Particle Swarm Optimisation for Feature Selection: A Size-Controlled Approach

Tony Butler-Yeoman, Bing Xue, and Mengjie Zhang

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

Abstract

Feature selection is a preprocessing step in classification tasks, which can reduce the dimensionality of a dataset and improve the classification accuracy and efficiency. However, many current feature selection algorithms select an unnecessarily large feature subsets, particularly on datasets with high dimensionality. This paper proposes a new particle swarm optimisation (PSO) based feature selection approach, where a new method is proposed to find the possible smallest size that potentially good feature subsets can have to guide the PSO algorithm to search for smaller feature subsets. The proposed algorithm is examined and compared with original PSO based feature selection and two typical feature selection method on twelve benchmark datasets of varying difficulty. The experimental results show that the proposed algorithm successfully further reduces the dimensionality of the dataset over original PSO and one of the conventional method, and maintains or even increases the classification performance in most cases. The proposed algorithm selects more features than the other conventional method, but achieves better classification performance in most cases, which shows that the proposed algorithm can balance the classification performance and the number of features in most cases. Furthermore, the proposed algorithm also shows better efficiency and consistency performance in terms of selecting consistent features across different stochastic runs.

Keywords: Classification, Feature Selection, Particle Swarm Optimisation

1 Introduction

In many classification tasks, the datasets often include a large number of features, so as to represent the target concept as completely as possible. However, as many features are redundant or irrelevant, this results in noise in the dataset that reduces the performance of many classification algorithms (Dash and Liu, 1997). Furthermore, the large number of features contribute to the "curse of dimensionality" (Dash and Liu, 1997; Guyon and Elisseeff, 2003), which is one of the major obstacles in classification. Feature selection is the process of choosing a subset of the relevant features from a large number of original features. The chosen feature subset should be small and accurately describe the target concept. As a preprocessing step, feature selection is a practical and well-known solution to the problems of high-dimensionality data, resulting in a fast and high-performing classification process.

Based on the evaluation criteria, feature selection algorithms are generally classified into two categories: *filter* approaches and *wrapper* approaches (Dash and Liu, 1997; Guyon and Elisseeff, 2003). Their main difference is that wrapper approaches include a classification/learning algorithm in the feature subset evaluation step. The classification algorithm is used to evaluate the goodness (i.e. the classification performance) of the selected features. A filter feature selection process is independent of any classification algorithm. Filter algorithms are often computationally less expensive and more general than wrapper algorithms. However, filters ignore the performance of the selected features on a classification algorithm while wrappers evaluate the feature subsets based on the classification performance, which usually results in better performance achieved by wrappers than filters for a particular classification algorithm (Dash and Liu, 1997; Mitra et al., 2002; Liu and Zhao, 2009; Liu et al., 2010).

Feature selection is a challenging task since the space of possible feature subsets is the power set of the features, hence there are 2^n possible feature subsets for a dataset with n features. If all features were completely independent, an efficient greedy algorithm could search this space fast by identifying and removing irrelevant features, leaving only the most useful features. However, since features are often interacting with each other, which leads to that individually relevant features may become redundant and individually weakly relevant features may become highly relevant when combined with other features (Guyon and Elisseeff, 2003). Therefore, a powerful global search algorithm that can consider all features at the same time is needed to find the optimal feature subset(s). Although different types of search techniques have been applied to features selection (Guyon and Elisseeff, 2003; Liu et al., 2010), existing approaches still suffer from the problem of being stagnation in local optima. Evolutionary computation (EC) techniques are well-known for their promising search ability, and have been applied to feature selection tasks with some success, such as genetic algorithms (GAs) (Zhu et al., 2007; Lin et al., 2014), genetic programming (Muni et al., 2006; Neshatian and Zhang, 2012; Espejo et al., 2010; Purohit et al., 2010), differential evolution (DE) (Bharathi P T, 2014*a*,*b*), and particle swarm optimisation (PSO) (Boubezoul and Paris, 2012; Xue et al., 2013*b*; Vieira et al., 2013). Compared with GAs and GP, PSO is easier to implement, has fewer parameters, computationally less expensive, and can converge more quickly (Engelbrecht, 2007). However, the search of the original PSO algorithm often selects a relatively large number of features, which may include redundant features. Further reduction on the number of features and analysis on the consistency of features selected across different stochastic runs are still needed.

1.1 Goals

The goal of this paper is to develop a new PSO based wrapper feature selection approach that can select a significantly smaller number of features and maintain the classification performance over using the original feature set and the features selected by standard PSO. To achieve this goal, a new method is proposed to find a target size which roughly indicates the smallest size of feature subsets that can achieve the highest or close to the highest classification performance. The target size is used to guide the search of PSO in addition to the original PSO search mechanism that focuses mainly on the finding the highest classification performance. Specifically, we will investigate:

- whether incorporating the target size method into PSO for feature selection can further reduce the number of features selected over standard PSO,
- whether the classification performance of PSO for feature selection can be maintained or even improved in the new approach,
- whether the proposed approach can outperform conventional feature selection methods, and
- whether the proposed approach can more consistently select key informative features than standard PSO.

2 Background

2.1 Particle Swarm Optimisation

Particle Swarm Optimisation is an evolutionary computation technique inspired by social behaviour proposed by Kennedy and Eberhart (Kennedy and Eberhart, 1995; Shi and Eberhart, 1998). PSO maintains a population of *particles*, called a *swarm*, each of which encodes a candidate solution in the search space. PSO initialises each particle in the swarm to a random position in the space, and iterates the position of each particle based on the experience of the particle and its neighbours. The position of particle i is represented by a vector, $x_i = (x_{i,1}, \ldots, x_{i,n})$, where n is the dimension of the search space. The velocity is represented by a similar vector $v_i = (v_{i,1}, \ldots, v_{i,n})$, where each component of the vector is limited to a predefined range $[-v_{max}, v_{max}]$. The best previous position (according to some fitness function) of particle i is recorded as the personal best, $pbest_i = (p_{i,1} \ldots p_{i,n})$, and the best position found by the population as a whole is recorded as the global best, $gbest = (g_1 \ldots g_n)$. At each iteration of the algorithm, PSO updates the velocity and position of each particle according to the following equations:

$$x_{i,d}^{t+1} = x_{i,d}^{t} + v_{i,d}^{t+1} \tag{1}$$

$$\begin{aligned} v_{i,d}^{t+1} &= w \cdot v_{i,d}^{t} + c_1 \cdot r_{1,i} \cdot (p_{i,d} - x_{i,d}^{t}) \\ &\quad + c_2 \cdot r_{2,i} \cdot (g_d - x_{i,d}^{t}) \end{aligned} \tag{2}$$

Here $0 < d \leq n$ denotes the component of the position or velocity vector, and t represents the t-th iteration of the algorithm. w is a predefined constant for the inertia weight, and c_1 and c_2 are predefined acceleration constants. Each $r_{1,i}$ and $r_{2,i}$ are random values uniformly distributed over $[0,1]$. This description of PSO is applicable to real-valued search spaces. However, feature selection, along with many other problems, occurs in a discrete search space and requires a modified algorithm. Binary PSO (Kennedy and Eberhart, 1997) is such an algorithm. In binary PSO, the values of the components of all position vectors (x_i, $pbest_i$, and $gbest_i$) are restricted to 0 or 1. Equation (2) is still used to update the velocity, each component of which now indicates the probability of the corresponding component in the position vector being 1. A sigmoid function $s(v_{i,d})$ is used to transform the components of the velocity into a unit range. Binary PSO updates the position of each particle according to the following equation:

$$x_{i,d} = \begin{cases} 1, & rand() < s(v_{i,d}) \\ 0, & otherwise \end{cases} \tag{3}$$

where

$$s(x) = \frac{1}{1 + e^{-x}}$$

2.2 Related Work

The following sections survey a relevant selection of recent and prominent work on feature selection.

2.2.1 Traditional Algorithms

FOCUS (Almuallim and Dietterich, 1994) and Relief (Kira and Rendell, 1992) are two founding feature selection algorithms. Each takes a filter approach and so a learning system is not used in evaluation. FOCUS exhaustively searches the space of feature subsets to find a small subset that accurately describes the target concept. This is highly effective, but of course exhaustive search is infeasible on large feature spaces. Relief does not perform exhaustive search, instead it scores each feature individually based on its relevance to the class label and selects a set of the best scoring features. This is much more efficient, but does not take into account feature interaction (Kononenko, 1994); this could lead to, for example, ignoring features that are only useful in combination, or selecting two highly redundant features.

Sequential Forward Selection (SFS) (Whitney, 1971) and Sequential Backward Selection (SBS) (Marill and Green, 1963) are two more widely-known algorithms, both using a wrapper approach. These perform a greedy search in the feature space, with SFS (SBS) starting with an empty (full) feature set and iteratively selecting features to add (remove). Because this is a greedy search, both algorithms are susceptible into local optima. To overcome this and other issues, the ‘plus-l-take-away-r’ method (Stearns, 1976) and sequential floating algorithms (SFFS and SBFS) (Pudil et al., 1994) were proposed, which are variants on SFS and SBS.

2.2.2 Feature Selection with PSO

Evolutionary computation techniques, including GAs (Lin et al., 2014) and GP (Neshatian and Zhang, 2012), have been broadly applied to feature selection problems. For brevity, this section will focus on using PSO for feature selection (Cervante et al., 2012; Xue et al., 2014; Lane et al., 2013, 2014; Xue et al., 2015; Nguyen et al., 2014; Tran et al., 2014).

Both continuous PSO and binary PSO have been used for both filter and wrapper, single objective and multi-objective feature selection. A number of new PSO algorithms have been proposed to improve performance on feature selection problems, including initialisation strategies, representation, fitness functions, and the search mechanisms. Xue et al. (Xue et al., 2013*a*) developed a new initialisation strategy to mimic the typical forward and backward feature selection methods in the PSO search process, which showed that good initialisation significantly increased the performance of PSO for feature selection. There are only a few works on developing new representations in PSO for feature selection. The typical representation has been slightly modified to simultaneously perform feature selection and parameter optimsation of a classification algorithm, mostly optimising the parameters in the kernel functions of SVMs (Lin et al., 2008; Huang and Dun, 2008; Vieira et al., 2013; Boubezoul and Paris, 2012). The length of the new representation is equal to the total number of features and parameters. The representation was encoded in three different ways, being continuous encoding (Lin et al., 2008), binary encoding (Vieira et al., 2013), and a mixture of binary and continuous encoding (Huang and Dun, 2008; Boubezoul and Paris, 2012). Since PSO was originally proposed for continuous optimisation, continuous encoding performed better than the other two encoding schemes. Lane et al. (Lane et al., 2013) proposed the use of PSO and statistical clustering (which groups similar features into the same cluster) for feature selection, where a new representation was proposed to incorporate statistical feature clustering information during the search process of PSO. In the new representation, features from the same cluster were arranged together and only a single feature was selected from each cluster. The proposed algorithm was shown to be able to significantly reduce the number of features.

Learning from neighbours' experience, i.e. social interaction through *gbest*, and learning from each individual's own experience through *pbest*, are the key ideas in PSO. Chuang et al. (Chuang et al., 2008) developed a *gbest* resetting mechanism by including zero features in order to guide the swarm to search for small feature subsets. Xue et al. (Xue et al., 2013*a*) considered the number of features when updating *pbest* and *gbest* during the search process of PSO, which could further reduce the number of features over the traditional updating *pbest* and *gbest* mechanism without deteriorating the classification performance.

The fitness function plays an important role in PSO for feature selection. Many existing works used only the classification performance as the fitness function (Liu et al., 2011; Zhang and Hu, 2005; Tang et al., 2005; Yong et al., 2015), which led to relatively large feature subsets. However, most of the fitness functions used different ways to combine both the classification performance and the number of features into a single fitness function (Huang and Dun, 2008; Fdhila et al., 2011; Ramadan and Abdel Kader, 2007). However, it is difficult to determine in advance the optimal balance between them without *a priori* knowledge. Most of the algorithms select a relatively large number of features.

Algorithm 1 Size-Controlled PSO Feature Selection

```
divide Dataset into a training set and a test set
randomly initialise position and velocity of each particle
while the stopping criterion is not met do
    evaluate the fitness (classification performance) of each particle on the training set
    for each particle $p$ do
        update the $pbest$ of $p$
        update the $gbest$ of $p$
    end for
    for each particle $p$ do
        update the velocity of $p$ according to Equation (4)
        update the position of $p$ according to Equation (1)
    end for
end while
Test process:
calculate the classification accuracy of the features selected by $gbest$ on the test set
return the position of $gbest$ (the selected feature subset)
return the training and test classification accuracies
```

3 Proposed Approach

3.1 Size-Controlled PSO for Feature Selection

In PSO for feature selection, PSO focuses on searching for the best classification performance according to the fitness function. However, since the search space is huge in most cases, besides the classification performance, PSO needs further guidance to search towards a feature subset with not only high classification accuracy, but also a small size. Therefore, we propose a size-controlled PSO for feature selection, where the search of PSO is also guided by a *target* size T. The influence of T is produced through the velocity updating equation, which is shown by Equation (4).

$$\begin{aligned} v_{i,d}^{t+1} &= w \cdot v_{i,d}^{t} \\ &+ c_1 \cdot r_{1,i} \cdot (p_{i,d} - x_{i,d}^{t}) \\ &+ c_2 \cdot r_{2,i} \cdot (g_d - x_{i,d}^{t}) \\ &+ c_3 \cdot r_{3,i} \cdot s(T - |p|) \end{aligned} \tag{4}$$

where,

$$s(x) = \frac{1}{1 + e^{-x}}$$

where $|p|$ shows the number of features selected by particle i, c_3 is a third acceleration constant, and each $r_{3,i}$ is a random value uniformly distributed in $[0, 1]$.

Algorithm 1 shows the pseudo-code for the size-controlled PSO. The position of each particle is still updated using Equation (1). To implement this size-controlled PSO, the key issue is how to determine the target size T.

Algorithm 2 Sequential Random Target Size Optimisation

```
1:  classification accuracy indicates how good a feature subset is;
2:  $Accuracy''$ indicates how good an integer $n$ is;
3:  $d$: the dimensionality of the data;
4:  initialise $range \leftarrow [1, d]$.
5:  Start:
6:  for iteration $i$ from 1 to maximum iterations do
7:      randomly sample a set of $N$ integers and their values are within $range$, i.e. randomly sample a set of candidate sizes;
8:      for each candidate integer $n$ from set $N$ do
9:          randomly sample $s$ different feature subsets and each subset contains $n$ features;
10:         evaluate the classification accuracies of the $s$ subsets;
11:         find the highest classification accuracy and assign it as the $Accuracy''$ of the candidate size $n$;
12:     end for
13:     $topAcc \leftarrow$ the best $Accuracy''$ so far regardless of size;
14:     find all possible candidate sizes with $Accuracy'' \in [topAcc - toleranceAcc]$, and assign the smallest candidate size to $bestSize$;
15:     $range \leftarrow [1, \frac{2*R*d}{i}]$;
16:     centre $range$ on $bestSize$;
17:     if $bestSize$ is not changed then
18:         exit;
19:     end if
20: end for
21: return $bestSize$ as target size $T$.
```

3.2 Sequential Random Target Size Optimisation

In this section, we propose a sequential random target size optimisation (SRTSO) method to find the target size T that the size-controlled PSO requires.

The Pseudocode of SRTSO can found in Algorithm 2. SRTSO aims to find a good size ($bestSize$) with the expectation that the feature subset(s) containing $bestSize$ features can achieve the optimal or near optimisal classification performance. Meanwhile, $bestSize$ is the smallest size that can achieve such good classification performance. In SRTSO, there are two evaluation criteria: (1) the classification accuracy shows how good a feature subset is; (2) $Accuracy''$ shows how good the size n is, and since SRTSO generates multiple feature subsets that contain n features, the best classification accuracy among them is used as the $Accuracy''$ value.

SRTSO follows an iteratively searching process. In each iteration, a set of N integers (candidate sizes) are randomly generated, where all integers are within $range$, Line 7. $range$ is initialised as $[1, d]$ and dynamically changes during the search as $range = [1, \frac{2Rd}{i}]$, where R is a constant value, d is the dimensionality and i means the ith iteration of SRTSO. $\frac{2Rd}{i}$ can ensure that the candidate size that SRTSO searches for becomes smaller along with the search process, i.e. the increase of i. Furthermore, $range$ is also determined/adjusted by the classification performance in each iteration through centering it around a value called $bestSize$. $bestSize$ is calculated from Line 8 to 14. $bestSize$ shows the smallest sizes which have $Accuracy'' \in [topAcc{-}toleranceAcc]$, where $topAcc$ is the highest classification SRTSO has found so far and $toleranceAcc$ is a very small percentage. Using the constraint of $Accuracy'' \in [topAcc - toleranceAcc]$ to determine the $bestSize$ value is to ensure that the $bestSize$ is a small value and also always has a highest $Accuracy''$.

When the maximum number of iteration has been reached or $bestSize$ is the same in two iterations, SRTSO is terminated. SRTSO returns $bestSize$ as the target size T, which guides PSO to search the regions where the size of feature subsets is T.

By applying the SRTSO method to determine T in Equation 4 in PSO, a new approach named SCTSOFS is proposed to solve feature selection problems.

Table 1: Datasets

Dataset	Number of features	Number of instances	Number of classes
WDBC	30	569	2
Ionosphere	34	351	2
Splice	61	3190	4
Hill Valley	100	606	2
Gas 6	128	1694	3
Musk 1	166	476	2
Semeion	256	1593	2
Arrhythmia	278	452	2
Madelon	500	2600	2
Isolet 5	617	1599	26
Multiple Features	649	2000	10
Amazon	10000	1599	50

4 Experimental Design

Twelve datasets (in Table 1) were chosen from the UCI machine learning repository (Bache and Lichman, 2013) as benchmark problems to evaluate the performance of SCTSOFS. These were chosen to represent a range of features, instances, and classes to represent different types of tasks. For each dataset, the instances are randomly divided into $^2/_3$ for the training set and $^1/_3$ for the testing set such that class distribution is approximately maintained.

To perform fitness evaluations in the feature selection process, a classification algorithm is needed to calculate the classification accuracy. There are many options for the algorithm, such as K-nearest neighbour (KNN), Decision Trees, Support Vector Machines, and Naive Bayes. KNN was chosen with $k = 1$ (1NN) due to its simplicity and wide use in existing papers. Each evaluation uses 10-fold cross validation on the training set (Guyon and Elisseeff, 2003).

The performance of SCTSOFS is compared with that of all features, standard PSO for feature selection and two conventional feature selection algorithms: SFS and SBS. The parameters in PSO and SCTSOFS follows common settings suggested in (Clerc and Kennedy, 2002): inertia weight $w = 0.7298$, acceleration constants $c_1 = c_2 = 1.49618$, maximum velocity $v_{max} = 6$. PSO uses a population size of 30 and the maximum iterations of 50 and SCTSOFS uses a population size of 10 and the maximum iterations of 30, and its third acceleration constant $c_3 = c_1$. For SCTSO, the maximum iteration is 2, $N = 30$, $s = 10$, $R = 0.3$, and $tolerance = 0.01$. By using such settings, it ensures that PSO and SCTSOFS have the same number of evaluations for fair comparison purposes. PSO and SCTSOFS are performed for 30 independent runs on each dataset. SFS and SBS are

Table 2: Accuracy (%) and size results for All features, standard PSO, and SCTSOFS

Dataset	Algorithm	Training Set Acc.			Test Set Acc.		
		Mean size	Mean (± stdev)	Best	Mean (± stdev)	Significance	Best
WDBC	All	30.0	93.9	93.9	96.3	–	96.3
	PSO	14.4	96.8 ± 0.2	97.4	96.8 ± 1.1	=	98.4
	SCTSOFS	9.8	96.2 ± 1.1	97.1	96.8 ± 1.1		98.9
Ionosphere	All	34.0	88.0	88.0	84.5	–	84.5
	PSO	12.8	94.0 ± 0.6	95.7	87.3 ± 2.0	=	90.6
	SCTSOFS	7.9	93.7 ± 0.9	95.7	86.6 ± 3.2		93.1
Splice	All	60.0	72.3	72.3	69.6	–	69.6
	PSO	23.9	78.9 ± 0.8	80.7	75.4 ± 1.8	–	79.0
	SCTSOFS	14.0	80.5 ± 2.4	85.0	78.0 ± 3.3		84.0
Hill Valley	All	100.0	57.2	57.2	56.5	+	56.5
	PSO	48.0	61.3 ± 0.4	62.1	55.2 ± 0.8	=	56.9
	SCTSOFS	25.7	60.2 ± 1.4	62.5	54.8 ± 1.7		57.9
Gas 6	All	128.0	100.0	100.0	99.8	+	99.8
	PSO	35.8	100.0 ± 0.0	100.0	99.8 ± 0.2	=	100.0
	SCTSOFS	3.4	100.0 ± 0.0	100.0	99.7 ± 0.2		100.0
Musk 1	All	166.0	84.2	84.2	74.7	–	74.7
	PSO	79.6	92.8 ± 0.8	94.6	79.4 ± 2.7	=	83.0
	SCTSOFS	48.6	90.3 ± 2.4	92.4	79.3 ± 3.1		85.5
Semeion	All	265.0	97.1	97.1	96.4	+	96.4
	PSO	136.2	98.2 ± 0.1	98.4	95.9 ± 0.6	+	97.2
	SCTSOFS	89.0	96.9 ± 1.1	98.0	95.3 ± 0.6		96.2
Arrhythmia	All	278.0	53.8	53.8	55.0	=	55.0
	PSO	135.5	64.4 ± 0.7	66.1	56.4 ± 2.2	=	60.3
	SCTSOFS	108.9	62.2 ± 2.8	67.1	55.8 ± 3.5		63.6
Madelon	All	500.0	54.0	54.0	54.3	–	54.3
	PSO	244.7	60.0 ± 0.6	60.9	55.6 ± 1.6	=	59.4
	SCTSOFS	150.2	59.1 ± 1.7	65.9	55.9 ± 2.3		60.3
Isolet 5	All	617.0	80.6	80.6	71.2	–	71.2
	PSO	304.2	84.1 ± 0.3	85.0	74.3 ± 1.0	+	76.9
	SCTSOFS	238.3	82.6 ± 1.1	83.9	72.7 ± 1.4		76.0
Multiple Features	All	649.0	97.8	97.8	96.9	+	96.9
	PSO	326.2	98.5 ± 0.1	98.7	97.1 ± 0.4	+	98.0
	SCTSOFS	168.7	97.4 ± 0.8	98.1	96.2 ± 0.9		97.7
Amazon	All	10000.0	13.8	13.8	11.4	–	11.4
	PSO	4919.1	18.1 ± 0.4	19.1	12.1 ± 0.9	–	14.0
	SCTSOFS	480.6	19.2 ± 0.8	20.7	13.3 ± 1.5		16.8

deterministic methods, which produce one single solution (feature subset) on each dataset and their settings follow common settings in Weka (Witten and Frank, 2005). A statistical significance test, Wilcoxon test with the significance level as 0.05, is performed on the test accuracies of different algorithms.

5 Results and Discussions

5.1 Comparisons with All Features

Table 2 summarises the classification accuracy and the feature subset size of "All" features, PSO and SCTSOFS, where "mean ± stdev" shows the average and the standard deviation of the accuracies from the 30 independent runs. In terms of the training set, the classification performance of SCTSOFS is better than using the full set of features on ten out of the twelve datasets. The comparison on test set accuracy between SCTSOFS and using a full set of features is generally positive. SCTSOFS performs significantly better on seven of the twelve datasets, worse on four, and equally on one. This indicates that feature selection using SCTSOFS is mostly beneficial to classification accuracy, but this may depend on the dataset.

In terms of the feature subset size, SCTSOFS substantially reduce the dimensionality to at least one third of the original number of features. For example, the number of features is reduced to around 150 on average from the original 500 on the Madelon dataset, but still increase the classification accuracy.

5.2 Comparisons with Standard PSO

According to Table 2, it can be seen that in terms of the subset size, SCTSOFS significantly outperforms the standard PSO feature selection algorithm. SCTSOFS selects a subset with slightly more than half the number of features that PSO selects. The best improvement of SCTSOFS over standard PSO is on the Gas 6 and Amazon datasets, i.e. selecting an order of magnitude fewer features. The results show that SCTSOFS has largely accomplished its goal of reducing feature subset size.

With regards to the accuracy on the test set, SCTSOFS is similar or significantly better than PSO on nine out of the twelve datasets, but significantly worse on three datasets. This indicates an overall trend of having similar test set performance. The results on the training set suggest there is no noticeable difference in the level of overfitting between the two algorithms. However, overfitting is suggested for *both* algorithms on some datasets, in particular Musk 1.

5.3 Comparison with Traditional methods

Table 3 shows the classification accuracy and subset size of SFS and SBS on each dataset, where the empty cells for the three large datasets mean that the algorithm (SFS or SBS) cannot produce a solution by running for a week. There is a symbol of +, −, or = for each algorithm on each dataset, representing the result of a significance test between the classification accuracy of the corresponding algorithm and SCTSOFS on the test set; a + indicates that the corresponding method is statistically significantly better, a − indicates SCTSOFS is better, and an = indicates they are not statistically distinguishable. Statistical significance tests for the size of the feature subset are not given, as the difference is clear.

According to Table 3, SCTSOFS very clearly outperforms SBS in terms of subset size, with SBS selecting the majority of features on most datasets. SBS has statistically equal or lower test set accuracy on ten of the twelve datasets. On these datasets, SCTSOFS is a superior algorithm in all respects.

In contrast, the comparison between SFS and SCTSOFS is mixed. SFS selects very small subsets on all datasets. On the test set, the accuracy of SCTSOFS is statistically equal to or better than SFS on eight of the twelve datasets. The possible reason is that although SCTSOFS uses a size control method, but the main focus is still to optimise the classification performance. Therefore, SCTSOFS outperforms SFS in terms of the classification performance in most cases.

5.4 Computational Cost

Table 4 summarises the running time of each algorithm and the empty cells for the three large datasets mean that the algorithm (SFS or SBS) cannot produce a solution by running for a week. It can be observed that SCTSOFS is faster than standard PSO on all the twelve datasets, although SCTSOFS has extra calculation in SCTSO to find the target size T. This is likely due to the evaluated subsets being, on average, much smaller in SCTSOFS than PSO. The speed difference ranges from near-identical to less than half the running time.

Comparing SCTSOFS with the two traditional methods, SFS runs more quickly on seven of the twelve datasets and SBS also takes orders of magnitude longer to run on almost all datasets, except for the smallest dataset. SBS on the three large datasets (Isolet 5, Multiple Features, and Amazon) and SFS on the two large datasets (Multiple Features and Amazon) cannot even produce a solution within a week. The main reason is that the number of evaluations in SCTSOFS is a fixed number, and the number of evaluations in SFS and SBS increases very quickly along with the number of features in the datasets.

5.5 Analysis of Selected Features

This section compares the features selected by the standard PSO algorithm and SCTSOFS, with the aim of comparing the *consistency* of the two algorithms. Here, consistency measures the similarity of the selected features across multiple runs. This is an important performance metric for a stochastic feature selection algorithm, as it indicates the ability to consistently identify high-quality features in a classification task. This can be analysed based on the frequency of each feature in a dataset being selected is analysed. A uniform frequency among all or most features indicates that the feature selection algorithm is highly inconsistent, indiscriminately selecting features. Similarly, if some features are very commonly selected and others are very rarely selected, the algorithm is highly consistent and selective in its outputs.

To show the consistency, the number of times that each feature is selected by PSO or SCTSOFS in the 30 independent runs are collected, and scaled to [0, 1] as frequency values. 0 means that the feature is not selected at all through the 30 runs and 1 means it is selected across all the 30 runs.

Table 3: Accuracy (%) and size results for SFS and SBS

Method	Size	Train Acc.	Sig.	Test Acc.	Sig.	Size	Train Acc.	Sig.	Test Acc.	Sig.
	WDBC					Ionosphere				
SFS	7.0	93.9	−	96.3	−	5.0	93.2	−	86.4	=
SBS	21.0	96.3	=	98.4	+	26.0	90.1	−	82.0	−
	Splice					Hill Valley				
SFS	6.0	88.9	+	88.2	+	1.0	56.3	−	49.5	−
SBS	50.0	75.1	−	69.8	−	90.0	59.0	−	55.2	=
	Gas 6					Musk 1				
SFS	2.0	100.0	+	99.3	−	14.0	93.7	+	79.3	=
SBS	3.0	100.0	+	99.3	−	84.0	93.4	+	80.4	=
	Semeion					Arrhythmia				
SFS						10.0	65.8	+	59.0	+
SBS	206.0	98.3	+	95.9	+	123.0	64.5	+	54.3	−
	Madelon					Isolet 5				
SFS	11.0	88.3	+	86.4	+	40.0	90.1	+	80.4	+
SBS	488.0	57.2	−	53.8	−					
	Multiple Features					Amazon				
SFS	14.0	98.9	+	95.9	=					
SBS										

Table 4: Running time (milliseconds) of algorithms

Algorithm	WDBC	Ionosphere	Splice
All	40	45	895
SFS	7854	3299	177342
SBS	13434	8564	2356288
PSO	15694 ± 1642	7007 ± 672	1072188 ± 88625
SCTSOFS	14877 ± 3070	5504 ± 703	686714 ± 209926

Algorithm	Hill Valley	Gas 6	Musk 1
All	62	199	57
SFS	11842	37932	97880
SBS	211320	2305076	1515172
PSO	43607 ± 2651	162846 ± 22318	43556 ± 5375
SCTSOFS	40780 ± 8940	109839 ± 18292	30494 ± 5738

Algorithm	Semeion	Arrhythmia	Madelon
All	709	81	7378
SFS	23204	70472	2519672
SBS	64670783	6230209	183583434
PSO	972074 ± 78612	76910 ± 3713	9640148 ± 650128
SCTSOFS	703928 ± 142299	60132 ± 13983	6454478 ± 2519219

Algorithm	Isolet 5	Multiple Features	Amazon
All	1482	3862	43292
SFS	8844293	4223189	
SBS			
PSO	1832157 ± 28898	5418443 ± 471782	54771372 ± 1488128
SCTSOFS	1502773 ± 290200	2933193 ± 459691	25809611 ± 5421677

Table 5: Q-measure of PSO and SCTSOFS

Algorithm	WDBC	Ionosphere	Splice
PSO	0.447	0.468	0.485
SCTSOFS	0.417	0.634	0.590
Algorithm	Hill Valley	Gas 6	Musk 1
PSO	0.345	0.241	0.282
SCTSOFS	0.435	1.109	0.308
Algorithm	Semeion	Arrhythmia	Madelon
PSO	0.211	0.210	0.185
SCTSOFS	0.245	0.214	0.209
Algorithm	Isolet 5	Multiple Features	Amazon
PSO	0.198	0.171	0.153
SCTSOFS	0.204	0.245	0.671

5.5.1 Statistics-based Analysis

The frequency values of the features being selected can be treated as distributions. The concept of consistency is characterised by the probability distribution of features being highly non-uniform, and so a comparison to the discrete uniform distribution of n elements, where n is the dimensionality of the data, gives a metric for an algorithm's consistency. Supposing $f_1 \dots f_n$ are the frequency values for the n feature over a number of experiments, the Q-value of those frequencies is defined as follows:

$$Q = \sum_{i=1}^{n} \left| \frac{f_i}{\sum_{i=1}^{n} f_i} - \frac{1}{n} \right|$$

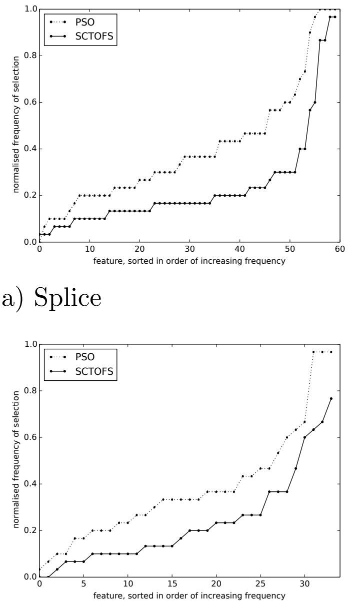
This, in essence, normalises the frequencies to form a proper probability distribution and then sums the absolute values of the difference between each feature and the worst-case values of the uniform distribution. A high Q-value indicates that the algorithm shows high consistency, and a low Q-value indicates inconsistency.

Table 5 gives the Q-value for the benchmark datasets. The results are near-universally favourable towards SCTSOFS, showing a Q-value greater than or equal to that of the standard PSO algorithm in all cases except for the WDBC dataset. This indicates that the new method SCTSOFS is more consistent in terms of finding relevant features.

5.5.2 Graph-based Analysis

To better show the results, Figure 1 takes the Splice and Ionosphere datasets as examples to plot the scaled values. Other datasets follow a similar pattern and they are not presented here due to the page limit.

In Figure 1, the scaled values are sorted in an ascending order. Following the horizontal axis, from the left to right, the scaled values are increasing, which shows the lowest frequency to the highest frequency of features being selected. The deeper the curve is the more consistent the algorithm is, because how deepness of the curve shows how well the algorithm distinguish different features. Both plots in Figure 1 show SCTSOFS positively in comparison to standard PSO, particularly, Splice shows over two-thirds of the features being selected no more than 20% of the time, in contrast with standard PSO. It is also shown that SCTSOFS as a more bowed curve, indicating a higher consistency. These results do, however, corroborate with the above statistics-based analysis. In addition, the area under the curve for SCTSOFS is significantly smaller than that of the standard PSO algorithm in both cases. This is expected, as the area represents the average size of the selected feature subsets.



(a) Splice

(b) Ionosphere

Figure 1: Sorted frequencies of features being selected.

6 Conclusions and Future Work

The goal of this paper was to develop a new PSO approach to feature selection with the expectation of significantly reducing the number of features and achieving the similar or even better classification than using all features and standard PSO. The goal was achieved by developing a sequential random target size optimisation method to find a rough target size (number of features), which was used to guide the PSO search towards feature subsets with high classification performance and small number of features.

The proposed algorithm, SCTSOFS, was compared with the standard PSO for feature selection and two typical traditional methods, SFS and SBS. The results show that in most cases, SCTSOFS substantially reduced the dimensionality of the dataset over the original feature set, the feature sets selected by standard PSO, and SBS, and the classification performance was maintained or even improved. Although SFS selected a smaller number of features in many cases, the classification performance of SCTSOFS was generally better, which suggested that although SCTSOFS seriously considered the number of features during feature selection process, but it did not sacrifice the classification performance. Furthermore, SCTSOFS used a shorter running time than PSO in all cases, a longer time than SFS and SBS on small datasets, but a much shorter time than SFS and SBS on large datasets. Since PSO and SCTSOFS are stochastic methods, we also analyse their consistency in terms of selecting features in different runs, the analysis showed that SCTSOFS selected more consistent features across different independent runs, which can provide a better suggestions to real-world users to find informative or key features in complex problems.

Several questions have been raised in the process of this research, such as the consistency analysis for stochastic feature selection algorithms. The analysis presented in this paper can provide straightforward illustrations, but is relatively simple and lack of theoretical prove. Further developments in this area would yield deeper insights into the operation, effectiveness, and applicability of these types of algorithms.

References

Almuallim, H. and Dietterich, T. G. (1994), 'Learning boolean concepts in the presence of many irrelevant features', *Artificial Intelligence* **69**, 279–305.

Bache, K. and Lichman, M. (2013), 'Uci machine learning repository'.
URL: *http://archive.ics.uci.edu/ml*

Bharathi P T, P. S. (2014*a*), 'Differential evolution and genetic algorithm based feature subset selection for recognition of river ice type', *Journal of Theoretical and Applied Information Technology* **7**(1), 254–262.

Bharathi P T, P. S. (2014*b*), 'Optimal feature subset selection using differential evolution and extreme learning machine', *International Journal of Science and Research (IJSR)* **3**, 1898–1905.

Boubezoul, A. and Paris, S. (2012), 'Application of global optimization methods to model and feature selection', *Pattern Recognition* **45**(10), 3676 – 3686.

Cervante, L., Xue, B., Shang, L. and Zhang, M. (2012), A dimension reduction approach to classification based on particle swarm optimisation and rough set theory, *in* '25nd Australasian Joint Conference on Artificial Intelligence', Vol. 7691 of *Lecture Notes in Computer Science*, Springer, pp. 313–325.

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

Clerc, M. and Kennedy, J. (2002), 'The particle swarm– explosion, stability, and convergence in a multidimensional complex space', *IEEE Transactions on Evolutionary Computation* **6**(1), 58–73.

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

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

Espejo, P., Ventura, S. and Herrera, F. (2010), 'A survey on the application of genetic programming to classification', *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* **40**(2), 121–144.

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

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

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

Kennedy, J. and Eberhart, R. (1995), Particle swarm optimization, *in* 'IEEE International Conference on Neural Networks', Vol. 4, pp. 1942–1948.

Kennedy, J. and Eberhart, R. (1997), A discrete binary version of the particle swarm algorithm, *in* 'IEEE International Conference on Systems, Man, and Cybernetics', Vol. 5, pp. 4104–4108.

Kira, K. and Rendell, L. A. (1992), 'A practical approach to feature selection', *Assorted Conferences and Workshops* pp. 249–256.

Kononenko, I. (1994), 'Estimating attributes: Analysis and extensions of relief', *Lecture Notes in Computer Science* **784**, 171.

Lane, M., Xue, B., Liu, I. and Zhang, M. (2013), Particle swarm optimisation and statistical clustering for feature selection, *in* 'AI 2013: Advances in Artificial Intelligence', Vol. 8272 of *Lecture Notes in Computer Science*, Springer International Publishing, pp. 214–220.

Lane, M., Xue, B., Liu, I. and Zhang, M. (2014), Gaussian based particle swarm optimisation and statistical clustering for feature selection, *in* 'Evolutionary Computation in Combinatorial Optimisation', Vol. 8600 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 133–144.

Lin, F., Liang, D., Yeh, C.-C. and Huang, J.-C. (2014), 'Novel feature selection methods to financial distress prediction', *Expert Systems with Applications* **41**(5), 2472–2483.

Lin, S. W., Ying, K. C., Chen, S. C. and Lee, Z. J. (2008), 'Particle swarm optimization for parameter determination and feature selection of support vector machines', *Expert Systems with Applications* **35**(4), 1817–1824.

Liu, H., Motoda, H., Setiono, R. and Zhao, Z. (2010), Feature selection: An ever evolving frontier in data mining, *in* 'FSDM', Vol. 10 of *JMLR Proceedings*, JMLR.org, pp. 4–13.

Liu, H. and Zhao, Z. (2009), Manipulating data and dimension reduction methods: Feature selection, *in* 'Encyclopedia of Complexity and Systems Science', Springer, pp. 5348–5359.

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

Marill, T. and Green, D. (1963), 'On the effectiveness of receptors in recognition systems', *IEEE Transactions on Information Theory* **9**(1), 11–17.

Mitra, P., Murthy, C. and Pal, S. (2002), 'Unsupervised feature selection using feature similarity', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(3), 301–312.

Muni, D., Pal, N. and Das, J. (2006), 'Genetic programming for simultaneous feature selection and classifier design', *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* **36**(1), 106–117.

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

Nguyen, H., Xue, B., Liu, I. and Zhang, M. (2014), PSO and statistical clustering for feature selection: A new representation, *in* 'Simulated Evolution and Learning', Vol. 8886 of *Lecture Notes in Computer Science*, pp. 569–581.

Pudil, P., Novovicova, J. and Kittler, J. V. (1994), 'Floating search methods in feature selection', *Pattern Recognition Letters* **15**(11), 1119–1125.

Purohit, A., Chaudhari, N. and Tiwari, A. (2010), Construction of classifier with feature selection based on genetic programming, *in* 'IEEE Congress on Evolutionary Computation (CEC'10)', pp. 1–5.

Ramadan, R. M. and Abdel Kader, R. F. (2007), 'Face recognition using particle swarm optimization-based selected features', *International Journal of Signal Processing, Image Processing and Pattern Recognition* **2**(2), 51–65.

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

Stearns, S. (1976), On selecting features for pattern classifier, *in* 'Proceedings of the 3rd International Conference on Pattern Recognition', IEEE Press, Coronado, Calif, USA, pp. 71–75.

Tang, E. K., Suganthan, P. and Yao, X. (2005), Feature selection for microarray data using least squares SVM and particle swarm optimization, *in* 'IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB'05)', pp. 1–8.

Tran, B., Xue, B. and Zhang, M. (2014), Overview of particle swarm optimisation for feature selection in classification, *in* 'Simulated Evolution and Learning', Vol. 8886 of *Lecture Notes in Computer Science*, Springer International Publishing, pp. 605–617.

Vieira, S. M., Mendonça, L. F., Farinha, G. J. and Sousa, J. M. (2013), 'Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients', *Applied Soft Computing* **13**(5), 3494–3504.

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

Witten, I. H. and Frank, E. (2005), *Data Mining: Practical Machine Learning Tools and Techniques (Second Edition)*, Morgan Kaufmann.

Xue, B., Cervante, L., Shang, L., Browne, W. N. and Zhang, M. (2014), 'Binary PSO and rough set theory for feature selection: A multi-objective filter based approach', *International Journal of Computational Intelligence and Applications* **13**(02), 1450009.

Xue, B., Zhang, M. and Browne, W. (2013*a*), Novel initialisation and updating mechanisms in PSO for feature selection in classification, *in* 'Applications of Evolutionary Computation', Vol. 7835 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 428–438.

Xue, B., Zhang, M. and Browne, W. N. (2013*b*), 'Particle swarm optimization for feature selection in classification: A multi-objective approach', *IEEE Transactions on Cybernetics* **43**(6), 1656–1671.

Xue, B., Zhang, M. and Browne, W. N. (2015), 'A comprehensive comparison on evolutionary feature selection approaches to classification', *International Journal of Computational Intelligence and Applications* **14**(02), 1550008.

Yong, Z., Dunwei, G., Ying, H. and Wanqiu, Z. (2015), 'Feature selection algorithm based on bare bones particle swarm optimization', *Neurocomputing* **148**, 150–157.

Zhang, C. and Hu, H. (2005), Using PSO algorithm to evolve an optimum input subset for a SVM in time series forecasting, *in* 'IEEE International Conference on Systems, Man and Cybernetics (SMC'05)', Vol. 4, pp. 3793–3796.

Zhu, Z., Ong, Y.-S. and Dash, M. (2007), 'Markov blanket-embedded genetic algorithm for gene selection', *Pattern Recognition* **40**(11), 3236–3248.