Multi-Objective Particle Swarm Optimisation (PSO) for Feature Selection

Bing Xue Mengjie Zhang Will N. Browne

School of Engineering and Computer Science
Victoria University of Wellington
PO Box 600, Wellington 6140, New Zealand
{Bing.Xue, Mengjie.Zhang,Will.Browne}@ecs.vuw.ac.nz

ABSTRACT

Feature selection (FS) is an important data preprocessing technique, which has two goals of minimising the classification error and minimising the number of features selected. Based on particle swarm optimisation (PSO), this paper proposes two multi-objective algorithms for selecting the Pareto front of non-dominated solutions (feature subsets) for classification. The first algorithm introduces the idea of non-dominated sorting based multi-objective genetic algorithm II into PSO for FS. In the second algorithm, multi-objective PSO uses the ideas of crowding, mutation and dominance to search for the Pareto front solutions. The two algorithms are compared with two single objective FS methods and a conventional FS method on nine datasets. Experimental results show that both proposed algorithms can automatically evolve a smaller number of features and achieve better classification performance than using all features and feature subsets obtained from the two single objective methods and the conventional method. Both the continuous and the binary versions of PSO are investigated in the two proposed algorithms and the results show that continuous version generally achieves better performance than the binary version. The second new algorithm outperforms the first algorithm in both continuous and binary versions.

Categories and Subject Descriptors

I.2 [**Computing Methodologies**]: Artificial Intelligence

General Terms

Algorithm, Performance

Keywords

Feature selection, particle swarm optimisation, multi-objective optimisation

1. INTRODUCTION

In many problems such as classification, a large number of features are introduced to well describe the target concepts. However, the large number of features causes the problem of "the curse of dimensionality", which is a major obstacle in classification. Meanwhile, not all of the features are useful for classification. Irrelevant and redundant features may even increase the classification error rate. Feature selection (FS) could reduce the number of features by eliminating irrelevant and redundant features, and thus result in enhanced efficiency and/or increased classification performance [5].

Based on whether a learning algorithm is included in the training process or not, existing FS approaches can be broadly classified into two categories: filter and wrapper approaches. A filter FS approach is conducted as a preprocessing procedure and the search process is independent of a learning algorithm. In wrapper approaches, a learning algorithm is part of the evaluation function to determine the goodness of the selected feature subset. Wrappers can usually achieve better results than filters while filters are more general and computationally less expensive than wrappers [14].

A FS algorithm explores the search space of different feature combinations to reduce the number of features and simultaneously optimise the classification performance. In FS, the size of the search space for n features is 2^n. So in most situations, it is impractical to conduct an exhaustive search [14]. Therefore, the search strategy is the key part in FS. Different search techniques have been applied to FS such as greedy search, but most of them suffer from the problem of becoming stuck in local optima and/or high computational cost [25, 17]. Therefore, an efficient global search technique is needed to develop a good FS algorithm.

Evolutionary computation techniques are well-known for their global search ability, and have been applied to the FS problems. These include particle swarm optimisation (PSO) [23, 26], genetic algorithms (GAs) [27] and genetic programming (GP) [20]. Compared with GAs and GP, PSO is easier to implement, has fewer parameters, computationally less expensive, and can converge more quickly [12]. Due to these advantages, two versions of PSO, namely continuous PSO and binary PSO, have been used for FS problems [23, 26, 19]. However, no study has been conducted to investigate the difference of using continuous PSO and binary PSO for FS. FS problems have two goals, which are maximising the classification performance (minimising the classification error rate) and minimising the number of features. These two objectives are usually conflicting and there is a trade-off between them. However, most of the existing FS approaches, including PSO based approaches, aim to maximise the classification performance only. Therefore, it is sought to use

PSO to develop a multi-objective FS approach to simultaneously minimising the classification error rate and minimising the number of features selected.

1.1 Goals

The overall goal of this paper is to develop a PSO based multi-objective FS approach to classification problems with the expectation of obtaining a set of non-dominated solutions, which contain a small number of features and achieve better classification performance than using all features. To achieve this goal, we will firstly develop two single objective FS methods with a fitness function considering classification error rate and the number of features. We will then develop two new FS methods based on two multi-objective PSO algorithms, *non-dominated sorting PSO (NSPSO)* using the idea of non-dominated sorting based multi-objective genetic algorithm II (NSGAII) and *CMDPSO* using the ideas of crowding, mutation and dominance. These proposed FS algorithms will be examined on benchmark problems of varying difficulty. Specifically, we will investigate

- whether single objective PSO based FS algorithms can select a small number of features and improve classification performance over using all features,
- whether the NSPSO based multi-objective FS algorithm can evolve a Pareto front of non-dominated solutions with a smaller number of features and better classification performance than single objective FS algorithms,
- whether a CMDPSO based multi-objective FS algorithm can obtain a better set of non-dominated solutions than NSPSO and outperform the single objective FS algorithms, and
- which of the two versions of PSO, continuous PSO and binary PSO, can achieve better performance for FS.

2. BACKGROUND

2.1 Particle Swarm Optimisation (PSO)

PSO is an evolutionary computation technique proposed by Kennedy and Eberhart in 1995 [10, 21]. In PSO, a population, called a *swarm*, of candidate solutions are encoded as particles in the search space. PSO starts with the random initialisation of a population of particles. The whole swarm move in the search space to search for the best solution by updating the position of each particle based on the experience of its own and its neighbouring particles [10, 21]. During movement, the current position of particle i is represented by a vector $x_i = (x_{i1}, x_{i2}, ..., x_{iD})$, where D is the dimensionality of the search space. The velocity of particle i is represented as $v_i = (v_{i1}, v_{i2}, ..., v_{iD})$, which is limited by a predefined maximum velocity, v_{max} and $v_{id}^t \in [-v_{max}, v_{max}]$. The best previous position of a 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 solution by updating the velocity and the position of each particle according to the following equations:

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \tag{1}$$

$$\begin{aligned} v_{id}^{t+1} &= w * v_{id}^t + c_1 * r_{1i} * (p_{id} - x_{id}^t) \\ &+ c_2 * r_{2i} * (p_{gd} - x_{id}^t) \end{aligned} \tag{2}$$

where t denotes the tth iteration. $d \in D$ denotes the dth dimension in the search space. w is inertia weight. c_1 and c_2 are acceleration constants. r_{1i} and r_{2i} are random values uniformly distributed in [0, 1]. p_{id} and p_{gd} represent the elements of *pbest* and *gbest* in the dth dimension.

PSO was originally proposed for solving continuous problems [10]. However, many problems, such as FS, occur in a discrete search space. Therefore, Kennedy and Eberhart [11] developed a binary PSO (BPSO) for discrete problems. In BPSO, x_{id}, p_{id} and p_{gd} are restricted to 1 or 0. The velocity in BPSO indicates the probability of the corresponding element in the position vector taking value 1. A sigmoid function $s(v_{id})$ is introduced to transform v_{id} to the range of (0, 1). BPSO updates the position of each particle according to the following formulae:

$$x_{id} = \begin{cases} 1, \text{ if } rand() < s(v_{id}) \\ 0, \ otherwise \end{cases} \tag{3}$$

where

$$s(v_{id}) = \frac{1}{1 + e^{-v_{id}}} \tag{4}$$

where $rand()$ is a random number selected from a uniform distribution in [0,1].

BPSO preserves the fundamental concept of the PSO algorithm. However, not all important characteristics of the PSO algorithm are completely present in BPSO [11].

2.2 Multi-Objective Optimisation

Multi-objective optimisation involves simultaneously optimising two or more conflicting objective functions. In mathematical terms, the formulae for a minimisition problem with multiple objective functions can be written as:

$$minimise \quad F(x) = [f_1(x), f_2(x), \ ... \ , f_k(x)] \tag{5}$$

subject to

$$g_i(x) \leqslant 0, \ i = 1, 2, \ ... \ m \tag{6}$$

$$h_i(x) \leqslant 0, \ i = 1, 2, \ ... \ l \tag{7}$$

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 minimised, $g_i(x)$ and $h_i(x)$ are the constraint functions of the problem.

The quality of a solution in multi-objective problems is explained in terms of trade-offs between two or more conflicting objectives. Let y and z be two solutions of the above k-objective minimisation problem. If the following conditions are met, one can say y dominates z (or z is dominated by y, or y is better than z):

$$\forall i : f_i(y) \leqslant f_i(z) \quad and \quad \exists f_i(y) < f_i(z) \tag{8}$$

When y is not dominated by any other solutions, y is referred to as a Pareto-optimal solution. The set of all Pareto-optimal solutions forms the trade-off surface in the search space, the Pareto front. An multi-objective algorithm is designed to search for a set of non-dominated solutions.

FS can be expressed as a two-objective minimisation problems with the objectives of minimising the number of features and the classification error rate.

2.3 Recent Work Related to FS

2.3.1 Classical FS Approaches

Sequential forward selection (SFS) [25] and sequential backward selection (SBS) [17] are two popular wrapper FS approaches, but suffer from the so-called nesting effect and easily trapped into local optima.

The FOCUS algorithm [1], a classical filter algorithm, exhaustively examines all feature subsets, then selects the smallest feature subset. However, the FOCUS algorithm is computationally inefficient because of the exhaustive search. The Relief algorithm [13] assigns a relevance 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.

2.3.2 Non-PSO Evolutionary computation Methods for FS

GP, GAs and ant colony optimisation (ACO) have been applied to FS. Based on GP and a variation of naïve Bayes, Neshatian and Zhang [20] propose a FS approach, where a bit-mask representation is used for feature subsets and a set of operators are used as primitive functions. GP is used to combine feature subsets and operators together to find the optimal subset of features. Experiments show that the dimensionality and processing time can be significantly reduced by the proposed algorithm.

Chakraborty [3] proposes a GA with fuzzy sets based fitness function to build a filter FS approach. This method have the same fitness function with BPSO based method in [4]. However, the performance of BPSO in [4] is better than that of this GA based algorithm.

Based on ACO and rough sets theory, He [18] proposes a filter based FS approach. The features included in the core of the rough sets is the starting point of the proposed method. Forward selection is adopted into the proposed method to search for the best feature subset. Experimental results show that the proposed approach achieves better classification performance with fewer features than a C4.5 based FS approach.

2.3.3 PSO based FS Approaches

PSO has recently gained more attention for solving FS problems. Based on continuous PSO and support vector machines (SVM), Azevedo et al. [2] propose a FS algorithm for keystroke dynamics systems. Experimental results show that the proposed algorithm produces better performance than a GA with SVM model regarding the classification error, processing time and feature reduction rate.

Mohemmed et al. [19] propose a hybrid method (PSOAdaBoost) which incorporated 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 would also speed up the process of training and increase the accuracy of weak classifiers in AdaBoost.

Based on BPSO, Unler and Murat [23] propose a FS algorithm with an adaptive selection strategy, where a feature is chosen not only according to the likelihood calculated by BPSO, but also to its contribution to the features already selected. Experimental results suggest that the proposed BPSO method outperforms the tabu search and scatter search algorithms.

Based on PSO and SVM, Huang et al. [9] propose a FS method in which BPSO is used to search the optimal feature subset and continuous PSO is used to simultaneously optimise the parameters in the kernel function of SVM. Experiments show that the proposed algorithm could determine the parameters and search the optimal feature subset simultaneously, and also achieve good classification performance.

Liu et al. [16] introduce a multi-swarm BPSO (MSPSO) algorithm to search for the optimal feature subset and optimise the parameters of SVM simultaneously. Experiments show that the proposed FS method could achieve higher classification accuracy than grid search, standard BPSO 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.

Many studies have shown that PSO is an efficient search technique for FS. However, most of the existing work aims to minimise the classification error and not much work has been conducted for solving a FS task as a multi-objective problem. Therefore, development of a multi-objective FS algorithm using PSO to simultaneously minimise the classification error and minimise the number of features is still an open issue.

3. PROPOSED APPROACHES

3.1 Single Objective PSO for FS

FS can be performed by BPSO as a single objective problem to minimise the classification error rate. The goal is to see whether BPSO can select a subset of features to achieve a lower classification error rate than using all features, and the results will be used to as a baseline to compare the performance of newly developed algorithms. The fitness function (Equation 9) is to minimise the classification error rate obtained by the selected feature subset during evolution.

$$Fitness_1 = ErrorRate = \frac{FP + FN}{TP + TN + FP + FN} \tag{9}$$

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

The representation of a particle in BPSO is a n-bit binary string, where n is the number of available features in the dataset and also the dimensionality of the search space. In the binary string, “1” represents that the feature is selected and “0” otherwise.

3.2 Single Objective PSO for Weighting Two Objectives in FS

The feature subset selected by BPSO may still contain potential redundancy because the fitness function (Equation 9) does not intend to minimise the number of features. A new fitness function (Equation 10) is proposed with the goals of minimising both the classification error rate and the number of features selected.

$$Fitness_2 = \alpha * \frac{\#Features}{\#All\ Features} + (1 - \alpha) * \frac{ErrorRate}{Error_0} \tag{10}$$

where $\alpha \in [0, 1]$, α and $(1 - \alpha)$ show the relative importance of the number of features and the classification error rate. $(1 - \alpha)$ is set larger than α because the classification performance is assumed more important than the number of features. However, the number of selected features $\#Features$ is usually much larger than the classification error rate $ErrorRate$. In order to balance these two components, $\#Features$ is divided by the total number of features $\#All\ Features$, which transforms the value to the range of (0, 1]. Meanwhile, the classification error rate is normalised via dividing it by the error rate using all available features $Error_0$.

The representation of a particle in this algorithm is the same as the n-bit binary string described in Section 3.1.

Algorithm 1: NSBPSO for Feature Selection

```
begin
    divide $Dataset$ into a training set and a test set; initialise each particle in the swarm ($Swarm$);
    while $Maximum\ Iterations\ is\ not\ met$ do
        evaluate two objective values of each particle ;        /* number of features and the training error rate */
        identify the non-dominated particles ($nonDomS$) in $Swarm$;
        calculate crowding distance of each particle in $nonDomS$;
        Sort particles in $nonDomS$ according to the crowding distance;
        copy all the particles in $Swarm$ to a form temporary swarm ($temSwarm$);
        for $i$=1 to $Population\ Size\ (P)$ do
            update the $pbest$ of particle $i$;
            randomly selecting a $gbest$ for particle $i$ from the highest ranked (least crowded) solutions in $nonDomS$;
            update the velocity and the position of particle $i$;
            add the updated particle $i$ to $temSwarm$;
        identify different levels of non-dominated fronts $F = (F_1, F_2, F_3, \ ...)$ in $temSwarm$;
        empty the current $Swarm$ for the next iteration;
        $i = 1$;
        while $|Swarm| < P$ do
            if $(|Swarm| + |F_i| \leqslant P)$ then
                calculate crowding distance of each particle in $F_i$;
                add $F_i$ to $Swarm$;
                $i = i + 1$;
            if $(|Swarm| + |F_i| > P)$ then
                calculate crowding distance of each particle in $F_i$;
                sort particles in $F_i$;
                add the $(P - |Swarm|)$ least crowded particles to $Swarm$;
    calculate the classification error rate of the solutions in the $F_1$ on the test set ;    /* $F_1$:achieved Pareto front */
    return the positions of particles in $F_1$;
    return the training and test classification error rates of the solutions in $F_1$;
```

3.3 NSPSO for FS

Equation 10 considers both the number of features and classification performance, but a proper value of α needs to be pre-determined. In most situations, users may want to make an informed decision from many available feature subsets. Therefore, it is needed to use multi-objective PSO to address FS problems with the objectives of minimising both the number of features and the classification error rate.

PSO was originally proposed as a single objective technique for continuous problems. In order to use PSO for multi-objective problems, we need to determine a good leader (*gbest*) for the swarm among a set of potential non-dominated solutions. Li [15] introduces the idea of non-dominated sorting in NSGAII [6] into PSO to develop a continuous multi-objective PSO algorithm and shows that the proposed algorithm can achieve promising results on the optimisation of several functions. However, this algorithm has never been applied to FS problems.

In this study we propose a multi-objective FS framework (NSPSO) and the continuous multi-objective PSO [15] is applied to search for the non-dominated solutions. Further, we propose a new binary multi-objective PSO using the idea of non-dominated sorting and applied in the proposed FS framework (NSPSO). Therefore, the FS framework, NSPSO, includes both the continuous and the binary versions of multi-objective PSO for FS. Algorithm 1 shows the pseudo-code of binary NSPSO for FS.

When using continuous NSPSO for FS, the representation of a particle is a vector of n real numbers and a threshold θ is needed to compare with the value x_i in the vector. Feature i is selected if $\theta > x_i$ and $\theta \leqslant x_i$ otherwise.

3.4 CMDPSO for FS

Although NSPSO has shown promising results on different problems [15], it has a potential limitation. By selecting particles from the combination of current swarm and the updated swarm, all non-dominated particles that share the same position will be added into the next iteration. Therefore, the diversity of the swarm might be lost during the evolutionary process. In order to better address FS problems, we develop another multi-objective FS framework, CMDPSO, based on multi-objective PSO using the ideas of crowding, mutation and dominance [22].

The ideas of crowding, mutation and dominance were used to in a continuous multi-objective PSO [22], but it has never been applied to FS and will employed in the proposed FS framework to develop a continuous versions of CMDPSO. Further, we develop a binary multi-objective PSO using the ideas of crowding, mutation and dominance to propose a binary CMDPSO FS algorithm. Except for the steps related to the selection of *gbest*, mutation and dominance, binay CMDPSO follows the basic steps of a PSO algorithm. In order to address the main issue of determining a good leader (*gbest*), a leader set is used to keep the non-dominated solutions as the potential leaders for each particle. The maximum size of the leader set is usually set as the number of individuals in the population. A crowding factor is employed to decide which non-dominated solutions should be added into the leader set and kept during the evolutionary process. Binary tournament selection based on the crowding factor is applied to choose a leader (*gbest*) for each particle from the leader set. A bit-flip mutation operator is adopted to keep the diversity of the swarm and improve the search ability of the algorithm. An archive is used to keep the non-dominated solutions and a dominance factor is adopt to determine the size of archive, which is also the number of non-dominated solutions that CMDBPSO reports.

In binary CMDPSO, the representation of each particle

Table 1: Datasets

Dataset	# features	# classes	# instances
Wine	13	3	178
Zoo	17	7	101
Vehicle	18	4	846
World Breast Cancer -Diagnostic (WBCD)	30	2	569
Sonar	60	2	208
Movementlibras	90	15	360
Hillvalley	100	2	606
Musk Version 1 (Musk1)	166	2	476
Isolet5	617	2	1559

is the same as the n-bit binary string described in Section 3.1. In continuous CMDPSO for FS, the representation of each particle is the same as the vector of n real numbers described in Section 3.3.

4. EXPERIMENTAL DESIGN

Table 1 shows the nine datasets used in the experiments, which are chosen from the UCI machine learning repository [7]. The nine datasets were selected to have different numbers of features (from 13 to 617), classes and instances as the representative samples of the problems that the proposed algorithms can address. In the experiments, the instances in each dataset are randomly divided into two sets: 70% as the training set and 30% as the test set.

All the algorithms are wrapper approaches, i.e. needing a learning algorithm in the evolutionary training process. Many learning algorithms can be used here and one of the simplest algorithms, K-nearest neighbour (KNN), was chosen in the experiments. We use K=5 in KNN (5NN) to simplify the evaluation process. The linear forward selection (LFS) [8] is used as a benchmark technique to examine the performance of the proposed approaches.

In both single objective and multi-objective algorithms, the parameters of PSO are set as follows: inertia weight $w = 0.7298$, acceleration constants $c_1 = c_2 = 1.49618$, maximum velocity $v_{max} = 6.0$, population size $P = 30$, maximum iteration $T = 100$. The fully connected topology is used in BPSO. These values are chosen based on the common settings in the literature [21, 24]. In the weighted single objective algorithm, $\alpha = 0.2$ is used in Equation 10 to give more weight to classification performance. The threshold used in the continuous version of PSO for FS is set to 0.6. For each dataset, each algorithm has been conducted for 40 independent runs.

In single objective algorithms, one single solution (feature subset) is obtained in each run while for multi-objective algorithms CMDPSO and NSPSO, a set of solutions (feature subsets) are obtained in each run. In order to compare these two kinds of algorithms, 40 results (from 40 runs) of each single objective algorithm are presented in the Section 5. 40 sets of feature subsets achieved by each multi-objective algorithm are firstly combined into one union set. In the union set, the classification error rate of feature subsets, which share the same number of features (e.g. m), are averaged. The average classification error rate is assigned as the classification performance of the subsets with m features. A set of average solutions is obtained by using the average classification error rates and the corresponding numbers. The set of average solutions is called the *average* Pareto front and presented in the Section 5. Besides the average Pareto front, the non-dominated solutions in the union set are also presented in the Section 5 to compare with the solutions achieved by single objective algorithms. Experiments have been conducted on nine datasets, but due to the page limit, results on six datasets are presented as other three datasets have similar results.

5. RESULTS AND DISCUSSIONS

Experimental results of the four algorithms on six of the nine datasets are shown in Figures 1 and 2. In each figure, 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 for classification and the vertical axis shows the classification error rate.

In Figure 1, "NSB-Ave" ("NSC-Ave") stands for the average Pareto front resulted from binary (continuous) NSPSO for FS in the 40 independent runs. "NSB-Best" ("NSC-Best") represents the non-dominated solutions of all solutions resulted from binary (continuous) NSPSO for FS. "Stand" represents the solutions of BPSO with the overall classification performance as the fitness function. "Weights" shows the solutions of BPSO with the fitness function considering both the number of features and the classification performance. In Figure 2, "CMDB- " and "CMDC- " show the results of binary and continuous CMDPSO.

For "Stand" and "Weights", in some datasets, the algorithm may evolve the same feature subset in different runs and they are shown in the same point in the chart. Therefore, for "Stand" or "Weights", 40 results are presented, but there may be less than 40 points shown in each chart.

5.1 Results of Single Objective PSO for FS

According to Figures 1, in all datasets, both "Stand" and "Weights" can evolve many feature subsets, which selected around half of the available features and achieved a lower classification error rate than using all features.

Comparing "Stand" with "Weights", in all datasets, the number of features evolved by "Weights" is usually smaller than that of "Stand", which is caused by the weight factor in the fitness function (Equation 10). There is no significant difference between the classification error rates achieved by "Stand" and "Weights". This shows that feature subsets evolved by "Stand" still have redundancy because the number of features was not considered in the fitness function.

The results suggest that in all datasets, BPSO with the overall classification error rate as fitness function can effectively select a feature subset that contains around half of the available features and increases classification performance. By considering the number of features in the fitness function, "Weights" can further reduce the number of features while maintaining the classification performance.

5.2 Results of NSPSO for FS

5.2.1 *Results of Binary NSPSO for FS*

According to Figure 1, in almost all datasets, the average Pareto front of binary NSPSO ("NSB-Ave") contains two or more solutions, which selected a small number of features and achieved a lower classification error than using all features. For the same number of features, there are a variety of combinations of features with different classification error rates, the feature subsets obtained in different runs are usually different. Therefore, although the solutions obtained in

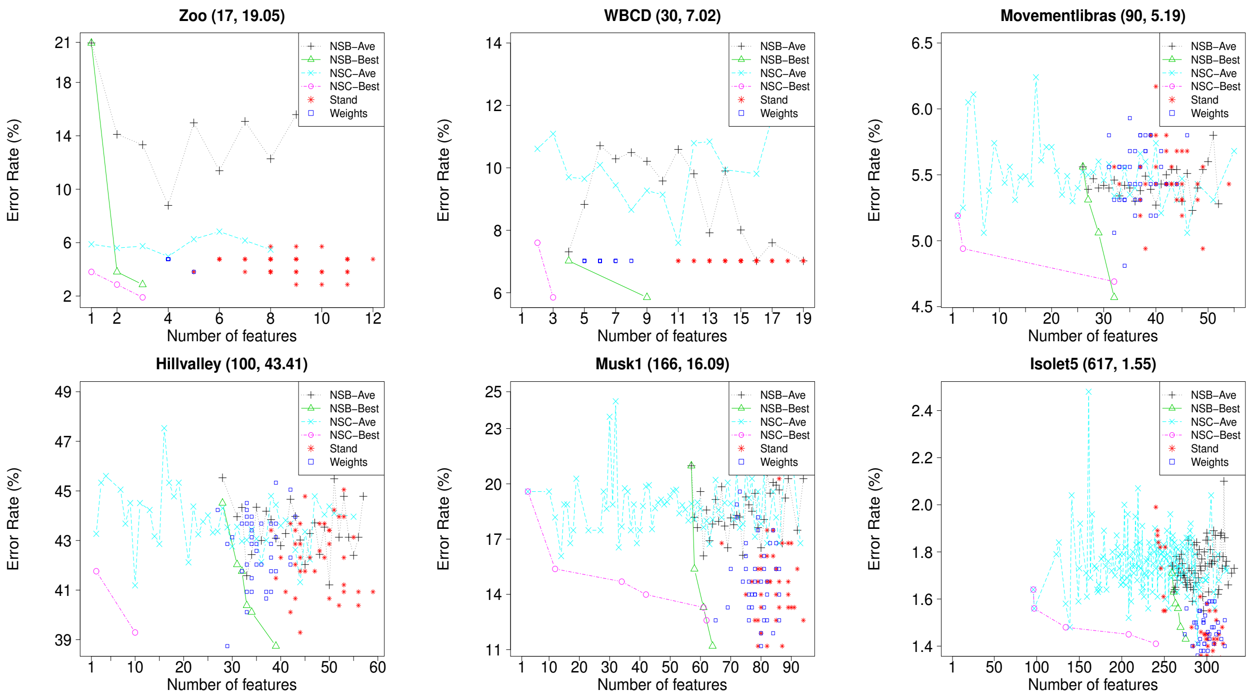

Figure 1: Results of continuous and binary NSPSO for FS and two single objective algorithms.

each run are non-dominated, some solutions in the average Pareto front may dominate others.

"NSB-Best" shows the non-dominated solutions of the binary NSPSO in the 40 independent runs. In almost all datasets, "NSB-Best" can evolve two or more feature subsets, which selected around only one third of the available features and achieved better classification performance than using all features (except for around 40% in the Isolet5 dataset).

Comparing binary NSPSO with "Stand" and "Weights", it can be seen that in some datasets, feature subsets obtained by "Stand" and "Weights" achieved a lower classification error rate than the average Pareto front ("NSB-Ave"). However, in almost all cases, the non-dominated solutions ("NSB-Best") of binary NSPSO are better than the feature subsets evolved by "Stand" and "Weights" in terms of both the number of features and the classification error rate.

5.2.2 *Results of Continuous NSPSO for FS*

According to Figure 1, in almost all cases, "NSC-Ave" contains two or more solutions, which selected a small number of features and achieved a lower classification error rate than using all features.

In all cases, "NSC-Best" evolved one or more feature subsets, which selected no more than 10% of the available features and achieved better classification performance than using all features (except for around 16% in the Isolet5 dataset). For example, in the Hillvalley dataset, by using the selected 1 or 10 features (from the available 100 features), 5NN can achieve a lower classification error rate than using all features.

Comparing continuous NSPSO with "Stand" and "Weights", it can be seen that in most cases, "NSC-Ave" successfully selected a smaller or much smaller number of features and achieved similar or better classification performance than "Stand" and "Weights". In all datasets, the solutions in "NSC-Best" are better than the feature subsets evolved by "Stand" and "Weights" in terms of both the number of features and classification performance.

The results suggest that in all datasets, both continuous and binary versions of NSPSO can evolve a set of feature subsets that selected only a small number of features but achieved better classification performance than using all features. The two versions of NSPSO as multi-objective techniques guided by two objectives can achieve better feature subsets than "Stand" and "Weights" in terms of the classification performance and the number of features.

5.3 Results of CMDPSO for FS

5.3.1 *Results of CMDPSO for FS*

According to Figure 2, in all datasets, "CMDB-Ave" contains two or more solutions, which successfully selected a small number of features and achieved better classification performance than using all features.

In most datasets, "CMDB-Best" contains one or more feature subsets, which selected around one third of the available features and achieved better classification performance than using all features.

Comparing the results of binary CMDPSO with that of "Stand" and "Weights", it can be seen that in almost all datasets, the classification error rate of "Stand" and "Weights" is similar with that of the solutions included in "CMDB-Ave". However, in all datasets, the non-dominated solutions ("CMDB-Best") of binary CMDPSO had a smaller number of features and achieved better classification performance than feature subsets evolved by "Stand" and "Weights".

5.3.2 *Results of Continuous CMDPSO for FS*

According to Figure 2, in all datasets, "CMDC-Ave" basically followed the same behaviours of "CMDB-Ave", but achieved better results than "CMDB-Ave" in most cases.

Comparing continuous CMDPSO with "Stand" and "Weights",

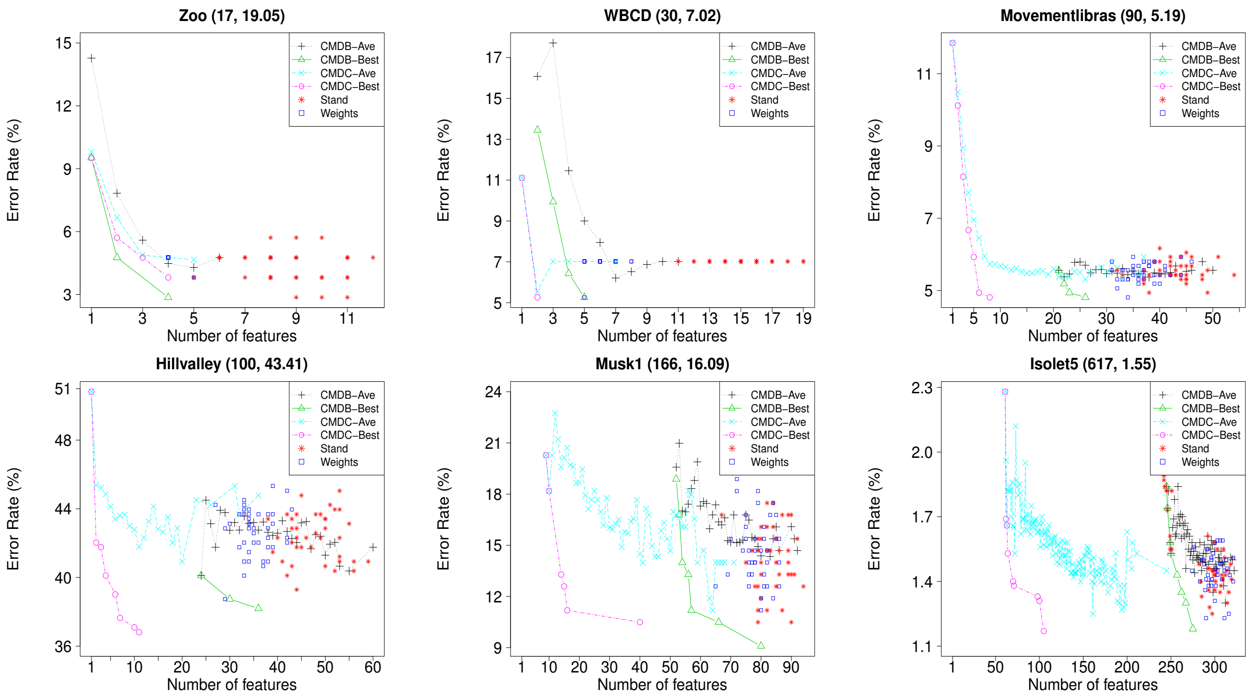

Figure 2: Results of continuous and binary CMDPSO for FS and two single objective algorithms

in almost all datasets, some of the feature subsets in "CMDC-Ave" selected a smaller number of features and achieved similar or even better classification performance than the feature subsets evolved by "Stand" and "Weights". In all datasets, the non-dominated solutions ("CMDC-Best") outperformed feature subsets evolved by "Stand" and "Weights".

The results suggest that in all datasets, both continuous and binary versions of CMDPSO can evolve a set of feature subsets with a small number of features and better classification performance than using all features. Although CMDPSO share the same parameter settings with "Stand" and "Weights", both versions of CMDPSO as multi-objective techniques can effectively explore the Pareto front and achieve better classification performance using fewer features than "Stand" and "Weights".

5.4 Comparisons Between Two Multi-objective Approaches

The results of "Stand" and "Weights" are shown in Figures 1 and 2. Therefore, "Stand" and "Weights" can be a baseline to compare the performance of the NSPSO with that of CMDPSO for FS.

Taking the WBCD dataset as an example, the average Pareto fronts of the two NSPSO algorithms could not achieve a lower classification error rate than "Stand" and "Weights". However, some solutions in the average Pareto fronts of the two CMDPSO algorithms have a smaller number of features and achieve better classification performance than "Stand" and "Weights". Comparing Figure 1 with Figure 2, in most cases, the performance of the average Pareto front of CMDPSO is better than that of NSPSO in both continuous and binary versions. In almost all cases, "CMDB-Best" and "CMDC-Best" outperformed "NSB-Best" and "NSC-Best", respectively, in terms of classification performance and the number of features.

Generally, CMDPSO can achieve better performance than NSPSO for FS. The reason may be that the way of setting the new positions for particles in NSPSO makes the swarm quickly loss diversity during the evolutionary training process and could not effectively search the solution space. We will investigate this in the future.

5.5 Comparisons Between Continuous and Binary PSO for FS

According to Figure 1, in almost all datasets, the average Pareto front, "NSC-Ave" outperformed "NSB-Ave". In the datasets with a relatively large number of features, continuous NSPSO can obtain a much smaller feature subset with similar or even better classification performance than binary NSPSO. The results of "NSC-Best" are also better than those of "NSB-Best" in most cases. According to Figure 2, continuous CMDPSO outperformed binary CMDPSO in almost all cases.

Generally, the two continuous versions of multi-objective PSO ("NSC","CMDC") achieved better performance than the two binary versions of multi-objective PSO ("NSB", "CMDB"). This might be because BPSO does not present all of the important characteristics of the PSO algorithm.

5.6 Comparisons with a Conventional FS Method

Taking the Zoo and WBCD datasets as examples, we check how well the four proposed algorithms perform compared with linear forward selection (LFS) [8]. LFS selected 6 features in Zoo with a classification error rate of 4.76% and 9 features in WBCD with a classification error rate of 11.70%. Comparing the results in Figures 1 and 2, "Stand", "Weights" can evolve two or more feature subsets that outperformed the subset selected by LFS. Results of NSPSO ("NSB-Ave", "NSB-Best", "NSC-Ave", and "NSC-Best") and CMDPSO ("CMDB-Ave", "CMDB-Best", "CMDC-Ave", and "CMDC-

Best") are much better than those of LFS. Other datasets have similar results, which show that PSO can effectively select a good feature subset with a small number of features and achieve high classification performance.

6. CONCLUSIONS

The goal of this paper was to develop a PSO based multi-objective FS approach to selecting a set of non-dominated feature subsets and achieving high classification performance. This goal was successfully achieved by developing two new multi-objective FS algorithms. The first algorithm employed multi-objective PSO using the idea of NSGAII to search for the Pareto front in FS problems. The second algorithm employed multi-objective PSO, which adopted the idea of crowding, mutation and dominance, as the seach technique for FS. The two proposed algorithms were examined and compared with two single objective FS algorithms and LFS on nine benchmark datasets of varying difficulty. The results suggest that two proposed algorithms can obtain a set of non-dominated solutions, which can successfully select a small number of features and achieve higher classification performance than using all features. In terms of both the number of features and classification performance, the two proposed algorithms outperformed the two single objective methods and LFS, and the second algorithm outperformed the first algorithm. In both two algorithms, using continuous PSO achieved better performance than using binary PSO, especially in the number of features selected.

The two propsed algorithms successfully reduced the number of features needed and achieved higher classification performance, but it is unknown whether the classification performance can be further increased without increasing the number of features, or whether the number of features can be further reduced with similar classification performance. In the future, we will further investigate the multi-objective PSO based FS approach to better exploring the Pareto front of non-dominated solutions in FS problems. We will intend to further investigate the why BPSO usually evolve a larger subset of features than continuous PSO, then develop a new BPSO algorithm to overcome the limitation of BPSO and address this problem.

7. REFERENCES

[1] H. Almuallim and T. G. Dietterich. Learning boolean concepts in the presence of many irrelevant features. *Artificial Intelligence*, 69:279–305, 1994.

[2] G. Azevedo, G. Cavalcanti, and E. Filho. An approach to feature selection for keystroke dynamics systems based on pso and feature weighting. In *IEEE Congress on Evolutionary Computation (CEC'07)*, pages 3577–3584, 2007.

[3] B. Chakraborty. Genetic algorithm with fuzzy fitness function for feature selection. In *IEEE International Symposium on Industrial Electronics (ISIE'02)*, volume 1, pages 315– 319, 2002.

[4] B. Chakraborty. Feature subset selection by particle swarm optimization with fuzzy fitness function. In *3rd International Conference on Intelligent System and Knowledge Engineering (ISKE'08)*, volume 1, pages 1038–1042, 2008.

[5] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(4):131–156, 1997.

[6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182 –197, 2002.

[7] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[8] M. Gutlein, E. Frank, M. Hall, and A. Karwath. Large-scale attribute selection using wrappers. In *IEEE Symposium on Computational Intelligence and Data Mining (CIDM '09)*, pages 332–339, 2009.

[9] C. L. Huang and J. F. Dun. A distributed pso-svm hybrid system with feature selection and parameter optimization. *Application on Soft Computing*, 8:1381–1391, 2008.

[10] J. Kennedy and R. Eberhart. Particle swarm optimization. In *IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, 1995.

[11] J. Kennedy and R. Eberhart. A discrete binary version of the particle swarm algorithm. In *IEEE International Conference on Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation.*, volume 5, pages 4104–4108, 1997.

[12] J. Kennedy and W. Spears. Matching algorithms to problems: an experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator. In *IEEE Congress on Evolutionary Computation (CEC'98)*, pages 78–83, 1998.

[13] K. Kira and L. A. Rendell. A practical approach to feature selection. *Assorted Conferences and Workshops*, pages 249–256, 1992.

[14] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97:273–324, 1997.

[15] X. Li. A non-dominated sorting particle swarm optimizer for multiobjective optimization. In *GECCO*, pages 37–48, 2003.

[16] Y. Liu, G. Wang, H. Chen, and H. Dong. An improved particle swarm optimization for feature selection. *Journal of Bionic Engineering*, 8(2):191–200, 2011.

[17] T. Marill and D. Green. On the effectiveness of receptors in recognition systems. *IEEE Transactions on Information Theory*, 9(1):11–17, 1963.

[18] H. Ming. A rough set based hybrid method to feature selection. In *International Symposium on Knowledge Acquisition and Modeling (KAM '08)*, pages 585–588, 2008.

[19] A. Mohemmed, M. Zhang, and M. Johnston. Particle swarm optimization based adaboost for face detection. In *IEEE Congress on Evolutionary Computation (CEC'09)*, pages 2494–2501, 2009.

[20] K. Neshatian and M. Zhang. Dimensionality reduction in face detection: A genetic programming approach. In *24th International Conference Image and Vision Computing New Zealand (IVCNZ'09)*, pages 391–396, 2009.

[21] Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *IEEE International Conference on Evolutionary Computation (CEC'98)*, pages 69–73, 1998.

[22] M. R. Sierra and C. A. C. Coello. Improving pso-based multi-objective optimization using crowding, mutation and epsilon-dominance. In *EMO*, pages 505–519, 2005.

[23] A. Unler and A. Murat. A discrete particle swarm optimization method for feature selection in binary classification problems. *European Journal of Operational Research*, 206(3):528–539, 2010.

[24] F. Van Den Bergh. *An analysis of particle swarm optimizers*. PhD thesis, Pretoria, South Africa, 2002.

[25] A. Whitney. A direct method of nonparametric measurement selection. *IEEE Transactions on Computers*, C-20(9):1100–1103, 1971.

[26] C. S. Yang, L. Y. Chuang, C. H. Ke, and C. H. Yang. Boolean binary particle swarm optimization for feature selection. In *IEEE Congress on Evolutionary Computation (CEC'08)*, pages 2093–2098, 2008.

[27] H. Yuan, S. S. Tseng, and W. Gangshan. A two-phase feature selection method using both filter and wrapper. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC'99)*, volume 2, pages 132–136, 1999.