Filter based Backward Elimination in Wrapper based PSO for Feature Selection in Classification

Hoai Bach Nguyen[1], Bing Xue[1], Ivy Liu[2] and Mengjie Zhang[1]
[1] School of Engineering and Computer Science
[2] School of Mathematics, Statistics and Operations Research
Victoria University of Wellington, PO Box 600, Wellington 6140, New Zealand
Email: {nguyenhoai2, Bing.Xue, Mengjie.Zhang} @ecs.vuw.ac.nz, Ivy.Liu@msor.vuw.ac.nz

***Abstract*—The advances in data collection increase the dimensionality of the data (i.e. the total number of features) in many fields, which arises a challenge to many existing feature selection approaches. This paper develops a new feature selection approach based on particle swarm optimisation (PSO) and a local search that mimics the typical backward elimination feature selection method. The proposed algorithm uses a wrapper based fitness function, i.e. the classification error rate. The local search is performed only on the global best and uses a filter based measure, which aims to take the advantages of both filter and wrapper approaches. The proposed approach is tested and compared with three recent PSO based feature selection algorithms and two typical traditional feature selection methods. Experiments on eight benchmark datasets show that the proposed algorithm can be successfully used to select a significantly smaller number of features and simultaneously improve the classification performance over using all features. The proposed approach outperforms the three PSO based algorithms and the two traditional methods.**

I. Introduction

Feature selection (also called dimension reduction), which is a process of finding a small subset of features from a given large feature set, has been of interest for many decades [1, 2]. In classification, feature selection is to find a subset of relevant features so that the dimensionality of the data can be reduced and the learning/classification process can be sped up while the overall classification performance can be maintained or improved [2].

A feature selection approach can be viewed as the combination of a search technique for finding an optimal feature subset (s), along with an evaluation criterion which evaluates the goodness of different feature subsets. Based on the evaluation measure, there are two categories of feature selection methods [2, 3]: wrapper approaches and filter approaches. Wrapper approaches employ a classification/learning algorithm to evaluate the goodness of the selected features, where a classifier is trained for each feature subset. Wrapper approaches are usually computationally intensive, but provide promising feature subsets for the particular classification algorithm. The evaluations in filter approaches are independent of any classification algorithm. They use information measure, consistency measure or other measures to evaluate feature subsets [2]. Filter approaches are often computationally cheaper and more general to different classification algorithms than wrapper approaches. Wrapper and filter methods can complement each other in that filter methods can search through the feature space efficiently while the wrappers provides good accuracy.

Feature selection has a large search space, which is 2^n for a dataset including n features. An effective and efficient search technique is necessary for developing a promising feature selection approach. Particle swarm optimisation (PSO) is an evolutionary computation (EC) method [4, 5], which is a powerful global search method and computationally cheaper than other EC algorithms [6]. There are two main types of PSO, which are continuous PSO [4, 5] and binary PSO [7]. Both of them have been successfully applied to feature selection [8, 9], but the research in [10] shows that continuous PSO can achieve better performance than binary PSO. Therefore, this work focuses mainly on investigating and improving the performance of continuous PSO for feature selection in classification.

In continuous PSO for feature selection, the position value shows the probability of the corresponding feature being selected. However, when evaluating the goodness of a particle, a feature is either selected or not selected and the probability information is not fully used. Meanwhile, feature selection is a complicated task with many local optima, which easily leads to the issue of being stuck in local optima. Combining local search with EC algorithms has shown to be effective in many fields, but this has seldom been investigated in PSO for feature selection.

A. Goals

The overall goal of this paper is to develop a new PSO based feature selection approach to reducing the dimensionality of data while maintaining or increasing the classification accuracy over using all features. To achieve this goal, a local search mimicking a typical backward elimination method is performed on the global best of the swarm in each iteration to improve its search ability. This "backward elimination" (or local search) is designed to be fast and effective by 1) using a mutual information based filter measure 2) further utilising the information given by the position value, and 3) performing the search on small sub-groups of the features selected by the global best, rather than on the whole set of selected features. The proposed approach is examined and compared with three recently developed PSO based algorithms and two traditional feature selection algorithms on eight datasets of

varying difficulty. Specifically, we will investigate:

- whether the new approach can successfully select a smaller number of features and achieve similar or higher classification accuracy than using all features,
- whether the new approach can outperform the three PSO based algorithms in terms of the classification performance and the number of features, and
- whether the new approach can achieve better performance than the two traditional algorithms.

II. Background

A. Particle Swarm Optimisation (PSO)

Particle swarm optimisation (PSO) [4, 5] is based on the idea of swarm intelligence, inspired by social behaviours, such as fish schooling and birds flocking. In PSO, a candidate solution is represented by a particle. A population of particles move ("fly") together in the search space to find the optimal solutions. During the movement, each particle (i) has a position shown by $x_i = (x_{i1}, x_{i2}, ..., x_{iD})$ and a velocity shown by $v_i = (v_{i1}, v_{i2}, ..., v_{iD})$, where D is the dimensionality of the search space. During the search process, each particle can remember its best position visited so far called personal best (denoted by *pbest*), and the best previous position visited so far by the whole swarm called global best (denoted by *gbest*). Based on *pbest* and *gbest*, PSO iteratively updates x_i and v_i of each particle to search for the optimal solutions according to Eqs. 1 and 2.

$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * r_{i1} * (p_{id} - x_{id}^{t}) + c_2 * r_{i2} * (p_{gd} - x_{id}^{t}) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (2)$$

where v_{id}^{t+1} shows the velocity of particle i in the dth dimension at the $(t+1)$th iteration. x_{id}^{t+1} shows the position value of particle i. w is the inertia weight reflecting the influence of the previous velocity. c_1 and c_2 are acceleration constants. r_{i1}, r_{i2} and $rand()$ are random values, which are uniformly distributed in [0, 1]. p_{id} and p_{gd} shows the values of *pbest* and *gbest* in the dth dimension.

B. Entropy and Mutual Information

Entropy and mutual information are two of the fundamental concepts in information theory [11], which provide a way to measure the information of random variables.

Let X be a random variable with discrete values, its uncertainty can be measured by entropy $H(X)$ defined as:

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log_2 p(x) \quad (3)$$

where $p(x) = Pr(X = x)$ is the probability density function of X. It is noted that entropy does not depend on actual values, but just the probability distribution of the random variable.

For two discrete random variables X and Y with their probability density function $p(x, y)$, the joint entropy $H(X, Y)$ is defined as:

$$H(X, Y) = -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log_2 p(x, y) \quad (4)$$

In the situation, where a variable is known while the other variables are unknown, the uncertainty can be evaluated by conditional entropy. Let Y be the given/known variable, the conditional entropy (denoted by $H(X|Y)$) of X with respect to Y is defined as:

$$H(X|Y) = -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log_2 p(x|y) \quad (5)$$

where $p(x|y)$ is the posterior probabilities of X given Y. From this definition, if X completely depends on Y, then $H(X|Y)$ is zero, which means that no more other information is required to describe X when Y is known. Otherwise, $H(X|Y) = H(X)$ denotes that knowing Y will provides no information about X.

Mutual information measures the information shared between two random variables. Given a variable X, the information gained from X about another variable Y is mutual information denoted by $I(X; Y)$.

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log_2 \frac{p(x, y)}{p(x)p(y)} \end{aligned} \quad (6)$$

According to Eq. 6, $I(X; Y)$ will be large if two variables X and Y are closely related. $I(X; Y) = 0$ if X and Y are totally unrelated.

Mutual information has been used as a filter measure in feature selection [12, 13] with which the algorithms can successfully reduce the dimensionality of the data and maintain or increase the classification accuracy over using all features.

C. Related Work on Feature Selection

1) Traditional Feature Selection Methods.: Feature ranking is a relax version of feature selection, where a score is assigned to each of the available features individually according to a pre-defined criterion [2]. Feature selection can be achieved by choosing the features with the best scores. However, it ignores the interaction between features. If the features in the dataset are highly correlated, this approach may not achieve good results because some of the selected features may be redundant [2].

Sequential feature selection is a basic traditional feature selection approach, including sequential forward selection (SFS) [14] and sequential backward selection (SBS) [15]. SFS (or SBS) selects features by incrementally adding (or removing) features from the empty (or full) feature set. However, they may suffer from the problem of "nesting", i.e. once a feature has been added to (or removed from) the feature set, it cannot be removed (or added) later. To solve this problem, "plus-l-take away-r" [16] was proposed to perform l times forward selection and then r times backward selection. However, the optimal values of (l, r) is hard to determine. Floating selection methods [1], namely sequential backward floating selection (SBFS) and sequential forward floating selection (SFFS), were proposed to automatically determine (l, r). Later, a linear forward selection (LFS) method was also proposed [17] based

on the best-first algorithm and SFFS. The number of features considered in each step is restricted, which can reduce the computational cost while maintaining the comparable classification accuracy.

2) EC Approaches for Feature Selection.: EC algorithms have been applied to feature selection problems, such as PSO, genetic algorithms (GAs) [18], genetic programming (GP) [19], and differential evolution (DE) [20].

Based on GP, Ahmed et al. [19] proposed a filter feature selection algorithm for biomarker discovery in LC-MS data. The results show that the proposed algorithm can significantly reduce the number of features and increase the classification accuracy. Al-Ani et al. [20] also proposed a DE based feature selection method, where features are distributed to a set of wheels and DE is employed to select features from each wheel. This algorithm can significantly reduce the number of features and improve the classification performance. Zhu et al. [18] proposed a feature selection method incorporating GA with local search (i.e. forms a memetic algorithm). Meanwhile, this algorithm combines filter ranking measure into a wrapper framework to take the advantages of both filter and wrapper approaches. The results show that this algorithm outperforms GA alone and other algorithms.

A PSO based filter-wrapper feature selection algorithm was proposed in [8], where a filter measure is used to encode the position of each particle and the classification performance is used in the fitness function. The experiments show that the proposed method slightly outperforms a binary PSO based filter method. However, it has not been compared with any wrapper algorithm, which can usually obtain higher classification performance than a filter algorithm. Fdhila et al. [21] applied multi-swarm PSO to solve feature selection problems. However, the computational cost of the proposed algorithm is high because it involves parallel evolutionary processes and multiple sub-swarms with a relative large number of particles.

Many existing studies improved PSO for feature selection algorithms by developing new $gbest$ updating mechanisms. Chuang et al. [9] proposed to reset $gbest$ elements to zero if it maintains the same values for several iterations. However, the performance of this algorithm is not compared with the standard PSO based feature selection algorithm. Chuang et al. [22] applied the so-called catfish effect to $gbest$ in PSO for feature selection, which introduces new particles into the swarm by re-initialising the worst particles when $gbest$ has not been improved for a number of iterations. The authors claimed that the introduced catfish particles could help PSO avoid premature convergence and lead to better results than sequential GA, SFS, SFFS and other methods. Recently, Xue et al. [23] developed a new $pbest$ and $gbest$ updating mechanism in PSO for feature selection by considering the number of features when updating them. The proposed algorithm can increase the classification accuracy and reduce both the number of features and the computational time.

Lin et al. [24] proposed a wrapper feature selection algorithm using PSO and support vector machine (SVM). This algorithm aims to optimise the parameters in SVM and search for the best feature subset simultaneously. Cervante et al. [13] developed a PSO based filter approach, where mutual information and entropy were used to form the fitness function. Experiments show that PSO using mutual information can effective address feature selection problems. Our recent work on PSO for feature selection can be seen in [25, 26, 27, 28]

Based on PSO and a statistical clustering method [29, 30] that groups features to different clusters and similar features to the same cluster, Lane et al. [31] proposed a feature selection algorithm, which uses PSO to select one feature from each cluster. The results show that by selecting a representative feature from each cluster, the proposed algorithm can significantly reduce the number of features and increase the classification performance. This shows the the statistical clustering information (i.e. feature clusters) can provide useful information in feature selection. Therefore, this work will also utilise such information to further develop the new approach (details can be seen in Section III-D).

Algorithm 1: Pseudo-code of PSOBE.

```
Input  : A Training set and a Test set, parameters in PSO;
Output : gbest (selected feature subset);
         Training and test classification accuracies.
1  begin
2      randomly initialise the position and velocity of each particle;
3      while Maximumiterations is not reached do
4          evaluate fitness of each particle ;   /* i.e. the
           classification error rate (Eq. 7) */
5          for i=1 to PopulationSize do
6              update the pbest of particle i;
7          end
8          update the gbest ;                    /* fully connected
           topology */
9          for i=1 to PopulationSize do
10             for d=1 to Dimensionality do
11                 update v_id according to Eq. 1;
12                 update x_id according to Eq. 2;
13             end
14         end
15         perform "backward elimination" on gbest ;
           /* According to Fig. 1 */
16     end
17     calculate the classification accuracy of the selected feature
       subset on the test set;
18     return the position of gbest (the selected feature subset), the
       training and test classification accuracies;
19 end
```

III. Proposed Approach

A. Overall Algorithm

Algorithm 1 shows the overall structure of the proposed algorithm named PSOBE. PSOBE follows the basic steps of a standard PSO algorithm except the "backward elimination" procedure on $gbest$.

In PSOBE, each particle is encoded as a vector of real numbers, $x_i = (x_{i1}, x_{i2}, .., x_{id}, .., x_{iD})$, where D is the dimensionality of the dataset. $0 \leq x_{id} \leq 1$ shows the probability of the dth feature being selected. A threshold θ is used to determine whether this feature is selected. If $\theta \leq x_{id}$, the dth feature is selected. Otherwise, the dth feature is not selected. The classification error rate (i.e. a wrapper measure) is used as the fitness function in PSOBE (shown by Eq. 7), which

aims to minimise the classification error rate (or maximise the classification accuracy) of the selected features.

$$ErrorRate = \frac{FP + FN}{TP + TN + FP + FN} \quad (7)$$

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

A "backward elimination" procedure is developed to mimic the typical backward elimination (or backward selection) algorithm [15]. This "backward elimination" is used as a local search based on $gbest$ to find a better solution, to improve the search ability and to avoid the algorithm becoming stagnation in local optima. Two of the key components to develop this "backward elimination" is the *measure* to evaluate the goodness of each feature and *how to perform* the "backward elimination". Meanwhile, as discussed before, the position values show the probability of the corresponding features being selected, but such information is not fully reflected by the evaluation because the feature can only be either selected or not selected when evaluating the classification error rate. Therefore, this "backward elimination" also aims to further utilise the information from the position values.

The "backward elimination" measure, the consideration of the position value and how to perform the "backward elimination" are described in detail in Sections III(B), (C) and (D).

B. "Backward elimination" measure

Since wrapper based feature selection approaches suffer from the problem of high computational cost, this "backward elimination" should not be computationally heavy, but can still capture the usefulness of features. Therefore, a fast filter measure, which is based on mutual information, is used here. More importantly, by using this filter measure, PSOBE is also expected to take the advantages of both filter methods and wrapper methods.

When using mutual information in feature selection [12, 13], features and the class label are treated as variables. Mutual information evaluates the shared information between a feature and the class label or between two features. Let F be a set of selected features, c be the class label, f_i $(1 \leq i < |F|)$ is any feature in F. The relevance between F and c can be measured by Eq. 8 [12, 13], which suggests that the contribution of f_i to the relevance can be shown by Eq. 9. The detailed calculation of $I(f_i; c)$ can be referred to Eq. 6.

$$Rel(F) = \sum_{f_i \in F} I(f_i; c) \quad (8)$$

$$\triangle Rel(f_i) = I(f_i; c) \quad (9)$$

The redundancy contained in F can be measured by Eq. 10 [12, 13], which suggests that the redundancy brought by f_i can be shown by Eq. 11.

$$Red(F) = \sum_{f_i, f_j \in F} I(f_i, f_j) \quad (10)$$

$$\triangle Red(f_i) = \sum_{f_j \in F; f_i \neq f_j} I(f_i, f_j) \quad (11)$$

In [12, 13], the goodness of the feature subset (F) is evaluated by $(Rel(F) - Red(F))$, i.e. the relevance of F minus the redundancy of F. Therefore, Eq. 12 is proposed here to measure the goodness of f_i in F, i.e. the relevance of f_i minus the redundancy brought by f_i. $|F|$ means the size of the feature subset F.

$$Fit(f_i) = \triangle Rel(f_i) - \frac{1}{|F| - 1} \triangle Red(f_i) \quad (12)$$

Eq. 12 is a maximisation function, the larger the better. For backward elimination, the feature that has the largest redundancy and the smallest relevancy, i.e. the smallest value of $Fit(f_i)$, should be removed. However, when $Fit(f_i) \geq 0$, the relevance of f_i is larger or at least the same as its redundancy, f_i should not be removed. Therefore, feature f_i is removed only when $Fit(f_i) < 0$ and $Fit(f_i)$ is the smallest value among all the possible features in F, i.e. $Fit(f_i) \leq Fit(f_j)$, where $j = 1, 2, ...|F|$ and $j \neq i$.

C. Consideration of the position value

The position value is considered in the "backward elimination" step to further utilise the information it carries (i.e. probability information). This is achieved by adding the position value of f_i, (i.e. x_i) to Eq. 12, which forms the new equation (Eq. 13). $\theta < x_i \leq 1.0$ since the corresponding feature is selected.

By adding x_i, Eq. 13 ensures that if two features has the same smallest value from Eq. 12, the one with a smaller position value (i.e. smaller probability) will be removed from F. Therefore, f_i is removed only when $Fit'(f_i) < 0$ and $Fit'(f_i) \leq Fit'(f_j)$, where $j = 1, 2, ...|F|$ and $j \neq i$.

$$Fit'(f_i) = \frac{1}{x_i} * (\triangle Rel(f_i) - \frac{1}{|F| - 1} \triangle Red(f_i)) \quad (13)$$

D. How to perform the "backward elimination"

When using the "backward elimination", the number of features being removed is an important factor, but it is difficult to pre-determine. It should be a dynamic value determined according to the features selected by $gbest$.

To solve this problem, a statistical clustering method [29, 30] is used to first group features to different clusters, where similar features are in the same cluster. The "backward elimination" is then performed on each cluster rather than on all the features selected by $gbest$. The intuition is that if a large number of features is selected from one cluster, removing one feature will not reduce the classification performance too much because features in the same cluster are similar features. Specifically, for each cluster, the features selected by $gbest$ are first collected to form the feature subset (i.e. F used above) on which the "backward elimination" may perform. Then the "backward elimination" is performed on F to reduce one feature only when there are more than $\sqrt{m} + 1$ features are selected by $gbest$ (i.e. $|F| > \sqrt{m}+1$), where m is the total

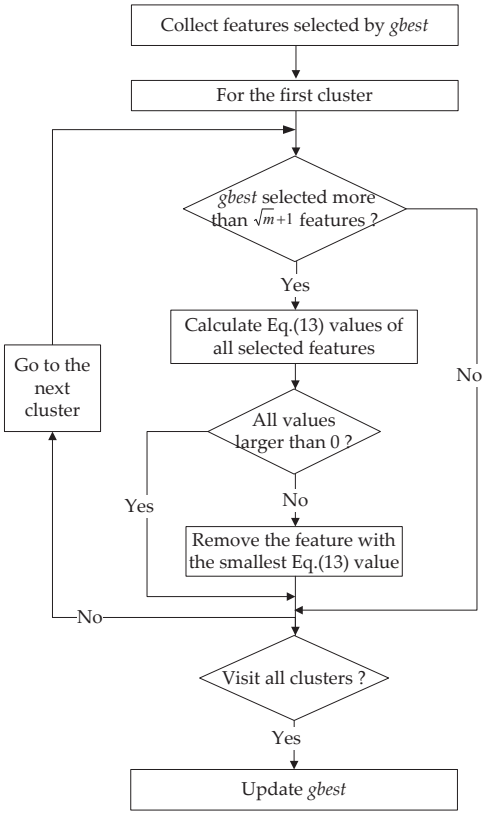

Fig. 1. The Flowchart of "Backward Elimination"

number of features included in this cluster. This can ensure that when a small proportion of features are selected by $gbest$, they will remain to preserve the useful information. On the other hand, if a large proportion of features are selected, the removal of one feature will not reduce the classification performance. Meanwhile, performing Eq. 13 on small feature clusters will also reduce the computation cost over performing it on a large feature set.

The flowchart of the "backward elimination" process is shown by Fig. 1. Note that although this "backward elimination" has many steps, their calculation is substantially faster than a wrapper evaluation involving training a classifier. In fact, reducing features from $gbest$ will guide the algorithm to search for small feature subsets, which needs shorter or much shorter time depending on the number of features reduced. Therefore, we hypothesise that using this "backward elimination" as local search on $gbest$ will reduce the number of features, improve the classification performance and also reduce the computational cost.

IV. Experiments Configuration

A. Benchmark Techniques

To examine the performance of PSOBE, two traditional wrapper feature selection methods and three PSO based algorithms (PSOFS, PSO2S [32] and PSO42 [23]) are used as benchmark techniques in the experiments. The results in [31] are not listed here due to the page limit, but comparing the results on the datasets used in both this paper and [31], PSOBE achieved better performance than the method in [31].

TABLE I
Datasets

Dataset	# Features	# **Clusters**	# Classes	# Instances
Wine	13	6	3	178
Vehicle	18	6	4	846
Ionosphere	34	11	2	351
Sonar	60	12	2	208
Musk Version 1 (Musk1)	166	14	2	476
Arrhythmia	279	15	16	452
Madelon	500	11	2	4400
Multiple Features	649	15	10	2000

The two traditional algorithms are LFS [17] and greedy stepwise backward selection (GSBS), which were derived from two typical greedy search based feature selection methods, i.e. SFS and SBS, respectively. LFS 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 better results. More details can be seen in the literature [17]. GSBS is a backward selection that starts with all available features and stops when the deletion of any remaining feature results in a decrease in the classification accuracy. PSOFS, PSO2S and PSO42 are three PSO based algorithms. PSOFS uses the standard continuous PSO for feature selection. PSO2S is a two-stage based algorithm and the details can be seen from [32]. Note that PSO2S in [32] is based on binary PSO and PSO2S here is based on continuous PSO to be consistent with PSOBE and others. PSO42 is based on an initialisation strategy and a $pbest$ and $gbest$ updating mechanism. The details of PSO42 can be seen from [23]. PSOFS, PSO2S and PSO42 use the classification error rate as the fitness function, which is the same as PSOBE.

B. Datasets and Parameter Settings

Eight datasets (Table I) chosen from the UCI machine learning repository [33] are used in the experiments. The eight datasets are chosen to have different numbers of features, classes and instances to be used as representatives of problems that the proposed algorithm can address. For each dataset, the instances are randomly divided into two sets: 70% as the training set and 30% as the test set. Since the statistical clustering method and mutual information only work on discrete data, the training data is first discretised using Weka [34]. The statistical clustering method only needs to perform once for each dataset and the number of clusters is shown in Table I.

In the experiments, K-nearest neighbour (KNN), where K=5, was used as the classification/learning algorithm. During the training process, KNN with 10-fold cross-validation is employed to evaluate the classification error rate of the selected feature subset on the training set, and then 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 [3].

Weka [34] is used to run the experiments of LFS and GSBS and all the settings are kept to the defaults. The parameters of PSOFS, PSO2S, PSO42 and PSOBE are set as follows: $w = 0.7298$, $c_1 = c_2 = 1.49618$, $v_{max} = 6.0$ [5]. the population size is 30, and the maximum iteration is 100. The

TABLE II
COMPARISONS WITH PSO BASED ALGORITHMS

Dataset	Method	Ave-Size	Best	Ave ± Std	T	T1	T2	T3
	All	13	76.54					
	PSOFS	7.93	98.77	95.6 ± 1.7953	+			
Wine	PSO2S	7.93	98.77	95.6 ± 1.7953	+			
	PSO42	6.73	98.77	94.86 ± 1.8628	+			
	PSOBE	3.67	100	97.82 ± 2.0826	+	+	+	+
	All	18	83.86					
	PSOFS	9.5	87.01	85.03 ± 0.8899	+			
Vehicle	PSO2S	8.63	87.01	84.91 ± 0.8762	+			
	PSO42	10.33	87.01	85.44 ± 0.8372	+			
	PSOBE	4.67	85.63	83.86 ± 0.8847	=	-	-	-
	All	34	83.81					
	PSOFS	12.47	93.33	88.41 ± 2.3079	+			
Ionosphere	PSO2S	11.83	91.43	88.16 ± 1.9931	+			
	PSO42	3.13	91.43	86.89 ± 1.6444	+			
	PSOBE	3.67	93.33	88.54 ± 2.3776	+	=	=	+
	All	60	76.19					
	PSOFS	26.1	84.13	77.3 ± 3.5765	=			
Sonar	PSO2S	24	84.13	77.46 ± 3.5582	+			
	PSO42	11.23	84.13	77.94 ± 3.2104	+			
	PSOBE	11.13	87.3	78.25 ± 4.2621	+	=	=	=
	All	166	83.92					
	PSOFS	85.93	88.81	84.61 ± 2.0568	=			
Musk1	PSO2S	85	88.81	84.52 ± 1.9819	=			
	PSO42	77.3	89.51	84.87 ± 2.7042	=			
	PSOBE	36.7	90.91	84.76 ± 2.8856	=	=	=	=
	All	279	94.46					
	PSOFS	118.73	95.14	94.56 ± 0.3517	=			
Arrhythmia	PSO2S	126.43	95.14	94.44 ± 0.263	=			
	PSO42	69.77	95.59	94.77 ± 0.4495	+			
	PSOBE	18.27	95.81	94.86 ± 0.4962	+	+	+	=
	All	500	70.90					
	PSOFS	259.07	78.97	76.35 ± 1.0909	+			
Madelon	PSO2S	257.33	79.36	76.39 ± 1.1957	+			
	PSO42	206.57	84.23	78.81 ± 3.1171	+			
	PSOBE	47.83	88.33	84.73 ± 2.3752	+	+	+	+
	All	649	98.63					
	PSOFS	297.07	99.20	99.00 ± 0.0934	+			
MultipleF	PSO2S	311.23	99.13	98.99 ± 0.0582	+			
	PSO42	314.5	99.20	99.00 ± 0.0935	+			
	PSOBE	51.1	99.10	98.86 ± 0.1356	+	-	-	-

fully connected topology is used. All the four PSO based algorithms share the same representation. The threshold θ is set as 0.6, which is to keep consistent with the value in [32, 23].

LFS and GSBS are deterministic methods, which produce a unique solution. The four PSO based algorithms are stochastic methods and each of them has been performed for 30 independent runs on each dataset. A statistical significance test, Wilson test, is performed between the classification accuracies achieved by different algorithms. The significance level of the Wilson test was selected as 0.05.

V. RESULTS AND DISCUSSIONS

This section firstly discusses the results of PSOBE and the other three PSO based feature selection algorithms (Table II), then compares the performance of PSOBE and the two traditional methods, i.e. LFS and GSBS (Table III).

In Table II, "All" means all the available features, "Size" means the average number of features selected by a PSO based algorithm. "Best", "Ave" and "Std" show the best, average and standard deviation of the testing accuracies achieved by PSOFS, PSO2S, PSO42 or PSOBE in the 30 independent runs. "T" shows the results of the significance tests between the testing accuracy of "All" and that of the three PSO based algorithms. "T1", "T2", and "T3" represent the results of the significance tests between the testing accuracy of PSOBE against that of PSOFS, PSO2S and PSO42, respectively. "+" ("-") means the accuracy of PSOBE is significantly higher (lower) than that of "All", PSOFS, PSO2S and PSO42. "=" means there is no significant difference.

A. Results of PSOBE

According to Table II, it can be seen that on the eight datasets, PSOBE achieved similar classification accuracy to all features on two datasets, and significantly higher accuracy than all features on six datasets. In all cases, PSOBE only selected less than 25% of the total number of features. For the datasets including a relatively large number of features, i.e. Arrhythmia, Madelon, and MultipleF, PSOBE removed more than 90% of the features and at the same time significantly increased the classification accuracy.

The results show that PSOBE is an effective feature selection algorithm, which can reduce the dimensionality of the data and at the same time improve the classification accuracy.

B. Comparisons with PSO based Algorithms

According to Table II, the "Size" and "T" values show that all the four PSO based algorithms can be successfully used to address feature selection problems, which selected a smaller number of features and maintain or even increase the classification accuracy.

Comparing PSOBE with PSOFS, Table II shows that in all cases, the number of features selected by PSOBE is significantly smaller than that of PSOFS. On six of the eight datasets, PSOBE achieved significantly higher or similar classification accuracy to PSOFS. Although on the Vehicle and MultipleF (Multiple Features) datasets, the classification accuracy of PSOBE is slightly lower than that of PSOFS, but the number of features is significantly smaller. Especially on the MultipleF dataset, PSOFS selected on average 297.07 features and PSOBE further reduced the number of features to only 51.1, but the average classification accuracy was only decreased for less than 0.14% (only 0.1% for the best accuracy).

Comparing PSOBE with PSO2S and PSO42, it can be observed that in almost all cases, PSOBE selected a smaller number of features than both PSO2S and PSO42. The only exception is that PSOBE selected a slightly larger number of features (only 0.5 on average) than PSO42 on the Ionosphere datasets (both are are between 3 and 4), but PSOBE achieved significantly better classification performance than PSO42. Meanwhile, the overall classification performance of PSOBE is similar or better than PSO2S and PSO42 in most cases.

The results show that PSOBE with the "backward elimination" step on the $gbest$ can help the algorithm better explore the solution space than PSOFS, PSO2S and PSO42. It results in a better features subset, which includes a smaller number of features and similar or better classification performance than all the other three algorithms.

TABLE III
RESULTS OF LFS ANG GSBS

Wine				Vehicle			
Method	Size	Acc	T4	Method	Size	Acc	T4
LFS	7	74.07	-	LFS	9	83.07	-
GSBS	8	85.19	-	GSBS	16	75.79	-
PSOBE	3.67	97.82		PSOBE	4.67	83.86	
Ionosphere				Sonar			
Method	Size	Acc	T	Method	Size	Acc	T
LFS	4	86.67	-	LFS	3	77.78	=
GSBS	30	78.1	-	GSBS	48	68.25	-
PSOBE	3.67	88.54		PSOBE	11.13	78.25	
Musk1				Arrhythmia			
Method	Size	Acc	T	Method	Size	Acc	T
LFS	10	85.31	=	LFS	11	94.46	-
GSBS	122	76.22	-	GSBS	130	93.55	-
PSOBE	36.7	84.76		PSOBE	18.27	94.86	
Madelon				MultipleF			
Method	Size	Acc	T	Method	Size	Acc	T
LFS	7	64.62	-	LFS	18	99.0	+
GSBS	489	51.28	-	GSBS			
PSOBE	47.83	84.73		PSOBE	51.1	98.86	

C. Comparisons with Traditional Methods

Table III compares the results of PSOBE with that of LFS and GSBS. LFS or GSBS produces a single solution, where number of features and classification accuracy are shown in the "Size" and "Acc" columns. For PSOBE, the average number of features and the *average* classification accuracy are listed in the "Size" and "Acc" columns. "T4" shows the significance tests between the classification accuracy of PSOBE and LFS or GSBS, where "-" ("+") means that LFS and GSBS is significantly worse (better) than PSOBE. Therefore, the more "-", the higher PSOBE's classification accuracy. The results of GSBS on the MultipleF dataset are not presented since the experiment cannot finish within a week.

According to Table III, PSOBE selected a larger number of features than LFS, but achieved significantly better classification accuracy than LFS in most cases. Although on the MultipleF dataset, LFS achieved better classification performance than PSOBE, but the highest classification accuracy of PSOBE (seen from Table II) is better than LFS. PSOBE outperformed GSBS in terms of both the classification accuracy and the number of features in all cases. The results show that PSOBE using PSO and the "backward elimination" can better explore the solution space to search for a better feature subset than the traditional methods, LFS and GSBS.

D. Comparisons on Computational Time

Table IV presents the average computational time used by PSOFS, PSO2S, PSO42 and PSOBE in a single run. The values in the table are expressed in minutes.

According to Table IV, it can be seen that in most cases, all the four PSO based algorithms can finish one run within 10 minutes except on the datasets with a large number of features and instances. The main reason of using long time on large datasets is that all the four algorithms are wrapper approaches. Most of their computational time was spent on fitness evaluations, where a classification process is conducted to get the training classification error rate as the fitness value of the feature subset. A larger number of features or instances needs a longer time for evaluating its fitness.

TABLE IV
COMPUTATIONAL TIME

	Wine	Vehicle	Ionosphere	Sonar	Musk1	Arrhythmia	Madelon	MultipleF
PSOFS	0.25	8.14	1.37	0.75	10.16	11.77	866.15	676.52
PSO2S	0.23	5.99	1.09	0.59	7.71	11.19	962.42	875.63
PSO42	0.31	8.6	1.03	0.54	6.47	8.84	810.27	703.72
PSOBE	0.19	7.56	1.69	0.71	4.16	3.69	322.56	157.05

From Table IV, it can also be observed that PSOBE usually used shorter time than other three algorithms, i.e. PSOFS, PSO2S, and PSO42. Although compared with the other three algorithms, PSOBE has extra steps regarding the "backward elimination", PSOBE is still computationally cheaper than the other three algorithms, which is consistent with our hypothesis. The reason is that the filter "backward elimination" is extremely fast compared with the wrapper fitness evaluation and the reduction of the features can significantly reduce the computational time. This pattern is particularly obvious on the two large datasets (i.e. Madelon and MultipleF), where the number of features selected by PSOBE is much smaller and it used a much shorter time than the other three algorithms.

Compared with LFS and GSBS, PSOBE is slower than LFS because LFS selected a smaller number of features. PSOBE is slower than GSBS on the datasets with a small number of features, but faster than GSBS on the datasets with a large number of features, e.g. Madelon and MultipleF. The main reason is that GSBS starts with all the available features and large feature subsets are involved during the search process, which needs a long time for each evaluation. Meanwhile, PSOBE has a fixed number of evaluations, but the number of evaluations in GSBS grows on the large datasets.

VI. CONCLUSIONS AND FUTURE WORK

The goal of this paper was to develop a new PSO approach to feature selection in classification to select a small subset of features and increase the classification accuracy. This goal has been achieved by proposing a "backward elimination" procedure mimicking a typical backward elimination method to improve the $gbest$ during the search process. The"backward elimination" uses a computationally cheap filter measure to incorporate with the wrapper fitness function aiming to take the advantages of both filter and wrapper approaches. The"backward elimination" is performed on clusters of features rather than all the features selected by $gbest$ to automatically determine the number of features being removed. The proposed approach was examined and compared with a standard PSO based algorithm, two recently developed PSO algorithms, and two traditional algorithms (LFS and GSBS) on eight datasets. The results showed that the proposed algorithm can be successfully used to reduce the number of features and increase the classification accuracy. It outperformed the three PSO based algorithms and GSBS in terms of both the classification performance and the number of features, and selected a slightly larger number of features than LFS, but achieved significantly higher classification accuracy than LFS on almost all datasets. In additional, the proposed approach is also computationally cheaper than the other three PSO based algorithms. This suggests that the proposed algorithm takes

the advantages of both filter and wrapper approaches, which helps increase the classification accuracy. Meanwhile, the filter "backward elimination" is very fast. It may cause a subtle increase in the computational time, but the reduction of the number of features can significantly reduce the time.

From the results, it can also be observed that when the number of features reaches a certain small value, it may sacrifice the classification performance, like on the Vehicle and MultipleF. datasets. Therefore, in the future, we will further improve the performance of the proposed algorithm and also investigate a multi-objective feature selection approach to search for a set of trade-off feature subsets to meet different requirements in real-world applications.

REFERENCES

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

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

[3] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, pp. 273–324, 1997.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *IEEE International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948.

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

[6] A. P. Engelbrecht, *Computational intelligence: an introduction (2. ed.)*. Wiley, 2007.

[7] 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.*, vol. 5, 1997, pp. 4104–4108.

[8] M. A. Esseghir, G. Goncalves, and Y. Slimani, "Adaptive particle swarm optimizer for feature selection," in *international conference on Intelligent data engineering and automated learning (IDEAL'10)*. Springer Verlag, 2010, pp. 226–233.

[9] L. Y. Chuang, H. W. Chang, C. J. Tu, and C. H. Yang, "Improved binary PSO for feature selection using gene expression data," *Computational Biology and Chemistry*, vol. 32, no. 29, pp. 29– 38, 2008.

[10] B. Xue, M. Zhang, and W. N. Browne, "Multi-objective particle swarm optimisation (PSO) for feature selection," in *Genetic and Evolutionary Computation Conference*, 2012, pp. 81–88.

[11] C. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Urbana:The University of Illinois Press, 1949.

[12] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[13] L. Cervante, B. Xue, M. Zhang, and L. Shang, "Binary particle swarm optimisation for feature selection: A filter based approach," in *IEEE Congress on Evolutionary Computation (CEC'12)*, 2012, pp. 881–888.

[14] A. Whitney, "A direct method of nonparametric measurement selection," *IEEE Transactions on Computers*, vol. C-20, no. 9, pp. 1100–1103, 1971.

[15] T. Marill and D. Green, "On the effectiveness of receptors in recognition systems," *IEEE Transactions on Information Theory*, vol. 9, no. 1, pp. 11–17, 1963.

[16] S. Stearns, "On selecting features for pattern classifier," in *Proceedings of the 3rd International Conference on Pattern Recognition*, 1976, pp. 71–75.

[17] 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)*. IEEE, 2009, pp. 332–339.

[18] Z. X. Zhu, Y. S. Ong, and M. Dash, "Wrapper-filter feature selection algorithm using a memetic framework," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 1, pp. 70–76, 2007.

[19] S. Ahmed, M. Zhang, and L. Peng, "Enhanced feature selection for biomarker discovery in lc-ms data using gp," in *IEEE Congress on Evolutionary Computation (CEC'13)*, 2013, pp. 584–591.

[20] A. Al-Ani, A. Alsukker, and R. N. Khushaba, "Feature subset selection using differential evolution and a wheel based search strategy," *Swarm and Evolutionary Computation*, vol. 9, pp. 15–26, 2013.

[21] R. Fdhila, T. Hamdani, and A. Alimi, "Distributed mopso with a new population subdivision technique for the feature selection," in *International Symposium on Computational Intelligence and Intelligent Informatics (ISCIII'11)*, 2011, pp. 81 –86.

[22] L. Y. Chuang, S. W. Tsai, and C. H. Yang, "Improved binary particle swarm optimization using catfish effect for feature selection," *Expert Systems with Applications*, vol. 38, pp. 12 699–12 707, 2011.

[23] B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms," *Applied Soft Computing*, p. Online, 2013.

[24] 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 Systems with Applications*, vol. 35, no. 4, pp. 1817–1824, 2008.

[25] B. Xue, L. Cervante, L. Shang, W. N. Browne, and M. Zhang, "A multi-objective particle swarm optimisation for filter based feature selection in classification problems," *Connection Science*, 2012.

[26] B. Xue, M. Zhang, and W. Browne, "Particle swarm optimization for feature selection in classification: A multi-objective approach," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1656–1671, 2013.

[27] B. Xue, M. Zhang, Y. Dai, and W. N. Browne, "PSO for feature construction and binary classification," in *Genetic and evolutionary computation conference*, 2013, pp. 137–144.

[28] B. Xue, L. Cervante, L. Shang, W. N. Browne, and M. Zhang, "Multi-objective evolutionary algorithms for filter based feature selection in classification," *International Journal on Artificial Intelligence Tools*, vol. 22, no. 04, p. 1350024, 2013.

[29] S. Pledger and R. Arnold, "Multivariate methods using mixtures: correspondence analysis, scaling and pattern detection," *Computational Statistics and Data Analysis (online: http://dx.doi.org/10.1016/j.csda.2013.05.013)*, 2013.

[30] E. Matechou, I. Liu, S. Pledger, and R. Arnold, "Biclustering models for ordinal data," 2011, presentation at the NZ Statistical Assn. Annual Conference, University of Auckland.

[31] M. Lane, B. Xue, I. Liu, and M. Zhang, "Particle swarm optimisation and statistical clustering for feature selection," in *AI 2013: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, 2013, vol. 8272, pp. 214–220.

[32] B. Xue, M. Zhang, and W. N. Browne, "New fitness functions in binary particle swarm optimisation for feature selection," in *IEEE Congress on Evolutionary Computation (CEC'12)*, 2012, pp. 2145–2152.

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

[34] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explorations*, vol. 11, pp. 931–934, 2009.