Particle Swarm Optimisation for Feature Selection in Classification: Novel Initialisation and Updating Mechanisms

Bing Xue, Mengjie Zhang and Will N. Browne

School of Engineering and Computer Science, Victoria University of Wellington, PO Box 600, Wellington 6140, New Zealand

Abstract

In classification, feature selection is an important data pre-processing technique, but it is a difficult problem due mainly to the large search space. Particle swarm optimisation (PSO) is an efficient evolutionary computation technique. However, the traditional personal best and global best updating mechanism in PSO limits its performance for feature selection and the potential of PSO for feature selection has not been fully investigated. This paper proposes three new initialisation strategies and three new personal best and global best updating mechanisms in PSO to develop novel feature selection approaches with the goals of maximising the classification performance, minimising the number of features and reducing the computational time. The proposed initialisation strategies and updating mechanisms are compared with the traditional initialisation and the traditional updating mechanism. Meanwhile, the most promising initialisation strategy and updating mechanism are combined to form a new approach (PSO(4-2)) to address feature selection problems and it is compared with two traditional feature selection methods and two PSO based methods. Experiments on twenty benchmark datasets show that PSO with the new initialisation strategies and/or the new updating mechanisms can automatically evolve a feature subset with a smaller number of features and higher classification performance than using all features. PSO(4-2) outperforms the two traditional methods and two PSO based algorithm in terms of the computational time, the number of features and the classification performance. The superior performance of this algorithm is due mainly to both the proposed initialisation strategy, which aims to take the advantages of both the forward selection and backward selection to decrease the number of features and the computational time, and the new updating mechanism, which can overcome the limitations of traditional updating mechanisms by taking the number of features into account, which reduces the number of features and the computational time.

Keywords: Particle Swarm Optimisation, Feature Selection, Classification

---

1. Introduction

In classification problems, a dataset usually involves a large number of features, often including 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, which is termed "the curse of dimensionality" [1, 2]. Feature selection is proposed to select a subset of relevant features from a large number of available features to achieve similar or even better classification performance than using all features [2]. 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]. Existing feature selection algorithms can be broadly classified into two categories [3, 4]: filter approaches and wrapper approaches. Filter approaches are independent of a learning algorithm and they are argued to be computationally cheaper and more general than wrappers. Wrapper approaches include a learning algorithm as part of the evaluation function. Therefore, wrappers can often achieve better results than filter approaches.

Feature selection is a difficult combinatorial problem. The best feature subset is usually a group of features with the presence of feature complementarity because of the feature interaction problem. There could be two-way or multi-way interactions among features [1, 5]. As a result, an individually relevant feature may become redundant when working together with other features so that eliminating some such features will remove or reduce unnecessary complexity. On the other hand, an individually redundant or weakly relevant feature may become highly relevant when working with others. Therefore, an optimal feature subset should be a group of complementary features. The feature selection task is challenging due mainly to the large search space. The size of the search space increases exponentially with respect to the number of available features in the dataset [1]. Therefore, an exhaustive search is practically impossible in most situations. Although many different search techniques have been applied to feature selection, most of these algorithms still suffer from the problems of stagnation in local optima or being computationally expensive [2, 1, 4]. 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 optimisation (PSO) [6, 7] is a relatively recent EC technique, which is computationally less expensive than some other EC algorithms. Therefore, PSO has been used as an effective technique in feature selection [8, 4]. However, there are a number of limitations about current PSO for feature selection. Firstly, PSO has not been tuned to the feature selection task. Gutierrez et al. [9] show that initialisation strategies in PSO perform differently in different problems with high dimensional search spaces. However, no existing initialisation strategies are specifically proposed for feature selection problems except our previous work [10]. Secondly, the traditional personal and global best updating mechanism may miss some feature subsets with high classification performance, but a small number of features (detailed discussions in Section 3.2). Therefore, the potential of PSO for feature selection has not been fully investigated and we will continue our previous work [10] to further study the initialisation and the updating mechanism in PSO for feature selection.

1.1. Goals

The overall goal of this paper is to propose a new PSO based feature selection approach to selecting a smaller number of features and achieving similar or even better classification performance than using all features and traditional/existing feature selection methods. In order to achieve this goal, we propose *three new initialisation strategies*, which are motivated by forward selection and backward selection, and *three new personal and global best updating mechanisms*, which consider both the number of feature and the classification performance to overcome the limitation of the traditional updating mechanism. Specifically, we will:

- propose new initialisation strategies in PSO to reduce the number of features without decreasing the classification performance of the feature subsets achieved by using traditional initialisation strategy,
- develop new updating mechanisms in PSO to guide the algorithm to search for the feature subsets with high classification performance and a

smaller number of features, and to outperform the traditional updating mechanism,

- develop a new PSO based feature selection algorithm using one of the proposed initialisation strategies and one of the proposed updating mechanisms, and
- investigate whether the proposed feature selection algorithm can obtain a feature subset with a smaller number of features and better classification performance than using all features, and outperform two traditional feature selection methods, the standard PSO based algorithm, and a PSO based algorithm using a single fitness function combining both the number of features and the classification performance.

1.2. Organisation

The remainder of the paper is organised as follows. Section 2 provides background information. Section 3 describes the proposed new initialisation strategies, the proposed personal best and global best updating mechanisms, and the pseudo-code of the proposed algorithm. Section 4 describes experimental design and Section 5 presents experimental results with discussions. Section 6 provides conclusions and future work.

2. Background

2.1. Particle Swarm Optimisation (PSO)

PSO is an EC technique proposed by Kennedy and Eberhart in 1995 [6, 7]. PSO simulates the social behaviour such as birds flocking and fish schooling. In PSO, a population, also 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. Particles move in the search space to search for the optimal solution by updating the position of each particle based on the experience of its own and its neighbouring particles. During the 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 called *pbest* and the best position obtained by the swarm so far is the global best called *gbest*. PSO searches for the optimal solution by updating the position and the velocity of each particle according to the following equations:

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

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

where t denotes the tth iteration in the evolutionary process. $d \in D$ 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.

2.2. Related Work on Feature Selection

1) Traditional Feature Selection Approaches

Relief [11] is a filter feature selection algorithm, which 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. FOCUS [12], also a filter algorithm, exhaustively examines all possible feature subsets, then selects the smallest feature subset. However, the FOCUS algorithm is computationally inefficient because of the exhaustive search.

Greedy search based sequential forward selection (SFS) [13] and sequential backward selection (SBS) [14] are two typical wrapper methods. SFS (SBS) starts with no features (all features), then candidate features are sequentially added to (removed from) the subset until the further addition (removal) does not increase the classification performance. However, these two methods suffer from the problem of so-called nesting effect, which means once a feature is selected (eliminated) it cannot be eliminated (selected) later. This problem can be solved by combining both SFS and SBS into one algorithm. Therefore, Stearns [15] proposes a "plus-l-take away-r" method, which performs l times forward selection followed by r times backward elimination. However, it is difficult to determine the optimal values of (l, r).

2) Evolutionary Computation Techniques for Feature Selection

EC techniques have been applied to feature selection problems, such as genetic algorithms (GAs), genetic programming (GP), ant colony optimisation (ACO) and PSO.

Zhu et al. [16] propose a feature selection method using a memetic algorithm that is a combination of local search and GA. In this algorithm, individual features are firstly ranked according to a filter measure. GA employs the classification accuracy as the fitness function and deletes or adds a feature according to the ranking information. The experiments show that this algorithm outperforms GA alone and other algorithms. Neshatian and Zhang [17] propose a GP based filter model as a multi-objective algorithm for feature selection in binary classification problems. Muni et al. [18] develop a multi-tree GP algorithm for feature selection (GPmtfs) to simultaneously select a feature subset and design a classifier using the selected features. For a problem with c classes, each classifier in GPmtfs has c trees. Comparisons suggest GPmtfs achieves better results than SFS, SBS and other methods. However, the number of features selected increases when there are (synthetically added) noisy features. There are also some other GP related works, which can be seen from [19, 20, 21]. Jensen and Shen [22] apply ACO to find a small reduct in rough set theory to address feature selection problems. Later, Ke et al. [23], [24], and Chen et al. [25] also successfully use ACO and rough set theory to solve feature selection problems. However, the datasets used in these papers have a relatively small number of features (the maximum number is 70).

Chuang et al. [26] develop a strategy for *gbest* in PSO for feature selection in which *gbest* will be reset to zero if it maintains the same value for several iterations. However, the proposed algorithm is only compared with one traditional method in terms of the classification performance and no PSO or EC based algorithms are used for comparisons. Chuang et al. [4] apply the so-called catfish effect to PSO for feature selection, which is to introduce new particles into the swarm by re-initialising the worst particles when *gbest* has not 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. Wang et al. [27] propose a filter feature selection algorithm based on an improved binary PSO and rough set. However, the classification performance of the feature subset is only tested on one learning algorithm, the LEM2 algorithm, which has some bias for rough set based algorithms.

Based on binary PSO, Iswandy and Koenig [28] develop a filter based algorithm. The proposed algorithm employs different weights to linearly combine three objectives, which are evaluated by three filter criteria, into a single fitness function. The results suggest that this algorithm improved the classification performance over using all the available features. Lin and Chen [8] propose a wrapper feature selection algorithm (PSOLDA) based on PSO and a linear discrimination analysis algorithm (LDA), which aims to maximise the classification performance evaluated by LDA. Different parameters are tuned to obtain the best settings for PSOLDA. Experimental results show that PSOLDA outperforms LDA using all features, LDA with principal components analysis (PCA), and LDA with forward and backward selection in almost all cases. However, PSOLDA is sensitive to parameter setting and the datasets in the experiments have a small number of features. Based on a filter measure and PSO, a filter-wrapper feature selection algorithm is proposed in [29], with the goal of integrating their advantages. The 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.

Marinakis et al. [30] propose a wrapper approach based on binary PSO and K-nearest neighbour (KNN) for a real-world medical diagnosis problem called Pap-smear cell classification. The results show that this method removes around half of the features and achieves good classification performance. Huang and Dun [31] propose a wrapper algorithm for feature selection and parameter optimisation in a support vector machine (SVM). In the proposed algorithm, each particle is encoded by two parts, where the first part represents the features in a dataset, which are optimised by binary PSO, and the second part is the parameters in SVM, where are evaluated by continuous PSO. However, only one dataset with a small number of features is used in the experiments, which can not demonstrate the full performance of the proposed algorithm. Fdhila et al. [32] apply a multi-swarm PSO algorithm to solve feature selection problems. However, the computational cost of the proposed algorithm is also high because it involves parallel evolutionary processes and multiple subswarms with a relative large number of particles.

3) Initialisation and Updating Mechanisms in PSO

Recently, many initialisation strategies have been proposed in PSO to improve its performance. Parsopoulos and Vrahatis [33] use the Nonlinear Simplex method (NSM) [34] to generate initial particles in PSO. Experiments show that the NSM based initialisation can improve the performance of PSO on 14 benchmark functions. Since NSM is slow and can be applied only to the problems with low dimensionality, this initialisation method may not be appropriate for feature selection problems, where the dimensionality is typically large. Richards and Ventura [35] propose an initialisation strategy for PSO based on centroidal Voronoi tessellations (CVTs). The goal of this CVTs based initialisation is to evenly initialise the particles' positions. This is clearly different from our new initialisation strategies, which aim to reduce the number of features and also reduce the computational cost of each evaluation. The evenly distributed positions generated by CVTs does not help reduce the number of features. Jabeen et al. [36] propose an opposition based initialisation strategy in PSO. The experiments show that the proposed opposition based PSO achieves better performance than random initialisation. Later, Wang et al. [37] propose an initialisation method based on space transformation search (STS) strategy by evaluating the solutions in both original and transformed spaces to get a better set of initial particles. Gutierrez et al. [9] assume that uniformly distributed particles in the initialisation can improve the performance of PSO. Three different initialisation strategies, the orthogonal array initialisation, a chaotic technique and the opposition-based initialisation in PSO are compared on problems with high dimensional search space. The experiments show that the three initialisation strategies perform differently on different problems. However, all these strategies are general methods and no existing initialisation strategies are specifically proposed for feature selection.

Some researchers also work on developing new *gbest* updating mechanisms. Wang et al. [38] apply a dynamic Cauchy mutation to *gbest*, where if the new *gbest* is better after the application of the mutation operator then it is replaced by the mutated *gbest*. However, the proposed algorithm does not work well on multi-modal problems and it may also not perform well for feature selection problems, which have many local optiman in the search space. Chuang et al. [26] propose a *gbest* updating mechanism, but it simply sets *gbest* to zero and can not be applied to *pbest*. Chen et al. [39] and Qi and Ding [40] also develop new *gbest* updating mechanisms in PSO. However, there has been no existing work that proposes a *pbest* updating mechanism in PSO specifically for feature selection to date.

Many studies have shown that PSO is an efficient search technique for feature selection. However, the potential of PSO for feature selection has not been fully investigated. This paper will work on developing a new initialisation strategy and a new *pbest* and *gbest* updating mechanism in PSO for feature selection to further reduce the number of features without decreasing or probably increasing the classification performance.

3. Proposed Approach

In this section, we will propose three new different initialisation strategies and three new *pbest* and *gbest* updating mechanisms in PSO for feature selection with the goals of increasing the classification performance, reducing the number of features and reducing the computational time.

PSO has two versions, which are continuous PSO [6, 7] and binary PSO [41]. Both of them have been applied in feature selection problems and achieve good results [26, 4, 31]. However, binary PSO is not developed as well as continuous PSO and does not follow the main principles of the standard PSO algorithm [42, 43], for example, 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. Therefore, we will use continuous PSO to propose a novel feature selection approach.

3.1. New Initialisation Strategies

Two commonly used wrapper feature selection algorithms are forward selection [13] and backward selection [14]. Forward selection starts with an empty feature set (no features) and firstly searches for a feature subset (S) with one feature by selecting the feature that achieves the highest classification performance. Then the algorithm selects another feature from the candidate features to add to S. Feature i is selected if adding i to S achieves the largest improvement in classification accuracy. The candidate features are sequentially added to S until the further addition of any feature does not increase the classification performance. On the other hand, backward selection starts with all the available features, then candidate features are sequentially removed from the feature subset until the further removal of any feature does not increase the classification performance. The main difference between them is their starting points (initialisation). Forward selection starts with an empty set of features while backward selection starts with all available features. Forward selection usually selects a smaller number of features and is computationally less than backward selection, but when the best feature subset contains a relatively large number of features, backward selection has a larger chance to obtain the best solution.

Hence, we propose three new initialisation strategies in PSO for feature selection, which are **small initialisation** motivated by forward selection, **large initialisation** motivated by backward selection and **mixed initialisation** aiming to take the advantages of forward and backward selection to avoid their disadvantages. To examine the influence of initialisation strategy in PSO for feature selection, we will test the performance of the three new initialisation strategies and compare them with the traditional random initialisation strategy. Details of these four strategies are as follows:

- **Traditional initialisation**: Each particle is randomly initialised in terms of both the number of features and the combination of individual features.
- **Small initialisation**: This is motivated by forward selection to initialise each particle using a small number of features, but random (different in most cases) combinations of features.
- **Large initialisation**: This is motivated by backward selection to initialise each particle using a large number of features, but random (different in most cases) combinations of features.
- **Mixed initialisation**: This initialisation strategy combines both the small initialisation and large initialisation strategies. In this proposed initialisation strategy, most particles are initialised using a small number of features (simulating forward selection) and other particles are initialised using large feature subsets (simulating backward selection). Meanwhile, through social interaction (updating *pbest* and *gbest*), PSO is expected to be able to reach and search the solution space with medium size feature subsets if these feature subsets can achieve better classification performance.

3.2. New pbest and gbest Updating Mechanism

Sharing information through *pbest* and *gbest* is an important factor in PSO, as it influences the behaviour of the swarm during the evolutionary process. Traditionally, the *pbest* and *gbest* are updated solely based on the fitness value of the particles (i.e. classification performance in feature selection problems). *pbest* of a particle is updated only when the fitness of the new position of the particle is better than the current *pbest*. In feature selection, the traditional updating mechanism has a potential limitation. If the classification performance of the particle's new position is the same as the current *pbest*, but the number of features is smaller, the particle's new position corresponds to a better feature subset. However, according to the traditional updating mechanism, the *pbest* will not be updated because their classification performance is the same.

In order to overcome this limitation, we propose three new *pbest* and *gbest* updating mechanisms, which now considers the number of features when updating *pbest* and *gbest*, to reduce the number of features. The three newly proposed *pbest* or *gbest* updating mechanisms are compared with the traditional updating mechanism and the detailed description of these four mechanisms are shown as follows.

- **Traditional updating mechanism**: If the classification performance of the particle's new position is better than *pbest*, *pbest* is updated. If the new *pbest* is better than *gbest*, *gbest* is updated (see Pseudo-code 1). The number of features is not considered and it can be increased, decreased or the same.

Pseudo-code 1: Traditional updating mechanism

```
if accuracy of particle i ($x_i$) is better than pbest then
    $pbest = x_i$ ;                                  // Update the pbest
if accuracy of any pbest is better than gbest then
    $gbest = pbest$ ;                                // Update the gbest
```

- **Classification performance as the first priority**: *pbest* and *gbest* are updated in two situations (see Pseudo-code 2). The first situation is that if the classification performance of the particle's new position is better than *pbest*, *pbest* will be updated and replaced by the new position. In this case, the number of features selected will be ignored as in the traditional updating mechanism. The second situation is that if the classification performance is the same as *pbest* and the number of features is smaller, the current *pbest* will be replaced by the particle's new position. After updating the *pbest* of each particle, *gbest* of each particle is updated in the same way by comparing *gbest* with the *pbest* of the particle and its neighbours.

- **Improve both the number of features and the classification performance**: *pbest* and *gbest* are updated in two situations (see

Pseudo-code 2: Classification performance as the first priority

```
if $accuracy\ of\ particle\ i\ (x_i)\ is\ better\ than\ pbest$ then
    $pbest = x_i$ ;                                        // Update the $pbest$
else if $accuracy\ of\ x_i\ is\ the\ same\ as\ pbest\ and\ |x_i| < |pbest|$ then
    $pbest = x_i$ ;                                        // Update the $pbest$
if $accuracy\ of\ any\ pbest\ is\ better\ than\ gbest$ then
    $gbest = pbest$ ;                                      // Update the $gbest$
else if $accuracy\ of\ any\ pbest\ is\ the\ same\ as\ gbest\ and\ |pbest| < |gbest|$ then
    $gbest = pbest$ ;                                      // Update the $gbest$
```

Pseudo-code 3). The first situation is that if the classification performance of the particle's new position is better than that of *pbest* and the number of features is not larger than *pbest*, *pbest* is updated. The second situation is that if the number of features is smaller than *pbest* and if the number of features becomes smaller and the classification performance of the new position is *not worse (the same or better)* than current *pbest*, *pbest* is updated. *gbest* is updated in the same way by comparing *gbest* with the *pbest* of the particle and its neighbours.

Pseudo-code 3: Improving both the number of features and the classification performance

```
begin
    if $accuracy\ of\ particle\ i\ (x_i)\ is\ better\ than\ pbest\ and\ |x_i| \leq |pbest|$ then
        $pbest = x_i$ ;                                    // Update the $pbest$
    else if $accuracy\ of\ x_i\ is\ the\ same\ as\ pbest\ and\ |x_i| < |pbest|$ then
        $pbest = x_i$ ;                                    // Update the $pbest$
    if $accuracy\ of\ any\ pbest\ is\ better\ than\ gbest\ and\ |pbest| \leq |gbest|$ then
        $gbest = pbest$ ;                                  // Update the $gbest$
    else if $accuracy\ of\ any\ pbest\ is\ the\ same\ as\ gbest\ and\ |pbest| < |gbest|$ then
        $gbest = pbest$ ;                                  // Update the $gbest$
```

- **Compromise between the classification performance and the number of features**: *pbest* and *gbest* are also updated in two situations (see Pseudo-code 4). The first situation is that if the classification performance is better than *pbest* and the number of features is not larger than *pbest*, *pbest* is updated. The second situation is that if the classification performance decreases by less than 5% and the number of features is smaller, *pbest* is updated. *gbest* is updated in the same way by comparing *gbest* with the *pbest* of the particle and its neighbours.

Pseudo-code 4: Compromise between the classification performance and the number of features

```
begin
    if accuracy of particle i ($x_i$) is better than pbest and $|x_i| \leq |pbest|$ then
        $pbest = x_i$ ;                                         // Update the pbest
    else if accuracy of $x_i$ is better than $pbest * 0.95$ and $|x_i| < |pbest|$ then
        $pbest = x_i$ ;                                         // Update the pbest
    if accuracy of any pbest is better than gbest and $|pbest| \leq |gbest|$ then
        $gbest = pbest$ ;                                       // Update the gbest
    else if accuracy of any pbest is better than $gbest * 0.95$ and $|pbest| < |gbest|$
    then
        $gbest = pbest$ ;                                       // Update the gbest
```

All these three new updating mechanisms include the traditional updating mechanism and add other situations in updating *pbest* and *gbest*. Where available, it will always select a better feature subset to be the *pbest* or *gbest*, which either has better classification performance or the same classification performance with a smaller number of features. This can help the algorithm filter out redundant features and make the feature subset with good classification performance and a small number of features to be the *leader* (*pbest* or *gbest*) of each particle and the whole swarm.

The proposed *pbest* and *gbest* updating mechanisms are similar to parsimony pressure used in GP. In GP, each individual can be represented as a tree. The size of the trees can be considered in the selection process, where the selection operator prefers smaller trees only when their fitnesses are equal, known as parsimony pressure [44]. However, the proposed updating mechanisms are different from parsimony pressure in two aspects. Firstly, the parsimony pressure in GP changes the size of the trees while the proposed *pbest* and *gbest* updating mechanisms do not change the size of the particles that is always the total number of features in the dataset. Secondly, the parsimony pressure is to control the size of the trees in GP, which was not designed for any problem domain, but the number of features considered in the proposed *pbest* and *gbest* updating mechanisms are particular for feature selection problems to optimise one of the two main objectives, i.e. minimising the number of features.

3.3. Pseudo-code of A Proposed Feature Selection Algorithm

Based on the three new initialisation strategies and the three new *pbest* and *gbest* updating mechanisms, new PSO based feature selection algorithms can be proposed by using one new initialisation strategy or/and one new updating mechanism. The fitness function, which measures the classification performance, is used in the proposed algorithms and shown by Equation 3.

$$Fitness_1 = ErrorRate = \frac{FP + FN}{TP + TN + FP + FN} \tag{3}$$

where FP, FN, TP and TN stand for false positives, false negatives, true positives, and true negatives, respectively. For example, in an binary classification problem with a class Positive and a class Negative, for an instance a, TP increases 1 when a is predicted as class Positive and a's true class label is also class Positive. FP increases 1 when a is predicted as Positive while a's true class label is Negative. TN increases 1 when a is predicted as Negative and a's true class label is also Negative. FN increases 1 when a is predicted as Negative, but a's true class label is Positive.

In this work, we will investigate the performance of the standard PSO using the traditional initialisation with traditional updating mechanism for feature selection (named PSO(1-1)). We will only investigate the performance of PSO using three new initialisation strategies with traditional updating mechanism, which are PSO(2-1) using small initialisation, PSO(3-1) using large initialisation and PSO(4-1) using mixed initialisation, and PSO using the traditional initialisation strategy with one of the three new updating mechanisms, which are PSO(1-2) treats the classification performance as the first priority, PSO(1-3) aims to improve both the classification performance and the number of features, and PSO(1-4) compromises between the classification performance and the number of features. In order to test the combination of new initialisation strategies and updating mechanisms, we also investigate the performance of PSO using the mixed initialisation and the *pbest* and *gbest* updating mechanism treating the classification performance as the first priority for feature selection (named PSO(4-2)). The reason is that the mixed initialisation is proposed to utilise the advantages and avoid the disadvantages of both forward selection and backward selection, and considers that the classification performance is more important than the number of

Algorithm 1: The pseudo-code of the proposed algorithm (PSO(4-2))

```
begin
    divide Dataset into a Training set and a Test set;
    initialise most of the particles using small feature subsets and the others
    particles using relatively large feature subsets;
    initialise the velocity of each particle;
    while Maximum Iterations or the stopping criterion is not met do
        evaluate the fitness (classification performance, i.e. error rate) of each
        particle on the Training set;
        for i=1 to Population Size do
            if accuracy of particle i ($x_i$) is better than that of pbest then
                $pbest = x_i$ ;                // Update the pbest of particle i
            else if accuracy of $x_i$ is the same as pbest and $|x_i| < |pbest|$ then
                $pbest = x_i$ ;                // Update the pbest of particle i
            if accuracy of any pbest is better than that of gbest then
                $gbest = pbest$ ;              // Update the gbest of particle i
            else if accuracy of any pbest is the same as gbest and
            $|pbest| < |gbest|$ then
                $gbest = pbest$ ;              // Update the gbest of particle i
        for i=1 to Population Size do
            update the velocity and the position of particle i
    calculate the classification accuracy of the selected feature subset on the Test
    set;
    return the position of gbest (the selected feature subset);
    return the training and test classification accuracies;
```

features in feature selection problems. Therefore, PSO(4-2) is expected to simultaneously increase the classification performance, reduce the number of features and reduce the computationally time. The pseudo-code of PSO(4-2) can be seen in Algorithm 1. The pseudo-code of the other new algorithms is similar to PSO(4-2) except for the initialisation and the *pbest* and *gbest* updating mechanism procedure.

4. Design of Experiments

4.1. Benchmark Techniques

To examine the performance of the proposed initialisation strategies, we will firstly compare PSO(2-1), PSO(3-1) and PSO(4-1) with PSO(1-1), all of which use the traditional *pbest* and *gbest* updating mechanism. We also compare PSO(1-2), PSO(1-3) and PSO(1-4) with PSO(1-1), all of which use the traditional random initialisation strategy. Meanwhile, we compare the proposed PSO(4-2) with two traditional wrapper feature selection methods, PSO(1-1), and a PSO based algorithm with a single fitness function combining both the classification performance and the number of features (PSO-No, defined below).

The two traditional methods are linear forward selection (LFS) and greedy stepwise backward selection (GSBS), which were derived from SFS and SBS, respectively. LFS [45] restricts the number of features that is 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 usually obtain good results. More details can be seen in the literature [45]. The greedy stepwise based feature selection algorithm can move either forward or backward in the search space [46]. Given that LFS performs a forward selection, a backward search is chosen in greedy stepwise to conduct a greedy stepwise backward selection (GSBS). GSBS starts with all available features and stops when the deletion of any remaining feature results in a decrease in classification performance.

PSO-No uses a single fitness function to combine both the classification performance and the number of features, where the weight for the number of features is very small. The fitness function is shown by Equation 4.

$$Fitness_2 = ErrorRate + \alpha * \#Features \tag{4}$$

where $ErrorRate$ means the training classification error of the selected features and $\#Features$ presents the number of features selected. α shows the relative importance of the number of features. α is designed to be an extremely small value to ensure that the second component in Equation 4 is always smaller than the first component. Therefore, the classification performance can dominate Equation 4 to lead PSO-No to search for feature subsets with low classification error rates. Since the datasets used in this paper do not have a huge number of features (over 10 000), for the purpose of letting the classification performance dominate Equation 4, α can be a fixed, small number (i.e. 1.0E-6). Note that the purpose of Equation 4 is different from a typical way of combing the classification performance and the number of features into a single fitness function [47, 48], where the relative importance of these two parts are not necessarily significantly different. Equation (4) aims to make the classification quality significantly more important than the number of features. This purpose is achieved by using the very small α in

Table 1: Datasets

Dataset	No. of Features	No. of Classes	No. of Instances	Dataset	No. of Features	No. of Classes	No. of Instances
Wine	13	3	178	Zoo	17	7	101
Vehicle	18	4	846	German	24	2	1000
WBCD	30	2	569	Ionosphere	34	2	351
Lung	56	3	32	Sonar	60	2	208
Movementlibras	90	15	360	Hillvalley	100	2	606
Musk Version1(Musk1)	166	2	476	Arrhythmia	279	16	452
Madelon	500	2	4400	Isolet5	617	2	1559

Equation (4), which make the classification quality 10^6 times more important than the number of features (and the total number of features in the dataset can be ignored).

Equation 4 is a typical way to optimise two objectives, where one objective is often more important than the other objective. Feature selection involves two goals of maximising the classification performance and minimising the number of features, but the classification performance is more important than the number of features. However, this typical way of combining two objectives into one single fitness function with a very small α **has not been applied** to feature selection. In this work, we applied it to feature selection to develop PSO-No and compare its performance with that of PSO(1-1) and PSO(4-2). The basic steps of PSO-No are the same as PSO(4-2) except for the fitness function, the initialisation strategy and the *pbest* and *gbest* updating mechanism.

4.2. Datasets and Parameter Settings

Fourteen datasets (Table 1) chosen from the UCI machine learning repository [49] are used in the main experiments and six additional datasets in Table 8 in Section 6.2 are used for further comparisons. The fourteen datasets are chosen to have different numbers of features, classes and instances as representatives of different types of problems that the proposed algorithms will be tested on. For each dataset, the instances are randomly divided into two sets: 70% as the training set and 30% as the test set.

As a wrapper approach, the proposed algorithms requires a learning algorithm. A simple and commonly used learning algorithm [26], KNN is used in the experiments and K=5 (5NN). Waikato Environment for Knowledge Analysis (Weka) [50] is used to run the experiments using LFS and GSBS. All the settings in LFS and GSBS are kept to the defaults. 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 10 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 [3].

Feature selection is a binary problem, but continuous PSO is used in this work because there are some limitations in the current version of binary PSO [43]. The representation of a particle is an "n-bit" string, where "n" is the total number of features in the dataset. The position value in each dimension (x_{id}) is in [0,1]. A threshold θ is needed to compare with the value of x_{id}. If $x_{id} > \theta$, the dth feature is selected. Otherwise, the dth feature is not selected. The parameters are selected according to common settings proposed by Clerc and Kennedy [51]. The common settings are used here because using them can clearly test whether the improvement of the performance is caused by the newly proposed mechanisms rather than other factors. The detailed settings are shown as follows: $w = 0.7298$, $c_1 = c_2 = 1.49618$, population size is 30, and the maximum iteration is 100. The fully connected topology is used. According to our preliminary experiments [52, 53], the threshold θ is set as 0.6 to determine whether a feature is selected or not. In PSO(2-1), all particles are initialised using a small number of features, which is around 10% of the total number of features in the dataset, but the combination of individual features are randomly selected. In PSO(3-1), all particles are initialised using a large number of features (more than half of the total number of features), where for one particle, a random number (e.g. m, where m is between half and the total number of features) is firstly generated and m features are randomly selected to initialise this particle. In PSO(4-1) and PSO(4-2), a major part of the swarm (2/3) is initialised using small feature subsets like PSO(2-1), and the other minor part of the swarm (1/3) is initialised using more than half of the total number of features like PSO(3-1). In PSO-No, $\alpha = 1.0E{-}6$, which is very small to ensure that the classification performance is always much more important than the number of features.

For each dataset, the experiments to examine the feature selection performance of each algorithm has been conducted for 50 independent runs. A statistical significance test, pairwise Student's T-test, is performed between the testing classification performance of different algorithms. The significance level in the T-tests is selected as 0.05 (or confidence interval is 95%).

Meanwhile, the non-parametric statistical significance test, Wilcoxon test, is also performed with the significance level of 0.05. The results of the Wilcoxon test is similar to that of T-test. Therefore, the results of the Wilcoxon test are not listed in the next section.

5. Experimental Results and Discussions

Table 2 shows the experimental results of PSO(1-1), PSO(2-1), PSO(3-1), and PSO(4-1). Table 3 shows the experimental results of PSO(1-1), PSO(1-2), PSO(1-3), and PSO(1-4). The results of PSO(4-2), PSO(4-3), and PSO(4-4) are shown in 4. Table 5 shows the experimental results of PSO(1-1), PSO(4-2), and the benchmark techniques.

5.1. Influence of the initialisation strategy in PSO for feature selection

Table 2 shows the experimental results of using the four different initialisation strategies with the traditional *pbest* and *gbest* updating mechanism. In both Table 2 and Table 3, "All" shows the total number of features in the dataset. "AveNO." represents the average number of features selected by each algorithm in 50 independent runs. "Ave", "Best" and "StdDev" indicate the average, the best and the standard deviation of the classification accuracy obtained from the 50 runs on each test set. "T1" shows the results of T-test between the classification performance of using all the available features and that of a PSO based feature selection algorithm, and "T2" shows the results of T-test between the classification performance achieved by PSO(1-1) and another algorithm. In both "T1" and "T2", "+" ("-") means that the classification performance of the feature selection algorithm is significantly better (worse) than that of all features or PSO(1-1). "=" indicates they are similar.

5.1.1. Results of random initialisation

According to Table 2, on 13 of the 14 cases, standard PSO(1-1) using traditional random initialisation strategy selected around half of the available features and achieved better classification performance than using all features. Only on the Movementlibras, where the classification performance of using all features was already very high (94.81%), the average classification accuracy of PSO(1-1) was slightly worse (94.51%), but its best classification accuracy (95.31%) is better than using all features and PSO(1-1) removed around two thirds of the available features. The results suggest that standard PSO with traditional initialisation and updating mechanism can be

Table 2: Experimental Results of New Initialisation Strategies

Dataset	Method	AveNO.	Ave(Best)	StdDev	T1	T2	Dataset	Method	AveNO.	Ave(Best)	StdDev	T1	T2
	All	13	76.54					All	17	80.95			
	1-1	7.24	95.95(100)	2.2	+			1-1	8.16	95.47(97.14)	0.77	+	
Wine	2-1	3.58	96.1(100)	2.46	+	=	Zoo	2-1	3.2	94.08(97.14)	1.62	+	-
	3-1	9.44	96.62(98.77)	2.3	+	=		3-1	9.96	95.62(97.14)	0.89	+	=
	4-1	8.1	96.15(98.77)	1.92	+	=		4-1	7.98	95.64(97.14)	0.71	+	=
	All	18	83.86					All	24	68.0			
	1-1	9.06	85.07(87.01)	0.87	+			1-1	12.42	68.47(72)	1.89	=	
Vehicle	2-1	3.6	81.7(85.24)	1.84	-	-	German	2-1	2.34	67.35(72)	6.46	=	=
	3-1	10.58	85.26(87.99)	0.96	+	=		3-1	16.82	69.13(71.67)	1.22	+	+
	4-1	9.54	85.06(86.61)	0.99	+	=		4-1	13.68	69.27(74.33)	1.78	+	+
	All	30	92.98					All	34	83.81			
	1-1	11.82	93.34(94.74)	0.52	+			1-1	8.74	88.86(93.33)	2.04	+	
WBCD	2-1	3.04	93.05(94.74)	2.43	=	=	Ionosphere	2-1	3.36	88.06(92.38)	2.15	+	=
	3-1	19.06	93.02(94.15)	0.36	=	-		3-1	18.38	86.5(94.29)	1.97	+	-
	4-1	8.12	93.78(94.74)	0.76	+	+		4-1	3.26	87.45(92.38)	2.38	+	-
	All	56	70.0					All	60	76.19			
	1-1	23.58	74(90)	7.48	+			1-1	22.04	78.64(85.71)	3.32	+	
Lung	2-1	2.86	79(90)	8.77	+	+	Sonar	2-1	6.18	76.83(85.71)	5.07	=	-
	3-1	37.16	73.2(90)	5.46	+	=		3-1	31.76	78.03(85.71)	3.69	+	=
	4-1	15.96	75.8(90)	7.77	+	=		4-1	11.3	77.4(87.3)	4.11	+	=
	All	90	94.81					All	100	56.59			
	1-1	37.96	94.51(95.31)	0.33	-			1-1	40.54	58.13(60.99)	1.35	+	
Movement	2-1	11.84	94.28(95.19)	0.44	-	-	Hillvalley	2-1	6.74	56.98(61.81)	2	=	-
	3-1	50.62	94.42(95.06)	0.25	-	=		3-1	60.56	57.78(60.71)	1.27	+	=
	4-1	27.12	94.52(95.19)	0.32	-	=		4-1	13.16	57.68(61.26)	1.85	+	=
	All	166	83.92					All	278	94.46			
	1-1	74.26	84.6(91.61)	2.23	+			1-1	120.22	94.54(95.14)	0.35	=	
Musk1	2-1	16.66	81.57(89.51)	3.12	-	-	Arrhythmia	2-1	14.36	94.4(95.36)	0.62	=	=
	3-1	107.14	85.51(90.21)	2.17	+	+		3-1	171.98	94.32(95.02)	0.26	-	-
	4-1	83.54	84.62(90.21)	2.32	+	=		4-1	81.72	94.69(95.59)	0.36	+	+
	All	500	70.9					All	617	98.45			
	1-1	235.2	76.93(79.87)	1.54	+			1-1	283.3	98.59(98.78)	0.09	+	
Madelon	2-1	16.32	77.22(89.23)	9.01	+	=	Isolet5	2-1	115.4	98.44(98.69)	0.13	=	-
	3-1	336.9	74.32(78.85)	1.58	-	-		3-1	395.64	98.54(98.77)	0.1	+	-
	4-1	267.36	76.54(83.85)	2.65	+	=		4-1	337.6	98.58(98.85)	0.1	+	=

successfully applied to feature selection problems to select a smaller number of features and achieve better classification performance than using all features.

5.1.2. Small initialisation

According to Table 2, PSO(2-1) further reduced the number of features selected by PSO(1-1), which was less than 20% of the available features. PSO(2-1) achieved similar or better classification performance than using all features in most cases. The classification performance of PSO(2-1) was similar to PSO(1-1) on five datasets, better than PSO(1-1) on two datasets, but worse than PSO(1-1) on six datasets. The results suggests that PSO(2-1) has the advantage of the forward selection to select a small number of features. However, for the datasets in which the best classification performance was achieved by a large number of features, PSO(2-1) achieved slightly worse classification performance than other three algorithms, such as in the Movementlibras dataset, Musk1, and German datasets. The main reason is that the number of features was not considered or evolved during the evolutionary training process. If two feature subsets have the same (high) classification performance, but have different numbers of features, PSO(2-1) will be more likely to reach the smaller feature subset first and keep it as *pbset* or *gbest*. Even if PSO(2-1) reaches the larger feature subsets, *pbset* or *gbest* will not be replaced. Therefore, the number of features in PSO(2-1) is usually small, but this may also limit PSO(2-1) to search for the space containing solutions with a larger number of features, which results in slightly worse classification performance than PSO(1-1) in some cases. Although increasing the number of iterations might address this limitation in some cases, it will also increase the computational cost.

5.1.3. Large initialisation

According to Table 2, PSO(3-1) selected a larger number of features than PSO(1-1) and PSO(2-1). PSO(3-1) achieved better classification performance than using all features on 11 of the 14 datasets, similar classification performance to PSO(1-1) on seven datasets, better classification performance than PSO(1-1) on two datasets, but worse than PSO(1-1) on five datasets. The results suggest that PSO(3-1), which simulates the backward selection, also suffers from the problem of selecting a relatively large number of features. The main reason is that *pbest* and *gbest* in PSO(3-1) are firstly assigned by feature subsets with a large number of features. Even if PSO(3-1) reaches a smaller feature subset with the same classification performance, *pbest* and *gbest* will not be updated. Therefore, this may limit PSO(3-1) to search for the space containing solutions with a smaller number of features, and also results slightly worse classification performance than PSO(1-1) in some cases.

5.1.4. Mixed initialisation

According to Table 2, PSO(4-1) achieved better classification performance than using all features on 13 of 14 datasets (except for the Movementlibras dataset). On 13 of the 14 datasets, PSO(4-1) achieved similar or better classification performance than PSO(1-1), while in most cases, the number of features selected by PSO(4-1) was larger than PSO(2-1) but smaller than PSO(1-1) and PSO(3-1). This might be because PSO(4-1) was proposed to simulate both forward and backward selection to utilise their advantages and avoid their disadvantages. By initialising particles using both a small number and a large number of features, PSO(4-1) can avoid the limitations of PSO(2-1) and PSO(3-1) as it achieved at least as good a classification performance as PSO(1-1), but selected a smaller number of features in most cases. Meanwhile, as the number of features selected is smaller, the computational time of PSO(4-1) is usually shorter than PSO(1-1) and PSO(3-1).

Generally, all these four methods using different initialisation strategies selected a smaller number of features and achieved better classification performance than using all features. Although the classification performance is slightly different, the main difference between the above four algorithms are the number of features. The main reason is that all these four algorithms do not consider (or evolve) the number of features during the evolutionary training process. Therefore, the initialisation of the number of features can *significantly* influence the final solutions. PSO(4-1) simulates both forward selection and backward selection in the initialisation procedure, which results in at least as good classification performance as PSO(1-1), but it selected a smaller number of features in eight of the 14 cases and also reduced the computational time. The results and comparisons suggest that the initialisation strategy is important in PSO for feature selection, and it should not be ignored.

5.2. Influence of pbset and gbest updating mechanism in PSO for feature selection

Table 3 shows the experimental results of PSO using the traditional initialisation strategy (random initialisation) and different *pbset* and *gbest* updating mechanisms for feature selection. The performance of PSO(1-1) has been discussed in Section 5.1 and will not be repeated here.

5.2.1. Classification performance as the first priority

According to Table 3, PSO(1-2) selected less than half or even less than one third of the available features and achieved significantly better or similar classification than using all features on 13 of the 14 datasets. The exception being the Movementlibras dataset, where PSO(1-2) selected around one third of the available features, but achieved slightly worse average classification performance (94.56%) than using all features (94.81%), although the

Table 3: Experimental Results of New *pbest* and *gbest* Updating Mechanisms

Dataset	Method	AveNO.	Ave(Best)	StdDev	T1	T2	Dataset	Method	AveNO.	Ave(Best)	StdDev	T1	T2
	All	13	76.54					All	17	80.95			
	1-1	7.24	95.95(100)	2.2	+			1-1	8.16	95.47(97.14)	0.77	+	
Wine	1-2	5.9	95.16(98.77)	2.19	+	=	Zoo	1-2	4.98	95.3(97.14)	0.55	+	=
	1-3	4.64	96.87(100)	2.78	+	=		1-3	4.04	95.2(96.19)	0.42	+	-
	1-4	4.58	96.72(100)	2.82	+	=		1-4	4.28	95.24(96.19)	0.5	+	=
	All	18	83.86					All	24	68.0			
	1-1	9.06	85.07(87.01)	0.87	+			1-1	12.42	68.47(72)	1.89	=	
Vehicle	1-2	9.16	85.13(87.01)	0.85	+	=	German	1-2	11.78	68.57(72.33)	2.05	=	=
	1-3	5.48	84.15(86.81)	1.26	=	-		1-3	6.38	68.95(72)	2.02	+	=
	1-4	4.48	83.68(86.81)	1.26	=	-		1-4	4.46	68.81(72)	1.97	+	=
	All	30	92.98					All	34	83.81			
	1-1	11.82	93.34(94.74)	0.52	+			1-1	8.74	88.86(93.33)	2.04	+	
WBCD	1-2	4.18	93.72(94.74)	0.85	+	+	Ionosphere	1-2	8.18	88.91(92.38)	2.04	+	=
	1-3	3.02	93.91(94.74)	1.43	+	+		1-3	3.5	88.53(95.24)	2.86	+	=
	1-4	2.54	93.63(94.74)	1.9	+	=		1-4	3.38	88.08(91.43)	2.12	+	=
	All	56	70.0					All	60	76.19			
	1-1	23.58	74(90)	7.48	+			1-1	22.04	78.64(85.71)	3.32	+	
Lung	1-2	12.62	75.4(90)	7.27	+	=	Sonar	1-2	17.98	78.03(85.71)	3.76	+	=
	1-3	5.12	79.8(90)	6.16	+	+		1-3	9.06	77.78(87.3)	3.8	+	=
	1-4	5.52	79.6(90)	6.31	+	+		1-4	7.86	77.46(87.3)	3.63	+	=
	All	90	94.81					All	100	56.59			
	1-1	37.96	94.51(95.31)	0.33	-			1-1	40.54	58.13(60.99)	1.35	+	
Movement	1-2	36.72	94.56(95.31)	0.32	-	=	Hillvalley	1-2	40.74	58.04(60.44)	1.24	+	=
	1-3	18.28	94.47(95.31)	0.39	-	=		1-3	18.6	57.98(60.16)	1.46	+	=
	1-4	11.84	94.42(95.19)	0.35	-	=		1-4	4.42	56.18(60.16)	1.96	=	-
	All	166	83.92					All	278	94.46			
	1-1	74.26	84.6(91.61)	2.23	+			1-1	120.22	94.54(95.14)	0.35	=	
Musk1	1-2	72.3	84.69(90.91)	2.24	+	=	Arrhythmia	1-2	108.52	94.44(95.14)	0.3	=	=
	1-3	38.1	82.84(88.11)	2.77	-	-		1-3	48.34	94.78(95.81)	0.38	+	+
	1-4	30.2	82.88(88.11)	2.69	-	-		1-4	20.74	94.68(95.59)	0.37	+	+
	All	500	70.9					All	617	98.45			
	1-1	235.2	76.93(79.87)	1.54	+			1-1	283.3	98.59(98.78)	0.09	+	
Madelon	1-2	233.94	77.08(80.64)	1.66	+	=	Isolet5	1-2	282.8	98.6(98.85)	0.09	+	=
	1-3	104.82	80.06(85.77)	2.33	+	+		1-3	149.4	98.56(98.95)	0.12	+	=
	1-4	73.14	81.74(86.41)	1.9	+	+		1-4	98.34	98.59(98.8)	0.11	+	=

best classification performance (95.31%) of PSO(1-2) was better than using all features. Compared with PSO(1-1), PSO(1-2) achieved similar or better classification performance on all datasets and a smaller number of features on 12 of the 14 datasets. The main reason is that PSO(1-2) takes the classification performance as the first priority, which can firstly guarantee PSO(1-2) achieve at least as good classification performance as PSO(1-1), and PSO(1-2) also considers the number of features if the classification performance is the same value, which can remove redundant or irrelevant features to reduce the number of features selected and may further improve the classification performance, such as on the WBCD dataset.

5.2.2. Improve both the number of features and the classification performance

According to Table 3, PSO(1-3) selected around a quarter (or less) of the available features and achieved significantly better or similar classification performance than using all features on 12 of the 14 datasets. On the Movementlibras and Musk1 datasets, although PSO(1-3) achieved slightly worse classification performance on average than using all features, its best classification performance was higher and the number of features was significantly reduced to less than a quarter of the total number of features. The number of features selected by PSO(1-3) is much smaller than that of PSO(1-1) and PSO(1-2). The classification performance of PSO(1-3) was similar to that of PSO(1-1) on six of the 14 datasets, slightly worse than PSO(1-1) on four datasets and slightly better on four datasets. This might be because in PSO(1-1) and PSO(1-2), if the classification performance was increased, the number of features was ignored. PSO(1-3) aims to guarantee both the classification performance was not reduced and the number of features was not increased when updating *pbest* or *gbest*, which can further reduce the number of features without decreasing the classification performance on 10 of the 14 datasets, but it might also cause the algorithm to miss the feature subsets with high classification performance and a large number of features.

5.2.3. Classification performance compromises the number of features

According to Table 3, in most datasets, PSO(1-4) selected around one fifth (or less) of the available features and achieved similar or even better classification performance than using all features. PSO(1-4) further reduced the number of features selected by PSO(1-1), PSO(1-2), and PSO(1-3). PSO(1-4) achieved similar classification performance to PSO(1-1) on six datasets, slightly worse than PSO(1-1) on five datasets and slightly better than PSO(1-1) on three datasets. The reason might be that when updating *pbest* or *gbest*, the number of features in PSO(1-4) was treated as more important than in PSO(1-1), PSO(1-2) and PSO(1-3), which guides PSO(1-4) to search for the feature subsets with a small number of features. Meanwhile, the classification performance in PSO(1-4) compromises the number of features to a very small extent. Therefore, the classification performance of PSO(1-4) was slightly worse than PSO(1-1), PSO(1-2) and PSO(1-3) in many cases.

Generally, all these four methods using the new *pbset* and *gbest* updating mechanisms can select a smaller number of features and achieve better classification performance than using all features. PSO(1-2) takes the classification performance as the first priority and considers the number of features only if the classification performance is the same. Therefore, PSO(1-2) achieved at least as good a classification performance as PSO(1-1), but selected a smaller number of features. PSO(1-3) aims to not reduce the classification performance and also not increase the number of features, which may results in missing the feature subset with high classification performance and a large number of features. In PSO(1-4), the classification performance was compromised by the requirement to reduce the number of features. Therefore, PSO(1-4) selected a smaller number of features than PSO(1-1), but the classification performance was worse than PSO(1-1) in some cases. The results and comparisons show that the *pbset* and *gbest* updating mechanism can *significantly* influence the performance of PSO for feature selection in terms of both the classification performance and the number of features. The results also show that the updating mechanism is more important than the initialisation strategy in PSO for feature selection. Therefore, to improve the performance of PSO for feature selection, the updating mechanism should be naturally considered first. Meanwhile, since the initialisation strategy is simple and easy to implement, we should combine them together to further improve the feature selection performance and reduce the computational cost.

5.2.4. Comparisons Between PSO(4-2), PSO(4-3) and PSO(4-4)

From Section 5.1, it can be seen that the mixed initialisation strategy outperformed the others strategies. Therefore, the performance of using the mixed initialisation and new updating mechanisms are investigated and Table 4 shows comparisons between PSO(4-2), PSO(4-3) and PSO(4-4).

According to Table 4, the feature subsets selected by PSO(4-2) was larger than PSO(4-3) on eight of the 14 datasets and larger than PSO(4-4) on 13 of the 14 datasets. The main reasons are the same as discussed before, which was PSO(4-3) and PSO(4-4) treat the number of features with higher weighting than PSO(4-2). On the other hand, PSO(4-2) treats the classification performance with higher emphasis than PSO(4-3) and PSO(4-4). The classification performance achieved by PSO(4-2) was at least similar to that of PSO(4-3) and PSO(4-4). It also significantly better than PSO(4-3) on 10 datasets and PSO(4-4) on nine datasets of the 14 datasets. Meanwhile, on the datasets including a large number of features (larger than 100), PSO(4-2) achieved significantly better classification performance than both PSO(4-3) and PSO(4-4). Since the classification performance is usually more important than the number of features, we choose PSO(4-2) as a representative method to compare with other methods.

Table 4: Comparisons Between PSO(4-3), PSO(4-4) and PSO(4-2)

Dataset	Method	AveNO.	Ave(Best)	StdDev	T-test	Dataset	Method	AveNO.	Ave(Best)	StdDev	T-test
	4-2	6.84	95.26(98.77)	1.75E0			4-2	6.46	95.52(97.14)	71.7E-2	
Wine	4-3	6.5	94.79(100)	3.37E0	=		4-3	5.88	95.05(98.1)	1.32E0	-
	4-4	6.42	94.4(100)	3.93E0	=	Zoo	4-4	5.7	95.14(98.1)	1.31E0	=
	4-2	10.16	85.3(87.01)	94.7E-2			4-2	12.76	68.47(70.67)	1.45E0	
Vehicle	4-3	8.12	84.7(87.99)	1.35E0	-		4-3	11.14	68.93(70.67)	1.61E0	=
	4-4	7.08	84.53(87.8)	1.52E0	-	German	4-4	2.78	67.21(71)	9.77E0	=
	4-2	3.46	93.98(94.74)	85.6E-2			4-2	3.26	87.27(91.43)	1.84E0	
WBCD	4-3	2.4	91.88(94.74)	2.81E0	-		4-3	2.4	87.03(93.33)	2.33E0	=
	4-4	2.5	90.96(94.74)	2.61E0	-	Ionosphere	4-4	2.34	86.57(91.43)	2.02E0	=
	4-2	6.74	78.4(90)	6.44E0			4-2	11.24	78.16(85.71)	3.18E0	
Lung	4-3	9.44	78.8(100)	9.93E0	=		4-3	8.98	74.54(85.71)	5.02E0	-
	4-4	9.44	78.8(100)	9.93E0	=	Sonar	4-4	6.1	73.21(85.71)	5.07E0	-
	4-2	27.46	94.58(95.19)	37.6E-2			4-2	12.22	57.77(60.71)	1.47E0	
Movement	4-3	39.08	94.41(95.06)	31E-2	-		4-3	3.02	55.62(59.62)	1.85E0	-
	4-4	26.64	94.4(94.94)	29.9E-2	-	Hillvalley	4-4	2.14	54.96(59.07)	1.84E0	-
	4-2	76.54	84.94(89.51)	2.52E0			4-2	70.02	94.77(95.59)	39.6E-2	
Musk1	4-3	77.36	83.02(90.21)	2.52E0	-		4-3	97.44	94.41(95.36)	39.3E-2	-
	4-4	62.26	82.76(90.91)	2.72E0	-	Arrhythmia	4-4	29.36	94.25(95.02)	47.3E-2	-
	4-2	203.32	78.86(84.23)	3.15E0			4-2	283.22	98.63(98.87)	11.7E-2	
Madelon	4-3	214.76	76.54(87.56)	3.03E0	-		4-3	321.4	98.45(98.65)	11.7E-2	-
	4-4	181.12	75.31(82.05)	2.61E0	-	Isolet5	4-4	260.58	98.34(98.69)	14.2E-2	-

5.3. Results of PSO(4-2)

Table 5 shows the experimental results of the proposed algorithm and the benchmark techniques. “T-test” shows the result of the T-test between a benchmark technique and PSO(4-2), where “+” (“-”) means that the classification performance of the benchmark technique is significantly better (worse) than that of PSO(4-2). “=” indicates they are similar.

5.3.1. Results of LFS and GSBS

According to Table 5, LFS selected a smaller number of features and achieved a similar or higher classification accuracy than using all features in most cases. GSBS could reduce the number of features, but could not improve the classification performance on many datasets. In most cases, LFS outperformed GSBS in terms of both the number of features and the classification performance. The results indicate that LFS as a forward selection algorithm is more likely to obtain good feature subsets with a small number of features than GSBS (backward selection) because of different starting points. However, in some cases, GSBS achieved better classification performance than LFS.

5.3.2. Results of PSO-No

According to Table 5, PSO-No evolved feature subsets with a small number of features and achieved better classification performance than using all

Table 5: Comparisons Between PSO(4-2) and Benchmark Methods

Dataset	Method	AveNO.	Ave(Best)	StdDev	T-test	Dataset	Method	AveNO.	Ave(Best)	StdDev	T-test
	All	13	76.54		-		All	17	80.95		-
Wine	LFS	7	74.07		-		LFS	8	79.05		-
	GSBS	8	85.19		-	Zoo	GSBS	7	80.0		-
	PSO(1-1)	7.24	95.95(100)	2.2E0	=		PSO(1-1)	8.16	95.47(97.14)	77.3E-2	=
	PSO-No	5.86	95.33(98.77)	2.13E0	=		PSO-No	4.86	95.18(96.19)	51.7E-2	-
	PSO(4-2)	6.84	95.26(98.77)	1.75E0			PSO(4-2)	6.46	95.52(97.14)	71.7E-2	
	All	18	83.86		-		All	24	68.0		-
	LFS	9	83.07		-		LFS	3	68.67		=
Vehicle	GSBS	16	75.79		-	German	GSBS	18	64.33		-
	PSO(1-1)	9.06	85.07(87.01)	86.9E-2	=		PSO(1-1)	12.42	68.47(72)	1.89E0	=
	PSO-No	8.96	85.15(87.01)	88.8E-2	=		PSO-No	11.52	68.37(74.33)	2.2E0	=
	PSO(4-2)	10.16	85.3(87.01)	94.7E-2			PSO(4-2)	12.76	68.47(70.67)	1.45E0	
	All	30	92.98		-		All	34	83.81		-
	LFS	10	88.89		-		LFS	4	86.67		-
WBCD	GSBS	25	83.63		-	Ionosphere	GSBS	30	78.1		-
	PSO(1-1)	11.82	93.34(94.74)	52.2E-2	-		PSO(1-1)	8.74	88.86(93.33)	2.04E0	+
	PSO-No	4.18	93.73(94.74)	84.6E-2	=		PSO-No	8.36	89.26(95.24)	2.27E0	+
	PSO(4-2)	3.46	93.98(94.74)	85.6E-2			PSO(4-2)	3.26	87.27(91.43)	1.84E0	
	All	56	70.0		-		All	60	76.19		-
	LFS	6	90.0		+		LFS	3	77.78		=
Lung	GSBS	33	90.0		+	Sonar	GSBS	48	68.25		-
	PSO(1-1)	23.58	74(90)	7.48E0	-		PSO(1-1)	22.04	78.64(85.71)	3.32E0	=
	PSO-No	11.82	74.8(90)	8.06E0	-		PSO-No	17.66	78.29(85.71)	3.67E0	=
	PSO(4-2)	6.74	78.4(90)	6.44E0			PSO(4-2)	11.24	78.16(85.71)	3.18E0	
	All	90	94.81		+		All	100	56.59		-
	LFS	14	95.06		+		LFS	8	57.69		=
Movement	GSBS	80	93.46		-	Hillvalley	GSBS	90	49.45		-
	PSO(1-1)	37.96	94.51(95.31)	32.8E-2	=		PSO(1-1)	40.54	58.13(60.99)	1.35E0	=
	PSO-No	36.24	94.52(95.31)	34.6E-2	=		PSO-No	40.38	57.93(60.71)	1.32E0	=
	PSO(4-2)	27.46	94.58(95.19)	37.6E-2			PSO(4-2)	12.22	57.77(60.71)	1.47E0	
	All	166	83.92		-		All	279	94.46		-
	LFS	10	85.31		=		LFS	11	94.46		-
Musk1	GSBS	122	76.22		-	Arrhythmia	GSBS	130	93.55		-
	PSO(1-1)	74.26	84.6(91.61)	2.23E0	=		PSO(1-1)	120.22	94.54(95.14)	35.1E-2	-
	PSO-No	71.52	84.76(89.51)	2.27E0	=		PSO-No	98.08	94.51(95.14)	32.9E-2	-
	PSO(4-2)	76.54	84.94(89.51)	2.52E0			PSO(4-2)	70.02	94.77(95.59)	39.6E-2	
	All	500	70.9		-		All	617	98.45		-
	LFS	7	64.62		-		LFS	24	98.34		-
Madelon	GSBS	489	51.28		-	Isolet5	GSBS	560	97.16		-
	PSO(1-1)	235.2	76.93(79.87)	1.54E0	-		PSO(1-1)	283.3	98.59(98.78)	8.79E-2	-
	PSO-No	232.86	76.98(80.64)	1.64E0	-		PSO-No	272.46	98.6(98.8)	10.4E-2	=
	PSO(4-2)	203.32	78.86(84.23)	3.15E0			PSO(4-2)	283.22	98.63(98.87)	11.7E-2	

features in 12 of the 14 cases. PSO-No outperformed PSO(1-1) in terms of the number of features in all cases and the classification performance in 8 of the 14 cases. The reason is that PSO-No considers both the number of features and the classification performance in the fitness function, but the weight for the number of features was much smaller than the classification performance. Therefore, the PSO-No can guarantee the classification performance, but if two feature subsets have the same classification performance, PSO-No will prefer the smaller one while PSO(1-1) will not do this.

According to Table 5, PSO(4-2) evolved feature subsets that selected less than half (or even close to 10% on four datasets) of the available features, but achieved significantly better classification performance than using all features on 13 of the 14 datasets. Only in the Movement dataset, the average classification performance obtained by PSO(4-2) (94.62%) was less, by 0.2%, than that of using all features (94.81%), but the best accuracy (95.19%) was higher than using all features.

5.3.3. Comparing PSO(4-2) with LFS and GSBS

PSO(4-2) usually achieved significantly better or similar classification performance to LFS. The number of features in PSO(4-2) was similar or slightly larger than LFS on the datasets with a relatively small number of features and much larger on the datasets with a large number of features. The main reason is that the selecting mechanism of LFS always selected a very small number of features while the number of features selected by PSO(4-2) increased when the total number of features increased. Comparing PSO(4-2) with GSBS, the number of features in PSO(4-2) was smaller than GSBS in all datasets and the classification performance of PSO(4-2) was significantly better than GSBS on 13 of the 14 datasets. This suggest that PSO(4-2) as a PSO based algorithm can search the solution space more effectively than both LFS and GSB. The initialisation strategy motivated by both forward selection and backward selection can help PSO(4-2) take their advantages to obtain feature subsets with a smaller number of features and better classification performance than LFS and GSBS in most cases.

5.3.4. Comparisons Between PSO(4-2) and PSO(1-1)

According to Table 5, PSO(4-2) selected feature subsets including a smaller number of features and achieved significantly better classification performance than (or similar to) PSO(1-1) on 13 of the 14 datasets except for the Ionosphere dataset, but the number of features in PSO(4-2) was around a quarter of that in PSO(1-1). This suggests that although PSO(1-1) and PSO(4-2) shared the same fitness function (Equation 3), the proposed initialisation strategy and *pbest* and *gbest* updating mechanism can help PSO(4-2) to effectively eliminate the redundant and irrelevant features to obtain a smaller feature subset with similar or significantly better classification performance than PSO(1-1).

5.3.5. Comparisons Between PSO(4-2) and PSO-No

According to Table 5, on 13 of the 14 datasets, the classification performance of PSO(4-2) was similar or significantly better than that of PSO-No. Only on the Ionosphere dataset, the classification performance of PSO(4-2) was slightly worse (by 2%) than PSO-No, but PSO(4-2) further removed 61% of the number of features in PSO-No. In most cases, the number of features in PSO(4-2) was smaller or much smaller than in PSO-No. The number of features in PSO-No was slightly smaller than in PSO(4-2) in some cases, but such datasets have a relatively small total number of features and feature selection is not as important and difficult as on the datasets with a large number of features. In many cases, PSO(4-2) outperformed PSO-No in terms of both the number of features and the classification performance. The reason is that although the weight for the number of features was very small in PSO-No, the reduction of the number of features still influences the classification performance in some cases. In PSO(4-2), the fitness function only includes the classification performance during the whole evolutionary process. This ensures that the reduction of the number of features in PSO(4-2) will not reduce the classification performance. Meanwhile, the proposed mixed initialisation strategy and *pbest* and *gbest* updating mechanism can help PSO(4-2) further remove the irrelevant or redundant features, which in turn may increase the classification performance.

Note that simply increasing the number of iterations cannot help PSO(1-1) and PSO-No achieve the same performance obtained by PSO(4-2). The main reason is that PSO(1-1) does not consider the number of features and PSO-No takes a trade-off between the classification performance and the number of features. PSO(4-2) simulates both forward and backward selection to duplicate their advantages, which helps PSO(4-2) pay more attention to small feature subsets, but does not miss the large feature subsets with high classification performance. Meanwhile, because of the proposed *pbest* and *gbest* updating mechanism, for two feature subsets with the same classification performance, PSO(4-2) will select the smaller one as the new *pbest* or *gbest*. PSO(1-1) using traditional updating mechanism will not do this during the evolutionary training process. Therefore, it is most likely PSO(1-1) and PSO-No can not achieve as good performance as PSO(4-2).

Table 6: Computational Time (minutes)

Method	Wine	Zoo	Vehicle	German	WBCD	Ionosp	Lung	Sonar	Movement	Hillva	Musk1	Arrhy	Madelon	Isolet5
1-1	0.25	0.11	8.08	12.85	3.92	1.35	0.02	0.75	3.32	42.28	10	11.9	871.5	374.4
2-1	0.22	0.06	5.41	5.56	2.36	0.95	0.01	0.31	1.51	14.12	2.63	2.62	100.42	143.45
3-1	0.34	0.1	8.58	15.08	5.18	1.93	0.03	0.98	3.66	47.74	13.71	20.98	1242.78	583.32
4-1	0.33	0.11	8.06	11.98	3.23	0.98	0.02	0.54	2.6	15.82	9.48	8.3	856.53	399.67
1-1	0.25	0.11	8.08	12.85	3.92	1.35	0.02	0.75	3.32	42.28	10	11.9	871.5	374.4
1-2	0.29	0.1	6.65	12.67	2.96	1.34	0.02	0.71	2.82	34.65	10.35	13.23	968.47	430.8
1-3	0.25	0.07	5.38	8.11	2.15	0.89	0.01	0.43	2.21	26.37	5.3	7.91	445.6	212.19
1-4	0.25	0.07	6.31	7.07	2.1	1.04	0.01	0.48	1.8	16.09	5.41	4.49	372.7	166.98
PSO-No	0.24	0.08	6.63	10.68	2.45	1.13	0.01	0.59	2.83	34.83	8.53	11.19	813.61	362.17
4-2	0.31	0.11	8.49	13.05	2.88	1.03	0.01	0.54	2.72	20.17	10.34	8.88	792.25	364.07

6. Further Discussions

6.1. Analysis on Computational Time

Table 6 shows the average computational time used by different methods in 50 independent runs. All the algorithms used in the experiments are wrapper based feature selection approaches. Therefore, most of their computational time was spent on the fitness evaluation, i.e calculating the classification performance of the selected features.

For the algorithms using a new initialisation strategy and traditional *pbest* and *gbest* updating mechanism, from Table 6, it can be seen that PSO(2-1) with the small initialisation strategy used the shortest time, PSO(3-1) with the large initialisation strategy used the longest time. PSO(4-1) used slightly less time than the traditional PSO(1-1). The main reason for the time differences is that a larger number of features needs longer time for classification. PSO(2-1) (or PSO(3-1)), which usually selected a smallest (or largest) number of features, used the shortest (or the longest) time.

For the algorithms using the traditional initialisation strategy and a new *pbest* and *gbest* updating mechanism, from Table 6, it can be seen that PSO(1-3) and PSO(1-4) used less computational time than PSO(1-1) and PSO(1-2) and the reason is that they selected a smaller number of features. On 10 of the 14 datasets, PSO(1-4) used less or the same time as PSO(1-3). On 8 datasets, PSO(1-1) used the longer time than PSO(1-2). The time used by these four algorithms also follow the same observations mentioned above, which is the algorithms selecting more features used longer time. However, On the Musk1, Arrhy, Madelon and Isolet5 datasets, PSO(1-2) selected a similar or slightly smaller number of features than PSO(1-1), but used longer time. This is because when updating *pbest* and *gbest*, PSO(1-2) has a procedure of comparing the number of features while PSO(1-1) does not.

Table 6 also shows that PSO(4-2) took less time than PSO-No on six datasets with a relatively large number of features. Although PSO-No took less time on seven datasets, five of them are datasets with a small number of features (≤ 30). PSO(4-2), which usually selected a smaller number of features, used less time than PSO-No. Meanwhile, PSO-No took less time than PSO(1-1) because the size of the feature subsets evolved by PSO-No was smaller than PSO(1-1) during the evolutionary training process. For the benchmark techniques, LFS usually used less time than the other methods because the forward selection strategy starts with a small number of features. GSBS costs less time than the other PSO based algorithms on the datasets with a small number of features, but more time on the datasets with a large number of features, such as the Madelon and Isolet5 datasets. The reason is that GSBS starts with the full set of features, which needs much longer time for each evaluation. The number of evaluations in GSBS substantially increased on such large datasets while the number of evaluations in PSO based algorithms still remains the same.

6.2. Comparisons with Other Existing Methods

In order to further test the performance of PSO(4-2), we compare its performance with four recently published methods from other researchers [8, 54, 29, 4]. The four papers from other researchers used datasets with a relatively small number of features, which was less than 100. Therefore, we choose only the six datasets that are used in both our experiments, and the results are shown in Table 7. We also compare PSO(4-2) with a recent method from us [47] and the results of the ten datasets that are used in this paper and our previous paper [47] are also shown in Table 7.

Compared with the methods from others, the results show that PSO(4-2) achieved better classification performance than two other methods on the Vehicle and Ionosphere datasets, *slightly* worse classification performance on the Wine, German and WBCD datasets. On the Sonar dataset, PSO(4-2) achieved better classification performance than one method, but worse than another two methods. However, for all the datasets, PSO(4-2) selected a smaller or much smaller number of features, such as on the Sonar dataset, the number of features selected by PSO(4-2) was only one third of the other three methods. The results also show that the performance of the proposed algorithms are competitive to the state-of-art methods, but since the number of features was smaller, the computational time was typically shorter.

Comparing PSO(4-2) with our previous method (named PSO2Stage) [47], Table 7 shows that PSO(4-2) selected a smaller number of features than

Table 7: Comparisons with Other Existing Methods

Dataset	Method	Classifier	AveNO. of Features	Ave Accuracy	Best Accuracy
Wine	Lin and Chen [8] (2009)	LDA	12.3		100
	Chuang et al. [4](2011)	KNN	8	99.44	99.44
	Boubezoul and Paris [54] (2012)	KNN	11.9	96.79	99.31
	PSO2Stage [47] (2012)	KNN	5.1	96.94	98.77
	PSO(4-2)	KNN	6.84	95.26	98.77
Vehicle	Lin and Chen [8] (2009)	LDA	15.5		79.4
	Chuang et al. [4](2011)	KNN	12	75.06	75.06
	PSO2Stage [47] (2012)	KNN	7.3	84.47	85.04
	PSO(4-2)	KNN	10.16	85.3	87.01
German	Lin and Chen [8] (2009)	LDA	22.4		75.6
	PSO2Stage [47] (2012)	KNN	8.62	68.93	73.67
	PSO(4-2)	KNN	12.76	68.47	70.67
WBCD	Chuang et al. [4](2011)	KNN	14.2	98.17	98.24
	PSO2Stage [47] (2012)	KNN	6.68	92.98	92.98
	PSO(4-2)	KNN	3.46	93.98	94.74
Ionosphere	Lin and Chen [8] (2009)	LDA	21.7		92.2
	Esseghir et al. [29] (2010)	ANN	13.79	87.52	88.47
	PSO2Stage [47] (2012)	KNN	8.9	89.52	93.33
	PSO(4-2)	KNN	3.26	87.27	91.43
Sonar	Lin and Chen [8] (2009)	LDA	38.1		90.5
	Esseghir et al. [29] (2010)	ANN	30.73	73.59	75.58
	Chuang et al. [4](2011)	KNN	30.2	96.92	97.12
	PSO(4-2)	KNN	11.24	78.16	85.71
Lung	PSO2Stage [47] (2012)	KNN	22.22	73.25	80
	PSO(4-2)	KNN	6.74	78.4	90
Hillvalley	PSO2Stage [47] (2012)	KNN	37.1	57.61	60.44
	PSO(4-2)	KNN	12.22	57.77	60.71
Musk1	PSO2Stage [47] (2012)	KNN	80.72	85.7	89.51
	PSO(4-2)	KNN	76.54	84.94	89.51
Madelon	PSO2Stage [47] (2012)	KNN	241.35	77.34	79.62
	PSO(4-2)	KNN	203.32	78.86	84.23
Isolet5	PSO2Stage [47] (2012)	KNN	302.55	98.57	98.77
	PSO(4-2)	KNN	283.22	98.63	98.87

PSO2Stage on seven of the 10 datasets and better classification performance on six datasets. PSO(4-2) achieved competitive results with PSO2Stage on datasets with a relatively small number of features, but on the five datasets that have a relatively large number of features (Lung with 56 features, Hill-valley with 100 features, Musk1 with 166 features, Madelon with 500 features and Isolet5 with 614 features), PSO(4-2) selected a smaller number of features than PSO2Stage in all cases, and achieved better classification performance than PSO2Stage on four of the five cases. Although all the datasets are commonly used benchmark problems in the literature, the performance of an algorithm on datasets with a larger number of features is clearly more important than on datasets with a smaller number of features. On the five datasets with a large number of features, PSO(4-2) outperforms PSO2Stage

Table 8: Datasets

Dataset	No. of Features	No. of Classes	No. of Instances	Dataset	No. of Features	No. of Classes	No. of Instances
Spect	22	2	267	Statlog	36	6	6435
Statlog	36	6	6435	Soybean Large	35	19	307
Waveform	40	3	5000	Splice	60	3	3190

in terms of the number of features in all cases and the classification performance in almost all cases. A smaller number of selected features will result in shorter computational time in PSO(4-2) than in PSO2Stage. Therefore, PSO(4-2) is clearly better than PSO2Stage on datasets with a large number of features.

6.3. Experiments on More Datasets

Since the datasets used in Section 5 are mainly continuous datasets, to further test the performance of the proposed algorithms, a set of experiments have been conducted on additional six discrete datasets, which are shown in Table 8. The results of PSO(1-1), PSO-No and PSO(4-2) are shown in Table 9. The other results show a similar pattern, so will not be presented here.

According to Table 9, it can be seen that all the three PSO based methods significantly reduced the number of features and achieved better similar or better classification performance than using all features. The classification performance of PSO(4-2) was at least as good as PSO(1-1) and PSO-No, which was significantly better on three datasets and similar on another three datasets. However, the PSO(4-2) achieved a smaller number of features than PSO(1-1) on only three datasets. On five of the six datasets, PSO-No selected a smaller number of features than PSO(1-1) and PSO(4-2). Since the number of features in Table 9 is less or equal to 60, comparing Table 9 with the results of datasets having a small number of features, similar patterns can be observed, where PSO(4-2) achieved better classification performance while PSO-No selected a smaller number of features.

7. Conclusions

The goal of this paper was to propose a PSO based feature selection approach to selecting a smaller number of features and achieving similar or even better classification performance than using all features. The goal was successfully achieved by developing three new initialisation strategies motivated by forward selection and backward word selection, and three new *pbest* and *gbest* updating mechanisms considering both the number of feature and

Table 9: Experimental Results on Other Datasets

Dataset	Method	AveNO.	Ave(Best)	StdDev	T-test	Dataset	Method	AveNO.	Ave(Best)	StdDev	T-test
Spect	All	22	65.0		=	Leddisplay	All	24	89.0		-
	PSO(1-1)	7.6	63.95(70)	4.35E0	=		PSO(1-1)	9.12	100(100)	1.82E-2	=
	PSO-No	7.6	63.68(71.25)	4.45E0	=		PSO-No	6.56	100(100)	0E0	=
	PSO(4-2)	6.02	64.32(72.5)	5.16E0			PSO(4-2)	7.16	99.99(100)	5.61E-2	
Statlog	All	36	90.0		-	Soybeanlarge	All	35	89.0		-
	PSO(1-1)	24.38	96.49(96.77)	14.1E-2	-		PSO(1-1)	22.1	99.13(99.38)	16.2E-2	-
	PSO-No	24.06	96.47(96.84)	16E-2	-		PSO-No	20.9	99.11(99.44)	19.6E-2	-
	PSO(4-2)	29.56	96.61(96.84)	10.8E-2			PSO(4-2)	25.42	99.19(99.49)	16.4E-2	
Waveform	All	40	80.0		-	Splice	All	60	60.0		-
	PSO(1-1)	22.14	86.79(88.18)	65.5E-2	=		PSO(1-1)	18.7	80.87(91.15)	3.46E0	-
	PSO-No	21.82	86.75(88.13)	59.4E-2	=		PSO-No	19.04	80.72(91.15)	3.46E0	-
	PSO(4-2)	28.9	86.62(87.96)	61.5E-2			PSO(4-2)	5.42	89.22(91.99)	2.04E0	

the classification performance to overcome the limitation of the traditional updating mechanism.

A novel algorithm (named PSO(4-2)) using the most promising initialisation strategy and the most promising updating mechanism was compared with two traditional feature selection methods, a standard PSO based method, and a PSO based algorithm (PSO-No) with a single fitness function combining the two objectives of maximising the classification performance and minimising the number of features. The results show that PSO with new initialisation strategies or/and new updating mechanisms could automatically evolve a feature subset with a smaller number of features and higher classification performance than using all features and also the standard PSO based algorithm. PSO(4-2) achieved significantly better classification performance than using all features, the two traditional methods and the standard PSO based algorithm. In most cases, this proposed algorithm achieved similar or better classification performance than PSO-No, but using a smaller or much smaller number of features and shorter computational time.

This study shows that PSO is an effective search technique for feature selection problems. The experiments show that the initialisation strategy can influence the number of features selected and the computational time. More importantly, this work also highlights that *pbest* and *gbest* are important factors to guide the search behaviour of a PSO algorithm. The updating mechanism of *pbest* and *gbest* can be utilised to optimise one objective (i.e. to reduce the number of features) in the problem without deteriorating the other objective (to maximise classification performance).

In the future, we will conduct further work on the updating mechanisms (e.g. the position and velocity updating equations) in PSO to solve feature selection in order to further reduce the number of features, increase the classification performance and reduce the computational time. We also intend to propose a PSO based multi-objective feature selection approach in order to simultaneously maximise the classification performance and minimise the number of features. Meanwhile, we also intend to investigate whether using *one* learning algorithm in a wrapper feature selection approach can select a good or near-optimal feature subset for *other* learning algorithms for classification tasks. We will also investigate the use of PSO for feature selection on datasets with a huge number of features (e.g. over 10 000).

References

[1] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, The Journal of Machine Learning Research 3 (2003) 1157–1182.

[2] M. Dash, H. Liu, Feature selection for classification, Intelligent Data Analysis 1 (1997) 131–156.

[3] R. Kohavi, G. H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1997) 273–324.

[4] L. Y. Chuang, S. W. Tsai, C. H. Yang, Improved binary particle swarm optimization using catfish effect for feature selection, Expert Systems with Applications 38 (2011) 12699–12707.

[5] I. A. Gheyas, L. S. Smith, Feature subset selection in large dimensionality domains, Pattern Recognition 43 (2010) 5–13.

[6] J. Kennedy, R. Eberhart, Particle swarm optimization, in: International Conference on Neural Networks, volume 4, IEEE, 1995, pp. 1942–1948.

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

[8] S.-W. Lin, S.-C. Chen, PSOLDA: A particle swarm optimization approach for enhancing classification accuracy rate of linear discriminant analysis, Appled Soft Computing 9 (2009) 1008–1015.

[9] A. Gutierrez, M. Lanza, I. Barriuso, L. Valle, M. Domingo, J. Perez, J. Basterrechea, Comparison of different PSO initialization techniques

for high dimensional search space problems: A test with FSS and antenna arrays, in: The 5th European Conference on Antennas and Propagation (EUCAP'11), IEEE, 2011, pp. 965–969.

[10] B. Xue, M. Zhang, W. Browne, Novel initialisation and updating mechanisms in pso for feature selection in classification, in: Applications of Evolutionary Computation, volume 7835 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2013, pp. 428–438.

[11] K. Kira, L. A. Rendell, A practical approach to feature selection, Assorted Conferences and Workshops (1992) 249–256.

[12] H. Almuallim, T. G. Dietterich, Learning boolean concepts in the presence of many irrelevant features, Artificial Intelligence 69 (1994) 279–305.

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

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

[15] S. Stearns, On selecting features for pattern classifier, in: Proceedings of the 3rd International Conference on Pattern Recognition, IEEE, Coronado, CA, 2003, pp. 71–75.

[16] Z. X. Zhu, Y. S. Ong, M. Dash, Wrapper-filter feature selection algorithm using a memetic framework, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 37 (2007) 70–76.

[17] K. Neshatian, M. Zhang, Pareto front feature selection: using genetic programming to explore feature space, in: Proceedings of the 11th Annual conference on Genetic and evolutionary computation (GECCO'09), New York, NY, USA, pp. 1027–1034.

[18] D. Muni, N. Pal, J. Das, Genetic programming for simultaneous feature selection and classifier design, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 36 (2006) 106–117.

[19] K. Neshatian, M. Zhang, P. Andreae, A filter approach to multiple feature construction for symbolic learning classifiers using genetic programming, IEEE Transactions on Evolutionary Computation 16 (2012) 645–661.

[20] K. Neshatian, M. Zhang, Using genetic programming for context-sensitive feature scoring in classification problems, Connection Science 23 (2011) 183–207.

[21] K. Neshatian, M. Zhang, Improving relevance measures using genetic programming, in: European Conference on Genetic Programming (EuroGP 2012), volume 7244 of *Lecture Notes in Computer Science*, Springer, 2012, pp. 97–108.

[22] R. Jensen, Q. Shen, Finding rough set reducts with ant colony optimization, in: Proceedings of the 2003 UK Workshop on Computational Intelligence, pp. 15–22.

[23] L. Ke, Z. Feng, Z. Ren, An efficient ant colony optimization approach to attribute reduction in rough set theory, Pattern Recognition Letters 29 (2008) 1351–1357.

[24] R. A. A. M. Mishra, D., T. Jena., Rough aco: A hybridized model for feature selection in gene expression data, International Journal of Computer Communication and Technology 1 (2009) 85–98.

[25] Y. Chen, D. Miao, R. Wang, A rough set approach to feature selection based on ant colony optimization, Pattern Recognition Letters 31 (2010) 226 – 233.

[26] L. Y. Chuang, H. W. Chang, C. J. Tu, C. H. Yang, Improved binary PSO for feature selection using gene expression data, Computational Biology and Chemistry 32 (2008) 29– 38.

[27] X. Wang, J. Yang, X. Teng, W. Xia, R. Jensen, Feature selection based on rough sets and particle swarm optimization, Pattern Recognition Letters 28 (2007) 459–471.

[28] K. Iswandy, A. Koenig, Feature-level fusion by multi-objective binary particle swarm based unbiased feature selection for optimized sensor

system design, in: IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 365 –370.

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

[30] Y. Marinakis, M. Marinaki, G. Dounias, Particle swarm optimization for pap-smear diagnosis, Expert Systems with Applications 35 (2008) 1645 – 1656.

[31] C. L. Huang, J. F. Dun, A distributed PSO-SVM hybrid system with feature selection and parameter optimization, Application on Soft Computing 8 (2008) 1381–1391.

[32] R. Fdhila, T. Hamdani, A. Alimi, Distributed MOPSO with a new population subdivision technique for the feature selection, in: 5th International Symposium on Computational Intelligence and Intelligent Informatics (ISCIII'11), IEEE, 2011, pp. 81–86.

[33] K. Parsopoulos, M. Vrahatis, Initializing the particle swarm optimizer using the nonlinear simplex method, Advances in intelligent systems, fuzzy systems, evolutionary computation 216 (2002) 1–6.

[34] J. A. Nelder, R. Mead, A simplex method for function minimization, The Computer Journal 7 (1965) 308–313.

[35] M. Richards, D. Ventura, Choosing a starting configuration for particle swarm optimization, in: International Joint Conference on Neural Networks, volume 3, IEEE, 2004, pp. 2309–2312.

[36] H. Jabeen, Z. Jalil, A. R. Baig, Opposition based initialization in particle swarm optimization (o-pso), in: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference, GECCO'09, ACM, 2009, pp. 2047–2052.

[37] H. Wang, Z. Wu, J. Wang, X. Dong, S. Yu, C. Chen, A new population initialization method based on space transformation search, in: Fifth International Conference on Natural Computation (ICNC'09), volume 5, IEEE Computer Society, 2009, pp. 332–336.

[38] H. Wang, H. Li, Y. Liu, C. Li, S. Zeng, Opposition-based particle swarm algorithm with cauchy mutation, in: IEEE Congress on Evolutionary Computation (CEC'07), IEEE, 2007, pp. 4750–4756.

[39] H.-C. Chen, O.-C. Chen, Particle swarm optimization incorporating a preferential velocity-updating mechanism and its applications in iir filter design, in: International Conference on Systems, Man and Cybernetics (SMC'06), volume 2, IEEE, 2006, pp. 1190–1195.

[40] J. Qi, Y. Ding, An improved particle swarm optimization with an adaptive updating mechanism, in: Proceedings of the Second international conference on Advances in swarm intelligence - volume Part I, ICSI'11, Springer-Verlag, 2011, pp. 130–137.

[41] J. Kennedy, 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, pp. 4104–4108.

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

[43] M. Khanesar, M. Teshnehlab, M. Shoorehdeli, A novel binary particle swarm optimization, in: Mediterranean Conference on Control Automation (MED'07), IEEE, 2007, pp. 1–6.

[44] S. Luke, L. Panait, Lexicographic parsimony pressure, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'02), Morgan Kaufmann, 2002, pp. 829–836.

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

[46] R. Caruana, D. Freitag, Greedy attribute selection, in: International Conference on Machine Learning, Morgan Kaufmann, 1994, pp. 28–36.

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

[48] X. Wang, J. Yang, R. Jensen, X. Liu, Rough set feature selection and rule induction for prediction of malignancy degree in brain glioma, Computer Methods and Programs in Biomedicine 83 (2006) 147156.

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

[50] T. Hastie, R. Tibshirani, J. Friedman, J. Franklin, The elements of statistical learning: data mining, inference and prediction, The Mathematical Intelligencer 27 (2005) 83–85.

[51] M. Clerc, J. Kennedy, The particle swarm– explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6 (2002) 58–73.

[52] B. Xue, M. Zhang, W. N. Browne, Multi-objective particle swarm optimisation (PSO) for feature selection, in: Genetic and Evolutionary Computation Conference (GECCO'12), pp. 81–88.

[53] B. Xue, M. Zhang, W. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics (2012) 1–16.

[54] A. Boubezoul, S. Paris, Application of global optimization methods to model and feature selection, Pattern Recognition 45 (2012) 3676 – 3686.