSPSOFS and CMDPSOFS follow the basic update strategies of standard PSO. As multi-objective algorithms, both NSPSOFS and CMDPSOFS employ a crowding distance to the nondominated solutions (potential $gbest$) to keep the diversity of the selected $gbest$ for particles. The main differences between NSPSOFS and CMDPSOFS are the following: 1) how to store the nondominated solutions (potential $gebst$). In NSPSOFS, there is no external set to store nondominated solutions, and all of the nondominated solutions are kept and updated within the swarm. CMDPSOFS includes an external leader set, which is used to store the nondominated solutions. The leader set is updated from iteration to iteration. 2) How to choose a $gbest$ for each particle. NSPSOFS ranks all of the nondominated solutions according to their crowding distance; then, a $gbest$ is randomly selected from the highest ranked (least crowded) nondominated solutions. In CMDPSOFS, a binary tournament selection is applied to select two nondominated solutions from the leader set, and the less crowded solution is used as $gbest$. 3) When updating the swarm to a new iteration, NSPSOFS combines the new solutions (after applying the velocity and position update equations) and the old solutions into a union. Different levels of nondominated solutions are identified from the union to form the swarm in the next iteration. The nondominated solutions in the last iteration are reported by NSPSOFS. In CMDPSOFS, two different mutation operators are applied together with the update equations, and the solutions in the archive are reported as final solutions. 4) Parameters, such as w, c_1, and c_2, in NSPSOFS are constants, and in CMDPSOFS, they are random values within known ranges.

IV. Experimental Design

A. Benchmark Techniques

In order to examine the performance of the two PSO-based multi-objective feature selection algorithms, two conventional wrapper feature selection methods, two single objective algorithms, and three well-known evolutionary multi-objective algorithms are used as benchmark techniques in the experiments. The two conventional methods are linear forward selection (LFS) and greedy stepwise backward selection (GSBS). The two single objective algorithms are ErFS and 2SFS, described in Section III. The three multi-objective algorithms are NSGAII, strength Pareto evolutionary algorithm 2 (SPEA2), and Pareto archived evolutionary strategy (PAES).

LFS and GSBS were derived from SFS and SBS, respectively. LFS [50] restricts the number of features that are considered in each step of the forward selection, which can reduce the number of evaluations. Therefore, LFS is computationally less expensive than SFS and can obtain good results. More details can be seen in the literature [50].

The greedy stepwise based feature selection algorithm can move either forward or backward in the search space [51]. Given that LFS performs a forward selection, a backward search is chosen in greedy stepwise to form a GSBS. GSBS starts with all available features and stops when the deletion of

TABLE I
DATA SETS

Dataset	# Features	# Classes	# Instances
Wine	13	3	178
Australian	14	2	690
Zoo	17	7	101
Vehicle	18	4	846
German	24	2	1000
World Breast Cancer -Diagnostic (WBCD)	30	2	569
Ionosphere	34	2	351
Lung Cancer	56	3	32
Hillvalley	100	2	606
Musk Version 1 (Musk1)	166	2	476
Madelon	500	2	4400
Isolet5	617	2	1559

any remaining feature results in a decrease in evaluation, i.e., the accuracy of classification.

NSGAII is one of the most popular evolutionary multi-objective algorithms proposed by Deb *et al.* [47]. The main principle of NSGAII is the use of fast nondominated sorting technique and the diversity preservation strategy. The fast nondominated sorting technique is used to rank the parent and offspring populations to different levels of nondominated solution fronts. A density estimation based on the crowding distance is adopted to keep the diversity of the population. More details can be seen in the literature [47].

SPEA2 is a popular evolutionary multi-objective algorithm proposed by Zitzler *et al.* [52]. The main principle is the fine-gained fitness assignment strategy and the use of an archive truncation method. In SPEA2, the fitness of each individual is the sum of its strength raw fitness and a density estimation. A new population is constructed by the nondominated solutions in both the original population and the archive. When the number of nondominated solutions is larger than the population size, the archive truncation method is applied to determine whether a nondominated solution should be selected or not according to the distance to its kth nearest neighbor. More details can be seen in the literature [52].

PAES is an evolutionary multi-objective algorithm proposed by Knowles and Corne [53]. The authors claimed that PAES may represent the simplest possible nontrival algorithm capable of generating diverse solutions in the Pareto front. The main idea of PAES is the use of a local search and the use of an archive of previously found nondominated solutions. PAES was proposed as a baseline approach for Pareto multi-objective algorithms, and it has been used to compare with SPEA2 in the literature [52].

B. Data Sets and Parameter Settings

Table I shows the 12 data sets used in the experiments, which were chosen from the UCI machine learning repository [54]. They were selected to have various numbers of features (from 13 to 617), classes (from 2 to 7), and instances (from 32 to 4400), and they are used as representative samples of the problems that the PSO-based multi-objective feature selection algorithms can address.

In the experiments, all of the instances in each data set are randomly divided into two sets: 70% as the training set and 30% as the test set. During the training process, each particle (individual) represents **one** feature subset. The classification performance of a selected feature subset is evaluated by 10-fold cross-validation on the training set. Note that 10-fold cross-validation is performed as an inner loop in the training process to evaluate the classification performance of a single feature subset on the training set and it does not generate ten feature subsets. After the training process, the selected features are evaluated on the test set to obtain the testing classification error rate. A detailed discussion of why and how 10-fold cross-validation is applied in this way is given by [15].

All of the algorithms are wrapper approaches, i.e., needing a learning algorithm in the evolutionary training process to evaluate the classification performance of the selected feature subset. Any learning algorithm can be used here, such as NB, DT, and SVM. One of the simplest and commonly used learning algorithms [39], [40], K-nearest neighbor (KNN), was chosen in the experiments. We use $K = 5$ in KNN (5NN) to simplify the evaluation process, and 5NN implemented in Java machine learning library (Java-ML) [55] is used here.

Waikato Environment for Knowledge Analysis (Weka) [56] is used to run the experiments using LFS and GSBS for feature selection. All of the settings in LFS and GSBS are kept to the defaults, except that backward search is chosen in the greedy stepwise approach to perform GSBS for feature selection. During the feature selection process, 5NN with 10-fold cross-validation in Weka is employed to evaluate the classification accuracy on the training set. In order to make fair comparisons, all of the feature subsets selected by LFS, GSBS, and other feature selection methods are tested by 5NN in Java-ML on the test sets.

In all of the PSO-based algorithms, the fully connected topology is used, the maximum velocity $v_{\max} = 0.6$, the population size $P = 30$, and the maximum iteration $T = 100$. In ErFS, 2SFS, and NSPSOFS, the inertia weight $w = 0.7298$, and the acceleration constants $c_1 = c_2 = 1.49618$. These values are chosen based on the common settings in the literature [8], [57]. In the CMDPSO, w is a random value in [0.1, 0.5], c_1 and c_2 are random values in [1.5, 2.0], and the mutation rate is $1/n$, where n is the number of available features (dimensionality). These values are based on the settings of an equivalent algorithm in the literature [49]. According to our previous experiments, the threshold θ in ErFS, 2SFS, NSPSOFS, and CMDPSOFS is set as 0.6. In 2SFS, the first 50 iterations are set as the first stage, and the last 50 iterations are the second stage. We assume that the number of features is less important than the classification performance. Therefore, $\alpha = 0.2$ in (8) in the second stage of 2SFS. In NSGAII, SPEA2, and PAES, the representation of each individual is the same as the commonly used GA for feature selection algorithms [22], [24], where each individual is encoded by an n-bit binary string and n is the number of available features. The bit with value "1" indicates that the feature is selected in the subset, and "0" otherwise. A bit-flip mutation operator is applied in three methods, and a single point crossover operator is used in NSGAII and SPEA2. The mutation rate is $1/n$, where n is the number of available features (dimensionality) and the crossover probability is 0.9. For each data set, all of the algorithms have been conducted for

40 independent runs, except for LFS and GSBS, because both of them are deterministic methods.

V. Results and Discussions

For each data set, LFS and GSBS obtain a unique solution (feature subset). ErFS and 2SFS obtain a single solution in each of the 40 independent runs. Multi-objective algorithms, NSPSOFS, CMDPSOFS, NSGAII, SPEA2, and PAES obtain a set of nondominated solutions in each run. In order to compare these three kinds of results, the single solution in LFS and GSBS, 40 solutions (from 40 runs) that resulted from ErFS and 2SFS are presented in this section. Forty sets of feature subsets achieved by one multi-objective algorithm are first combined into one union set. In the union set, for the feature subsets including the same number of features (e.g., m), their classification error rates are averaged. The average classification error rate is assigned as the average classification performance of the subsets with m features. Therefore, a set of average solutions is obtained by using the average classification error rates and the corresponding numbers (e.g., m). The set of average solutions is called the *average* Pareto front and presented here. Besides the average Pareto front, the nondominated solutions in the union set are also presented to compare with the solutions achieved by single objective algorithms. Since PAES achieved similar results to SPEA2, only the results of NSGAII and SPEA2 are shown in this section.

TABLE II
Experimental Results of LFS, GSBS, ErFS, and 2SFS

	Wine					Australian				
Method	All	LFS	GSBS	ErFS	2SFS	All	LFS	GSBS	ErFS	2SFS
AveSize	13	7	8	8	8	14	4	12	3.88	3.42
AveEr	23.46	25.93	14.81	4.04	4.04	29.95	29.95	30.43	14.52	15.76
BestEr				0	0				12.56	12.56
StdEr				1.83E-2	1.83E-2				3.6E-2	4.56E-2
	Zoo					Vehicle				
Method	All	LFS	GSBS	ErFS	2SFS	All	LFS	GSBS	ErFS	2SFS
AveSize	17	8	7	9.18	9.18	18	9	16	9.52	8.65
AveEr	19.05	20.95	20	4.5	4.5	16.14	16.93	24.21	15	15.05
BestEr				2.86	2.86				12.99	12.99
StdEr				90.1E-4	90.1E-4				79E-4	77.8E-4
	German					WBCD				
Method	All	LFS	GSBS	ErFS	2SFS	All	LFS	GSBS	ErFS	2SFS
AveSize	24	3	18	13.48	11.92	30	10	25	13.42	5
AveEr	32	31.33	35.67	30.59	30.85	7.02	11.11	16.37	6.61	6.46
BestEr				28	28				5.26	5.26
StdEr				1.33E-2	1.18E-2				55.8E-4	75.3E-4
	Ionosphere					Lung				
Method	All	LFS	GSBS	ErFS	2SFS	All	LFS	GSBS	ErFS	2SFS
AveSize	34	4	30	12.58	12.05	56	6	33	27.35	27.38
AveEr	16.19	13.33	21.9	11.6	11.86	30	10	10	28	27.75
BestEr				6.67	8.57				20	10
StdEr				2.14E-2	1.89E-2				6E-2	6.89E-2
	Hillvalley					Musk1				
Method	All	LFS	GSBS	ErFS	2SFS	All	LFS	GSBS	ErFS	2SFS
AveSize	100	8	90	47.32	47.05	166	10	122	86.48	85.58
AveEr	43.41	35.38	50.55	42.46	42.43	16.08	14.69	23.78	15.42	15.42
BestEr				38.19	38.19				11.19	11.19
StdEr				1.52E-2	1.55E-2				2.04E-2	1.99E-2
	Madelon					Isolet5				
Method	All	LFS	GSBS	ErFS	2SFS	All	LFS	GSBS	ErFS	2SFS
AveSize	500	7	489	258.1	256.48	617	24	560	318.7	315.62
AveEr	29.1	35.38	48.72	23.45	23.48	1.55	1.66	2.84	1.43	1.43
BestEr				20.51	20.64				1.23	1.25
StdEr				1.22E-2	1.26E-2				9.98E-4	9.65E-4

A. Results of Benchmark Techniques: LFS, GSBS, ErFS, and 2SFS

Table II shows the experimental results of LFS, GSBS, ErFS, and 2SFS. In the tables, "All" shows the results of using all available features for classification. "AveSize" represents the average size of the feature subsets evolved by PSO in the 40 runs. "AveEr," "BestEr," and "StdEr" show the average, the smallest, and the standard deviation of the test classification error rates in the 40 runs, respectively.

1) Results of LFS and GSBS: Based on Table II, in most cases, LFS selected a small number of features and achieved a similar or even lower classification error rate than using all available features. GSBS could reduce the number of features but only reduced the classification error rate on a few data sets. In most cases, LFS outperformed GSBS in terms of both the number of features and the classification performance. The results suggest that LFS as a forward selection algorithm is more likely to obtain some optimality of the small feature subsets than GSBS (backward selection) because of the different starting points. Feature subsets selected by GSBS may still have redundancy.

2) Results of ErFs: According to Table II, in almost all data sets, PSO with the basic fitness function (ErFS) evolved a feature subset, which only contained around half (or less than half in six data sets) of the available features and achieved a lower classification error rate than using all features. All of the standard deviation values (StdEr) are smaller than 0.05, except in the Lung data set, which only has a small number of instances and the classification performance changes more than in a data set with more instances. All of the standard deviation values in Table II are very small, which indicates that all algorithms are considerably stable and statistically significant testing is not necessary here. As the objective functions of LFS, GSBS, and ErFS are to minimize the classification error rate only, the results also suggest that PSO as an evolutionary search technique can obtain a better feature subset than LFS (in almost all cases) and GSBS (in all cases).

3) Results of 2SFS: According to Table II, 2SFS evolved a feature subset with around half (or less than half) of the available features and achieved a lower classification error rate than using all features in all data sets. Comparing with LFS, 2SFS achieved a lower classification error rate than LFS in most cases and a smaller number of features in some cases. 2SFS outperformed GSBS in terms of both the number of features and the classification performance in almost all data sets. 2SFS evolved a smaller number of features and achieved similar or even better classification performance than ErFS in almost all data sets.

The results in Table II suggest that PSOs with different fitness functions in ErFS and 2SFS obtain different feature subsets with different numbers of features and classification error rates. 2SFS further improves the feature subset obtained by ErFS in almost all cases. In the first stage, the fitness function (8) could guide PSO to search for the feature subset with the minimum classification error rate; then in the second stage, it guide PSO to search for the smallest feature subset with the already achieved high classification performance. Therefore, 2SFS can successfully remove redundant features and achieve high classification performance in most cases. However, in the German and Ionosphere data sets, 2SFS could further reduce the number of features, but the classification performances are slightly worse than ErFS. Therefore, Pareto front multi-objective feature selection algorithms are needed to better address the problems.

B. Results of NSPSOFS and CMDPSOFS

As 2SFS usually achieved better performance than the other methods, only the results of 2SFS are included in this section as a baseline to test the performance of NSPSOFS and CMDPSOFS. Fig. 2 shows the experimental results of NSPSOFS, CMDPSOFS, and 2SFS. In Fig. 2, each chart corresponds to one of the data set used in the experiments. On the top of each chart, the numbers in the brackets show the number of available features and the classification error rate using all features. In each chart, the horizontal axis shows the number of features selected, and the vertical axis shows the classification error rate.

In Fig. 2, "-A" stands for the average Pareto front resulted from NSPSOFS or CMDPSOFS in the 40 independent runs. "-B" represents the nondominated solutions resulted from NSPSOFS or CMDPSOFS in the 40 independent runs. "2SFS" shows the 40 solutions of 2SFS. In some data sets, 2SFS may evolve the same feature subset in different runs, and they are shown in the same point in the chart. Therefore, although 40 results are presented, there may be less than 40 distinct points shown in a chart.

1) Results of NSPSOFS: As shown in Fig. 2, in most cases, the average Pareto front of NSPSOFS (NSPSOFS-A) includes two or more solutions, which selected a smaller number of features and achieved a lower classification error rate than using all features. For the same number of features, there are a variety of combinations of features with different classification performances. The feature subsets obtained in different runs may include the same number of features but different classification error rates. Therefore, although the solutions obtained in each run are nondominated, some solutions in the average Pareto front may dominate others.

According to Fig. 2, the nondominated solutions (NSPSOFS-B) include one or more feature subsets, which achieved better classification performance than using all features in all data sets. In most data sets, NSPSOFS evolved a feature subset that only selected one or two features but achieved a lower classification error rate than using all features. For example, NSPSOFS selected only around 1.8% of the available features (1 from 56) in the Lung data set and selected 2% of the available features (2 from 100) in the Hillvalley data set but achieved higher classification performance than using all features. In almost all cases, the number of features was reduced to 10% or less, except for around 17.64% in Zoo, 26.4% in Madelon, and 21.72% in Isolet5.

Comparing NSPSOFS with 2SFS, it can be seen that the classification error rates of 2SFS and the average Pareto front (NSPSOFS-A) are similar in many data sets, but the number of features in NSPSOFS-A is usually smaller than that in 2SFS. In almost all cases, NSPSOFS-B included fewer features and achieved a lower classification error rate than 2SFS.

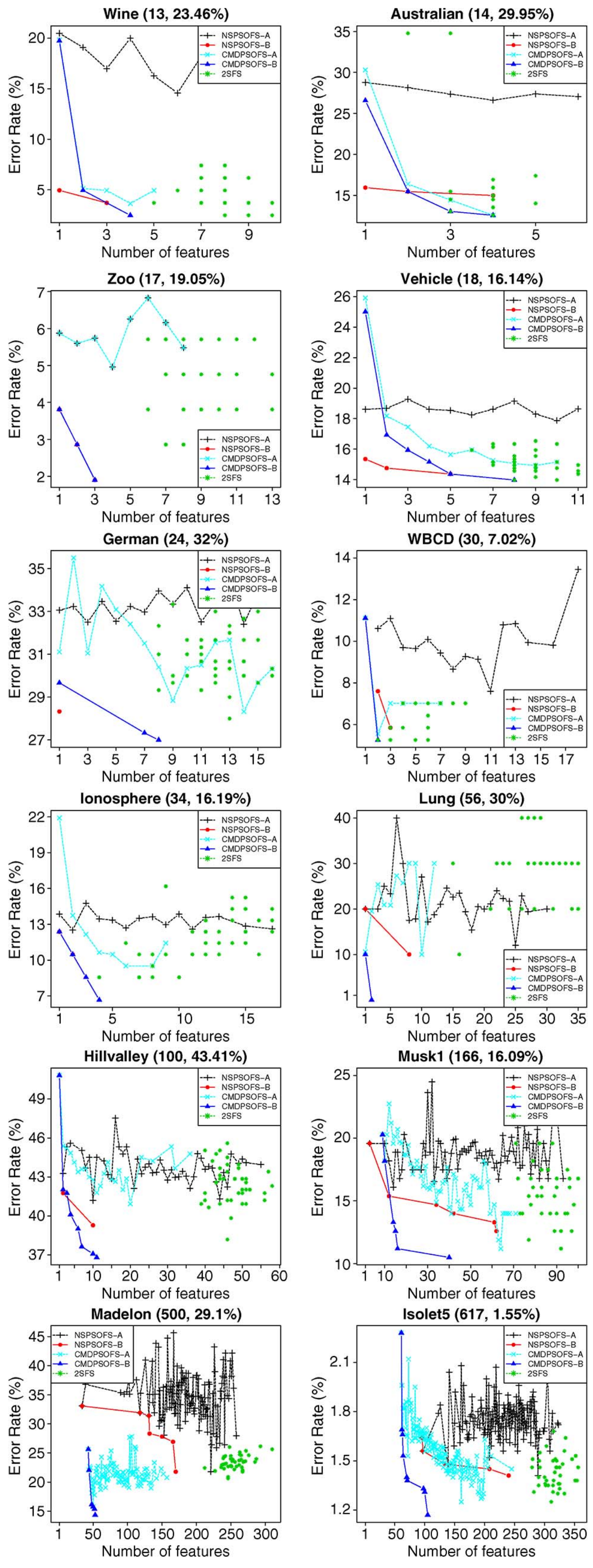

Fig. 2. Experimental results of NSPSOFS, CMDPSOFS, and 2SFS.

The results suggest that, although NSPSOFS shares the same parameter settings as 2SFS, NSPSOFS as a multi-objective technique can effectively explore the Pareto front and obtain feature subsets, which include a smaller number of features and achieve better classification performance than 2SFS.

2) Results of CMDPSOFS: According to Fig. 2, in all cases, CMDPSOFS-A includes two or more feature subsets, which successfully selected a smaller number of relevant features and achieved a lower classification error rate than using all features.

As shown in Fig. 2, CMDPSOFS-B includes one or more solutions, which achieved a lower classification error rate than using all features in all data sets. In most data sets, CMDPSOFS evolved a feature subset, which only selected one or two features but achieved a better classification performance than using all features. For example, CMDPSOFS selected only around 1.8% of the available features (1 from 56) in the Lung data set and selected only 2% of the available features (2 from 100) in the Hillvalley data set but achieved a lower classification error rate than using all features. In almost all data sets, the number of features selected is less than 10% of the total number of features, except for around 16.67% in the Vehicle data set and 10.37% in the Isolet5 data set.

Comparing CMDPSOFS with 2SFS, in all data sets, the classification performance of CMDPSOFS-A is similar to that of 2SFS, but the number of features in CMDPSOFS-A is smaller than that in 2SFS. Moreover, CMDPSOFS-B outperformed 2SFS in terms of both the number of features and the classification performance in all data sets.

The results suggest that CMDPSOFS as a multi-objective technique guided by the two objectives can effectively explore the Pareto front and achieve better feature subsets than 2SFS. Note that proper settings of the α value and the division of the two stages in 2SFS might increase its performance on either the number of features or the classification performance, but this requires prior knowledge and is problem specific. This is a disadvantage of 2SFS compared with Pareto front multi-objective algorithms.

C. Comparisons Between NSPSOFS, CMDPSOFS, NSGAII, and SPEA2

In order to test the performance of NSPSOFS and CMDPSOFS, they are compared with three popular evolutionary multi-objective algorithms, namely, NSGAII, SPEA2, and PAES. PAES achieved similar results to SPEA2 in terms of the number of features and the classification performance. Therefore, the results of PAES are not presented in this section. Comparisons between NSPSOFS, NSGAII, and SPEA2 are shown in Fig. 3. Comparisons between CMDPSOFS, NSGAII, and SPEA2 are shown in Fig. 4.

1) Comparisons Between NSPSOFS, NSGAII, and SPEA2: According to Fig. 3, in most cases, the average Pareto fronts, NSPSOFS-A, NSGAII-A, and SPEA2-A include two or more feature subsets, which selected a smaller number of features but achieved better classification performance than using all features. There are also some dominated solutions in the three average Pareto fronts, and the reason is the same as discussed in Section V-B.

Comparing NSPSOFS-A with NSGAII-A and SPEA2-A, in most cases, the classification performance of three methods are similar. Although in a few cases the classification error rates

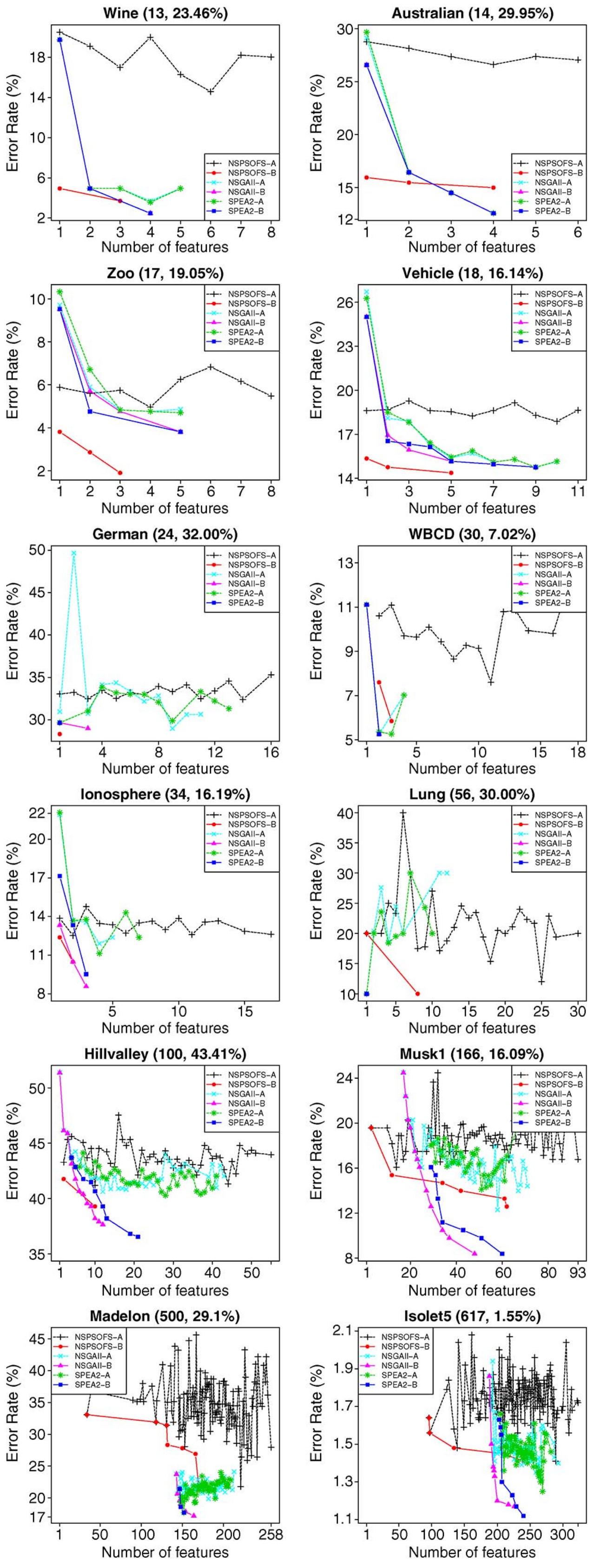

Fig. 3. Comparisons between NSPSOFS, NSGAII, and SPEA2.

of NSPSOFS-A are slightly higher than that of NSGAII-A and SPEA2-A, the number of features is usually smaller in NSPSOFS-A than that in NSGAII-A and SPEA2-A.

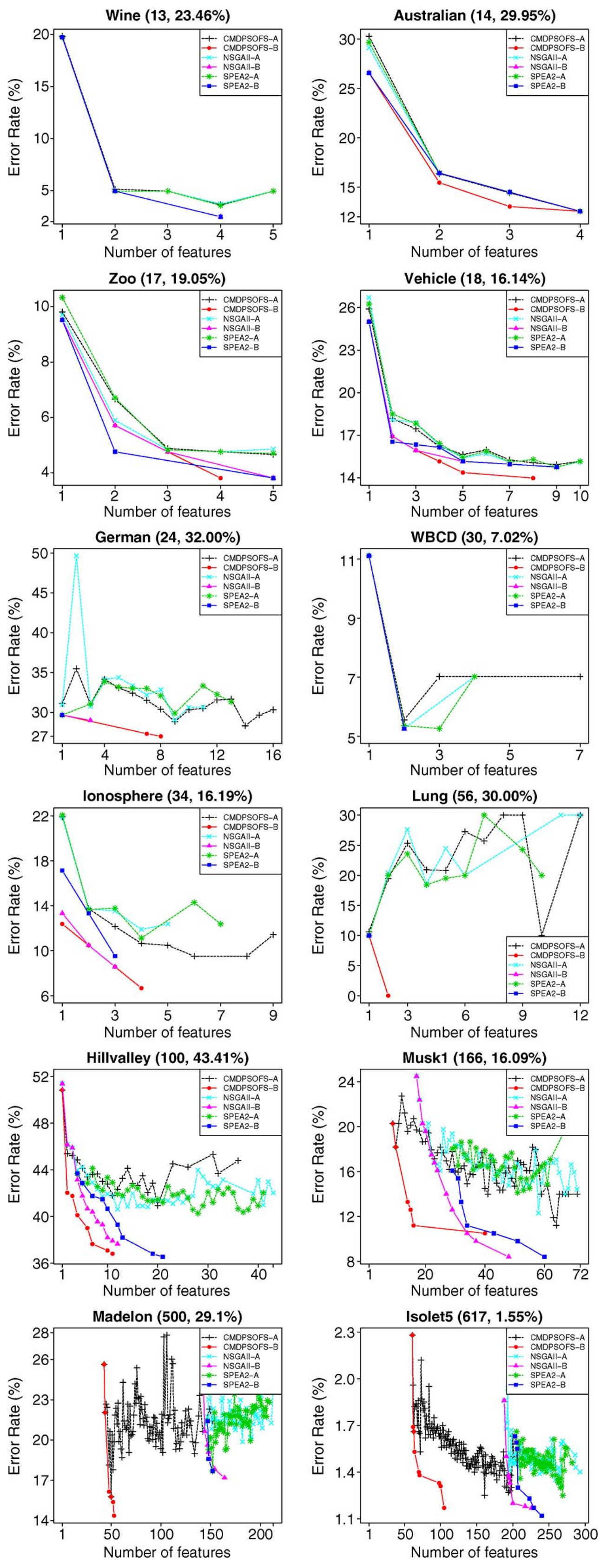

Fig. 4. Comparisons between CMDPSOFS, NSGAII, and SPEA2.

In terms of the nondominated solutions (NSPSOFS-B, NSGAII-B, and SPEA2-B), results in different data sets show different patterns. Specifically, in three data sets (Zoo, Vehicle, and German), NSPSOFS-B dominated NSGAII-B and SPEA2-B, while in two data sets (WBCD and Lung), NSPSOFS-B was dominated by NSGAII-B and SPEA2-B. In the other data sets, there are always solutions in NSPSOFS-B, which dominate solutions in NSGAII-B and SPEA2-B, although there are also solutions in NSPSOFS-B dominated by solutions in NSGAII-B and SPEA2-B. In most cases, NSGAII-B outperformed SPEA2-B in terms of the classification performance and the number of features.

The results in Fig. 3 suggest that NSPSOFS, NSGAII, and SPEA2 are generally competitive with each other. However, as discussed in Section III-C, NSPSOFS has a potential limitation of quickly losing the diversity of the swarm because of the updating mechanism. The performance of a PSO-based multi-objective algorithm could be improved if this limitation can be addressed.

2) Comparisons Between CMDPSOFS, NSGAII, and SPEA2: As shown in Fig. 4, the average Pareto fronts, CMDPSOFS-A, NSGAII-A, and SPEA2-A achieved similar classification performance in all data sets. However, the number of features in CMDPSOFS-A is usually smaller than that of NSGAII-A and SPEA2-A, especially in the Madelon and Isolet5 data sets.

Comparing the nondominated solutions CMDPSOFS-B with NSGAII-B and SPEA2-B, it can be seen that, in almost all data sets, CMDPSOFS-B achieved better results than NSGAII-B and SPEA2-B in terms of both the number of features and the classification performance. In a data set with a large number of features, the better performance of CMDPSOFS is more obvious, especially for the number of feature criterion. For example, in the Madelon data set, the number of features in NSGAII-B and SPEA2-B is around 150, while this number in CMDPSOFS-B is only around 50, which means that CMDPSOFS further reduced by two thirds of the number of features selected.

The results show that CMDPSOFS can address the limitation in NSPSOFS and achieve better performance than NSPSOFS, NSGAII, and SPEA2 in terms of both the number of features and the classification performance.

D. Results of Hypervolume Indicator

In order to further compare the results of multi-objective algorithms, NSPSOFS, CMDPSOFS, NSGAII, SPEA2, and PAES, the hypervolume indicator [58] is used in the experiments. In each run, each method obtained two Pareto fronts, which are a *training Pareto front* according to the training classification performance and the number of features, and a *testing Pareto front* according to the testing classification performance and the number of features. Therefore, for each method, we calculated two sets of hypervolume values based on the Pareto-fronts on the training process and the testing process, respectively. Therefore, for each method, 40 hypervolume values on the training process and 40 hypervolume values on the testing process were calculated. As the calculation of hypervolume needs the true Pareto front, which is not available in the tested data sets, we first combine the training (or testing) Pareto front of these five methods into a union and then identify the Pareto

TABLE III
T-TEST ON HYPERVOLUME RATIOS ON TESTING ACCURACY

Dataset	Wine		Australian		Zoo		Vehicle		German		WBCD	
	NS	CMD	NS	CMD	NS	CMD	NS	CMD	NS	CMD	NS	CMD
NSPSOFS		=		-		=		=		=		?
CMDPSOFS	=		+		=		=		=		?	
NSGAII	=	=	+	=	=	=	=	=	=	=	?	?
SPEA2	=	=	+	=	=	=	=	=	=	=	?	?
PAES	=	=	=	=	=	=	=	=	=	=	?	?
Dataset	Lung		Ionosphere		Hillvalley		Musk1		Madelon		Isolet5	
	NS	CMD	NS	CMD	NS	CMD	NS	CMD	NS	CMD	NS	CMD
NSPSOFS		?		+		+		+		=		+
CMDPSOFS	?		-		-		-		=		-	
NSGAII	?	?	-	=	-	+	-	+	=	=	+	+
SPEA2	?	?	=	+	-	+	-	+	=	=	+	+
PAES	?	?	-	=	-	+	-	+	=	=	-	+

TABLE IV
T-TEST ON HYPERVOLUME RATIOS ON TRAINING ACCURACY

Dataset	Wine		Australian		Zoo		Vehicle		German		WBCD	
	NS	CMD	NS	CMD	NS	CMD	NS	CMD	NS	CMD	NS	CMD
NSPSOFS		+		+		+		+		+		+
CMDPSOFS	-		-		-		-		-		-	
NSGAII	-	=	-	=	-	-	-	-	-	=	-	=
SPEA2	-	=	-	=	-	=	-	=	-	=	-	=
PAES	-	=	-	=	-	-	-	-	-	-	-	=
Dataset	Lung		Ionosphere		Hillvalley		Musk1		Madelon		Isolet5	
	NS	CMD	NS	CMD	NS	CMD	NS	CMD	NS	CMD	NS	CMD
NSPSOFS		+		+		+		+		+		+
CMDPSOFS	-		-		-		-		-		-	
NSGAII	-	-	-	-	-	+	-	+	=	+	+	+
SPEA2	-	-	-	-	=	+	-	+	=	+	+	+
PAES	-	-	-	-	-	-	-	-	-	-	-	-

front in the union as the "true Pareto front" to calculate the hypervolume values. The hypervolume values are normalized to hypervolume ratios, which is the division of the hypervolume value of a Pareto front and the hypervolume of the "true Pareto front." In order to compare NSPSOFS and CMDPSOFS with the other three algorithms, NSGAII, SPEA2, and PAES, the Student's T-test was performed on their hypervolume ratios, where the significance level was set as 0.05 (or confidence interval is 95%).

1) Results of Hypervolume on the Testing Process: Table III shows the results of the T-test between NSPSOFS, CMDPSOFS, NSGAII, SPEA2, and PAES on the hypervolume ratios in the *testing* process, where "NS" and "CMD" represent NSPSOFS and CMDPSOFS. In Table III, "+" ("-") indicates that NSPSOFS or CMDPSOFS is significantly better (worse) than another corresponding algorithm. "=" means that they are similar. In the WBCD and Lung data sets, "?" means that the hypervolume ratio could not be obtained because the extracted "true Pareto front" only contains two points and its hypervolume value is zero.

Table III shows that, compared with CMDPSOFS, NSGAII, SPEA2, and PAES, NSPSOFS achieved similar results in most cases, although NSPSOFS achieved better results on the Australian data set and worse results on the Hillvalley and Musk1 data sets. Table III also shows that CMDPSOFS achieved similar results with other methods in most cases. In the data sets with a large number of features, such as Hillvalley, Musk1, and Isolet5, CMDPSOFS achieved significantly better results than NSPSOFS, NSGAII, SPEA2, and PAES.

2) Results of Hypervolume on the Training Process: Table IV shows the results of the T-test between NSPSOFS, CMDPSOFS, NSGAII, SPEA2, and PAES on the hypervolume value ratios in the *training* process. It can be seen that NSPSOFS achieved slightly worse results than other methods in most cases, but NSPSOFS achieved better results than NSGAII and SPEA2 in the Isolet5 data set, where the number of features is large. Table IV also shows that, in the data sets with a relatively small number of features, CMDPSOFS usually achieved similar results to NSGAII, SPEA2, and PAES. In the data sets with large numbers of features, such as Hillvalley, Musk1, Madelon, and Isolet5, CMDPSOFS achieved significantly better results than NSPSOFS, NSGAII, and SPEA2. Although CMDPSOFS achieved slightly worse results than PAES on the training set, CMDPSOFS achieved similar or better results than PAES on the test set (shown in Table III), which is considered due to the overfitting problem in PAES.

TABLE V
COMPARISONS ON COMPUTATIONAL TIME (IN MINUTES)

	Wine	Australian	Zoo	Vehicle	German	WBCD
NSPSOFS	0.29	4.83	0.11	7.77	12.61	4.34
CMDPSOFS	0.25	4	0.09	6.59	9.3	2.71
NSGAII	0.24	3.99	0.09	6.59	9.32	2.67
SPEA2	0.2	3.24	0.07	5.53	7.64	2.13
PAES	0.21	3.95	0.07	6.35	8.85	2.01
	Lung	Ionosphere	Hillvalley	Musk1	Madelon	Isolet5
NSPSOFS	0.02	1.79	46.04	10.66	868.75	374.63
CMDPSOFS	0.01	1.54	23.49	6.02	394.45	200.71
NSGAII	0.01	1.06	28.88	7.46	721.71	338.23
SPEA2	0.01	0.87	30.23	6.69	694.91	331.91
PAES	0.01	1.04	24.88	5.73	560.83	276.97

E. Comparisons on Computational Time

Table V shows the average computational time (in minutes) used by NSPSOFS, CMDPSOFS, NSGAII, SPEA2, and PAES in one run.

From Table V, it can be seen that, for data sets that have a small number of features and instances, SPEA2 and PAES generally use less time than the other three methods. However, all algorithms can perform one run in relatively short time, a few minutes or even less than one minute, such as the Wine, Zoo, and Lung data sets. For data sets with a large number of features and instances, CMDPSOFS and PAES used shorter time than the other three methods, especially for the Madelon and Isolet5 data sets, where CMDPSOFS used much less time than NSPSOFS, NSGAII, and SPEA2. In such large data sets, computational time is more important than that in small data sets. CMDPSOFS can finish the evolutionary training process in much shorter time and achieve better results, which suggests that this method is a better choice than the other four methods in real-world applications, where a large number of features and instances are involved.

All of the methods have the same number of evaluations as they have the same number of individuals and iterations during the evolutionary process. NSGAII and NSPSOFS generally consumed more time, which is probably caused by the different levels of nondominated ranking mechanism and the calculation of crowding distances. CMDPSOFS also involves ranking, but it only happens in the small leader set. Therefore, ranking in CMDPSOFS does not cost as much time as NSGAII and NSPSOFS. More importantly, during the evolutionary training process, CMDPSOFS selected smaller numbers of features than the other four algorithms, which cost much less time for the

10-fold cross-validation to calculate the training classification performance in each evaluation, especially for the data sets with a large number of features.

F. Further Discussions

Experimental results show that the NSPSOFS and CMDPSOFS can be successfully used for feature selection, but NSPSOFS could not achieve as good results as CMDPSOFS. The main reasons are that feature selection problems are difficult problems with many local optima. CMDPSOFS employs different mechanisms to maintain the diversity of both the leader set and the swarm. Specifically, it selects and filters out crowded leaders and uses different mutation operators to maintain the diversity of the swarm to avoid stagnation in local optima.

As discussed in Section III, CMDPSOFS has an external leader set to store the nondominated solutions, which are used as potential leaders for each particle. Different from other multi-objective evolutionary techniques, CMDPSOFS employs a crowding factor to maintain and update the leader set from generation to generation, which helps in filtering out some crowded potential leaders. This mechanism will be more helpful for the data sets with a large number of features, where most nondominated solutions in the leader set may have similar numbers of features and slightly different classification error rates. These crowded nondominated solutions have a chance to be selected as a leader, which will limit the exploration ability of the algorithm. Eliminating such solutions helps the algorithm to explore the solution space more effectively to search for better results. Meanwhile, when selecting a leader for a particle, CMDPSOFS employs a binary tournament to select two nondominated solutions from the leader set, and the less crowded one will be selected as the leader. This mechanism attempts to keep the diversity of the swarm in future iterations and further avoids particles from converging to local optima. Moreover, CMDPSOFS employs different mutation operators for different groups of particles to maintain the diversity of the swarm and to balance its global and local search abilities. By contrast, NSPSOFS is less effective in terms of avoiding stagnation in local optima. NSPSOFS employs different levels of Pareto fronts to store the already found nondominated solutions. Therefore, all of the nondominated solutions will be kept in the swarm from generation to generation. Such nondominated solutions may be duplicated, and the swarm may lose diversity quickly, which will lead to the problem of premature convergence. Although NSGAII employs the same mechanism to store the nondominated solutions, NSGAII also employs mutation and crossover operators to keep the diversity. Therefore, NSPSOFS usually could not achieve as good results as CMDPSOFS and NSGAII.

G. How to Chose a Single Solution

In multi-objective problems, a set of Pareto front (nondominated) solutions is obtained, which are tradeoffs between different objectives. However, selecting a single solution from these solutions is an important issue. In feature selection problems, the two main objectives are minimizing the number of features and maximizing the classification performance, and the decision is a tradeoff between these two objectives. If the Pareto front was "smooth" in that adding each additional feature would reduce the classification error rate by a small, but significant margin, users could weigh the tradeoff criteria to determine their preferred solutions. However, the results produced show that this is not usually the case. Adding features beyond a known limit number does not increase the classification performance. For example, the Musk1 data set in Fig. 2, the subset with the lowest classification error rate in CMDPSOFS-B, has 40 features. Adding more features does not further increase the classification performance because it does not increase the relevance but increases the redundancy and the dimensionality. Meanwhile, removing features may not lead to a decrease in classification error rate as relevant features may be removed. In the Musk1 data set, the solution that stands in the "elbow" of CMDPSOFS-B would be a good choice. Therefore, visually seeing these possible solutions in the Pareto front assists users in determining their preferred compromises. This is actually the main reason why solving feature selection problems as multi-objective tasks is important.

VI. Conclusion and Future Work

This paper has conducted the first study on multi-objective PSO for feature selection. Specifically, we considered two PSO-based multi-objective feature selection algorithms, NSPSOFS and CMDPSOFS. The two feature selection algorithms were examined and compared with two conventional methods (LFS and GSBS), a single objective algorithm (ErFS), a two-stage method (2SFS), and three well-known multi-objective algorithms (NSGAII, SPEA2, and PAES) on 12 benchmark data sets of varying difficulty. Experimental results show that both NSPSOFS and CMDPSOFS can achieve more and better feature subsets than LFS, GSBS, ErFS, and 2SFS. NSPSOFS achieved similar (or slightly worse in some cases) results to NSGAII, SPEA2, and PAES in most cases. CMDPSOFS outperformed all other methods mentioned previously not only on the number of features but also on the classification performance. In particular, for the data sets with a large number of features, CMDPSOFS achieved better classification performance using fewer features and shorter computational time than the other four multi-objective algorithms.

This paper finds that, as multi-objective algorithms, NSPSOFS and CMDPSOFS can search the solution space more effectively to obtain a set of nondominated solutions instead of a single solution. Examining the Pareto front achieved by the multi-objective algorithms can assist users in choosing their preferred solutions to meet their own requirements. Meanwhile, this paper discovers that the potential limitation of losing the diversity of the swarm quickly in NSPSOFS limits its performance for feature selection. More importantly, this paper highlights the benefits of the strategies of maintaining the diversity of the swarm in CMDPSOFS. A crowding factor together with a binary tournament selection can effectively select and filter out some crowded nondominated solutions in the leader set. Different mutation operators in different groups of particles can effectively keep the diversity of the swarm and balance its global and local search abilities. These strategies account for the superior performance of CMDPSOFS over NSPSOFS, NSGAII, SPEA2, and PAES, especially on the data sets with large numbers of features.

The Pareto front multi-objective algorithms, NSPSOFS and CMDPSOFS, can achieve a set of good feature subsets, but it is unknown whether the achieved Pareto fronts can be improved or not. In the future, we will further investigate the multi-objective PSO-based feature selection approach to better explore the Pareto front of nondominated solutions in feature selection problems. We intend to investigate the use of binary multi-objective PSO for feature selection and compare its performance with that of continuous multi-objective PSO. Meanwhile, we will also investigate whether using a given learning algorithm in a wrapper feature selection approach can select a good or near-optimal feature subset for other learning algorithms for classification tasks.

References

[1] I. A. Gheyas and L. S. Smith, "Feature subset selection in large dimensionality domains," *Pattern Recognit.*, vol. 43, no. 1, pp. 5–13, Jan. 2010.

[2] M. Dash and H. Liu, "Feature selection for classification," *Intell. Data Anal.*, vol. 1, no. 1–4, pp. 131–156, 1997.

[3] A. Unler and A. Murat, "A discrete particle swarm optimization method for feature selection in binary classification problems," *Eur. J. Oper. Res.*, vol. 206, no. 3, pp. 528–539, Nov. 2010.

[4] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Mar. 2003.

[5] A. Whitney, "A direct method of nonparametric measurement selection," *IEEE Trans. Comput.*, vol. C-20, no. 9, pp. 1100–1103, Sep. 1971.

[6] T. Marill and D. Green, "On the effectiveness of receptors in recognition systems," *IEEE Trans. Inf. Theory*, vol. IT-9, no. 1, pp. 11–17, Jan. 1963.

[7] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, 1995, vol. 4, pp. 1942–1948.

[8] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE Int. CEC*, 1998, pp. 69–73.

[9] Y. Liu, G. Wang, H. Chen, and H. Dong, "An improved particle swarm optimization for feature selection," *J. Bionic Eng.*, vol. 8, no. 2, pp. 191–200, Jun. 2011.

[10] A. Mohemmed, M. Zhang, and M. Johnston, "Particle swarm optimization based AdaBoost for face detection," in *Proc. IEEE CEC*, 2009, pp. 2494–2501.

[11] J. Kennedy and R. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Proc. IEEE Int. Conf. Syst., Man, Cybern., Comput. Cybern. Simul.*, 1997, vol. 5, pp. 4104–4108.

[12] L. Y. Chuang, H. W. Chang, C. J. Tu, and C. H. Yang, "Improved binary PSO for feature selection using gene expression data," *Comput. Biol. Chem.*, vol. 32, no. 1, pp. 29–37, Feb. 2008.

[13] C. L. Huang and J. F. Dun, "A distributed PSO-SVM hybrid system with feature selection and parameter optimization," *Appl. Soft Comput.*, vol. 8, no. 4, pp. 1381–1391, Sep. 2008.

[14] A. P. Engelbrecht, *Computational Intelligence: An Introduction.*, 2nd ed. Chichester, U.K.: Wiley, 2007.

[15] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artif. Intell.*, vol. 97, no. 1/2, pp. 273–324, Dec. 1997.

[16] S. C. Yusta, "Different metaheuristic strategies to solve the feature selection problem," *Pattern Recognit. Lett.*, vol. 30, no. 5, pp. 525–534, Apr. 2009.

[17] S. Stearns, "On selecting features for pattern classifier," in *Proc. 3rd Int. Conf. Pattern Recog.*, Coronado, CA, 1976, pp. 71–75.

[18] P. Pudil, J. Novovicova, and J. V. Kittler, "Floating search methods in feature selection," *Pattern Recognit. Lett.*, vol. 15, no. 11, pp. 1119–1125, Nov. 1994.

[19] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Proc. 9th Int. Workshop Mach. Learn.*, 1992, pp. 249–256.

[20] C. Cardie, "Using decision trees to improve case-based learning," in *Proc. 10th ICML*, 1993, pp. 25–32.

[21] H. Almuallim and T. G. Dietterich, "Learning Boolean concepts in the presence of many irrelevant features," *Artif. Intell.*, vol. 69, no. 1/2, pp. 279–305, Sep. 1994.

[22] B. Chakraborty, "Genetic algorithm with fuzzy fitness function for feature selection," in *Proc IEEE ISIE*, 2002, vol. 1, pp. 315–319.

[23] B. Chakraborty, "Feature subset selection by particle swarm optimization with fuzzy fitness function," in *Proc. 3rd Int. Conf. ISKE*, 2008, vol. 1, pp. 1038–1042.

[24] T. M. Hamdani, J.-M. Won, A. M. Alimi, and F. Karray, "Multi-objective feature selection with NSGA II," in *Proc. 8th ICANNGA Part I*, 2007, vol. 4431, pp. 240–247.

[25] M. Banerjee, S. Mitra, and H. Banka, "Evolutionary rough feature selection in gene expression data," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 37, no. 4, pp. 622–632, Jul. 2007.

[26] Z. X. Zhu, Y. S. Ong, and M. Dash, "Wrapper–filter feature selection algorithm using a memetic framework," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 1, pp. 70–76, Feb. 2007.

[27] D. Muni, N. Pal, and J. Das, "Genetic programming for simultaneous feature selection and classifier design," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 1, pp. 106–117, Feb. 2006.

[28] K. Neshatian and M. Zhang, "Genetic programming for feature subset ranking in binary classification problems," in *Proc. Eur. Conf. Genetic Program.*, 2009, pp. 121–132.

[29] K. Neshatian, M. Zhang, and P. Andreae, "Genetic programming for feature ranking in classification problems," in *Simulated Evolution and Learning*, vol. 5361, *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2008, pp. 544–554.

[30] K. Neshatian and M. Zhang, "Pareto front feature selection: Using genetic programming to explore feature space," in *Proc. 11th Annu. GECCO*, New York, 2009, pp. 1027–1034.

[31] K. Neshatian and M. Zhang, "Unsupervised elimination of redundant features using genetic programming," in *Proc. 22nd Australas. Joint Conf. Artif. Intell.*, vol. 5866, *Lecture Notes in Computer Science*, 2009, pp. 432–442.

[32] K. Neshatian and M. Zhang, "Using genetic programming for context-sensitive feature scoring in classification problems," *Connect. Sci.*, vol. 23, no. 3, pp. 183–207, 2011.

[33] K. Neshatian and M. Zhang, "Improving relevance measures using genetic programming," in *Proc. EuroGP*, vol. 7244, *Lecture Notes in Computer Science*, 2012, pp. 97–108.

[34] H. H. Gao, H. H. Yang, and X. Y. Wang, "Ant colony optimization based network intrusion feature selection and detection," in *Proc. Int. Conf. Mach. Learn. Cybern.*, 2005, vol. 6, pp. 3871–3875.

[35] H. Ming, "A rough set based hybrid method to feature selection," in *Proc. Int. Symp. KAM*, 2008, pp. 585–588.

[36] X. Wang, J. Yang, X. Teng, W. Xia, and R. Jensen, "Feature selection based on rough sets and particle swarm optimization," *Pattern Recognit. Lett.*, vol. 28, no. 4, pp. 459–471, Mar. 2007.

[37] G. Azevedo, G. Cavalcanti, and E. Filho, "An approach to feature selection for keystroke dynamics systems based on PSO and feature weighting," in *Proc. IEEE CEC*, 2007, pp. 3577–3584.

[38] S. W. Lin, K. C. Ying, S. C. Chen, and Z. J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines," *Expert Syst. Appl.*, vol. 35, no. 4, pp. 1817–1824, Nov. 2008.

[39] C. S. Yang, L. Y. Chuang, and J. C. Li, "Chaotic maps in binary particle swarm optimization for feature selection," in *Proc. IEEE Conf. SMCIA*, 2008, pp. 107–112.

[40] L. Y. Chuang, S. W. Tsai, and C. H. Yang, "Improved binary particle swarm optimization using catfish effect for feature selection," *Expert Syst. Appl.*, vol. 38, no. 10, pp. 12 699–12 707, Sep. 2011.

[41] H. N. A. Hamed, N. K. Kasabov, and S. M. Shamsuddin, "Quantum-inspired particle swarm optimization for feature selection and parameter optimization in evolving spiking neural networks for classification tasks," *Evol. Algorithms*, vol. 1, pp. 132–148, 2011.

[42] M. A. Esseghir, G. Goncalves, and Y. Slimani, "Adaptive particle swarm optimizer for feature selection," in *Proc. Int. Conf. IDEAL*, 2010, pp. 226–233.

[43] B. Xue, M. Zhang, and W. N. Browne, "Single feature ranking and binary particle swarm optimisation based feature subset ranking for feature selection," in *Proc. 35th ACSC*, vol. 122, *Lecture Notes in Computer Science*, Melbourne, Australia, 2012, pp. 27–36.

[44] L. Cervante, B. Xue, L. Shang, and M. Zhang, "A dimension reduction approach to classification based on particle swarm optimisation and rough set theory," in *Proc. 25th Australas. Joint Conf. Artif. Intell.*, *Lecture Notes in Computer Science*, Dec. 4, 2012.

[45] L. Cervante, B. Xue, M. Zhang, and L. Shang, "Binary particle swarm optimisation for feature selection: A filter based approach," in *Proc. IEEE CEC*, 2012, pp. 1–8.

[46] B. Xue, M. Zhang, and W. N. Browne, "New fitness functions in binary particle swarm optimisation for feature selection," in *Proc. IEEE CEC*, 2012, pp. 1–8.

[47] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[48] X. Li, "A non-dominated sorting particle swarm optimizer for multiobjective optimization," in *Proc. Annu. GECCO*, 2003, pp. 37–48.

[49] M. R. Sierra and C. A. C. Coello, "Improving PSO-based multi-objective optimization using crowding, mutation and epsilon-dominance," in *Proc. EMO*, 2005, pp. 505–519.

[50] M. Gutlein, E. Frank, M. Hall, and A. Karwath, "Large-scale attribute selection using wrappers," in *Proc. IEEE Symp. CIDM*, 2009, pp. 332–339.

[51] R. Caruana and D. Freitag, "Greedy attribute selection," in *Proc. ICML*, 1994, pp. 28–36.

[52] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," in *Proc. Evol. Methods Des., Optim., Control Appl. Ind. Probl.*, 2001, pp. 95–100.

[53] J. Knowles and D. Corne, "The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimisation," in *Proc. CEC*, 1999, vol. 1, pp. 98–105.

[54] A. Frank and A. Asuncion, UCI Machine Learning Repository , 2010.

[55] T. Abeel, Y. V. de Peer, and Y. Saeys, "Java-ml: A machine learning library," *J. Mach. Learn. Res.*, vol. 10, pp. 931–934, Dec. 2009.

[56] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The elements of statistical learning: Data mining, inference and prediction," in *The Mathematical Intelligencer*, vol. 27. New York: Springer-Verlag, 2005, pp. 83–85.

[57] F. Van Den Bergh, "An analysis of particle swarm optimizers," Ph.D. dissertation, Faculty Nat. Agricult. Sci., Univ. Pretoria, Pretoria, South Africa, 2001.

[58] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler, "Theory of the hypervolume indicator: Optimal distributions and the choice of the reference point," in *Proc. 10th ACM SIGEVO Workshop FOGA*, New York, 2009, pp. 87–102.

Bing Xue (M'10) received the B.Sc. degree in management from the Henan University of Economics and Law, Zhengzhou, China, in 2007 and the M.Sc. degree in management from Shenzhen University, Shenzhen, China, in 2010. She has been working toward the Ph.D. degree in computer science in the Evolutionary Computation Research Group, Victoria University of Wellington, Wellington, New Zealand, since October 2010.

She currently holds a Victoria Doctoral Scholarship. Her current research interests include evolutionary computation, particularly particle swarm optimization, multi-objective optimization, and feature selection and dimension reduction for classification in machine learning and data mining. She serves as a reviewer of international conferences, including IEEE Congress on Evolutionary Computation and Australia Joint Conference on Artificial Intelligence, and international journals, including IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION AND NEUROCOMPUTING.

She is a member of ACM and the ACM SIGEVO group.

Mengjie Zhang (SM'10) received a BE and an ME in 1989 and 1992 from Artificial Intelligence Research Center, Agricultural University of Hebei, China, and a PhD in computer science from RMIT University, Australia in 2000.

Since 2000, he has been working at Victoria University of Wellington, New Zealand. He is currently Professor of Computer Science and heads the Evolutionary Computation Research Group. His research is mainly focused on evolutionary computation, particularly genetic programming and particle swarm optimisation with application areas of image analysis, multi-objective optimisation, job-shop scheduling, classification with unbalanced data, bioinformatics, and feature selection and dimension reduction for classification with high dimensions. He has published over 200 academic papers in refereed international journals and conferences. He has been serving as an associated editor or editorial board member for five international journals including IEEE Transactions on Evolutionary Computation, Evolutionary Computation Journal (MIT Press) and Genetic Programming and Evolvable Machines (Springer) and as a reviewer of over fifteen international journals. He has been serving as a steering committee member and a program committee member for over eighty international conferences. He has supervised over thirty postgraduate research students.

Prof Zhang is a Senior Member of IEEE, a member of the IEEE Computer Society, the IEEE CI Society and the IEEE SMC Society. He is also a member of the IEEE CIS Evolutionary Computation Technical Committee, a member of the IEEE CIS Intelligent System Application Technical Committee, a vicechair of the IEEE CIS Task Force on Evolutionary Computer Vision and Image Processing, and a committee member of the IEEE New Zealand Central Section. He is a member of ACM and the ACM SIGEVO group.

Will N. Browne received a BEng Mechanical Engineering, Honours degree from the University of Bath, UK in 1993, MSc in Energy (1994) and EngD (Engineering Doctorate scheme, 1999) University of Wales, Cardiff. After eight years lecturing in the Department of Cybernetics, University Reading, UK, he was appointed Senior Lecturer, School of Engineering and Computer Science, Victoria University of Wellington, NZ in 2008. Dr Browne's main area of research is Applied Cognitive Systems. This includes Learning Classifier Systems, Cognitive Robotics, Modern Heuristics for industrial application. Blue skies research includes analogues of emotions, abstraction, memories, Small-Worlds phenomenon, dissonance and machine consciousness.

He has been serving as Track chair (Co) of Genetics-based Machine Learning for Genetic and Evolutionary Computation Conference. Organising committee of the International Workshop on Learning Classifier Systems (IWLCS) from 2009/2010. Editor in chief for the Australasian Conference on Robotics and Automation 2012 and organised the Cognitive Robotics Intelligence and Control EPSRC (UK)/NSF (USA) sponsored workshop 2006. Programme committee membership includes the LCS and other GBML track at Genetic and Evolutionary Computation Conference [2004-present], Congress on Evolutionary Computation [2004-present], Hybrid Intelligent Systems [2003-present], Parallel Problem Solving from Nature [2004-present] and Robot and Human Interactive Communication [2005-present]. Journal reviewer: Journal of Soft Computing, IEEE Transactions on Evolutionary Computation, IEEE Trans. Systems Man and Cybernetics, Journal of Engineering Manufacture, Journal of Pattern Analysis and Applications, Cognitive Computation.

Dr. Browne is a member of IMechE, ACM and the ACM SIGEVO group. He has published over 50 academic papers in books, refereed international journals and conferences.