# Gaussian Transformation based Representation in Particle Swarm Optimisation for Feature Selection

Hoai Bach Nguyen[1], Bing Xue[1], Ivy Liu[2], Peter Andreae[1], and Mengjie Zhang[1]

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

**Abstract.** In classification, feature selection is an important but challenging task, which requires a powerful search technique. Particle swarm optimisation (PSO) has recently gained much attention for solving feature selection problems, but the current representation typically forms a high-dimensional search space. A new representation based on feature clusters was recently proposed to reduce the dimensionality and improve the performance, but it does not form a smooth fitness landscape, which may limit the performance of PSO. This paper proposes a new Gaussian based transformation rule for interpreting a particle as a feature subset, which is combined with the feature cluster based representation to develop a new PSO-based feature selection algorithm. The proposed algorithm is examined and compared with two recent PSO-based algorithms, where the first uses a Gaussian based updating mechanism and the conventional representation, and the second uses the feature cluster representation without using Gaussian distribution. Experiments on commonly used datasets of varying difficulty show that the proposed algorithm achieves better performance than the other two algorithms in terms of the classification performance and the number of features in both the training sets and the test sets. Further analyses show that the Gaussian transformation rule improves the stability, i.e. selecting similar features in different independent runs and almost always selects the most important features.

**Keywords:** Particle swarm optimisation, Feature selection, Representation, Gaussian distribution, Classification.

## 1 Introduction

In real-world classification problems, the dataset often has a large number of features, but not all features are relevant to the target concept. Irrelevant and redundant features are not useful for classification, but may reduce the performance due to "the curse of dimensionality" [1]. Feature selection is the process of selecting a small subset of relevant features to reduce the dimensionality with the goal of increasing or at least maintaining the classification performance while speeding up the classification process [2].

Feature selection is a challenging task due mainly to two reasons: a large solution space and feature interaction. 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, feature interaction means that individually relevant features may become redundant and individually weakly relevant features may become highly relevant when combined with other features [1]. Therefore, finding an effective way to deal with feature interaction is critical in feature selection. Research in statistical data analysis also involves interaction between features. A newly developed statistical model [4, 5] considers interaction between features to group similar features into clusters and dissimilar features into different clusters. Using this feature clustering information has shown to improve feature selection performance [6, 7].

Even with such information, a powerful search algorithm is still needed to find the optimal feature subsets. Although different types of search techniques have been applied to features selection [1, 2], 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) [8], particle swarm optimisation (PSO) [9–12], and differential evolution (DE) [13]. However, many EC algorithms (including PSO, GAs, and DE) use a representation scheme with the same length/dimensionality as the number of features $n$, which forms a huge search space and limits the performance of these algorithms particularly when it is large.

Nguyen et al. [7] propose a feature cluster based representation which significantly reduces the dimensionality of the search space. The proposed algorithm, named PSOR, [7] achieved better performance than a standard PSO-based algorithm, a PSO-based algorithm published in 2014 [9], and two typical conventional non-EC based feature selection algorithms. However, there is a potential limitation in that representation because the fitness landscape is not smooth (Details in Section 2.2). The performance of PSO can be significantly improved if the problem is encoded/represented in a search space with a smooth fitness landscape. This work aims to address this limitation and proposes a new algorithm to further improve the performance of PSO for feature selection.

### 1.1 Goals

The overall goal of this paper is to develop a new PSO algorithm for feature selection to reduce the dimensionality of the data and increase the classification performance. To achieve this goal, Gaussian distribution is introduced to the feature cluster based representation to smooth the fitness landscape, based on which a new PSO-based approach is proposed. The proposed algorithm involves the *feature cluster based representation* and the *Gaussian distribution*. It is examined and compared with two PSO-based algorithms: PSOR [7] that uses the feature cluster based representation without Gaussian distribution, and a PSO-based algorithm named GPSO [6] that uses a Gaussian distribution based updating mechanism but with the conventional representation. Specifically, we seek to show that

– the *feature cluster based representation* can improve feature selection performance by comparing the proposed algorithm with GPSO,

– the *Gaussian distribution* can improve the representation by comparing the proposed algorithm with PSOR, and
– the combination of the feature cluster based representation and the Gaussian distribution can further improve feature selection performance.

## 2 Background

### 2.1 Particle Swarm Optimisation (PSO)

Particle swarm optimisation (PSO) [14] is an EC method inspired by social behaviours, such as bird flocking and fish schooling. When using PSO to solve a problem, the solution is optimised by using a population, swarm, of candidate solutions, which are called particles. Each particle moves in the search space by updating its position as well as velocity. The current position of particle $i$ is represented by a vector $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, where $D$ is the dimensionality of the search space. These positions are updated by using another vector, called velocity $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$, where each element is limited by a predefined maximum velocity: $v_{id} \in [-v_{max}, v_{max}]$. During the search process, each particle maintains a record of the position of its best performance as far, called $pbest$. It also records the best solution among the $pbest$s of its neighbours, called $gbest$. At the $t^{th}$ iteration in the search process, the velocity and position of each particle are updated according Equations (1) and (2):

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

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

where $d$ is the $d^{th}$ dimension, $w$ is inertia weight, $c_1$ and $c_2$ are acceleration constants, $r_{i1}$ and $r_{i2}$ are random values uniformly distributed in [0,1].

### 2.2 Representation in PSO for Feature Selection

In PSO for feature selection, each particle represents a subset of features and its position is typically specified by an $n-$dimensional vector, where each dimension corresponds to one feature in the dataset. Each element of the position vector is a real valued number between [0, 1], which represents a confidence level that the corresponding feature is selected. The feature subset is typically constructed by selecting each feature if the confidence level is above a predefined threshold $\theta$. Most EC techniques, such as GAs, DE, and artificial bee colony, use a similar representation. However, such a representation has a potential limitation in that it leads to a very high-dimensional search space when $n$ is large.

Although the representation is an important component in PSO (or EC), there has been little work on developing new representations in PSO for feature selection. One of the major challenges is that it is hard to develop a representation suitable for the updating mechanisms. Only small modifications have been made on the original representation. Some work has added further dimensions to the search space by including the

parameters of the classifier in the position vector [10, 15, 12]. This cannot overcome the limitation of the original representation, but further increases the dimensionality and expands the search space even more.

Nguyen et al. [7] proposed a new feature cluster based representation with the dimensionality much smaller than $n$, where a statistical feature clustering model [4, 5] was applied as a pre-processing step to partition the features into $c$ clusters. PSO was then used to select features from each cluster with a limit of at most $\sqrt{n_j}$ features from a cluster $j$ containing $n_j$ features. The size of the clusters were $n_1$, $n_2$, ... , $n_c$, then the maximum number of selected features was $\sum_{j=1}^{c} \sqrt{n_j}$, which was also the dimensionality of the particle position vectors. This is usually much smaller than $n$. The dimensions of the position vector were partitioned for the clusters so that the dimensions from $\sum_{k=1}^{j-1}(\sqrt{n_k}+1)$ to $\sum_{k=1}^{j}(\sqrt{n_k}+1)$ correspond to cluster $j$. To interpret a particle position vector as a feature subset, each element of the position vector specified a feature from its corresponding cluster. The interval $x_i \in [0, 1]$ was segmented to $(n_j + 1)$ subranges, each corresponding to one feature in the $j$ cluster except the last subrange that corresponds to no feature. If a position value $x_{id}$ belongs to one of the first $n_j$ subranges, the corresponding feature will be selected. If $x_{id}$ belongs to the last subrange, no feature is selected from cluster $j$. More details about this representation can be found in [7].

The representation in [7] can successfully reduce the dimensionality of the search space and improve the performance, but has a limitation that it forms a unsmooth fitness landscape, which significantly influences the performance of PSO. In particular, as a position element changes within a subrange, there is no difference to the feature subset, but there will be sudden change to chose a different feature when the element crosses the boundary of the subrange. Suppose a feature $f$ corresponds to the range [0.4, 0.6]. For three position values, $p_1 = 0.42$, $p_2 = 0.58$ and $p_3 = 0.62$, $p_1$ and $p_2$ will select feature $f$ and $p_3$ will not. However, $p_3$ is closer to $p_2$ than $p_1$. This means during the search process of PSO, a relatively big change from $p_1$ to $p_2$ on a particle's position will not result in any change in its fitness value, but a relatively small change from $p_2$ to $p_3$ will suddenly change the fitness value. In other words, using the transformation rule, the fitness landscape of the feature selection problem is not smooth. Given that the proposed feature cluster based representation has achieved significantly better performance than PSO using the conventional representation [7], addressing this limitation is highly likely to further improve the performance.

## 3 Proposed Algorithm: GPSOR

This section proposes a new transformation rule for interpreting a position vector as a feature subset to address the limitation in the cluster based representation of PSOR. The new representation is expected to add more meaning to the movement of particles during the search process and form a smoother fitness landscape. In this work, the same statistical feature clustering model [4, 5] is applied as a pre-processing step to partition features into different clusters.

The length of the representation in this paper is the same as in [7], which is $\sum_{j=1}^{c} \sqrt{n_j}$. The $n_j$ features in cluster $j$ are represented by $\sqrt{n_j}$ dimensions in the position vector.

Each element of the vector specifies one of the features in the corresponding cluster to be included in the solution. The key difference is that the value of a position vector element is used to specify a feature in a probabilistic manner, using a Gaussian distribution to obtain a smoother search space. The interval [0, 1] is also divided into $(n_j + 1)$ subrange and the length of each subrange is $s = 1/(n_j + 1)$. The first $n_j$ subranges correspond to the $n_j$ features in cluster $j$ and the last subrange corresponds to no feature being selected (in case all features in the cluster are irrelevant). A position value still has a corresponding feature depending on which subrange it belongs to. The difference here from [7] relies on developing a new transformation rule, which is a key factor in the representation to smooth the fitness landscape. To achieve this, Gaussian distribution (i.e. normal distribution, Equation 3) is introduced here to interpret position values. Instead of directly determining which feature is selected, a position value is used to calculate the probabilities of features being selected through the Gaussian distribution.

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{3}$$

### 3.1 Constructing Gaussian Transformation Rule

The PSOR algorithm interprets a value in the position vector of a particle as a feature to be included in the feature subset. Which feature to be included is determined by identifying which subrange of [0,1] the position value lies in. The limitation of the PSOR algorithm is that small changes in a position value have no effect on the feature subset (and therefore the fitness value) until the value crosses the boundary of the subrange into the subrange corresponding to the next feature, at which point there is a sudden (non-smooth) change in the feature subset and the fitness value. GPSOR addresses this limitation by using the same representation as PSOR, but adding a small amount of Gaussian noise to each value in the position vector before identifying which subrange it is in, and therefore the feature it is specifying. This Gaussian noise means that as a value in the position vector approaches the boundary of a subrange, it has an increasing probability of specifying a neighbouring feature. The effect is to smooth the search space and provide an early indication of when the particle is moving towards a better (or worse) position in the search space.

The Gaussian noise for a value $x_{id}$ is currently generated from a Gaussian distribution with a standard deviation fixed at 1% of the width of subranges (ie, $\frac{\sigma_d=1}{100*(1+\sqrt{(n_j)})}$, where $n_j$ is the size of the feature cluster corresponding to dimension $d$ ). This ensures that if a position vector $x_{id}$ is in the center of a subrange, the probability of selecting the feature corresponding to that subrange is about 99%, but as $x_{id}$ gets close to the boundary of the subrange, the probability decreases smoothly to around 50%. Fig. 1 illustrates the distribution of $x_{id} = 0.58$ with the Gaussian noise for a feature cluster containing 4 features.

### 3.2 Selection of Features

The introduction of Gaussian distribution makes the position value and the move of particles more meaningful than directly using the position value to select a feature.

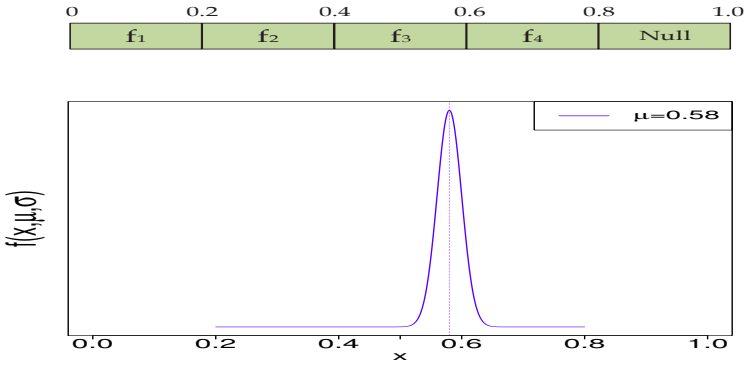


Fig. 1: Effect of applying the Gaussian distribution

This section discusses the details of how to use the Gaussian transformation rule to translate position values to features. For a position value corresponds to cluster $j$ with $n_j$ features, the selection of features from cluster $j$ is determined according to following three steps:

- build a Gaussian distribution function, $f(x, \mu, \sigma)$, using the position value as $\mu$ and $0.01 * s$, where $s = 1/(n_j + 1)$, as $\sigma$;
- use the inverse transform sampling method to generate a random number $r \in (0, 1)$ according to $f(x, \mu, \sigma)$;
- $r$ is used to determine which feature is selected from a certain cluster. A feature is selected if $r$ falls into its corresponding sub-interval.

Based on the built Gaussian distribution, the random number $r$ is likely to fall into the same sub-interval with the original position value, but has a chance with different probabilities to fall into other sub-intervals to select other features. For three position values, $p_1 = 0.5$, $p_2 = 0.58$ and $p_3 = 0.62$, their corresponding features are $f_3$, $f_3$ and $f_4$, respectively. If using the transformation rule in [7], $p_1$ and $p_2$ will select $f_3$, and $p_3$ will select $f_4$. This means a big move in the position from $p_1$ to $p_2$ does not change the solution, but a small move from $p_2$ to $p_3$ changes the solution. By using the proposed transformation rule, $p_1$ will be mostly likely to generate a random number to select $f_3$. $p_2$ will have a much lower probability than $p_1$ to select $f_3$, but $p_2$ has a similar probability to $p_3$ to select $f_3$ (or $f_4$). This means that solutions are gradually changed according to the position value, which makes the move of particles more meaningful and is expected to achieve better performance.

### 3.3 Pseudo-code of the Algorithm

Based on the proposed Gaussian based transformation rule, a PSO-based feature selection algorithm named GPSOR is proposed here. The pseudo-code of GPSOR is shown in Algorithm 1. The fitness function of GPSOR is to minimise the classification error rate of the classifier built using the selected features, which is calculated by the Equation (4).

**Algorithm 1** : Pseudo-code of GPSOR

```
1: begin
2: randomly initialise the position and velocity of each particle;
3: initialise a feature subset BFS with the empty feature set;
4: while Maximum iteration is not reached do
5:     for each particle in the swarm do
6:         decode the position to a feature subset using the Gaussian distribution;
7:         evaluate the fitness based on the selected features;
8:         replace BFS with the feature subset decoded from gbest if it is better;
9:     end for
10:    update pbest and gbest of each particle;
11:    for i = 1 to Population size do
12:        update v_i of particle i according to Equation 1;
13:        update x_i of particle i according to Equation 2;
14:    end for
15: end while
16: calculate the training and testing classification accuracy using BFS
17: return BFS, the training and testing classification accuracies;
18: end
```

$$Fitness = \frac{fp + fn}{tp + tn + fp + fn} \tag{4}$$

where tp, tn, fp, and fn denote true positives, true negatives, false positives, and false negatives, respectively.

## 4 Experimental Design

The proposed algorithm (GPSOR) is examined and compared with two PSO approaches, PSOR [7] using the cluster based representation without using Gaussian distribution, and GPSO [6] using the traditional representation and Gaussian distribution, where Gaussian distribution was used to determine how many features to be selected from a feature cluster. All the three methods use the statistical clustering model [4, 5] as a pre-processing step to partition features into different clusters. The statistical model is not described here since it is not the focus of this paper and details can be seen from [4, 5]. Since PSOR [7] has shown to be superior to two recent PSO algorithms without using statistical clustering and two conventional methods, GPSOR is also better than them if GPSOR can achieve better performance than PSOR. Therefore, the results of these algorithms are not listed here due to page limit.

Eight datasets (Table 1) chosen from the UCI machine learning repository [16] are used in the experiments. These datasets have different numbers of features, classes and instances. For each dataset, all instances are randomly divided into a training set and a test set, which contains 70% and 30% of the instances, respectively. In the experiments, the classification/learning algorithm is K-nearest neighbour (KNN) where K = 5. The parameters of PSO are set as follows [17]: $w$ = 0.7298, $c_1$ = $c_2$ = 1.49618, $v_{max}$ = 6.0, population size is 30, the maximum number of iterations is 100. The fully connected topology is used. All the three algorithms have been run for 30 independent times on

Table 1: Datasets

| Dataset | #features | #clusters | #classes | #instances |
|---|---|---|---|---|
| Wine | 13 | 6 | 3 | 178 |
| Vehicle | 18 | 6 | 4 | 846 |
| Ionosphere | 34 | 11 | 2 | 351 |
| Sonar | 60 | 12 | 2 | 208 |
| Musk1 | 166 | 14 | 2 | 476 |
| Arrhythmia | 279 | 15 | 16 | 452 |
| Madelon | 500 | 11 | 2 | 4400 |
| Multiple Features | 649 | 15 | 10 | 2000 |

each dataset. A statistical significance test, Wilcoxon test with significance level as 0.05, is performed to compare between the classification accuracies of different algorithms.

## 5 Experimental Results

Table 2 shows the experimental results, where "All" means that all the available features are used for classification. "Ave-size" shows the average number of selected features over the 30 runs. "Ave" and "Std" illustrate the average and standard deviation of the training or testing accuracies over the 30 independent runs. "Test" shows the results of the statistical significance tests between the accuracy of GPSOR and other algorithms. "+" or "-" means that the compared algorithm is significantly better or worse than GPSOR. "=" means there is no significant difference.

### 5.1 Effectiveness of the GPSOR Search

To evaluate the effectiveness of the new representation in searching for an optimal fitness value, we need to look at the performance of GPSOR on the training set.

All the three PSO-based algorithms aim to minimise the training classification error rate (i.e. maximise the training accuracy). From Table 2, it can be observed that GPSOR achieved significantly better training performance than using all features, GPSO, and PSOR in almost all cases. Only on the Multiple Features dataset, there is no significant difference between the training accuracy of using all features, GPSO and GPSOR. The reason is that the training accuracy is already very high when using the original feature set, i.e. 99.35%, which is hard to make significant improvement. The training results in Table 2 show that GPSOR using the representation with a lower dimensionality and the Gaussian distribution based transformation rule can better represent the problem and facilitate the search to significantly improve the performance of PSO for feature selection.

### 5.2 Comparison with PSO-based Algorithms on the Test Set

According to Table 2, the number of features selected by GPSOR is much smaller than the total number of features, but using the selected features only, the KNN classification algorithm achieved significantly better classification accuracy than using all features in

Table 2: Experimental Results

| Dataset | Method | Ave-Size | Test Set | | Training Set | |
|---|---|---|---|---|---|---|
| | | | Ave±Std | Test | Ave±Std | Test |
| Wine | All | 13 | 76.54 | - | 87.63 | - |
| | GPSO | 5.4 | 96.59 ± 2.76 | - | 96.71 ± 7.77E-14 | - |
| | PSOR | 4.75 | 96.70 ± 3.1 | - | 95.05 ± 0.58 | - |
| | GPSOR | 4.60 | 97.70 ± 2.52 | | 97.37 ± 0.42 | |
| Vehicle | All | 18 | 83.86 | - | 88.17 | - |
| | GPSO | 8.94 | 84.30 ± 0.62 | - | 86.11 ± 0.2 | - |
| | PSOR | 5.87 | 84.72 ± 0.87 | = | 84.61 ± 0.56 | - |
| | GPSOR | 7.30 | 84.74 ± 0.49 | | 90.10 ± 0.4 | |
| Ionosphere | All | 34 | 83.81 | - | 85.77 | - |
| | GPSO | 7.66 | 89.5 ± 1.68 | + | 91.59 ± 0.47 | - |
| | PSOR | 9.7 | 88.63 ± 1.68 | + | 90.04 ± 0.99 | - |
| | GPSOR | 3.17 | 86.89 ± 1.8 | | 93.90 ± 0.67 | |
| Sonar | All | 60 | 76.19 | - | 83.44 | - |
| | GPSO | 17.64 | 78.19 ± 4.14 | = | 86.74 ± 0.94 | - |
| | PSOR | 14.33 | 78.94 ± 4.02 | = | 87.01 ± 2 | - |
| | GPSOR | 10.17 | 78.25 ± 2.96 | | 90.67 ± 1.6 | |
| Musk1 | All | 166 | 83.92 | - | 92.19 | - |
| | GPSO | 39.64 | 84.95 ± 2.73 | + | 90.02 ± 0.6 | - |
| | PSOR | 35.03 | 83.12 ± 3.41 | = | 89.78 ± 1.25 | - |
| | GPSOR | 38.93 | 83.29 ± 2.48 | | 93.22 ± 1.37 | |
| Arrhythmia | All | 279 | 94.46 | - | 94.79 | - |
| | GPSO | 45.5 | 94.85 ± 0.34 | - | 94.87 ± 0.09 | - |
| | PSOR | 44.17 | 94.96 ± 0.38 | - | 95.11 ± 0.2 | - |
| | GPSOR | 42.03 | 95.12 ± 0.34 | | 95.75 ± 0.18 | |
| Madelon | All | 500 | 70.9 | - | 83.24 | - |
| | GPSO | 36.08 | 85.68 ± 1.1 | + | 85.45 ± 0.73 | - |
| | PSOR | 54.39 | 83.40 ± 2 | - | 83.73 ± 1.74 | - |
| | GPSOR | 51.17 | 84.06 ± 1.65 | | 89.20 ± 1.41 | |
| Multiple Features | All | 649 | 98.63 | - | 99.35 | = |
| | GPSO | 91.4 | 99.01 ± 0.13 | = | 99.38 ± 0.38 | = |
| | PSOR | 51.07 | 98.84 ± 0.18 | = | 99.17 ± 0.09 | - |
| | GPSOR | 51 | 98.86 ± 0.17 | | 99.36 ± 0.07 | |

all cases. For example, on the Madelon dataset, GPSOR selected on average 51 features from the original 500 features, but achieved a significant increase in the classification accuracy of 13%. The results suggest that the proposed GPSOR algorithm using Gaussian transformation rule based representation can successfully explore the search space to remove redundant and irrelevant features and increase the classification accuracy.

Comparing GPSOR with GPSO both of which used the statistical clustering information and Gaussian distribution, it can be observed that GPSOR selected fewer features than GPSO on seven out of the eight datasets, where the only exception is the Madelon dataset. GPSOR achieved significantly higher or similar classification accuracy than GPSO on five of the eight datasets. On the Ionosphere, Musk1 and Madelon datasets, GPSOR selected a smaller number of features than GPSO on the Ionosphere and Musk1 datasets, where GPSO achieved higher accuracy than GPSOR. On

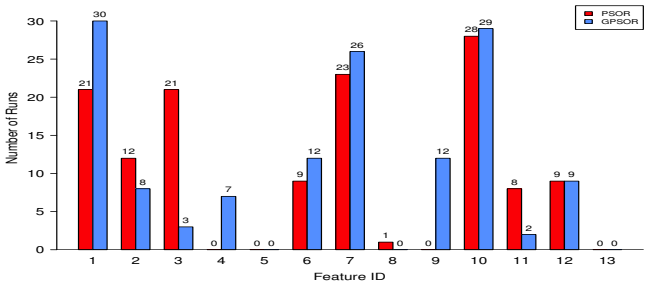


Fig. 2: NO. of runs that each feature was selected by PSOR and GPSOR.

the Madelon dataset, GPSO outperformed GPSOR in terms of both the number of features and the test accuracy, but according the performance on the training set, an overfitting problem may happen since GPSOR achieved much better training classification accuracy than GPSO. On the Multiple Features datasets, since the accuracy of using all features is very high, both GPSO and GPSOR could not significantly increase the training performance. However, since GPSOR removed redundant and irrelevant features, the selected small features subsets have better generalisation on the unseen test set than the original large feature set and significantly improved the test accuracy. The results suggest that GPSOR using the representation with a lower dimensionality can make the problem to be better addressed by PSO to achieve higher classification performance. Meanwhile, the representation in GPSOR limits the maximum number of features selected from each cluster, which successfully helped it reduce the chance of selecting a large feature subset.

Comparing GPSOR with PSOR both of which using the statistical clustering information and the cluster based representation, GPSOR selected a smaller number of features than PSOR on six of the eight datasets, and achieved similar or significantly higher classification accuracy on seven of the eight datasets. The only excepted dataset where PSOR achieved better classification performance than GPSOR is the Ionosphere dataset, but the number of features selected by GPSOR is about three times smaller than that of PSOR. The results suggest that in most cases, by introducing Gaussian distribution to the representation to transform a position vector to a feature subset, GPSOR can further improve the performance in terms of the both the classification performance and the number of features.

### 5.3 Further Discussions

Fig. 2 takes the Wine dataset as an example showing the features selected by PSOR and GPSOR in the 30 independent runs. The horizontal axis shows the index of the 13 features and the vertical axis shows the number of runs (numbers above the bars) that a feature is selected by PSOR or GPSOR in the 30 independent runs.

According to Fig. 2, the features selected by PSOR from the highest to the lowest frequency are Features 10, 7, 1, 3, 2, 6, 12, 11, 8, 4, 9, 5, and 13, while the order is Features 1, 10, 7, 6, 9, 12, 2, 4, 3, 11, 8, 5, and 13 for GPSOR. It can be observed that Features 10, 7 and 1 are important features since they were frequently selected by

both PSOR and GPSOR. In contrast, Features 5 and 13 are most likely useless since they were not selected by the two algorithms at all. To further confirm this, a further comparison with an existing paper [18], where the importance of individual features in the Wine dataset were discussed through a single feature ranking method (not a PSO-based method). It shows that from the most important to the least important, the ranking is Features 7, 10, 12, 1, 9, 11, 6, 2, 13, 5, 4, 3, and 8. Therefore, it further suggests that both PSOR and GPSOR can select the important features (e.g. Features 7 and 10) to reduce the dimensionality of the dataset and improve the classification performance.

According to Table 2, the average number of features selected by PSOR and GPSOR in the 30 runs are almost the same, which are 4.75 and 4.60. However, as can be seen from Fig. 2, the important features, Features 10, 7 and 1, were more frequently selected by GPSOR than by PSOR. This means that PSOR has a higher probability of selecting less important features than GPSOR in some runs. It also shows that the similarity (or consistency) between the 30 feature subsets selected by GPSOR in the 30 runs is higher than that of PSOR, which suggests GPSOR is more stable than PSOR. The high stability of GPSOR over PSOR was also reflected in Table 2 by the smaller standard deviation values. High stability is very important for a feature selection algorithm, which is key for users to identify and choose important features. The superior stability of GPSOR is mainly contributed by the introduction of the Gaussian distribution, which gives immediate and effective feedback to small movements of particles during the search process. In contrast, in PSOR, if a movement is not big enough to jump from one sub-interval to another, the fitness value of the particle will not change. Small movements toward a promising region in the search space might not be encouraged in PSOR. This limits the search ability and it is easy to be stuck into different local optima, which leads to more various feature subsets (i.e. lower stability) produced by PSOR than GPSOR.

## 6 Conclusions and Future Work

The goal of this paper was to develop a new PSO-based approach to feature selection with the expectation of reducing the number of features and increasing the classification performance. The goal has been successfully achieved by proposing a Gaussian distribution based transformation rule to interpret position vectors as feature subsets in a feature cluster based representation scheme, which smooths the fitness landscape and makes the movement of the particles more meaningful. The proposed algorithm, GPSOR, was compared with two recently developed algorithms, PSOR and GPSO, on commonly used classification tasks of varying difficulty. The experimental results show that the proposed algorithm was able to achieve better performance than these two algorithms in terms of the number of features and the classification performance. Further analyses showed that GPSOR using the Gaussian based transformation rule also increased the stability, i.e. selecting features more consistent than PSOR through different runs.

In many areas, such as gene analyses, the number of features often reaches to thousands or tens of thousands, which are referred to large scale problems. Feature selection is more important and more challenging than with hundreds of features. A novel representation to reduce the dimensionality is essential. Therefore, we intend to develop

novel EC-based approaches to solving such large-scale feature selection problems by investigating novel representation schemes and search mechanisms in the future.

## References


1. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. The Journal of Machine Learning Research **3** (2003) 1157–1182
2. Liu, H., Motoda, H., Setiono, R., Zhao, Z.: Feature selection: An ever evolving frontier in data mining. In: FSDM. Volume 10 of JMLR Proceedings. (2010) 4–13
3. Marill, T., Green, D.: On the effectiveness of receptors in recognition systems. IEEE Transactions on Information Theory **9**(1) (1963) 11–17
4. Matechou, E., Liu, I., Pledger, S., Arnold, R.: Biclustering models for ordinal data. Presentation at the NZ Statistical Assn. Annual Conference, University of Auckland (2011)
5. Pledger, S., Arnold, R.: Multivariate methods using mixtures: Correspondence analysis, scaling and pattern-detection. Computational Statistics & Data Analysis **71** (2014) 241–261
6. Lane, M., Xue, B., Liu, I., Zhang, M.: Gaussian based PSO and statistical clustering for feature selection. In: Evolutionary Computation in Combinatorial Optimisation (EvoCOP 2014). Volume 8600 of Lecture Notes in Computer Science. (2014) 133–144
7. Nguyen, H., Xue, B., Liu, I., Zhang, M.: PSO and statistical clustering for feature selection: A new representation. In: Simulated Evolution and Learning (SEAL 2014). Volume 8886 of Lecture Notes in Computer Science. (2014) 569–581
8. Zhu, Z., Ong, Y.S., Dash, M.: Markov blanket-embedded genetic algorithm for gene selection. Pattern Recognition **40**(11) (2007) 3236–3248
9. Xue, B., Zhang, M., Browne, W.: Novel initialisation and updating mechanisms in PSO for feature selection in classification. In: App. of Evolutionary Computation (EvoApps 2013). Volume 7835 of Lecture Notes in Computer Science. (2013) 428–438
10. Boubezoul, A., Paris, S.: Application of global optimization methods to model and feature selection. Pattern Recognition **45**(10) (2012) 3676 – 3686
11. Xue, B., Zhang, M., Browne, W.N.: Particle swarm optimization for feature selection in classification: A multi-objective approach. IEEE Transactions on Cybernetics **43**(6) (2013) 1656–1671
12. Vieira, S.M., Mendonça, L.F., Farinha, G.J., Sousa, J.M.: Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients. Applied Soft Computing **13**(5) (2013) 3494–3504
13. Xue, B., Fu, W., Zhang, M.: Multi-objective feature selection in classification: A differential evolution approach. In: Simulated Evolution and Learning (SEAL 2014). Volume 8886 of Lecture Notes in Computer Science. (2014) 516–528
14. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: IEEE International Conference on Neural Networks. Volume 4. (1995) 1942–1948
15. Lin, S.W., Ying, K.C., Chen, S.C., Lee, Z.J.: Particle swarm optimization for parameter determination and feature selection of support vector machines. Expert Systems with Applications **35**(4) (2008) 1817–1824
16. Bache, K., Lichman, M.: Uci machine learning repository (2013)
17. Clerc, M., Kennedy, J.: The particle swarm– explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation **6**(1) (2002) 58–73
18. Xue, B., Zhang, M., Browne, W.N.: Single feature ranking and binary PSO based feature subset ranking for feature selection. In: Australasian Computer Science Conference (ACSC 2012) Volume 122, (2012) 27–36