Similarity based Multi-objective Particle Swarm Optimisation for Feature Selection in Classification

Hoai Bach Nguyen, Bing Xue, and Mengjie Zhang

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

Abstract. This paper presents a particle swarm optimisation (PSO) based multi-objective feature selection approach to evolving a set of non-dominated feature subsets and achieving high classification performance. Firstly, a pure multi-objective PSO (named MOPSO-SRD) algorithm, is applied to solve feature selection problems. The results of this algorithm is then used to compare with the proposed a multi-objective PSO algorithm, called MOPSO-SiD. MOPSO-SiD is specially designed for feature selection problems, in which the similarity distance in the feature space is used to select a leader for each particle in the swarm. This distance measure is also used to update the archive set, which will be the final solution for a MOPSO algorithm. The results show that both algorithms successfully evolve a set of non-dominated solutions, which select a small number of features while achieving similar or better performance than using all features. In addition, in most case MOPSO-SiD selects smaller number of features than MOPSO-SRD, and outperforms single objective PSO for feature selection and two traditional feature selection methods.

Keywords: Feature Selection, Classification, Multi-objective Optimisation, Particle Swarm Optimisation

1 Introduction

Classification is one of the most important tasks in machine learning, which aims to predict the class label of an instance based on the value of instance's features. In the learning process, a set of instances, called the training set, is used to train a classification algorithm, which is tested on an unseen dataset, called the test set. In many problems, a large number of features is used to well describe the instances. Unfortunately, due to "the curse of dimensionality", the larger a set of features is, the longer time the training process takes. In addition, relevant features are often unknown without prior knowledge. Therefore, a large number of features often contains irrelevant or redundant features, which are not useful for classification. Those features might lower the quality of the whole feature set [4], because they usually conceal the useful information from the relevant features. Feature selection methods [4, 8] are used to remove those redundant and irrelevant features, which will not only speed up the learning/classification process but also maintain or even increase the classification performance over using all features. However, due to the complex interaction between features and the huge search space, it is hard to develop a good feature selection approach.

The main goal of feature selection is finding a small feature subset from a large set of original features to achieve similar or even better classification performance than using all features. In feature selection, suppose there are n features introduced, then the total number of possible subsets is 2^n. It can be seen that over the large search space, the exhaustive search is too slow to perform in most situations. In order to reduce the searching time, some greedy algorithms such as sequential forward selection [17] and sequential backward selection [10] are developed. However, these methods easily get stuck at the local optima. Because of the global search ability, evolutionary computation (EC) techniques, such as genetic programming (GP) [11], genetic algorithm (GAs) [7] and particle swarm optimization (PSO) [15], have been applied to solve the feature selection problem. Compared with GA and GP, PSO is more preferable because it is simple and easy to implement. In addition, PSO not only uses fewer parameter but also converge more quickly.

Feature selection can be viewed as a multi-objective problem because it needs to maximize the classification accuracy and simultaneously minimize the dimensionality of the selected subset. However, with fewer features being used for classification, the classification accuracy is likely decreased. Those two objectives often conflict with each other and the searching process needs to consider the trade-off between them. EC techniques are particularly suitable for multi-objective optmisation since their population based mechanism can produce multiple trade-off solutions in a single run. However, directly using existing multi-objective approaches to feature selection problems may not achieve promising performance since feature selection has a very complex search space, which requires a specifically designed multi-objective algorithm to solve the problem.

1.1 Goals

The overall goal of this study is to develop a PSO based multi-objective feature selection approach, which can produce a set of non-dominated solutions that select a small number of features and achieve better classification performance than using all features. To achieve this goal, we firstly directly apply a very recently developed multi-objective PSO (MOPSO), called MOPSO-SRD [6] to solve feature selection problem. After that, we develop a new MOPSO algorithm, called MOPSO-SiD, which is specifically designed for feature selection problems. This algorithm will then be compared with MOPSO-SRD results. Specifically, we will investigate

- whether two multi-objective PSO algorithms (MOPSOs) can be applied to evolve a set of non-dominated solutions with a small number of features and better classification performance than using all features and single objective feature selection methods;
- whether MOPSO-SiD, a MOPSO algorithm is designed specifically for feature selection problems, can produce better *Pareto front* than MOPSO-SRD.

2 Background

2.1 Particle Swarm Optimisation (PSO)

Particle Swarm Optimisation (PSO) [5] is an evolutionary computation method, which is inspired by social behaviours such as bird flocking and fish schooling.

In PSO, a problem is optimized by using a population, called swarm, of candidate solutions, which are called particles. In order to find the optimal solution, each particle moves around the search space by updating its position as well as velocity. Particularly, the current position of particle 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 $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$, which is limited by a predefined maximum velocity, v_{max} and $v_{id} \in [-v_{max}, v_{max}]$. During the search process, each particle maintains a record of the position of its previous best performance, called *pbest*. The best position of its neighbours is also recorded, which is called *gbest*. The position and velocity of each particle are updated according to the following equations:

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

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

where t denotes the t^{th} iteration in the search process, d is the d^{th} dimension in the search space, i is the index of particle, 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], p_{id} and p_{gd} represent the position entry of *pbest* and *gbest* in the d^{th} dimension, respectively.

2.2 Related Work on Feature Selection

Traditional Feature Selection Methods Sequential search techniques are also applied to solve feature selection problems. In particular, sequential forward selection (SFS) [17] and sequential backward selection (SBS) [10] are proposed. At each step of selection process, SFS (or SBS) adds (or removes) a feature from an empty (full) feature set. Although these local search techniques achieve better performance than the feature ranking method, they might suffer "nesting" problem, in which once a feature is added (or removed) from the feature set, it cannot be removed (or added) later. In order to avoid nesting effect, Stearns [14] proposed a "plus-l-takeaway-r" method in which SFS was applied l times forward and then SBS was applied for r back tracking steps. However, it is challenge to determine the best values of (l,r). This problem is addressed by sequential backward floating selection (SBFS) and sequential forward floating selection (SFFS), proposed by Pudil, et al.[12]. In SBFS ad SFFS, the values (l, r) are dynamically determined rather than being fixed in the "plus-l-takeaway-r" method.

EC Approaches to Feature Selection EC techniques have recently been used to solve feature selection problems due to their powerful global search abilities, such as GAs [7], GP [11], and PSO [15]. Muni et al. [11] developed a wrapper feature selection model based on multi-tree GP, which simultaneously selected a good feature subset and learned a classifier using the selected features. Two new crossover operations were introduced to increase the performance of GP for feature selection. Based on the two crossover operations introduced by Muni et al. [11], Purohit et al. [13] further introduced another crossover operator, which was randomly performed for selecting a subtree from the first parent and finding its best place in the second parent. Lin et al. [7] proposed a GA-based feature selection algorithm adopting domain knowledge of financial distress prediction, where features were classified into

different groups and a GA was used to search for subsets consisting of top candidate features from each group.

To avoid premature convergence in PSO, Chuang, et al. [2] proposed a new *gbest* updating mechanism, which resets *gbest* elements to zero if it maintains the same value after several iterations. However, the performance of this algorithm is not compared with other PSO based algorithms. Tran et al. [15] used the *gbest* resetting mechanism in [3] to reduce the number of features and performed a local search process on *pbest* to increase the classification performance of PSO for feature selection. Each evaluation in the local search was sped up by calculating fitness based only on the features being changed (from selected to not selected or from not selected to selected) instead of based on all the selected features. The proposed algorithm further reduced the number of features and improved the classification performance over [3] and standard PSO. PSO with multiple swarms to share experience has also been applied to feature selection [9], but may lead to the problem of high computational cost.

Two multi-objective PSO algorithms were used to solve feature selection problems [18]. The first algorithm applied the idea of non-dominated sorting based multi-objective genetic algorithm II (NSGAII) into PSO for feature selection. The other algorithm bases on the idea of crowding, mutation and dominance to evolve the Pareto front solutions. According to the experimental results, both algorithms can select a small number of features while achieving better classification performance than using all features. However, the above algorithms did not propose any specific design for feature selection problems. Therefore , this work will propose a new multi-objective PSO algorithm, which is specifically designed for feature selection problems.

3 Proposed approach

In feature selection problems, there are two main goals, which are minimising the number of selected features while maximising the classification performance. However these objectives are usually conflicting and there is trade-off between them. Particle Swarm Optimisation (PSO) is originally proposed to deal with single objective problems, therefore some multi-objective PSO approaches (MOPSO) are proposed to solve multi-objective problems. In MOPSO algorithms, instead of recording *gbest* for each particle, an archive set is used to maintain a set of non-dominated solutions being discovered so far. Most of the existing MOPSO algorithms are different in terms of the way to control this archive set as well as how to select a good leader (*gbest*) for the swarm among the archive set.

Although there are many works which apply MOPSO to solve feature selection problems, most of them do not consider the properties of feature selection problems. For example, in the normal MOPSO approach, if two particles have exactly same objective values, they are considered identical particles. Therefore one of the particle will not be added into the archive set. Meanwhile, in feature selection problems, two particles might select the same number of features as well as achieve the same classification accuracy, but the features being selected by each particle might be different. From the perspective of feature selection problems, these particles are still different in term of which features being selected. In feature selection problems,

beside the two main objectives, which features being selected is also important. Therefore, in this study we propose a new algorithm called MOPSO using the Similarity Distance (MOPSO-SiD). This algorithm makes two main contributions by adding a new leader selection algorithm and a new control mechanism for the archive set.

3.1 Leader Selection Algorithm

The main difference between PSO and MOPSO is how each particle selects its *gbest*. In PSO, each particle records its own *gbest*, which is the best position being discover by it and its neighbours. However in MOPSO, each particle will select its *gbest* from an archive set, which contains all non-dominated solutions being discovered so far. In MOPSO-SiD, for each generation, each particle freely selects its own leader by using the similarity distance calculation. Given two particles p_1 and p_2, the similarity distance (SiD) between these particles is calculated according to the Equation 3.

$$SiD(p_1, p_2) = \sum_{i=1}^{n} \sqrt{(x_{1i} - x_{2i})^2} \tag{3}$$

where n is the total number of features (i.e. length of each position vector), x_{1i}, x_{2i} are the i^{th} position entries of two particles p_1, p_2 respectively.

In each generation, for each particle in the search swarm, the similarity distance (SiD) between the particle and all archive members is calculated. After that, the archive member with the shortest SiD is chosen as the leader of that particle.

This distance measure (SiD) is especially good at the early iterations comparing with SRD in MOPSO-SRD. As mentioned above, MOPSO-SRD selects the leader basing on the distance of objective values. In other word, MOPSO-SRD or even MOPSO-CD only consider the objective space. For example, in MOPSO-SRD, a particle might select a archive member, which is the closest to it in the objective space. However in the feature space (search space), the selected archive member might be very far way from the particle if their selected features are different. Comparing with MOPSO-SRD, MOPSO-SiD provides more exploitation ability by selecting the closest archive member in terms of the position distance rather than the objective distance.

3.2 Archive Control Algorithm

Controlling the archive set is also an important part of a MOPSO algorithms. The controlling mechanism aims to decide whether or not a solution is added to the archive set or which solution should be removed from the archive set when this set is full. In general, a solution S is added to the archive set if it is not dominated by any archive members. This rule is still applied in MOPSO-Sid. However, if there is at least one archive member, which has the same objective values as the solution S, whether or not S will be added into the archive set. In MOPSO-SRD and MOPSO-CD, S will not be added to the archive set since these algorithm only consider the objective values. In feature selection problems, the situation can be different. Suppose that two particles might select the same number of features and achieve

the same classification, their selected features can still be different. This mean that those particles might be at the same position in the objective space but they are on different position in the feature space (search space). Compare with MOPSO-SRD and MOPSO-CD, MOPSO-SiD does consider about this problem. In particular, if there is an archive member, called A, which has the same objective values as S (solution to be added), MOPSO-SiD will further check the features being selected by both A and S. If the selected features of A and S are different, S will be added into the archive set, otherwise S will be discarded. Once more, MOPSO-SiD considers not only the objective space but also the feature space (search space).

Beside adding solutions, removing solutions from the archive set is also important in a MOPSO algorithms. In general, each MOPSO approach will have a measure to rank all solutions with in an archive set. For example, MOPSO-CD uses crowding distance (CD) to rank the solutions to improve the diversity of the archive set. Meanwhile, MOPSO-SRD ranks the archive members according to the square root distance (SRD). However these distance measures only considers the objective space, which might not be sufficient in feature selection problems. For instance, two particles which are close in the feature space (similar selected features) usually have similar classification accuracy as well as the number of selected features. However, two particles which are close in the objective space (similar classification accuracy and number of selected features) might select very different features. Therefore instead of using the crowding (CD) or square root distance (SRD), MOPOS-SiD uses the similarity between particles in the feature space to rank all archive members.

In particular, when the archive set is full, the similarity distance between each pair of archive members are calculated according to the Equation 3. After that, MOPSO-SiD will select a pair of archive members with the shortest similarity distance, which means that these members are the most similar pair in terms of feature being selected. Since in feature selection problems, the classification accuracy is preferable when the number of selected features is similar, MOPSO-SiD will remove the archive member with lower classification accuracy among the above selected pair of archive members. In general, MOPSO-SiD considers not only the objective values but also which features are selected by each particle, which is also important in feature selection problems.

3.3 The MOPSO-SiD Agorithm

In MOPSO-SiD for feature selection, the similarity distance (SiD) and the continuous multi-objective PSO is applied to search for the non-dominated solutions. The representation of each particle is a vector of n real numbers, where n is the total number of features. Each position entry $x_i \in [0, 1]$ corresponds to the i^{th} feature in the original feature set. A threshold θ is used to decide whether or not a feature is selected. In particular, the i^{th} feature is selected if and only if $\theta < x_i$. The two objective is to minimise the number of features and the classification performance. Algorithm 1 shows the pseudo-code of MOPSO-SiD.

4 Experimental Design

Eight datasets (Table 1) chosen from the UCI machine learning repository [1] are used in the experiments. These datasets have different number of features, classes

Algorithm 1 : Pseudo-code of MOPSO-SiD

begin
initialize the swarm and Archive A = {};
while *Maximum iterations* is not reached **do**
  for each particle i in the swarm **do**
    update the *pbest* of particle i;
    select the archive member with the shortest SiD as its *gbest*;
    update the velocity and the position of particle i;
    mutation;
    evaluate particles;
    if the i^{th} particle is not dominated by any archive members **then**
      insert $i^{th} particle into A$;
    end if
  end for
  if A is full **then**
    compute SiD between all pairs of archive members;
    select a pair with the shortest SiD;
    remove the archive member (among the selected pair) with lower accuracy;
  end if
end while
calculate the testing classification error rate of the solutions in A (archive set);
return the position of particles in A;
return the training and test classification error rates of the solutions in A; **end**

Table 1: Datasets.

Dataset	#features	#classes	#instances
Vehicle	18	4	946
WBCD	30	2	569
Ionosphere	34	2	351
Lung	56	4	32
Sonar	60	2	208
Movementlibras	90	15	360
Musk1	166	2	476
Arrhythmia	279	16	452

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.

In both MOPSO-SRD and MOPOS-SiD, the parameters are set as follows [16]: $w = 0.7298, c_1 = c_2 = 1.49618, v_{max} = 0.2$, population size is 30, the maximum number of iterations is 100. The threshold used in the continuous version of PSO for feature selection is set to 0.6. For each dataset, each algorithm has been conducted for 50 independent runs. After each run a set of non-dominated solutions are obtained. In order to compare MOPSO-SRD and MOPSO-SiD, firstly all 50 archive

sets are combined together to create an union set. In this union set, the classification error rate of feature subsets, which share the same number of features, are averaged. A set of average solutions is obtained by using the average classification error rate and the corresponding number of features. This average set is called the *average Pareto front*. In addition, for each dataset, all non-dominated solutions are selected from the union set to create a set of *best* solutions, called *best set*. A pure continuous PSO for feature slection problems is also ran 50 independent times on these above datasets. Each independent run produces a set of selected features. The average as well as the best solutions are obtained from the 50 solutions.

5 Results And Discussions

Experiment results of two algorithms on training and test set are shown in Figures 1 and 2, respectively. In each figure, the total number of original features and the classification error rate when all features being used are shown in the brackets. In each chart, the horizontal axis shows the number of selected features and the vertical axis shows the classification error rate. In figure 1, "SRD-Train-Ave" ("SiD-Train-Ave") stands for the average Pareto front resulted from MOPSO-SRD (MOPSO-SiD) in the 50 independent runs. "SRD-Train-Best" ("SiD-Test-Best") represents the non-dominated solutions of all solutions resulted from MOPSO-SRD (MOPSO-SiD). The results of single objective PSO for feature selection is shown as SOPSO in the figure. Figure 2 shows the same information as in Figure 1 but the classification error rates are calculated on the test set. The results of SFS are shown in Table 2.

5.1 MOPSO-SiD vs All Features

According to Figure 2, in all datasets, "SRD-Test-Ave" and "SiD-Test-Ave"contain at least one solution, which selects no more than 30% of the available features and achieves similar or better performance than using all features. In all datasets, both "SRD-Test-Best" and "SiD-Test-Best" contains one or more solution, which select around 8% of the available features and achieves similar or better performance than using all features.

The results suggest that in all datasets, both MOPSO-SRD and MOPSO-SiD can evolve a set of features subsets with a small number of features and better classification performance than using all features.

5.2 MOPSO-SiD vs SOPSO

As can be seen from both Figure 1 and 2, on most of datasets, both MOPSO-SRD and MOPSO-SiD can evolve at least one solution, which selects smaller number of features while achieving better classification accuracy than SFS approach. On Musk1 and Arrhythmia dataset, although SFS selects smaller number of features than the multi-objective PSO approaches, its solutions' classification accuracy is even worse than the worst solution of both MOPSO algorithms. This is because MOPSO does consider about the interaction between features, which SFS skips.

Comparing with pure PSO for feature selection, on most of datasets, both MOPSO approaches can find out better solutions. In particular, MOPSO approaches

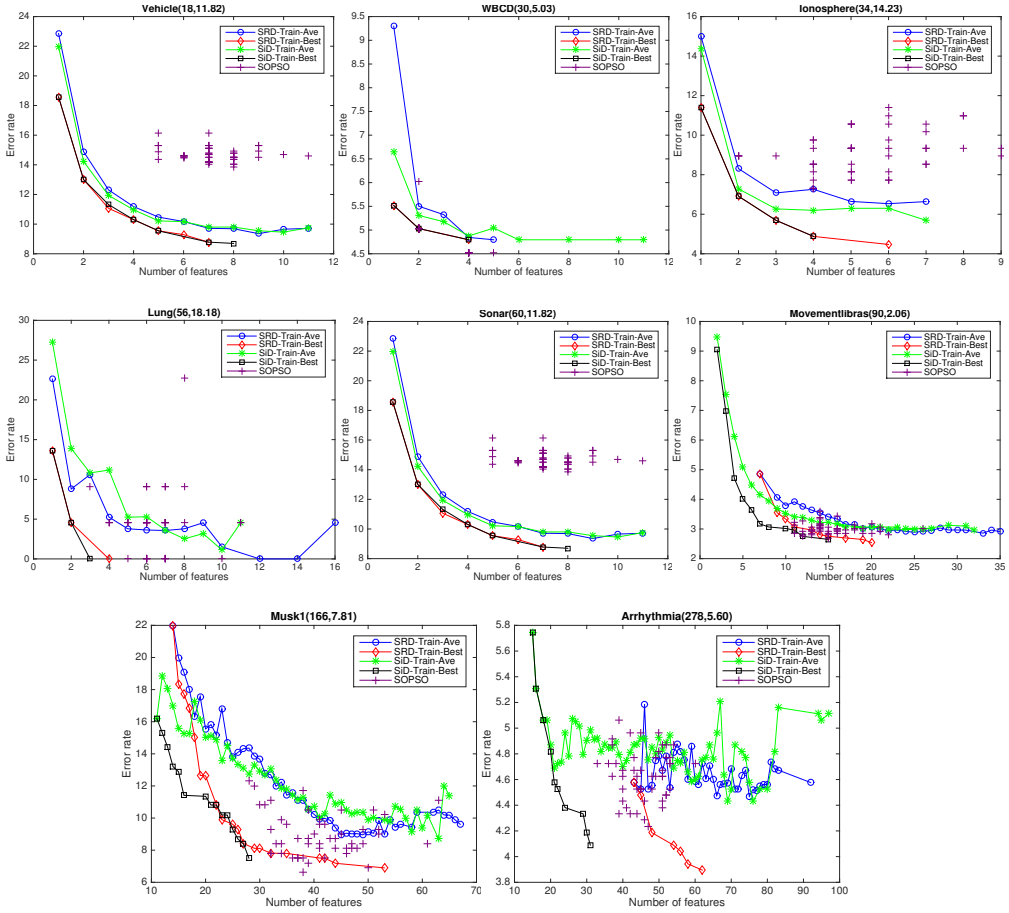

Fig. 1: Results of MOPSO-SRD and MOPSO-SiD on training set (In colour).

can evolve at least one solution that selects smaller number features and achieves better performance than the best solution evolved by the pure PSO.

5.3 MOPSO-SiD vs MOPSO-SRD

Firstly, let consider the training results in Figure 1, which show the searching ability of these two algorithms. As can be seen in Figure 1, the patterns of both "SiD-Train-Ave" and "SRD-Train-Ave" are similar. However, "SiD-Train-Ave" oscillates more then "SRD-Train-Ave", which is due to the *gbest* selection mechanism. On the one hand, MOPSO-SRD concentrates more on the objective values to select *gbest*. Meanwhile, MOPSO-SiD selects *gbest* by mainly using the similarity in the feature search space. In addition, in all dataset, the "SiD-Train-Ave" line is mostly on the left of "SRD-Train-Ave" line, which means that with MOPSO-SiD usually selects smaller number of features than MOPOS-SRD to achieve similar classification performance.

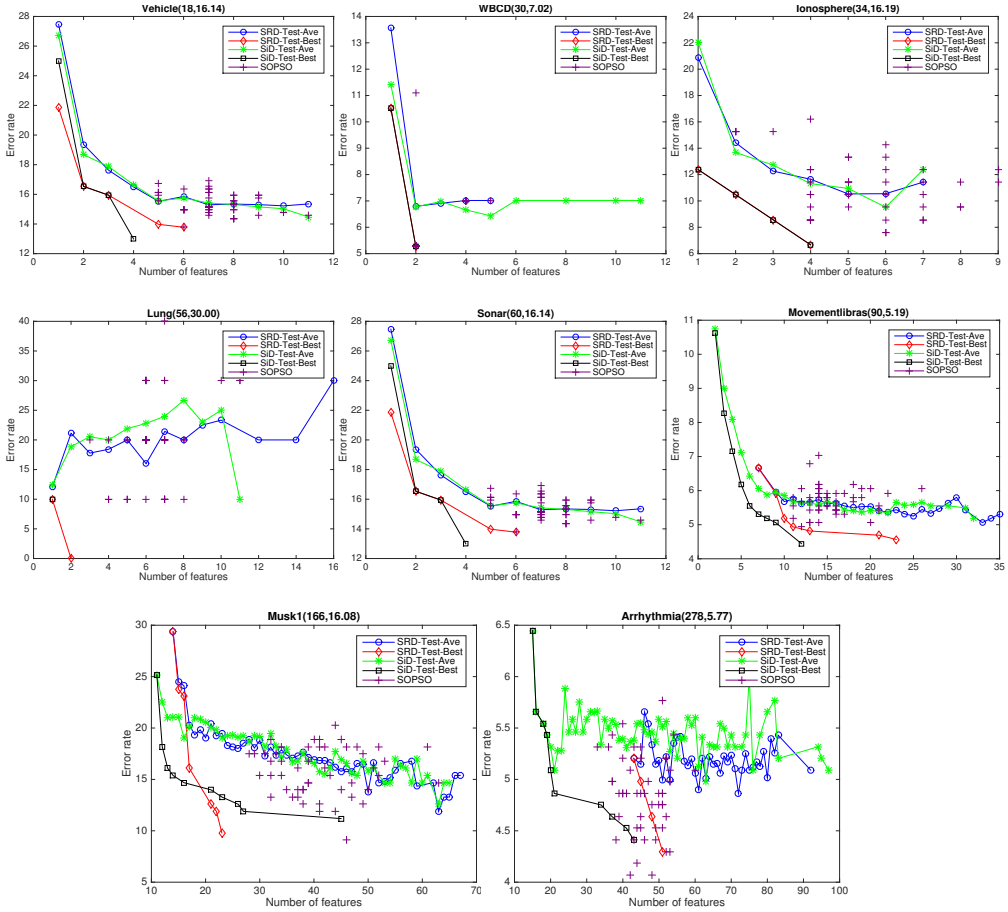

Fig. 2: Results of MOPSO-SRD and MOPSO-SiD on test set (In colour).

In terms of the best solutions, MOPSO-SiD outperforms MOPSO-SRD. As can be seen from Figure 1, in all datasets, the "SiD-Train-Best" lines are at the same position or on the left of the "SRD-Train-Best" lines, which indicates that MOPSO-SiD can evolve a smaller subset of features while achieving similar performance as MOPSO-SRD.

Figure 2 shows the results on the test set. In this figure, both lines "SRD-Test-Ave" and "SiD-Test-Ave" are even closer and more similar than in Figure 1. On two small datasets (WBCD and Movementlibras), "SiD-Test-Best" is at the same position or on the left of "SRD-Test-Best". So for the same number of features, MOPSO-SiD can achieve better classification performance than MOPSO-SRD. However, in the Musk1 dataset, when the number of features exceeds about 13% of the available features, MOPSO-SRD achieves better performance than MOPSO-SiD. These points with high classification accuracy and a large number of selected features are usually discovered at the end of each run. The similarity distance is very helpful

Table 2: Results of SFS.

Dataset	#Features	Train Error(%)	Test Error(%)
Vehicle	5	13.0	18.3
WBCD	1	5.5	11.1
Ionosphere	2	9.4	21.0
Lung	1	13.6	10.0
Sonar	5	19.4	27.0
Movementlibras	7	3.9	7.7
Musk1	1	22.8	27
Arrhythmia	3	5.4	7.0

at the beginning of each run, when most of particles in the swarm are at different position and the exploitation ability is more important. However, in the later iterations, when the particles in the swarm and archive set become similar, the exploration ability is more important. Compare with MOPSO-SiD, MOPSO-SRD provides more exploration ability. For instance, in MOPSO-SiD, a particle in the swarm always selects the closest archive member in the feature space as its leader. At the end of a run, the leader might be very similar to the particle, and therefore the particle is trapped at that position. On the other hand, in MOPSO-SRD, a leader is selected by using the square root distance in the objective space. In this case, although the particle and its leader are similar in term of objective values, they still can select very different features (different positions in feature space). Therefore, the particle has a chance to get out of the current position (probably a local optima). This explains why on the large datasets, MOPSO-SRD can discover points on the objective space, where the number of selected features is high and the classification accuracy is better.

6 Conclusion and Future Work

The goal of this study was to develop a PSO based multi-objective Feature Selection approach to evolving a set of non-dominated feature subsets and achieving high classification performance. Firstly, MOPSO-SRD, a pure multi-objective PSO, is applied to solve feature selection problems. The results of this algorithm is then used to compare with my proposed MOPSO algorithm, called MOPSO-SiD. MOPSO-SiD is specially designed for feature selection problems, in which the similarity distance in the feature space is used to select a *leader* for each particle in the swarm. This distance measure is also used to update the archive set, which will be the final solution for a MOPSO algorithm. The results show that both algorithms successfully evolve a set of non-dominated solutions, which select a small number of features while achieving similar or better performance than using all features. In addition, in most case MOPSO-SiD selects smaller number of features than MOPSO-SRD but still achieves similar classification performance.

This works starts incorporates the characteristics of feature selection problems into the multi-objective search to find a better Pareto front, which show some success. In the future, we will further improve the exploration and exploitation

abilities of multi-objective PSO for feature selection by embedding some genetic operators or a local search during the search process.

References

1. Bache, K., Lichman, M.: Uci machine learning repository (2013), `http://archive.ics.uci.edu/ml`
2. Chuang, L.Y., Chang, H.W., Tu, C.J., Yang, C.H.: Improved binary pso for feature selection using gene expression data. Computational Biology and Chemistry 32(1), 29–38 (2008)
3. Chuang, L.Y., Chang, H.W., Tu, C.J., Yang, C.H.: Improved binary PSO for feature selection using gene expression data. Computational Biology and Chemistry 32(29), 29– 38 (2008)
4. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. The Journal of Machine Learning Research 3, 1157–1182 (2003)
5. Kennedy, J., Eberhart, R., et al.: Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks. vol. 4, pp. 1942–1948. Perth, Australia (1995)
6. Leung, M.F., Ng, S.C., Cheung, C.C., Lui, A.: A new strategy for finding good local guides in mopso. In: Evolutionary Computation (CEC), 2014 IEEE Congress on. pp. 1990–1997 (July 2014)
7. Lin, F., Liang, D., Yeh, C.C., Huang, J.C.: Novel feature selection methods to financial distress prediction. Expert Systems with Applications 41(5), 2472–2483 (2014)
8. Liu, H., Motoda, H., Setiono, R., Zhao, Z.: Feature selection: An ever evolving frontier in data mining. In: Feature Selection for Data Mining. JMLR Proceedings, vol. 10, pp. 4–13. JMLR.org (2010)
9. Liu, Y., Wang, G., Chen, H., Dong, H.: An improved particle swarm optimization for feature selection. Journal of Bionic Engineering 8(2), 191–200 (2011)
10. Marill, T., Green, D.M.: On the effectiveness of receptors in recognition systems. Information Theory, IEEE Transactions on 9(1), 11–17 (1963)
11. Muni, D., Pal, N., Das, J.: Genetic programming for simultaneous feature selection and classifier design. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 36(1), 106–117 (2006)
12. Pudil, P., Novovičová, J., Kittler, J.: Floating search methods in feature selection. Pattern recognition letters 15(11), 1119–1125 (1994)
13. Purohit, A., Chaudhari, N., Tiwari, A.: Construction of classifier with feature selection based on genetic programming. In: IEEE Congress on Evolutionary Computation (CEC'10). pp. 1–5 (2010)
14. Stearns, S.D.: On selecting features for pattern classifiers. In: Proceedings of the 3rd International Conference on Pattern Recognition (ICPR 1976). pp. 71–75. Coronado, CA (1976)
15. Tran, B., Xue, B., Zhang, M.: Improved pso for feature selection on high-dimensional datasets. In: Simulated Evolution and Learning, Lecture Notes in Computer Science, vol. 8886, pp. 503–515. Springer International Publishing (2014)
16. Van Den Bergh, F.: An analysis of particle swarm optimizers. Ph.D. thesis, University of Pretoria (2006)
17. Whitney, A.W.: A direct method of nonparametric measurement selection. Computers, IEEE Transactions on 100(9), 1100–1103 (1971)
18. Xue, B., Cervante, L., Shang, L., Browne, W.N., Zhang, M.: A multi-objective particle swarm optimisation for filter-based feature selection in classification problems. Connection Science 24(2-3), 91–116 (2012)