VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@ecs.vuw.ac.nz

Feature Selection using Particle Swarm Optimisation: A Filter Approach

Liam Cervante

Supervisor: Mengjie Zhang

October 19, 2012

Submitted in partial fulfilment of the requirements for
Bachelor of Science with Honours in Computer Science.

Abstract

Often large numbers of features are introduced into data to describe patterns, because of this it can be difficult to extract and construct relevant features to perform a particular class, such as classification. Particle swarm optimisation has been shown to be a promising search strategy in feature selection. This project aims to test the effectiveness of rough set theory as a fitness function in a filter approach in conjunction with single and multi-objective particle swarm optimisation. Probabilistic rough set theory has been tested and combined with additional measures aimed to reduce the number of features further. Experimental results have shown that the number of features can be reduced down to 33% of the total number, while improving accuracy. Where overall improvement was not possible, it was possible to maintain the classification performance while reducing the number of features further. Multi-objective particle swarm optimisation can further improve upon the single objective approach. This project represents the first work using particle swarm optimisation and probabilistic rough set theory in a filter approach to feature selection.

Acknowledgements

I would like to thank Bing Xue has been working with me closely, in addition to Mengjie Zhang. Also, Lin Shang helped me form an initial understaning of rough set theory. I would also to thank the reviewers for their feedback throughout the year.

Contents

1 Introduction **1**
- 1.1 Goals . . . 2
- 1.2 Major Contributions . . . 2
- 1.3 Organisation . . . 3

2 Background **5**
- 2.1 Machine Learning and Classification . . . 5
- 2.2 Evolutionary Computation . . . 6
- 2.3 Particle Swarm Optimisation . . . 7
  - 2.3.1 Binary Particle Swarm Optimisation . . . 7
- 2.4 Rough Set Theory . . . 8
  - 2.4.1 Deterministic Rough Set Theory . . . 8
  - 2.4.2 Probabilistic Rough Set Theory . . . 9
- 2.5 Multi-objective Optimisation . . . 9
- 2.6 Related Work . . . 10
  - 2.6.1 Traditional Feature Selection Approaches . . . 10
  - 2.6.2 EC Algorithms (non-PSO) for Attribute Selection . . . 10
  - 2.6.3 PSO Based Approaches to Feature Selection . . . 11

3 Single Objective Feature Selection Algorithms based on PSO and RS **13**
- 3.1 Introduction . . . 13
  - 3.1.1 Chapter Objectives . . . 13
- 3.2 Rough Set Theory for FS . . . 13
  - 3.2.1 Deterministic Rough Set Theory for Feature Selection . . . 14
  - 3.2.2 Probabilistic Rough Set Theory for Feature Selection . . . 14
  - 3.2.3 New Algorithm: $RSPSO_1$ . . . 14
  - 3.2.4 New Algorithm: $RSPSO_2$ . . . 15
  - 3.2.5 Summary of the New Algorithms . . . 15
- 3.3 Experiments and Results . . . 15
  - 3.3.1 Experimental Design . . . 16
  - 3.3.2 Overall Results . . . 17
  - 3.3.3 Varying Parameters for $RSPSO_1$ . . . 18
  - 3.3.4 Varying Parameters for $RSPSO_2$ . . . 21
  - 3.3.5 Comparison of $RSPSO_1$ and $RSPSO_2$ . . . 22
  - 3.3.6 Comparison with Traditional FS Methods . . . 23
- 3.4 Chapter Summary . . . 23

4 Multi-Objective FS Based on PSO and RS **25**
4.1 Introduction . . . 25
4.1.1 Chapter Objectives . . . 25
4.2 Proposed Multi-objective Feature Selection Algorithm . . . 25
4.2.1 New Algorithm: $RSMOPSO_1$ . . . 26
4.2.2 New Algorithm: $RSMOPSO_2$ . . . 26
4.2.3 Summary of New Algorithms . . . 27
4.3 Experiments and Results . . . 28
4.3.1 Experimental Design . . . 28
4.3.2 Overall Results . . . 28
4.3.3 Varying Parameters for $RSMOPSO_1$ . . . 30
4.3.4 Varying Parameters for $RSMOPSO_2$ . . . 32
4.3.5 Comparison of $RSMOPSO_1$ and $RSMOPSO_2$ . . . 32
4.3.6 Comparison with Single-Objective Approach . . . 33
4.4 Chapter Summary . . . 34

5 Conclusions and Future Work **37**
5.1 Conclusions . . . 37
5.1.1 Single Objective Particle Swarm Optimisation . . . 37
5.1.2 Multi-Objective Particle Swarm Optimisation . . . 38
5.2 Future Work . . . 38

A Summer Research **43**

Chapter 1

Introduction

Classification is an important task in machine learning and data mining, which involves classifying a set of instances as one of a number of potential classes. Each instance is represented by a number of features that will be used by a machine learning algorithm to learn a relationship, or a function or classifier, between these features and the class labels. This classifier can then be given to unseen instances for which the true class is not known. In order to better describe these instances and to facilitate the discovery and extraction of patterns, large numbers of features are introduced. This can, however, lead to "the curse of dimensionality" [11] and poorly chosen features have a negative impact. For example, noisy and irrelevant features can degrade the classification performance while redundant features can increase computation cost with no significant gain.

Feature selection (FS), also called dimension reduction and attribute reduction, aims to reduce the noisy, irrelevant and redundant features while preserving or improving the classification performance [14]. By removing the unnecessary attributes, dimension reduction can reduce the training time of a learning algorithm and simplify the learnt classifier [8, 46]. Feature selection is a difficult task, where the size of the search space grows exponentially along with the number of attributes in the dataset. In order to select the best feature subset an efficient search technique is needed to explore the solution space and an evaluation criterion needs to be defined to rate and select the best feature subset.

Many different search techniques have been applied to feature selection. Simple greedy approaches such as forward selection and backward elimination [8] have been proposed. Forward selection starts with an empty set of features and greedily chooses the next best feature, while backward elimination starts with the full set of features and removes features gradually [8]. These greedy approaches suffer from the problem of stagnation in local optima, alternate *brute force* approaches also suffer from being computationally expensive [8, 55]. In order to better address dimension reduction problems, an efficient global search technique is needed. Evolutionary computation (EC) techniques are well-known for their global search ability. Particle swarm optimisation (PSO) [17, 42] is a relatively recent EC technique, which is computationally less expensive than other EC algorithms. Therefore, PSO has been used as an effective technique in dimension reduction [46, 26, 30].

Based on whether a learning/classification algorithm is included in the fitness evaluation, existing FS algorithms can be broadly classified into two categories: wrapper approaches and filter approaches [8, 6]. Wrapper approaches embed a classification algorithm as part of the evaluation criterion while filter approaches operate independently of a learning algorithm. It has been argued that because of the difference, wrapper approaches can often achieve better results than filter approaches, but they are computationally more expensive and less general in comparison [19]. PSO has been successfully applied to address dimension reduction problems. However, most of the existing PSO based feature selec-

tion algorithms rely on a wrapper approach. Although wrappers can achieve better performance, the use of such algorithms is limited in real-world applications because of their high computational cost. The development of PSO based filter feature selection approaches still remains an open issue.

Numerous mathematical theories have been applied to filter fitness functions, including information measures [10], dependency measures [53], consistency measures [54] and distance measures [24]. Rough set theory has been applied to attribute reduction [47] as a filter approach. However, traditional rough set theory has certain limitations [52]. Probabilistic rough set can overcome such limitations and from a theoretical point of view, Yao and Zhao [52] have shown that probabilistic rough set can be a good measure in attribute reduction, but its performance for dimension reduction has not been reported.

1.1 Goals

This project aims to investigate a new filter approach to feature selection for classification using particle swarm optimisation and rough set theory. The main goal is to develop new fitness functions that can be used to reduce the number of features and improve classification performance. Specific milestones in the project are:

1. Develop a single objective filter algorithm to FS based on PSO and RS, and investigate whether such a measure can outperform the conventional information gain measures for feature selection.

2. Propose a multi-objective method for filter FS based on PSO and rough set theory, and investigate whether the multi-objective PSO method can select a small number of features and achieve a similar or even better classification performance.

1.2 Major Contributions

This work shows how a novel filter based single objective feature selection approach based on PSO and rough set can be developed. Two algorithm are proposed that linearly combines two objectives, maximising the classification performance and minimising the number of features, into one fitness function. A rough set measure is used in both algorithms as a filter fitness measure to estimate the goodness of the features. The first algorithm considers the count of the number of selected features as the second objective. The second algorithm uses the definitions of rough set theory to minimise the number of features rather than a direct count. In several benchmark datasets the number of features used in analysis could be decreased while maintaining or improving the classification performance. In the Chess and Dermatology datasets it was possible to reduce the number of features to 33% of the original size while improving the performance across different classification algorithms.

The single objective approach has been accepted by 25th Australasian Joint Conference on Artificial Intelligence:

- Liam Cervante, Bing Xue, Lin Shang and Mengjie Zhang. "A Dimension Reduction Approach to Classification Based on Particle Swarm Optimisation and Rough Set Theory". *Proceedings of the 25th Australasian Joint Conference on Artificial Intelligence. Lecture Notes in Artificial Intelligence*. Springer. Sydney, Australia, December 2012. (To appear)

This work also shows how a new filter based multi-objective feature selection approach based on PSO and rough set. Two algorithms are again proposed, these are based on the single objective approaches. Both multi-objective approaches share the rough set measure

of fitness as one objective. The first algorithm uses the number of features as the second objective, while the second uses the definitions of rough set theory to measure the number of features. The multi-objective approaches return a set of solutions rather than the single solution returned by the single objective approach. This set can be used to trade off between performance and efficiency. The best performing solutions returned by the multi-objective approach can be used if classification performance is the key issue. The multi-objective approaches further lower the number of features in solutions, beyond the reduction achieved by the single objective approach.

Another paper is under preparation for EvoStar entitled "A Multi-Objective Feature Selection Approach using Particle Swarm Optimisation and Rough Set Theory for Classification". This paper will present the results of the multi-objective approaches

1.3 Organisation

The report is organised as follows. Chapter 2 presents from theoretical and mathematical background to classification, rough set theory and evolutionary computing. This chapter also presents some related work on feature selection using approaches based in multiple areas. Chapter 3 presents the single objective algorithms that linearly combine two objectives. Provided is the experimental design and results of using these algorithms. Chapter 4 presents the multi-objective approaches that search and return a pareto front. Chapter 5 concludes the report and presents a small section on possible future work. The bibliography and appendices follow.

Chapter 2

Background

This chapter provides background information about machine learning, particle swarm optimisation, rough set theory, typical feature selection algorithms, and existing work closely related to this project.

2.1 Machine Learning and Classification

Classification problems are usually solved by supervised learning techniques. The precise and correct output for a given input is known, in the case of classification the class labels of the instances are known [40, 29]. Typically classification algorithms are trained using a *training set* to produce a classifier or learned model. This learned model can then be passed to new and unseen instances and attempt to classify them. The performance of a classification algorithm can be evaluated on a *testing set*, which contains a set of never before seen instances.

There are many types of machine learning and classification algorithms. The *k-nearest neighbour* (kNN) classifier can be considered an instance-based learning algorithm [20]. Instances within a dataset that exist in close proximity will share similar properties. The unseen instances can be classified based on the instances they are close to in the training set, based on some distance measure such as Euclidean distance. kNN is considered an instance-based algorithm because a model or classification function is not produced in training, the instances in the training set are used directly. The *naive Bayes* classifier produces a model than can be used to classify the instances in the testing set [20]. The produced model is based on Bayesian probability, what is the likelihood of an instance being a certain class based on the values of the attributes.

The *decision tree* algorithm [40] will build a tree that can be used to evaluate instances. The leaves of a decision tree then return a particular class label. The tree is built using the training set, when an unseen instance is to be evaluated the tree can be traced down based on the attribute values in the unseen instance. When the trace reaches a leaf the class label of the leaf is returned as the class of the instance. A measure of impurity or uncertainty, such as Shannon's entropy [41], can be used to decide on which attributes to consider at each node. The attributes that provide the most information about a given class should be considered earlier to minimise the size of the tree. Decision tree classifiers are simple to understand and interpret and have the ability to perform internal feature selection. Not all attributes are guaranteed to be used in the decision tree.

Machine learning is the process of developing an algorithm or parameters to an algorithm over time [40]. Typically it involves three steps.

1. The generation of a solution to a problem

2. The evaluation of the solution

3. The solution, if it is not good enough, is improved in some way and step (2) is repeated.

Machine learning can be applied to classification problems. The performance of the built classifier can be evaluated, and if the classification performance was not adequate the parameters or algorithm can be changed slightly to improve performance further.

Feed-forward neural networks are an example of a machine learning technique [39], based on the model of a biological brain. In classification the feature values will be passed to some input nodes, the input layer, of the network. Each node in the network has parameters that are to perform transformation to the input before they are passed to the next set of nodes. The layers pass the input parameters along applying transformation at each stage. The output of the final nodes are used to interpret as the final classification label, typically for an n class classification problem there will be n output nodes, the output node with the largest value is the final classification of the input.

2.2 Evolutionary Computation

Evolutionary Computation (EC) techniques rely on Darwinian principles for automated problem solving. In evolutionary computation a population of potential solutions, called individuals, are generated. These individuals then interact with each other to explore different solutions and return the best. Some examples of evolutionary computation are provided here.

Genetic algorithms (GAs) attempt to mimic the process of natural evolution [12, 13, 16]. Solutions are encoded into bit strings and evolutionary concepts such as inheritance, mutation and crossover are then applied to the population. The solutions are evaluated and the best solutions, as measured by a fitness function, are selected to move into the *breeding pool*. These solutions are used to generate the next generation by the evolutionary concepts mentioned above, points in the bit string can be selected and mutated or swapped with other solutions. The evaluation is then repeated, in this way the best individuals are selected and improved upon until the preferred solution is found. GAs can be naturally applied to feature selection, the bit strings can represent which features have been selected and the fitness would be the classification performance of the selected features using a classification algorithm.

Genetic Programming (GP) is a specialisation of genetic algorithm where each individual is a computer program rather than a bit string [23, 22, 21]. The progams are evaluated using a fitness function as for GAs. The evolutionary concepts of GAs are also applied, but the process is somewhat more complex given the representation of a solution in GP is more complex than the GA bit strings. Traditionally GP solutions are represented as tree based structures [7]. Nodes and leaves can then be selected randomly for the breeding process, rather than points in the bit string.

Evolutionary algorithms, as mentioned above, typically use concepts inspired by evolution. Swarm intelligence involves the solutions interacting locally with each other and the environment. Rather than individuals merging and producing offspring, the individuals update themselves based on their surroundings and other individuals. The concepts involved in swarm intelligence are inspired by the social interactions of biological creatures. Examples of natural swarm intelligence include ant colonies, birds flocking and fish schooling.

Ant colony optimization (ACO) is a swarm intelligence graph search technique inspired by the behaviour of ants searching for a source of food [9]. A *searcher* ant will find a source

of food and return to the colony, leaving a trail of pheromones for other ants to follow. Over time other searcher ants may find the same food source by a different route, leading to two paths to the same food source. Ants searching for food follow these pheromone trails, when food is found they return and strengthen them. As the pheromone trail is volatile, the shorter route will gain a stronger pheromone trail. The longer trail will lose pheromone faster as the ants take longer to leave additional pheromones. This will encourage more ants to follow the shorter route, optimizing the path to the food source. This behaviour can be used to search for the ideal route in a graph search problem.

2.3 Particle Swarm Optimisation

Particle swarm optimisation (PSO) is an evolutionary computation technique inspired by social behaviours of birds flocking and fish schooling [17, 42]. In PSO, each candidate solution is represented as a particle in the swarm and PSO starts with a number of randomly generated particles. All the particles move in the search space to find the optimal solutions. During the movement, each particle (i.e., particle i) has a position and velocity, which are represented by vectors $x_i = (x_{i1}, x_{i2}, ..., x_{iD})$ and $v_i = (v_{i1}, v_{i2}, ..., v_{iD})$, respectively, where D is the dimensionality of the search space. A particle can remember the best positions it visits so far, which is called personal best *pbest*. The best position obtained by the population thus far is called the global best *gbest*, based on which a particle can share information with its neighbours. A particle iteratively updates its position and velocity to search for the optimal solutions based on *pbest* and *gbest* according to the following equations:

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \tag{2.1}$$

$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * r_1 * (p_{id} - x_{id}^{t}) + c_2 * r_2 * (p_{gd} - x_{id}^{t}) \tag{2.2}$$

where t represents the tth iteration in the evolutionary process. $d \in D$ represents the dth dimension in the D-dimensional search space. w is the inertia weight, which can balance the local search and global search abilities of the algorithm. c_1 and c_2 are acceleration constants. r_1 and r_2 are random constants uniformly distributed in [0, 1]. p_{id} and p_{gd} denote the values of *pbest* and *gbest* in the dth dimension. v_{id}^{t+1} is limited by a predefined maximum velocity, v_{max} and $v_{id}^{t+1} \in [-v_{max}, v_{max}]$. The algorithm stops when a predefined criterion is met, which could be a good fitness value or a predefined maximum number of iterations.

PSO is an example of swarm intelligence, the particles learn from their neighbours the location of *gbest* and update their position internally based on the above equations. The particles do not produce offspring by crossover or mutation as an evolutionary algorithm approach would require. PSO differs from ACO because rather than optimising a path from A to B, PSO searches for the optimal point in the search space.

2.3.1 Binary Particle Swarm Optimisation

PSO was originally proposed to address problems in real-number continuous search spaces. In order to extend PSO to address discrete problems Kennedy and Eberhart [18] developed *binary particle swarm optimisation* (BPSO). In BPSO, x_{id}, p_{id} and p_{gd} are restricted to 1 or 0. The velocity is still updated according to Equation (2.2), but it indicates the probability of the position in the corresponding dimension taking value 1, rather than changing the current position based on the velocity. Therefore, a sigmoid function is used to transform v_{id} to

the range of (0, 1). BPSO updates the position of each particle according to the following formula:

$$x_{id} = \begin{cases} 1, \text{ if } rand() < s(v_{id}) \\ 0, \text{ o}therwise \end{cases} \tag{2.3}$$

where

$$s(v_{id}) = \frac{1}{1 + e^{-v_{id}}} \tag{2.4}$$

and $rand()$ is a random number selected from a uniform distribution in [0,1].

2.4 Rough Set Theory

Take a set of objects, called the universe, that can be partitioned. It may not always be possible to distinguish these objects from each other under certain conditions because they may share similar traits and features. Rough set theory (RS) developed by Pawlak [36] provides a way of describing a conventional set in the above situation. The rough set is described by the lower and upper bounds of the conventional set, the lower bound contains all objects definitely in the set and the upper bound contains the objects that may be in the set. In this way the rough set can describe the set of objects that are definitely in and those that may possibly be in a given partion. Let $\mathbb{U}$ be the universe, the set of instances (objects), and let $\mathbb{A}$ be the set of attributes that describe these instances. Also, let $a(x)$ specify the value of attribute $a \in \mathbb{A}$ in instance $x \in \mathbb{U}$.

For any $P \subseteq \mathbb{A}$, a subset of the total available attributes, it is possible to decide on sets of objects that are indistinguishable, this gives the equivalence relation $IND(P)$:

$$IND(P) = \{(x, y) \in \mathbb{U}^2 \mid \forall a \in P.\, a(x) = a(y)\} \tag{2.5}$$

If $(x, y) \in IND(P)$ then x and y are indistinguishable according to P. The above relation can define equivalence classes, these are denoted $[x]_P$. This means that $y \in [x]_P \Leftrightarrow (x, y) \in IND(P)$.

2.4.1 Deterministic Rough Set Theory

Let $X \subseteq \mathbb{U}$ be the set we want to represent with P, the target set. We can define the lower and upper bounds of X according to P, $\underline{P}X$ and $\overline{P}X$ respectively, as

$$\underline{P}X = \{x \mid [x]_P \subseteq X\} \tag{2.6}$$

$$\overline{P}X = \{x \mid [x]_P \cap X \neq \emptyset\} \tag{2.7}$$

The rough set is then the tuple: $\langle \underline{P}X, \overline{P}X \rangle$. We can quantify the *accuracy* of the rough set using

$$\alpha_P(X) = \frac{|\underline{P}X|}{|\overline{P}X|}$$

which effectively measures how well the attributes in P separate the target set, X, from the rest of $\mathbb{U}$. If P is poorly chosen then few instances will be in the lower bound while many will be in the upper bound.

2.4.2 Probabilistic Rough Set Theory

Deterministic rough set theory, as defined above, specifies strict criteria for entry into the lower and upper bounds. For an instance to gain entry into the lower bound, everything in the equivalence class of that instance must also be in the target set. For it to gain entry into the upper bound only one instance in the equivalence class needs to be in the target set. It is possible that a minority of instances could share identical attributes but have different class labels. In practice this could happen due to minorities that are exceptions to common patterns or even human error in inputing the data. This being the case having even one instance labeled differently means that it becomes impossible to find a satisfying assignment because of one mistake. To overcome this problem we can introduce probabilistic rough set approximations [51]. Rather than having a strict lower bound, we can relax it with varying degree using a value α. The same applies to the upper bound, we can relax it with varying values for β.

In probabilistic rough set theory, $\mu_P[x]$ (See Equation 2.8) is defined as a way to measure the fitness of a given instance $x \in X$.

$$\mu_P[x] = \frac{|[x]_P \cap X|}{|[x]_P|} \tag{2.8}$$

The lower approximation is defined as Equation 2.9 and the upper approximation as Equation 2.10.

$$\underline{apr}_P X = \{x | \mu_P[x] \geq \alpha\} \tag{2.9}$$

$$\overline{apr}_P X = \{x | \mu_P[x] > \beta\} \tag{2.10}$$

where α and β can be adjusted to restrict or relax the lower or upper approximations. $\underline{apr}_P X$ and $\overline{apr}_P X$ loosen the boundaries of the rough set. If a large number of instances are in the target set X but a small number are not in a given equivalence class, it will include/exclude them in the lower/upper approximations. Note that when $\alpha = 1.0$ and $\beta = 0.0$, these definitions become the same as the strict bounds defined in Section 2.2.1.

2.5 Multi-objective Optimisation

Multi-objective optimisation involves minimising or maximising conflicting objectives. For k-objectives we want to

$$minimise\ F(x) = [f_1(x), f_2(x), \ldots, f_k(x)] \tag{2.11}$$

where x is the vector of decision variables, a potential solution, and $f_i(x)$ is a function on x. It is these functions that should be optimised.

The quality of a solution is then explained by trade-offs between the objectives. Given two solutions, y and z, in the above k-objective minimisation problem, we can only decide that y dominates z if

$$\forall i : f_i(y) \leq f_i(z) \wedge \exists j : f_j(y) < f_j(z)$$

that is y is superior to z in at least one objective and equal to z in all objectives it is not superior in, that is to say that y dominates z. When y is not dominated by any other solutions, y is a Pareto-optimal solution. The set of all Pareto-optimal solutions, known as the Pareto front, contains solutions that are not dominated by any others. Multi-objective optimisation is the search for the Pareto front.

2.6 Related Work

A number of feature selection algorithms have been proposed in recent years [8, 11, 26]. Typical feature selection algorithms are reviewed in this section.

2.6.1 Traditional Feature Selection Approaches

A traditional filter dimension reduction approach is principal components analysis (PCA), which constructs a low-dimensional representation of the data by finding a few orthogonal linear combinations of the original variables with the largest variance [1]. Due to its conceptual simplicity and being relatively efficient, PCA has been widely used in practice. However, PCA increases the dimensionality of the data in some cases due to its unsupervised nature. Decision trees (DT) use only relevant attributes that are required to completely classify the training set and remove all other attributes. Cardie [2] proposes a filter based dimension reduction algorithm that uses a decision tree algorithm to remove unnecessary attributes for a nearest neighbourhood algorithm.

Two commonly used wrapper methods are SFS [48] and SBS [27]. SFS (SBS) starts with no attributes (all attributes), then candidate attributes are sequentially added to (removed from) the initial attribute subset until the further addition (removal) does not increase the classification performance. However, both SFS and SBS suffer from the problem of nesting effect, because if an attribute is selected (eliminated) it cannot be eliminated (selected) later [55]. The "plus-l-take away-r" method proposed by Stearns [45] could overcome this limitation by performing l times forward selection followed by r times backward elimination. However, the determination of the optimal values of (l, r) is a difficult problem. To address this challenge, two floating attribute reduction algorithms are proposed by Pudil et al. [37], namely sequential forward floating selection (SFFS) and sequential backward floating selection (SBFS). SFFS and SBFS are developed to automatically determine the values for (l, r). These two floating methods are regarded to be at least as good as the best sequential method, but they also suffer from the problem of stagnation in local optima [55].

2.6.2 EC Algorithms (non-PSO) for Attribute Selection

Evolutionary computation techniques have been applied to address attribute reduction problems, such as GAs, GP, ant colony optimisation (ACO) and PSO.

Based on GAs, Chakraborty [3] proposes a attribute reduction algorithm using a fuzzy sets based fitness function. However, PSO with the same fitness function in [4] achieve better performance than this GA based algorithm. Hamdani et al. [15] developed a multi-objective FS algorithm using non-dominated sorting based multi-objective genetic algorithms (NSGAII), but its performance has not been compared to alternate FS algorithms. Also based on GA, Yuan et al. [54] propose a two-phase feature selection method using both filter and wrapper methods. In the filter phase, GA is employed with an inconsistency criterion to evaluate the fitness of solutions to remove irrelevant features. The wrapper phase starts with a feedforward neural network whose input nodes are features obtained in the filter phase. However, without considering feature interaction, features that would form the best feature subset may be removed in the filter phase.

Muni et al. [31] develop a multi-tree GP algorithm to simultaneously select a feature subset and design a classifier using the selected features called GPmtfs. For a c-class problem, each classifier requires c trees. Comparisons suggest this can achieve better results than simpler methods such as SFS and SBS but the number of features selected increases when dealing with a large amount of noise. Also based on GP, and using naïve Bayes (NB),

Kourosh and Zhang [32] propose a feature selection algorithm, where a bit-mask representation is used for feature subsets and a set of operators are used as primitive functions. GP is used to combine feature subsets and operators together to find the optimal subset of features. Kourosh and Zhang [33] also propose a GP relevance measure (GPRM) to evaluate and rank subsets of features in binary classification tasks, and GPRM is also efficient in terms of feature selection.

Ming [28] proposes an attribute reduction method based on ACO and rough set theory. Experimental results show that the proposed algorithm achieves better classification performance with fewer attributes than a C4.5 based attribute reduction algorithm.

2.6.3 PSO Based Approaches to Feature Selection

As an evolutionary computation technique, PSO has recently gained more attention for solving attribute reduction problems. Wang et al. [47] propose a filter attribute reduction algorithm based on an improved binary PSO and rough set. However, the classification performance of the reduct was only tested on one learning algorithm, the LEM2 algorithm, which originally is specific used for rough set and have some bias for the proposed rough set based algorithm. Meanwhile, only using one learning algorithm cannot show the advantage that filter algorithms is more general. Lin et al. [25] propose a wrapper attribute reduction algorithm (PSO+SVM) using PSO and SVM to optimise the parameters in SVM and search for the best attribute subset simultaneously. Mohemmed et al. [30] propose a hybrid method (PSOAdaBoost) that incorporates PSO with an AdaBoost framework for face detection. PSOAdaBoost aims to search for the best attribute subset and determine the decision thresholds of AdaBoost simultaneously, which speeds up the training and increase the accuracy of weak classifiers in AdaBoost.

Chuang et al. [6] apply the so-called catfish effect to PSO for attribute reduction, which is to introduce new particles into the swarm by initialising the worst particles when *gbest* has not improved for a number of iterations. The introduced catfish particles could help PSO avoid premature convergence. Another approach presented by Chuang et al. [5] is to simply reset *gbest* if it maintains the same value after several iterations. Yang et al. [49] also propose a strategy to renew the *gbest* during the search process to keep the diversity of the population in BPSO. When the *gbest* is identical after three iterations, a Boolean operator 'and(.)' will 'and' each bit of the *pbest* of all particles in an attempt to create a new *gbest*. Experimental results illustrate that the proposed method usually achieves higher classification accuracy with fewer features than GA and standard BPSO.

Liu et al. [26] introduce a multi-swarm PSO (MSPSO) algorithm to search for the optimal attribute subset and optimise the parameters of SVM simultaneously. Experiments show that MSPSO could achieve higher classification accuracy than grid search, standard PSO and GA. However, MSPSO is computationally more expensive than the other three methods because of the large population size and complicated communication rules between different subswarms. Based on PSO, Unler and Murat [46] propose a attribute reduction algorithm with an adaptive selection strategy, where an attribute is chosen not only according to the likelihood calculated by PSO, but also to its contribution to the attributes already selected. Experiments suggest that the proposed method outperforms the tabu search and scatter search algorithms.

PSO has been shown to be an efficient search technique for feature selection and dimension reduction by many existing studies. However, most of the existing approaches are wrappers, which are computationally expensive and less general than filter approaches. Therefore, investigation of an effective PSO based filter dimension reduction algorithm is still an open issue. Probabilistic rough set was claimed to be a good way for dimension

reduction problems [52], but its real performance has not been investigated. Therefore, it is thought to investigate the performance of probabilistic rough set and PSO for filter dimension reduction.

Chapter 3

Single Objective Feature Selection Algorithms based on PSO and RS

3.1 Introduction

This chapter presents how RS theory can be used as an evaluation criteria for a filter approach, and investigates its effectiveness. Two new algorithms are presented, $RSPSO_1$ and $RSPSO_2$, based on RS and PSO. Both algorithms use a RS based approach to measure the fitness of a given set of features. This rough set measure is linearly combined with additional objectives in an attempt to remove additional numbers of features, without lowering the classification accuracy further.

The chapter is laid out as follows. Fitness measures using deterministic and probabilistic rough set theory are presented in Section 3.2. Section 3.2. then presents the $RSPSO_1$ and $RSPSO_2$ algorithms. Pseudo-code of the proposed algorithms is provided in Section 3.2.5. Section 3.3 describes and discusses the experimental design and results. Section 3.3. presents the overall results achieved, before highlighting how changing the parameters of the new algorithms can affect performance. The last part of Section 3.3. compares $RSPSO_1$ and $RSPSO_2$ with some traditional approaches while Section 3.4 concludes the chapter.

3.1.1 Chapter Objectives

This chapter has the following objectives.

1. Investigate the applicability of both deterministic RS and probabilistic rough set and PSO for filter feature selection problems.

2. Investigate whether additional components combined with the RS theory approach can reduce the number of features further.

3. Investigate whether the RS and PSO can perform better than traditional FS approaches.

3.2 Rough Set Theory for FS

This section presents rough set theory as an evaluation criteria for feature selection. The new proposed algorithms, $RSPSO_1$ and $RSPSO_2$, are presented.

3.2.1 Deterministic Rough Set Theory for Feature Selection

Rough set theory provides a natural way to evaluate feature sets, given the definitions. Partition the universe using the class labels, each partition becomes a target set. The lower bound of each target set then measures the number of instances that have been completely separated from instances of other classes, remember the lower bound of a target set is the set of instances that definitely know are in that target set. Assume the universe, $\mathbf{U}$, has been partitioned into target sets using the class labels: $\{U_1, ..., U_n\}$, each target set contains only the instances that are in the same class. An evaluation criterion for a subset of features, $G \subseteq F$, is then:

$$Fitness_1(G, \mathbb{U}) = \frac{\sum_{i=1}^{n} |\underline{G}U_i|}{|\mathbf{U}|} \tag{3.1}$$

The evaluation criterion measures the number of instances that have been separated from instances of other classes by the features, a score of 1.0 means that G completely divides the classes. We do not consider the upper bound in our measurement as we use the size of the universe as the upper bound and we want to see what proportion of instances can/cannot be separated from others of differing classes in the whole universe across all target sets.

3.2.2 Probabilistic Rough Set Theory for Feature Selection

As discussed in the previous chapter, Section 2.2.2, the definitions of lower approximation and upper approximation limit the application of rough set theory. In classification problems, it may happen that two or more instances might have the same attribute values but be classified in different classes. This is possibly because incorrect values are entered or one instance is an exception to a class. Therefore, it is impossible to achieve the $Fitness_1(G) = 1.0$ in Equation 3.1. A set of attributes could be adequate, but erroneous or unusual values prevent certain instances being included in the lower bound. This problem can be addressed by relaxing the definitions of lower and upper approximations in probabilistic rough set theory. Therefore, a new filter feature selection algorithm based on BPSO and probabilistic rough set theory [26] is proposed.

The same thoery as in deterministic RS can be applied to probabilistic RS.

$$PotentialFitness_2(G, \mathbb{U}) = \frac{\sum_{i=1}^{n} |\underline{apr}_G U_i|}{|\mathbf{U}|} \tag{3.2}$$

Probabilistic rough set theory is equal to the original definition when $\alpha = 1.0$, this means we can include a test for the first fitness function while using only the probabilistic definition.

3.2.3 New Algorithm: RSPSO$_1$

The fitness measures defined in the previous section have quite a significant drawback when taken in isolation. Consider two sets of features that each achieve a perfect score of 1.0 using the above fitness functions. Assume that one of these does so using fewer features, this is clearly the preferred selection, one of our primary goals is to select as small a subset as possible. The simple fitness functions above do not consider this information.

A simple way to achieve this goal is to add additional components into the fitness function, with new components representing a check on the number of features. A potential fitness function would be:

$$RSPSO_1(G, \mathbb{U}) = \gamma * \frac{\sum_{x=1}^{n} |\underline{apr}_G X_i|}{|\mathbb{U}|} + (1 - \gamma) * (1 - \frac{\#features}{\#totalFeatures}) \tag{3.3}$$

where $\gamma \in [0,1]$ shows the relative importance of the representation power of the purity while $1-\gamma$ shows the relative importance of the number of attributes. This is a typical way to combine two attributes into one single fitness function. $RSPSO_1$ means that given two sets of features some consideration will be given to the number of features selected, leading to a feature-minimising fitness function.

3.2.4 New Algorithm: RSPSO$_2$

While this would achieve the goal of minimising the number of features there is large potential for a drop in classification accuracy on the testing set. The first component is essentially a measure of purity in the equivalence classes. A score of 1.0 means that each equivalence class is pure, that is every instance in each equivalence class are of the same class. Consider a situation where each equivalence class is very small, it is quite likely these small equivalence classes will be either pure or have little impurity. Such a situation could still score highly in $RSPSO_1$, a small number of features could reasonably describe a large number of equivalence classes.

If lots of small equivalence classes are extracted it is likely that these same patterns do not exist in the population, only in the training data. If however several large equivalence class with low impurity is extracted that information is likely to exist in the population, because a lot of instances obey that rule. To this end a better second component would be one that attempts to maximise the size of the equivalence classes. Note that larger equivalence classes imply fewer features, because more features mean it is easier for instances to differ.

$$RSPSO_2(G,\mathbb{U}) = \frac{\sum_{x=1}^{n} |\underline{apr}_G X_i|}{|\mathbb{U}|} + \frac{\sum_{x \in \{the\ equivalence\ classes\}} \frac{|x|}{|\mathbb{U}|}}{\#\ of\ equivalence\ classes} \tag{3.4}$$

The second measure in the above equation attempts to maximise the size of the equivalence classes by finding the average proportion of instances in each equivalence class.

3.2.5 Summary of the New Algorithms

Both the two algorithms, $RSPSO_1$ and $RSPSO_2$ attempt to *linearly combine* two objectives. The first objective uses RS to measure how good a set of features are. The second objective attempts to bring the number of selected features into consideration. The γ parameter in $RSPSO_1$ allows the original RS criteria to be evaluated independently. With $\gamma = 1.0$ the second component is not considered. $RSPSO_2$ attempts to use the theory behind the RS component to minimise the number of features, which considers the size of the equivalence classes that the training set is broken into.

Algorithm 1 presents the pseudo code of the proposed algorithms $RSPSO_1$ and $RSPSO_2$, which can be used in conjunction with the two evaluation criteria. The data is firstly split into training and testing sets, before the particles are then initialised with random velocities and and positions. These positions are then evaluated using the training set and *pbest* for each particle and *gbest* are assigned. The evolution process can then begin, at each stage the velocity and position are updated using the equations in the previous chapter before the new positions and *pbest* and *gbest* are again updated.

3.3 Experiments and Results

This section presents the experimental design and results of $RSPSO_1$ and $RSPSO_2$ approaches described above. The experimental design discusses the parameters used for the PSO equations. Following the experimental design section the overall results are discussed, before the

Algorithm 1: BPSO feature selection algorithm

```
1  begin
2      divide the data into train and test sets ;
3      initialise particles;
4      for i = 0 to number_of_particles do
5          for d = 0 to number_of_features do
6              v_id^0 = rand();
7              x_id^0 = rand();
8      evaluate fitness of each particle on train using Equation RSPSO_1 or RSPSO_2 ;
9      initialise p_id for each particle ;
10     initialise p_gd for each particle ;
11     for t = 0 to maxIteration do
12         for i = 0 to number_of_particles do
13             for d = 0 to number_of_features do
14                 update v_id^t using Equation 2.2. ;
15                 update x_id^t using Equation 2.3. ;
16         evaluate fitness of each particle on train using Equation RSPSO_1 or RSPSO_2 ;
17         for i = 0 to number_of_particles do
18             for d = 0 to number_of_features do
19                 update p_id ;
20                 update p_gd ;
21     accuracy ← classifaction accuracy of gbest on test ;
22     return ⟨accuracy, gbest⟩ ;
```

effects of varying the *RSPSO* parameters are evaluated. The proposed algorithms are finally compared with some traditional approaches.

3.3.1 Experimental Design

In order to test the algorithms described above, 7 datasets were taken from the UCI repository. Statlog and Waveform were discretized using the Weka discretize filter while the remaining datasets were in discrete form when accessed. Each dataset was split into $\frac{2}{3}$ training and $\frac{1}{3}$ testing data. Three classification algorithms were used to test the classification accuracy of the selected features, *decision tree* (DT), *Naive Bayes* (NB), and *5-Nearest Neighbour* (NN). The reported classification performance is the performance of the classification algorithms on the unseen testing data using only the selected features. The evolution and learning process was completed using only the training data. The datasets are described in Table 3.1.

For each dataset, the algorithm was conducted for 30 independent runs. In each run, the fully connected topology is used, $v_{max} = 6.0$, the population size is 30 and the maximum iteration is 500. $w = 0.7298$, $c_1 = c_2 = 1.49618$. These values are chosen based on the common settings in the literature [42]. The tables at the end of this section provide information about the output of the algorithm. The values reported represent the mean $\pm$ standard deviation (maximum) across all 30 runs with the average number of features also reported.

The experiments were performed using a cycle-stealing computational grid provided by the School of Engineering and Computer Science (ECS). The gird is managed by the Sun

Dataset	#Attributes	#Classes	#Instances
Lymphography (Lymph)	18	4	148
Spect	22	2	267
Dermatology (Derm)	33	6	366
Soybean Large (Soy)	35	19	307
Chess	36	2	3196
Statlog (Stat)	36	6	6435
Waveform (Wave)	40	3	5000

Table 3.1: Datasets

Grid Engine (SGE) and compirses of approximately 200 Dell Optiplex 990 Arch Linux workstations with 4096 MBytes RAM and Intel Core i5 2400 CPU @ 310 GHz 4 core processors. The propgrams were written and compiled using Java6, Linux bash and shell scripts were used to distribute jobs to the grid. For both algorithms, different values of α needed to be considered as they rely on the probabilistic rough set theory. When α is set to 1.0 the probabilistic rough set theory will perform in the same way as the deterministic approach, in this way we can test the traditional rough set theory. $RSPSO_1$ also requires values for γ, to set the relative importance of the two components.

3.3.2 Overall Results

Table 3.2. presents the results for the general parameters. $RSPSO_1(1.0)$ and $RSPSO_1(0.5)$ presents the results using $RSPSO_1$ with $\alpha = 1.0$ and γ as 1.0 and 0.5 respectively. This allows for a comparison of the RS measure without the second component, as $\gamma = 1.0$. $RSPSO_2$ considers with $\alpha = 1.0$. As both $RSPSO_1(0.5)$ and $RSPSO_2$ are giving equal weights to the two components, the effects of the second components can be evaluated. As $\alpha = 1.0$, it is the traditional rough set approach that is tested, not the probabilistic measures.

The results can be evaluated from the perspective of the two key goals in feature selection, the first one is too reduce the number of features and the second one is too improve the classification accuracy. In some cases, reducing the number of features did not lead to an improved performance. With the DT classifier it is possible to reduce the number of features and at least maintain or improve performance, in the Chess, Dermatology, Spect and Waveform datasets, the best classification performance was achieved using $RSPSO_1$ in only the Waveform dataset, in the remaining three $RSPSO_2$ achieved the best best performance. In the remaining datasets $RSPSO_1(1.0)$ and $RSPSO_2$ achieved similar performance, but $RSPSO_2$ often did so with fewer features and a more stable final solution. Consider the Soybean dataset, $RSPSO_1(1.0)$ achieved an average accuracy of 0.803 using 21.5 features while $RSPSO_2$ achieved 0.802 but with 17.5 features. The standard deviation for the $RSPSO_1$ approaches are also larger, suggesting a greater spread in the results, $RSPSO_2$ provided greater stability.

In the remaining classifiers, NB and NN, different trends are evidenced. The difference between the NB and DT classifiers are that DT do further feature selection while NB makes the assumption that the features are not dependent on each other. As fewer features are selected, the assumption of NB could come closer to being true, while the feature selection ability of DT are reduced. If we consider the Chess dataset, $RSPSO_1(0.5)$ achieves the best performance in the NB classifier, with an average of only 11.2 features. The features selected by $RSPSO_1(0.5)$ could share little correlation meaning the assumption in the NB classifier is correct. The Spect dataset shares this trait with the Chess dataset. $RSPSO_2$ performs strongly on the DT classifier but is outperformed by $RSPSO_1(0.5)$ when the NB classifier is used. Of the seven datasets, only Chess and Spect make improvements over the baseline of all features uniformly across the 3 classifiers. The Dermatology and Waveform datasets

	Algorithm	# Features	DT	NB	NN
Chess	All	36	0.985	0.879	0.932
	$RSPSO_1(1.0)$	30.7	0.983 ± 0.003 (0.987)	0.885 ± 0.015 (0.917)	0.941 ± 0.008 (0.959)
	$RSPSO_1(0.5)$	11.2	0.972 ± 0.006 (0.979)	0.908 ± 0.017 (0.939)	0.918 ± 0.018 (0.945)
	$RSPSO_2$	28.3	0.985 ± 0.001 (0.987)	0.895 ± 0.022 (0.923)	0.946 ± 0.005 (0.960)
Derm	All	33	0.828	0.959	0.951
	$RSPSO_1(1.0)$	21.0	0.860 ± 0.048 (0.975)	0.935 ± 0.032 (0.984)	0.919 ± 0.033 (0.975)
	$RSPSO_1(0.5)$	6.9	0.701 ± 0.080 (0.967)	0.763 ± 0.057 (0.926)	0.731 ± 0.057 (0.869)
	$RSPSO_2$	9.8	0.92 ± 0.028 (0.959)	0.923 ± 0.021 (0.959)	0.898 ± 0.036 (0.975)
Lymph	All	18	0.755	0.878	0.816
	$RSPSO_1(1.0)$	11.7	0.724 ± 0.068 (0.796)	0.846 ± 0.035 (0.918)	0.776 ± 0.046 (0.857)
	$RSPSO_1(0.5)$	5.0	0.673 ± 0.000 (0.673)	0.776 ± 0.000 (0.776)	0.755 ± 0.000 (0.755)
	$RSPSO_2$	6.6	0.722 ± 0.063 (0.796)	0.814 ± 0.014 (0.837)	0.772 ± 0.025 (0.857)
Spect	All	22	0.809	0.764	0.820
	$RSPSO_1(1.0)$	17.5	0.810 ± 0.023 (0.843)	0.773 ± 0.018 (0.809)	0.811 ± 0.018 (0.854)
	$RSPSO_1(0.5)$	7.8	0.801 ± 0.010 (0.831)	0.824 ± 0.017 (0.843)	0.830 ± 0.009 (0.843)
	$RSPSO_2$	14.5	0.82 ± 0.000 (0.82)	0.777 ± 0.006 (0.798)	0.812 ± 0.008 (0.831)
Soy	All	35	0.819	0.903	0.907
	$RSPSO_1(1.0)$	21.5	0.803 ± 0.046 (0.872)	0.848 ± 0.031 (0.899)	0.802 ± 0.052 (0.868)
	$RSPSO_1(0.5)$	7.7	0.706 ± 0.039 (0.846)	0.741 ± 0.037 (0.859)	0.663 ± 0.034 (0.775)
	$RSPSO_2$	17.5	0.802 ± 0.032 (0.85)	0.811 ± 0.025 (0.859)	0.74 ± 0.032 (0.793)
Stat	All	36	0.864	0.826	0.901
	$RSPSO_1(1.0)$	25.7	0.855 ± 0.006 (0.871)	0.821 ± 0.004 (0.827)	0.893 ± 0.0040 (0.902)
	$RSPSO_1(0.5)$	9.0	0.836 ± 0.007 (0.85)	0.803 ± 0.0070 (0.814)	0.865 ± 0.0070 (0.882)
	$RSPSO_2$	19.9	0.854 ± 0.008 (0.87)	0.82 ± 0.004 (0.83)	0.888 ± 0.004 (0.897)
Wave	All	40	0.748	0.797	0.791
	$RSPSO_1(1.0)$	24.5	0.748 ± 0.019 (0.772)	0.777 ± 0.020 (0.813)	0.752 ± 0.026 (0.803)
	$RSPSO_1(0.5)$	7.0	0.702 ± 0.022 (0.744)	0.709 ± 0.021 (0.744)	0.667 ± 0.028 (0.710)
	$RSPSO_2$	18.4	0.726 ± 0.036 (0.767)	0.748 ± 0.046 (0.813)	0.72 ± 0.050 (0.792)

Table 3.2: Comparison of classification accuracy on the test set using all features, $RSPSO_1(\gamma)$, and $RSPSO_2$ with $\alpha = 1.0$ and $\gamma = 0.5$ or 1.0

at least maintain, if not improve, accuracy using the DT classifier but not the NN and NB classifiers.

While the average classification performance is not always greater than the baseline of all features. The PSO search did find better subsets in at least one of the 30 runs in all cases except the Soybean dataset. Since PSO is a stochastic approach, each run may find a different set after a certain number of iterations. Some of the runs producing superior sets suggests that the RS measure is valuing good sets of feature highly, or these would not have been found in any of the searches. Even when in some cases, the classification accuracy achieved by the proposed algorithm is lower, the number of features has been significantly reduced. In many cases the corresponding decrease in accuracy was small in comparison. The classification accuracy of the Statlog dataset is slightly reduced. 0.864, 0.826 and 0.901, in the DT, NB and NN classifiers respectively, to 0.855, 0.821 and 0.893 with $RSPSO_1(1.0)$. This drop in accuracy corresponded with the number of features dropping from 36 to 25.7. In some cases this increase in efficiency may be acceptable at such a cost. The results discussed so far, only considered simple parameters. By varying the values of α and γ the search capabilities may be extended further.

3.3.3 Varying Parameters for RSPSO$_1$

The parameters can be further refined, α and γ can be used to change the relative importance of components to improve the accuracy in $RSPSO_1$. By relaxing α we can get higher fitnesses when using smaller sets of features, and by increasing γ we can give higher importance to the purity of the equivalence classes. Tables 3.3 and 3.4. show the results of changing the values for α and γ across both $RSPSO_1$.

α	γ	# Features	DT	NB	NN
			Chess Dataset		
1.0	1.0	30.7	0.983 ± 0.003 (0.987)	0.885 ± 0.015 (0.917)	0.941 ± 0.008 (0.959)
1.0	0.75	16.8	0.979 ± 0.003 (0.985)	0.914 ± 0.012 (0.945)	0.937 ± 0.013 (0.955)
1.0	0.5	11.2	0.972 ± 0.006 (0.979)	0.908 ± 0.017 (0.939)	0.918 ± 0.018 (0.945)
0.75	1.0	30.3	0.985 ± 0.001 (0.987)	0.884 ± 0.015 (0.911)	0.942 ± 0.006 (0.951)
0.75	0.75	7.7	0.961 ± 0.019 (0.977)	0.932 ± 0.009 (0.953)	0.821 ± 0.114 (0.921)
0.75	0.5	4.9	0.931 ± 0.013 (0.938)	0.931 ± 0.013 (0.941)	0.602 ± 0.198 (0.892)
0.5	1.0	28.8	0.981 ± 0.005 (0.987)	0.885 ± 0.017 (0.915)	0.941 ± 0.006 (0.953)
0.5	0.75	7.0	0.955 ± 0.020 (0.977)	0.927 ± 0.018 (0.957)	0.798 ± 0.161 (0.912)
0.5	0.5	5.1	0.933 ± 0.013 (0.959)	0.932 ± 0.014 (0.941)	0.643 ± 0.186 (0.928)
			Dermatology Dataset		
1.0	1.0	21.0	0.860 ± 0.048 (0.975)	0.935 ± 0.032 (0.984)	0.919 ± 0.033 (0.975)
1.0	0.75	7.7	0.732 ± 0.074 (0.926)	0.780 ± 0.054 (0.910)	0.748 ± 0.047 (0.836)
1.0	0.5	6.9	0.701 ± 0.080 (0.967)	0.763 ± 0.057 (0.926)	0.731 ± 0.057 (0.869)
0.75	1.0	21.0	0.860 ± 0.048 (0.975)	0.935 ± 0.032 (0.984)	0.919 ± 0.033 (0.975)
0.75	0.75	7.7	0.743 ± 0.085 (0.926)	0.786 ± 0.064 (0.910)	0.766 ± 0.073 (0.893)
0.75	0.5	6.4	0.752 ± 0.093 (0.951)	0.783 ± 0.075 (0.959)	0.725 ± 0.083 (0.943)
0.5	1.0	20.7	0.860 ± 0.051 (0.975)	0.925 ± 0.045 (0.984)	0.912 ± 0.036 (0.975)
0.5	0.75	6.3	0.698 ± 0.059 (0.836)	0.742 ± 0.055 (0.869)	0.702 ± 0.054 (0.828)
0.5	0.5	5.6	0.652 ± 0.079 (0.910)	0.705 ± 0.057 (0.902)	0.661 ± 0.051 (0.762)
			Spect Dataset		
1.0	1.0	17.5	0.810 ± 0.023 (0.843)	0.773 ± 0.018 (0.809)	0.811 ± 0.018 (0.854)
1.0	0.75	12.8	0.811 ± 0.011 (0.820)	0.795 ± 0.016 (0.809)	0.796 ± 0.019 (0.831)
1.0	0.5	7.8	0.801 ± 0.010 (0.831)	0.824 ± 0.017 (0.843)	0.830 ± 0.009 (0.843)
0.75	1.0	15.6	0.818 ± 0.008 (0.820)	0.759 ± 0.016 (0.787)	0.811 ± 0.011 (0.843)
0.75	0.75	7.1	0.798 ± 0.012 (0.831)	0.797 ± 0.029 (0.843)	0.805 ± 0.040 (0.843)
0.75	0.5	4.6	0.786 ± 0.026 (0.843)	0.796 ± 0.025 (0.843)	0.739 ± 0.248 (0.843)
0.5	1.0	16.6	0.800 ± 0.021 (0.843)	0.767 ± 0.021 (0.820)	0.813 ± 0.021 (0.843)
0.5	0.75	3.6	0.842 ± 0.006 (0.843)	0.754 ± 0.040 (0.843)	0.843 ± 0.000 (0.843)
0.5	0.5	2.0	0.843 ± 0.000 (0.843)	0.838 ± 0.024 (0.843)	0.815 ± 0.151 (0.843)

Table 3.3: $RSPSO_1$ performance on Chess, Dematology and Spect datasets with varying values for α and γ

While uniform improvement was achieved across all classifiers in the Chess and Spect datasets in Table 3.2, this often required the use of $RSPSO_2$. Table 3.3 shows the results of the different parameters in these two datasets. The Chess dataset, $RSPSO_1$ with $\alpha = 0.75$ and γ set to 1.0, so no second component, matches the accuracy of 0.985 achieved by $RSPSO_2$ although it does select more features. With $\alpha = \gamma = 0.5$ the NB classifier hits a high of 0.932 with an average of only 5.1 features, a large improvement over the accuracy in Table. 3.2. The Spect dataset experiences the same changes, but to a greater extent, the DT and NB classifiers hit the largest classification performance with $\alpha = \gamma = 0.5$ of 0.843 and 0.838 respectively, using only 2 features. A number this low is only possible because the Spect dataset has a low number of instances. The Dermatology dataset demonstrates that this trend is not always the case, across all 3 classifiers the peak is always at $\alpha = \gamma = 1.0$. This indicates adding the additional components is not always beneficial.

Table 3.4 presents the in depth results of the remaining datasets. The Lymph and Statlog datasets show improvement when α is reduced but γ is kept at 1.0. A small drop in performance can sometimes lead to a large drop in the number of features, consider the difference between $\gamma = 1.0$ and $\gamma = 0.75$ in the Lymph dataset when $\alpha = 1.0$, the DT classification accuracy falls from 0.727 to 0.712, but the feature count drops from 11.37 to only 4.0. The same can be seen in the Statlog dataset. If the primary concern is efficiency, reducing γ can vastly improve this while having only a small impact on the classification accuracy.

α	γ	# Features	DT	NB	NN
			Lymph Dataset		
1.0	1.0	11.7	0.724 ± 0.068 (0.796)	0.846 ± 0.035 (0.918)	0.776 ± 0.046 (0.857)
1.0	0.75	5.00	0.673 ± 0.000 (0.673)	0.776 ± 0.000 (0.776)	0.755 ± 0.000 (0.755)
1.0	0.5	5.00	0.673 ± 0.000 (0.673)	0.776 ± 0.000 (0.776)	0.755 ± 0.000 (0.755)
0.75	1.0	11.8	0.723 ± 0.068 (0.796)	0.846 ± 0.035 (0.918)	0.776 ± 0.047 (0.857)
0.75	0.75	5.00	0.673 ± 0.000 (0.673)	0.776 ± 0.000 (0.776)	0.755 ± 0.000 (0.755)
0.75	0.5	4.00	0.714 ± 0.000 (0.714)	0.816 ± 0.000 (0.816)	0.796 ± 0.000 (0.796)
0.5	1.0	11.4	0.727 ± 0.040 (0.796)	0.836 ± 0.042 (0.918)	0.778 ± 0.045 (0.857)
0.5	0.75	4.00	0.712 ± 0.047 (0.735)	0.792 ± 0.037 (0.857)	0.722 ± 0.052 (0.796)
0.5	0.5	3.00	0.680 ± 0.015 (0.714)	0.711 ± 0.038 (0.796)	0.687 ± 0.030 (755)
			Soybean Dataset		
1.0	1.0	21.5	0.803 ± 0.046 (0.872)	0.848 ± 0.031 (0.899)	0.802 ± 0.052 (0.868)
1.0	0.75	8.7	0.715 ± 0.038 (0.775)	0.752 ± 0.047 (0.833)	0.670 ± 0.034 (0.749)
1.0	0.5	7.7	0.706 ± 0.039 (0.846)	0.741 ± 0.037 (0.859)	0.663 ± 0.034 (0.775)
0.75	1.0	21.6	0.804 ± 0.043 (0.872)	0.849 ± 0.029 (0.899)	0.806 ± 0.043 (0.868)
0.75	0.75	8.8	0.713 ± 0.043 (0.775)	0.747 ± 0.031 (0.811)	0.668 ± 0.033 (0.749)
0.75	0.5	7.5	0.713 ± 0.039 (0.802)	0.761 ± 0.042 (0.833)	0.670 ± 0.033 (0.727)
0.5	1.0	20.4	0.801 ± 0.042 (0.868)	0.839 ± 0.029 (0.890)	0.792 ± 0.047 (0.859)
0.5	0.75	7.13	0.690 ± 0.054 (0.806)	0.728 ± 0.075 (0.846)	0.646 ± 0.054 (0.731)
0.5	0.5	6.27	0.664 ± 0.058 (0.775)	0.706 ± 0.071 (0.802)	0.619 ± 0.056 (0.700)
			Waveform Dataset		
1.0	1.0	24.5	0.748 ± 0.019 (0.772)	0.777 ± 0.02 (0.813)	0.752 ± 0.026 (0.803)
1.0	0.75	8.0	0.697 ± 0.025 (0.741)	0.709 ± 0.028 (0.753)	0.658 ± 0.038 (0.723)
1.0	0.5	7.0	0.702 ± 0.022 (0.744)	0.709 ± 0.021 (0.744)	0.667 ± 0.028 (0.71)
0.75	1.0	24.5	0.748 ± 0.019 (0.772)	0.777 ± 0.02 (0.813)	0.752 ± 0.026 (0.803)
0.75	0.8	8.0	0.701 ± 0.037 (0.763)	0.712 ± 0.038 (0.779)	0.668 ± 0.044 (0.738)
0.75	0.5	7.0	0.702 ± 0.025 (0.736)	0.71 ± 0.025 (0.756)	0.667 ± 0.031 (0.722)
0.5	1.0	24.5	0.748 ± 0.019 (0.772)	0.777 ± 0.02 (0.813)	0.752 ± 0.026 (0.803)
0.5	0.75	6.0	0.706 ± 0.03 (0.755)	0.708 ± 0.031 (0.755)	0.666 ± 0.042 (0.724)
0.5	0.5	6.0	0.704 ± 0.032 (0.756)	0.712 ± 0.029 (0.768)	0.667 ± 0.039 (0.743)
			Statlog Dataset		
1.0	1.0	25.7	0.855 ± 0.006 (0.871)	0.821 ± 0.004 (0.827)	0.893 ± 0.004 (0.902)
1.0	0.75	11.5	0.842 ± 0.009 (0.855)	0.809 ± 0.007 (0.825)	0.876 ± 0.007 (0.887)
1.0	0.5	8.9	0.836 ± 0.007 (0.85)	0.803 ± 0.007 (0.814)	0.865 ± 0.007 (0.882)
0.75	1.0	25.5	0.856 ± 0.006 (0.871)	0.82 ± 0.003 (0.825)	0.894 ± 0.004 (0.902)
0.75	0.75	10.3	0.846 ± 0.009 (0.864)	0.808 ± 0.008 (0.823)	0.877 ± 0.007 (0.891)
0.75	0.5	7.0	0.839 ± 0.010 (0.859)	0.801 ± 0.011 (0.817)	0.859 ± 0.010 (0.888)
0.5	1.0	23.5	0.857 ± 0.007 (0.867)	0.820 ± 0.004 (0.827)	0.892 ± 0.005 (0.899)
0.5	0.9	10.1	0.84 ± 0.008 (0.855)	0.803 ± 0.007 (0.816)	0.871 ± 0.005 (0.881)
0.5	0.5	5.6	0.834 ± 0.009 (0.858)	0.798 ± 0.009 (0.811)	0.843 ± 0.013 (0.865)

Table 3.4: $RSPSO_1$ performance on Lymph, Soybean, Waveform and Statlog datasets with varying values for α and γ

3.3.4 Varying Parameters for RSPSO$_2$

Table 3.5. shows the performance of $RSPSO_2$ with different values for α. There is no weighting on the components in $RSPSO_2$ which means the only parameter is the value for α. Changing the value for α did not always reduces the number of features further. In many datasets the number of features increases as α is set to 0.75, but drops when it is lowered further. Changing α will result in a different search, and it is possible that alternate solutions with a stronger second component may now have a higher ranking in the first component that means it becomes favourable. The reduction effect of relaxing α can sometimes have a positive effect.

α	# Features	DT	NB	NN
		Chess		
1.0	28.3	0.985 $\pm$ 0.0010 (0.987)	0.895 $\pm$ 0.022 (0.923)	0.946 $\pm$ 0.005 (0.96)
0.75	28.9	0.985 $\pm$ 0.0010 (0.987)	0.892 $\pm$ 0.02 (0.923)	0.946 $\pm$ 0.006 (0.956)
0.5	24.4	0.985 $\pm$ 0.0010 (0.987)	0.908 $\pm$ 0.017 (0.93)	0.945 $\pm$ 0.010 (0.963)
		Dermatology		
1.0	9.8	0.920 $\pm$ 0.028 (0.959)	0.923 $\pm$ 0.021 (0.959)	0.898 $\pm$ 0.036 (0.975)
0.75	10.0	0.922 $\pm$ 0.029 (0.959)	0.93 $\pm$ 0.020 (0.975)	0.909 $\pm$ 0.039 (0.975)
0.5	9.0	0.909 $\pm$ 0.045 (0.951)	0.918 $\pm$ 0.037 (0.975)	0.864 $\pm$ 0.040 (0.943)
		Spect		
1.0	14.5	0.820 $\pm$ 0.000 (0.82)	0.777 $\pm$ 0.006 (0.798)	0.812 $\pm$ 0.008 (0.831)
0.75	14.1	0.820 $\pm$ 0.000 (0.82)	0.775 $\pm$ 0.003 (0.775)	0.812 $\pm$ 0.005 (0.820)
0.5	11.9	0.818 $\pm$ 0.010 (0.843)	0.801 $\pm$ 0.020 (0.82)	0.822 $\pm$ 0.010 (0.843)
		Lymph		
1.0	6.567	0.722 $\pm$ 0.063 (0.796)	0.814 $\pm$ 0.014 (0.837)	0.772 $\pm$ 0.025 (0.857)
0.75	6.533	0.722 $\pm$ 0.063 (0.796)	0.816 $\pm$ 0.012 (0.837)	0.773 $\pm$ 0.028 (0.857)
0.5	1.633	0.565 $\pm$ 0.106 (0.735)	0.582 $\pm$ 0.131 (0.816)	0.184 $\pm$ 0.296 (0.796)
		Soybean		
1.0	17.467	0.802 $\pm$ 0.032 (0.85)	0.811 $\pm$ 0.025 (0.859)	0.74 $\pm$ 0.032 (0.793)
0.75	17.8	0.804 $\pm$ 0.033 (0.85)	0.813 $\pm$ 0.024 (0.855)	0.742 $\pm$ 0.026 (0.797)
0.5	13.7	0.789 $\pm$ 0.033 (0.833)	0.792 $\pm$ 0.044 (0.863)	0.72 $\pm$ 0.032 (0.767)
		Waveform		
1.0	18.433	0.726 $\pm$ 0.036 (0.767)	0.748 $\pm$ 0.046 (0.813)	0.72 $\pm$ 0.05 (0.792)
0.75	18.433	0.726 $\pm$ 0.036 (0.767)	0.748 $\pm$ 0.046 (0.813)	0.72 $\pm$ 0.05 (0.792)
0.5	11.1	0.697 $\pm$ 0.041 (0.758)	0.713 $\pm$ 0.049 (0.789)	0.681 $\pm$ 0.058 (0.786)
		Statlog		
-	36	0.864	0.826	0.901
1.0	19.9	0.854 $\pm$ 0.0080 (0.87)	0.82 $\pm$ 0.0040 (0.83)	0.888 $\pm$ 0.0040 (0.897)
0.75	19.933	0.855 $\pm$ 0.0080 (0.868)	0.822 $\pm$ 0.0040 (0.83)	0.89 $\pm$ 0.0060 (0.899)
0.5	16.5	0.849 $\pm$ 0.0070 (0.866)	0.818 $\pm$ 0.0070 (0.832)	0.883 $\pm$ 0.0060 (0.897)

Table 3.5: $RSPSO_2$ performance on all datasets with varying values for α

Consider the Chess dataset, with $\alpha = 0.5$ an average of 24.4 features are selected. The

DT classification accuracy stays at 0.985 and the NN average classification accuracy drops by 0.001 to 0.945. The best performance in the NN classifier, however, increased to 0.963 suggesting the potential for a positive impact. In the NB classifier the average performance improved to 0.908 while selecting fewer features. In all cases the accuracy has stayed above the performance of both all features and the independent RS measure while reducing the number of features even further.

The Lymph dataset experiences a large drop in performance suggesting that having α too low can have very negative effects, despite this the performance peaks slightly at $\alpha = 0.75$ before dropping. In most of the datasets the value for α can be used to improve performance a small amount. The Spect dataset peaks in the NB and NN classifiers at $\alpha = 0.5$. The Soybean and Dermatology datasets peak at $\alpha = 0.75$, in these cases the number of features increased as α is lowered. By changing α solutions that performed poorly before become more viable. A set that scores well in the second component could be held back by the strict criteria of $\alpha = 1.0$. When α is relaxed the set improves the score in the first component making it a valid option. This could explain the increase in the number of features. The changes in performance again demonstrate probabilistic rough sets can be used to improve performance.

3.3.5 Comparison of RSPSO$_1$ and RSPSO$_2$

An argument could be made that $RSPSO_2$ is better than $RSPSO_1$ as a feature selection algorithm. $RSPSO_1$ performs very strongly on the Spect dataset, however, the other datasets nearly all experience the strongest performance when $\gamma = 1.0$. This means that the second component of the $RSPSO_1$ evaluation criterion is often hindering the performance. Note, however, that $RSPSO_2$ can often match or outperform $RSPSO_1$ with $\gamma = 1.0$ in fewer features. The second component of $RSPSO_2$ is having a positive effect on the algorithm. Take the Chess dataset as an example, $RSPSO_2$ maintains an accuracy of 0.985 in the DT classifier and 0.945-0.946 in the NN classifier. Only in the NB classifier does $RSPSO_1$ do well, and this is an exception to the pattern outside of the Spect dataset, as with the Spect dataset we see an improvement when the additional component is added. The Dermatology dataset peaks at 0.86, 0.885 and 0.919, for the DT, NB and NN classifiers respectively, in the $RSPSO_1$ approach, this is using an average of 21.0 features and it is all when $\gamma = 1.0$. $RSPSO_2$ peaks at 0.92, 0.93 and 0.909 using only 10 features, the second component of $RSPSO_2$ has halved the total number of features at little cost to the classification accuracy, the DT accuracy even improved. This does occur sometimes in $RSPSO_1$, as seen above, but is much more frequent in $RSPSO_2$.

It is noticeable that $RSPSO_1$ vastly reduces the number of total features when the second component is added, more so than $RSPSO_2$. In only the Spect dataset does this have a positive effect, in $RSPSO_2$ we see that the size of the subset does not drop so significantly from $RSPSO_1$ with $\gamma = 1.0$ which is when only the first component is considered. There is often a trade-off between number of selected features and the classification accuracy. There is an optimal subset of features that achieves the highest performance. This will often not include all features but reducing beyond this number leads to a decline. As way to enable the trade off between these two objectives we can introduce the additional measures. Sometimes the additional measure can have a positive effect on performance, the original component may not be finding an optimal subset. However, as a way to reduce the size beyond the optimal set with regards to classification $RSPSO_2$ provides a gradual way of doing this. The drop in the number of features is not so dramatic as with $RSPSO_1$.

While adding the second component in $RSPSO_2$ only led to an improvement in the Chess and Spect datasets, it provides a much better feature count reduction approach than the sec-

ond component in $RSPSO_1$. Adding a γ measurement into $RSPSO_2$ could further improve the dimension reduction aspect, given greater control to the size of the feature subset.

3.3.6 Comparison with Traditional FS Methods

Tables 3.7 and 3.7. present the results using forward selection and backward elimination algorithms as implemented by the Weka tool kit. The performance are the accuracy of the selected features using a DT classifier. Both approaches performed better than all features on the Dermatology and Waveform datasets. The backward elimination also performed better on the Soybean dataset. The results of these traditional algorithms can be compared with the *RSPSO* approaches.

Dataset	Chess		Derm		Lymph		Spect	
Method	Size	Accuracy	Size	Accuracy	Size	Accuracy	Size	Accuracy
CfsF	5	0.781	17	0.873	8	0.733	4	0.7
CfsB	5	0.781	17	0.873	8	0.733	4	0.7

Table 3.6: Peformance of Traditional Algorithms on the Chess, Derm, Lymph and Soy datasets

Dataset	Soy		Stat		Wave	
Method	Size	Accuracy	Size	Accuracy	Size	Accuracy
CfsF	12	0.805	22	0.833	17	0.749
CfsB	14	0.854	22	0.833	17	0.749

Table 3.7: Performance of Traditional Algorithms on the Spect, Stat and Wave datasets

The *RSPSO* approaches approaches outperform the traditional approaches on the following datasets: Chess, Spect and Statlog. $RSPSO_2$ also gets a higher accuracy on the Dermatology datasets. In the remaining datasets, while the average accuracy was not greater in at least one of the 30 runs PSO was able to find a better performing subset based on the best accuracy. This indicates the traditional methods are not finding the optimal subset in any of the datasets, while the PSO approaches can get closer to the optimal solution.

3.4 Chapter Summary

In this chapter, two new single objective feature selection algorithms were proposed. These are $RSPSO_1$ and $RSPSO_2$ based on probabilistic rough set theory and particle swarm optimisation. $RSPSO_2$ considers an aspect of rough set theory in the second component, while $RSPSO_1$ uses only a count of the number of selected features. Experimental results show that the single objective approach can find good subsets of features, and the use of additional components can further reduce the features. In some cases it can lead to an improvement in performance but often to reduce the size of the set further leads to a drop in performance. RSPSO2 can reduce the size without a great impact on performance.

With different parameters $RSPSO_1$ can achieve improved performance in the Spect dataset. In the remaining datasets, different parameters in $RSPSO_1$ led to a pronounced drop in the number of features selected, this was often accompanied by a drop in the classification performance. $RSPSO_2$ was largely more successful at reducing the number of features than $RSPSO_1$. In some cases it maybe beneficial to lower the number of features providing the performance is not heavily impacted. $RSPSO_2$ often dropped the total number of features

without having the same large impact on the performance of $RSPSO_1$. Multi-objective optimisation, the search for the pareto front, should be able to improve this approach further by returning the pareto front. Solutions could then be selected from the pareto front based on the relative importance of accuracy and efficiency.

When being compared to traditional approaches, the PSO approaches suffer from being stochastic approaches. The search does not always find the same subset, given long enough different searches may trend to the same solution but if the training time does not continue for an extended period of time the sets may not be found. In all cases at least one of the PSO runs was able to find a better set of features than the traditional approaches. The average maybe brought down by searches that did not return the same datasets, but given longer search times these would have found the better subsets. The traditional approaches would always return the non-optimal subset, it is not optimal as the PSO approaches did find superior solutions.

Chapter 4

Multi-Objective FS Based on PSO and RS

4.1 Introduction

This section introduces multi-objective particle swarm optimisation. Two multi-objective algorithms are introduced $RSMOPSO_1$ and $RSMOPSO_2$ as multi-objective extensions to $RSPSO_1$ and $RSPSO_2$. Rather than being forced to linearly combine the two objectives, as in Chapter 3., they are considered and evaluated separately. The search aims to find the Pareto front of non-dominated solutions which can then be used to select a balance between performance and efficiency.

This chapter is laid as follows. Section 4.2 presents the multi-objective particle swarm optimisation approach and introduces the new algorithms $RSMOPSO_1$ and $RSMOPSO_2$, pseudo-code and a background to the OMOPSO algorithm is also provided. The remaining sections discuss and present the results and experimental design of the investigation before the final section concludes the chapter.

4.1.1 Chapter Objectives

This chapter has the following objectives.

1. Investigate whether the probabilistic RS and the number of features in OMOPSO for feature selection can achieve a set of non-dominated solutions, which can achieve better classification performance than using all features.

2. Investigate whether using the probabilistic RS and the number of equivalence classes in OMOPSO for feature selection can achieve a set of non-dominated solutions and outperform the single objective algorithm and the above multi-objective algorithm.

4.2 Proposed Multi-objective Feature Selection Algorithm

The single objective fitness functions attempt to take the two objectives and combine them, in the first approach weights are applied to each component to balance the importance of the multiple objectives. As mentioned in the background chapter multi-objective optimisation allows for multiple objectives to be considered without needing to combine them into one. The objectives from the single objective evaluation criterion can be taken and used to design objectives for a multi-objective approach. These objectives are not identical to the components in the single objective approach. The components used in the single objective

fitness functions are scaled to be between 0 and 1, the number of selected features is divided by the total number of features as an example. This scaling puts the components into the same range as the purity measure so that they be combined without the result of one being more powerful than the other. This is not a consideration for the multi-objective approach as the objectives are considered and evaluated separate of each other.

Given that the search is no longer for a single global solution there are certain considerations to consider in the transformation from single objective to multi-objective. In MOPSO [38], first, *gbest* is replaced with a set of leaders, called the *external archive*. This is an archive in which all the non-dominated solutions are stored. Upon updating its position a particle will consult this archive, rather than a single *gbest* value. It is also this archive that is returned as the result of the algorithm, the set of all non-dominated solutions found in the search.

In traditional PSO the particles use the *gbest* solution as a guide when being evolved. As the *gbest* has been replaced with the external archive there is no longer one solution to use as the guide. One question is how can it be ensured that diversity is maintained by the swarm. The swarm should not converge to only one non-dominated solution. There is also the issue of when to replace *pbest* for each particle, as new solutions could represent non-dominated solutions. The first issue can be solved by selecting one particular solution from the archive to act as a *gbest* for each particle. Diversity can be ensured by selecting good solutions from the external archive. Usually, *pbest* is replaced when we find a new solution that is incomparable to the current *pbest* (both non-dominated with respective to each other).

4.2.1 New Algorithm: RSMOPSO$_1$

The first objective to consider is the component that is constant across all prior fitness measurements, that is Equation 2.12.

$$Obj_1 = 1 - \frac{\sum_{x=1}^{n} |\underline{apr}_G X_i|}{|\mathbb{U}|}$$

As with the single objective approaches in the previous chapter, the impact of the number of features needs to be measured. The first approach directly considers the number of features in a solution, this can be used as one of objectives.

$$Obj_2 = \#\, of\ Selected\ Features \tag{4.1}$$

As with the single objective approach, the first evaluation criterion will consider the rough set measure against the number of selected features, as with $RSPSO_1$.

$$RSMOPSO_1 = [Obj_1, Obj_2] \tag{4.2}$$

4.2.2 New Algorithm: RSMOPSO$_2$

The second single-objective approach, $RSPSO_2$, considers the size of the equivalence classes, we want larger numbers of instances in the equivalence classes. Larger equivalence classes means there will be fewer in total.

$$Obj_3 = \#\, of\ Equivalence\ Classes \tag{4.3}$$

The multi-objective search can aim to minimise a set of these objectives. Obj_2 and Obj_3 are not exclusive, increasing one does not mean a decrease in the other which is likely to be

the case when comparing both to Obj_1. It is implied that a smaller number of equivalence classes will lead to a larger average number of instances in each equivalence class.

The second evaluation criterion considers the rough set measure against the number of equivalence classes, as with $RSPSO_2$.

$$RSMOPSO_2 = [Obj_1, Obj_3] \tag{4.4}$$

Algorithm 2: MOPSO feature selection algorithm

```
1  begin
2      divide the data into train and test sets ;
3      initialise particles;
4      for i = 0 to number_of_particles do
5          for d = 0 to number_of_features do
6              v_id^0 = rand();
7              x_id^0 = rand();
8      initialise the set of leaders, leader set ;
9      initialise the external archive ;
10     for t = 0 to maxIteration do
11         for i = 0 to number_of_particles do
12             gbest ← select leader from leader set ;
13             update v_id^t for all d using Equation 2.2. ;
14             update x_id^t for all d using Equation 2.3. ;
15             apply bit-flip mutation ;
16             evaluate the two objectives ;
17             update p_id ;
18         identify the non-dominated solutions to update leader set ;
19         send leaders to external archive ;
20         calculate the crowding distance of each member in leader set ;
21     calculate the classification accuracy of the solutions in the external archive ;
22     return the solutions in the external archive and their classification accuracy ;
```

4.2.3 Summary of New Algorithms

The algorithm used for the MOPSO approach is known as OMOPSO [44, 38]. OMOPSO attempts to solve several of the questions presented by multi-objective PSO, highlighted in the previous paragraphs. OMOPSO uses two external archives, a *leader set* and an *external archive*, to store the current set of leaders from which *gbest* is selected and the archive to store the final solutions. A crowding factor is used to filter out the leaders when the set becomes too large, crowding is used to measure how similar solutions are to each other so similar/identical solutions can be identified [44]. This allows the *external archive* to store all the best solution, while the *leader set* can still be restrained in size. This crowding is also used in the selection of *gbest* for each particle, aiding the goal of maintaining diversity. Finally, mutation factors are introduced to further increase the search capacity of the swarm. OMOPSO has shown favourable performance over other multi-objective approaches such as PSO approaches derived from NSGAII [38].

Algorithm 2. provides the pseudo-code for OMOPSO approach, this can be used with the objectives above to measure the performance. The first steps are as with the BPSO approach described above, the dataset is split first into training and test data and the swarm

Dataset	#Attributes	#Classes	#Instances
Lymphography (Lymph)	18	4	148
Spect	22	2	267
Dermatology (Derm)	33	6	366
Soybean Large (Soy)	35	19	307
Chess	36	2	3196
Waveform (Wave)	40	3	5000

Table 4.1: Datasets

is initialised. Next, the external archives are initialised by introducing the non-dominated solutions into the *leader set*. With the swarm and external archives initialised we can begin the search. At each iteration for every particle we select a leader from the leader set to act as *gbest* before updating the position and velocity of each dimension. The mutation factors are then applied before the two objectives are evaluated using our choice of $RSMOPSO_1$ and $RSMOPSO_2$. After each particle has been evaluated we can update *pbest* for each particle and handle the external archives. First, the *leader set* is updated from the population. Next, we update the *external archive* containing the final solutions from the *leader set* and then remove particles from the *leader set* using the crowding factor. Finally, the classification accuracy for the solutions in the *external archive* is calculated and returned.

4.3 Experiments and Results

This section presents the experimental design and results of the new algorithms, $RSMOPSO_1$ and $RSMOPSO_2$.

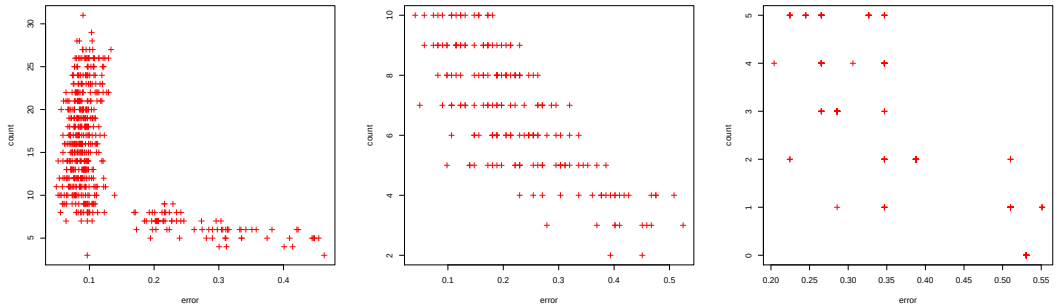
4.3.1 Experimental Design

The experiments were carried out using the jMetal framework, while the above algorithm is not provided it is relatively straight forward to extend the NSPSO algorithms that are provided. As was suggested by the authors of the OMOPSO algorithm the following parameters were used [44]. The acceleration constants, c_1 and c_2, are given a random number in the range of $[1.5, 2.0]$, note this is different to the constant value in single objective approach, $c_1 = c_2 = rand(1.5, 2.0)$. The inertia was given a random value in the range of $[0.1, 0.5]$, again instead of a single constant value, $w = rand(0.1, 0.5)$. Finally $v_{max} = 6.0$, the same as for the single objective approach. Similar datasets were used as for the *RSPSO* approach. 30 independent runs were executed for each experiment, the results shown are the combined external archives of each of these 30 runs. For each experiment the same ECS SGE Grid was used as for the single objective approaches.

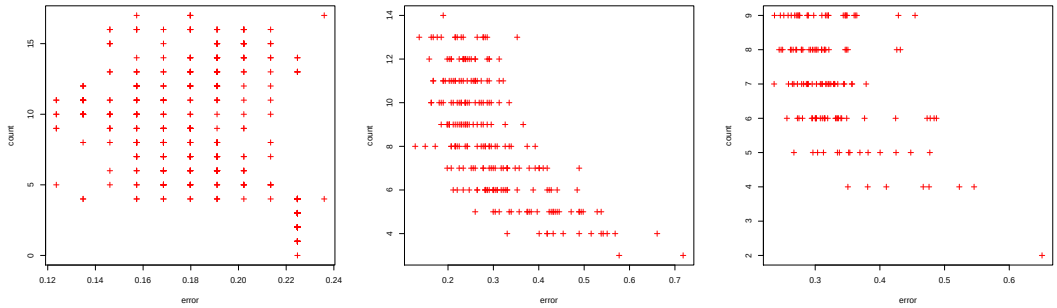
4.3.2 Overall Results

The results cannot be presented in the same format as the single objective approach. *RSMOPSO* returns the *external archive*, a set of non-dominated solutions, which should represent the Pareto front. Each solution will have a score for both the objectives, and the solutions can then be put through a classifier to test their performance on the training set. The results will be presented in graph form, with the objectives graphed against each other and the classification performance of the naive Bayes classifier graphed against the number of features in each solution.

The alternate classifiers are not shown as they would take up too much space. Naive Bayes was selected as decision trees perform additional feature selection as they are being built, it is easier to see the goodness of the selected features on a naive Bayes classifier as some are not being ignored. Nearest Neighbour was not used as the naive Bayes classifier is better suited to discrete data.



(a) Chess (b) Derm (c) Lymph



(d) Spect (e) Soy (f) Waveform

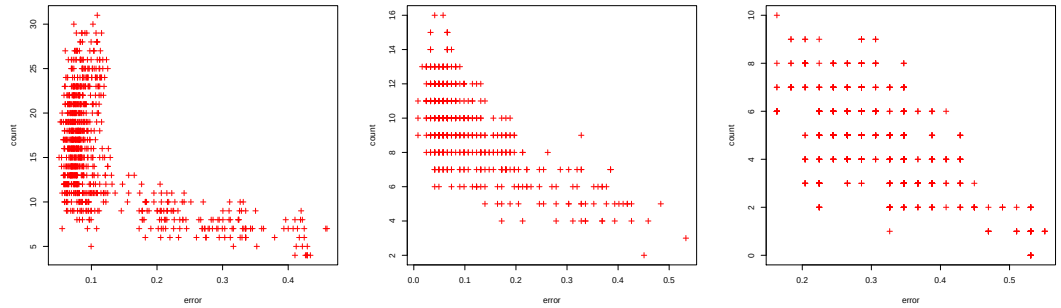
Figure 4.1: $RSMOPSO_1$ with $\alpha = 1.0$

Figure 4.1. presents the results using $RSMOPSO_1$ with $\alpha = 1.0$ while Figure 4.2. presents the same α values but with $RSMOPSO_2$. It is also important to note that both the Lymphography and Spect datasets have a wider spread of values than the other datasets. This could be because the Spect and Lymphography datasets have fewer total attributes and instances than the others, making the patterns harder to extract. Another important aspect of the results is that in nearly all circumstances the results peaked before dropping at higher numbers of features. In $RSMOPSO_1$ only the Dermatology dataset does not show this trend.

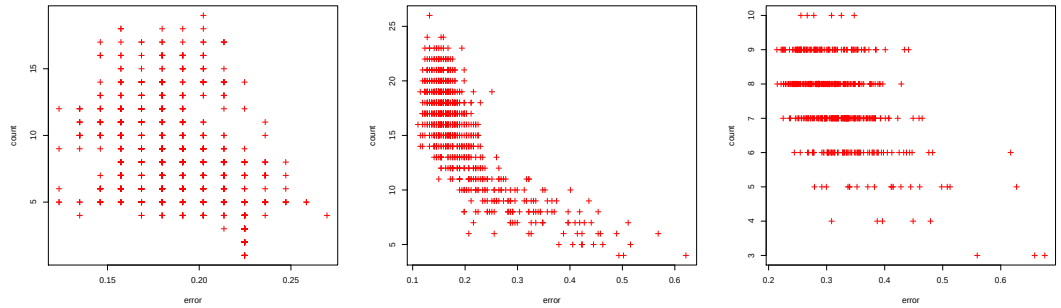
The results for the $RSMOPSO_2$ show the same trends as $RSMOPSO_1$, but to a greater extent. The Dermatology dataset now displays the same behaviour, the objectives in $RSMOPSO_2$ found a wider spread of results. This is also shown by the change in the Lymph dataset, the graph produced by $RSMOPSO_1$ was sparse in comparison to the graph resulting from $RSMOPSO_2$. $RSMOPSO_2$ also found better performing subsets than $RSMOPSO_1$ in the Lymph dataset. There is also improvement in the Soybean dataset, the curve is much more pronounced in $RSMOPSO_2$.

One thing to remember is that $RSMOPSO_2$ is considering the number of equivalence classes as the second objective while $RSMOPSO_1$ is considering the number of features. The number of equivalence classes could be any where up to the number of instances, if each instance was unique using the set of features, whereas the number of features is far more constrained. It seems likely then that $RSMOPSO_2$ has a greater potential for finding additional results, since the second objective can take a lot more values in $RSMOPSO_2$. This could account for the larger number of results returned by the $RSMOPSO_2$ approach when compared to the $RSMOPSO_1$ approach and explain the more pronounced curve in the Soybean and Dermatology datasets.

In the Spect dataset, the above appears to not have been an issue. If a lot of the instances share similarities this reduces the potential for a large number of equivalence classes. It is clear that the $RSPSO_2$ has been returning a greater number of results overall. When the objectives are converted into count and accuracy, neither of the algorithms consider



(a) Chess (b) Derm (c) Lymph



(d) Spect (e) Soy (f) Waveform

Figure 4.2: $RSMOPSO_2$ with $\alpha = 1.0$

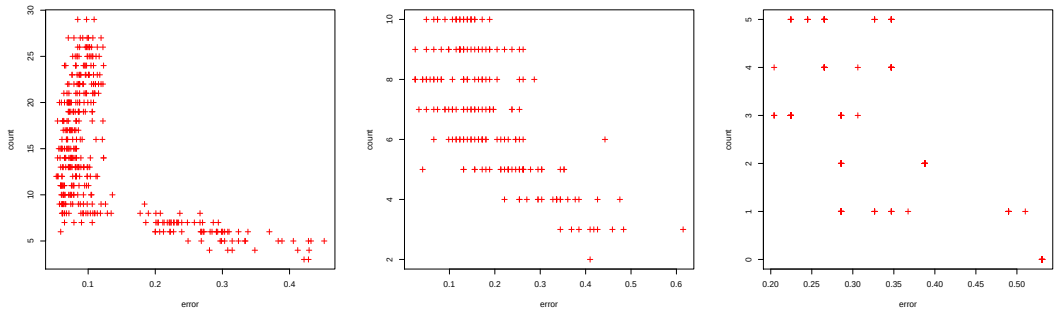
the accuracy only the RS estimate, what could have been a wide range of results along the number of equivalence classes becomes smaller as the two different solutions with a different number of equivalence class could have been using the same number of features.

Revisiting the shape of the curve, the data often peaks in classification accuracy before dropping in the sets with large features. This highlights that often there may exist an optimal set of features, that doesn't contain all features, and removing features from this will only lower performance. The Chess dataset using $RSMOPSO_1$ and $\alpha = 1.0$ is a good example, it peaks at an error rate of 0.049 using 11 features. More features have a negative impact, using all features achieved a larger error rate of 0.121. But when the number of features decreased below this set, the accuracy again began to drop. The same can be seen in $RSMOPSO_2$, also in the Chess dataset.

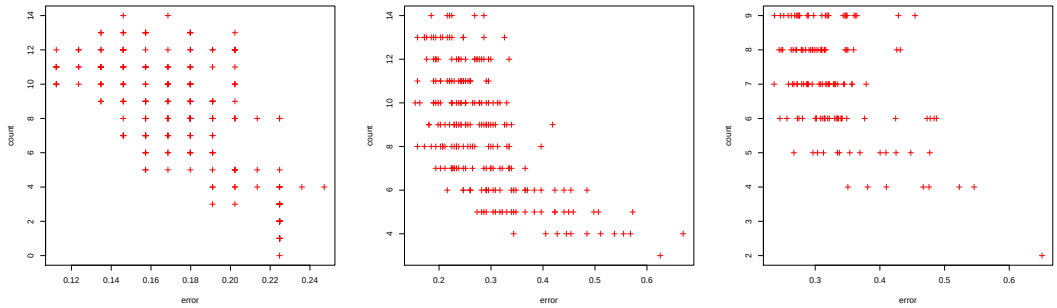
4.3.3 Varying Parameters for RSMOPSO$_1$

Again the potential for change in α needs to be considered. There is no γ to be considered, weights are not being assigned to the objectives. Figures 4.3 and 4.4. show the results for $alpga$ equalling 0.75 and 0.5 respectively when $RSMOPSO_1$ is used. Relaxing the value for α appears to bring the Dermatology dataset closer to the shape seen in $RSMOPSO_2$, the curve is more noticeable. Also when $\alpha = 0.75$ the Dermatology dataset found the strongest performing dataset, Table 4.2. (below) goes into more detail on the best performing solutions.

Of the datasets the large datasets, Chess was the least affected. There are two noticeable *clusters* in the Chess data plot which have remained largely untouched when $alpha = 0.75$. Sets that place in between the clusters begin to appear when α is 0.5, one could argue that this lower value of α has performed better because the search entered a space that was not searched by the other values. In general, however, $\alpha = 0.75$ appeared to add additional features at the extremes of the curves, making them more pronounced. Reducing α to 0.5 often increases the confusion, results are not added at the extremes but behind the front. This highlights, again, that the value for α is important.

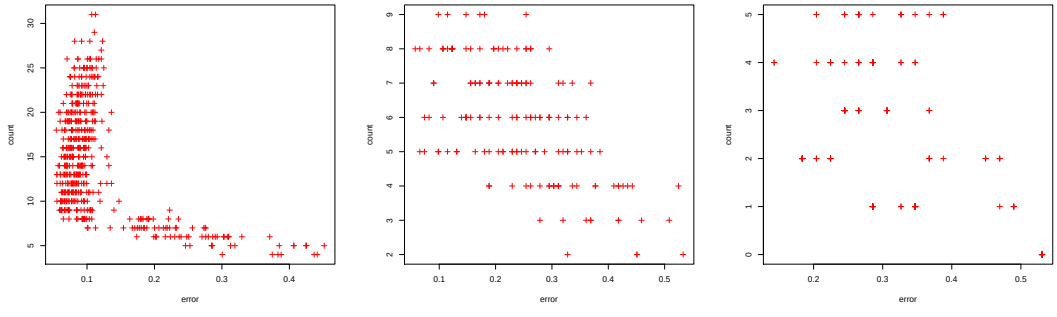


(a) Chess (b) Derm (c) Lymph

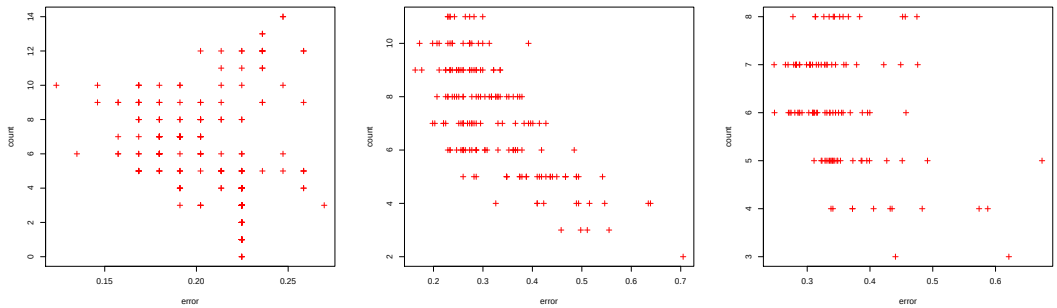


(d) Spect (e) Soybean (f) Waveform

Figure 4.3: $RSMOPSO_1$ with $\alpha = 0.75$



(a) Chess (b) Derm (c) Lymph

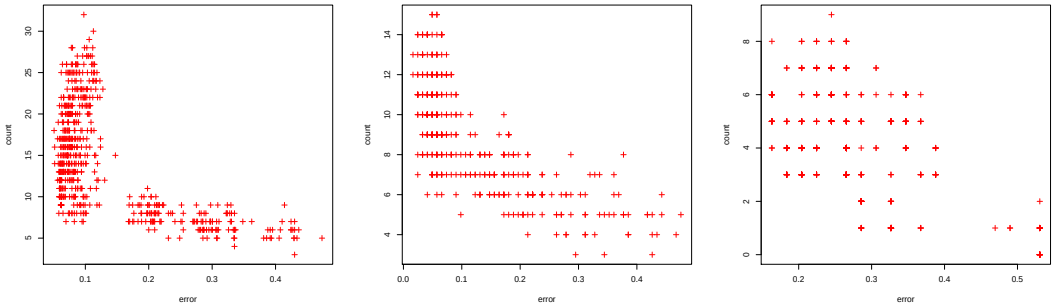


(d) Spect (e) Soybean (f) Waveform

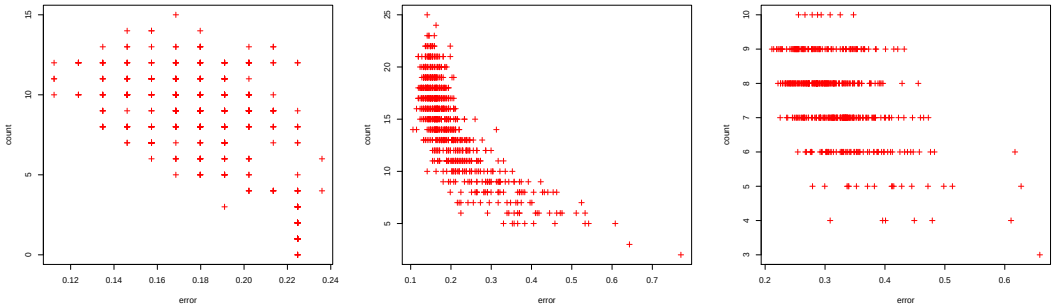
Figure 4.4: $RSMOPSO_1$ with $\alpha = 0.5$

4.3.4 Varying Parameters for RSMOPSO$_2$

Figures 4.5 and 4.6 show $RSMOPSO_2$ with α equalling 0.75 and 0.5 respectively. Similar trends can be seen in $RSMOPSO_2$ as in $RSMOPSO_1$. The multi-objective approach provides a set of returned solutions. The set allows for multiple aims to be met. If efficiency is the primary goal, the solution that achieves the most acceptable trade off between efficiency and classification performance will be selected. That lowering α increase the number of outliers means the Pareto front is extended, providing greater choice. The aim of the multi-objective search is to maximise the spread along the Pareto front. Relaxing α can increase the spread but, as the Dermatology dataset demonstrated, too far can have a negative effect. This was true in the single objective approaches and this effect can be seen in $RSMOPSO_2$ as well as $RSMOPSO_1$.



(a) Chess (b) Derm (c) Lymph



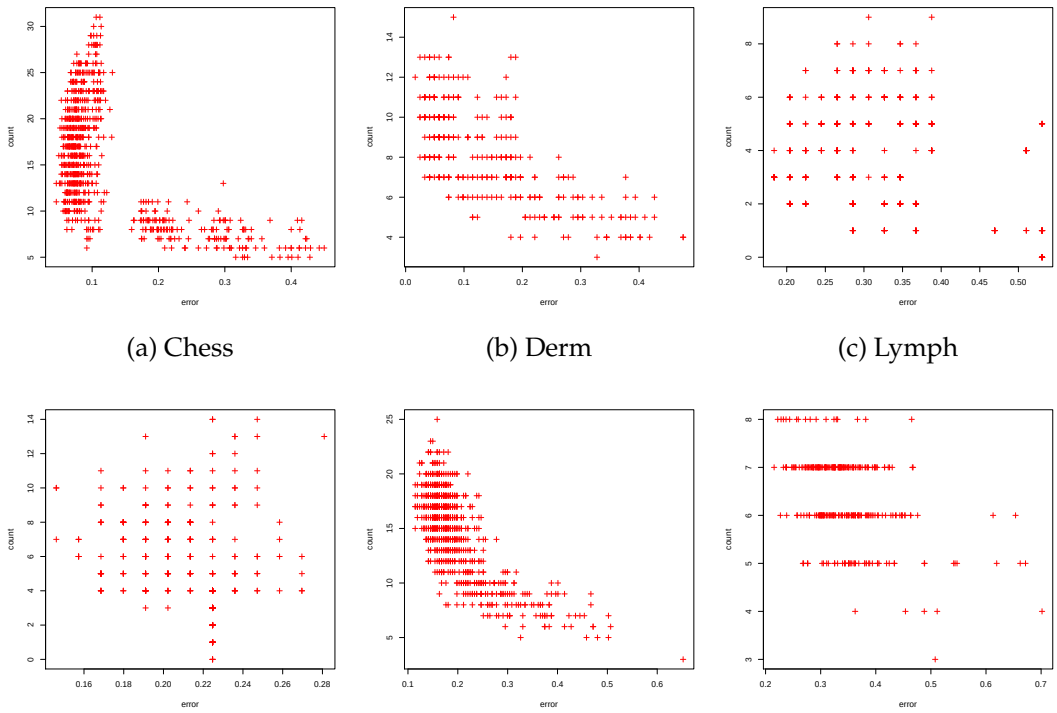
(d) Spect (e) Soybean (f) Waveform

Figure 4.5: $RSMOPSO_2$ with $\alpha = 0.75$

4.3.5 Comparison of RSMOPSO$_1$ and RSMOPSO$_2$

The main aim of multi-objective searches is to return a Pareto front, a solution can be chosen from this Pareto front that best suits the needs of the user. The searches for $RSMOPSO_2$ found a larger number of potential solutions. This is a potential advantage for $RSMOPSO_2$, the second objective in $RSMOPSO_2$ has a larger range than the second objective in $RSMOPSO_1$ allowing a greater spread of solutions. This demonstrated well by the Soybean dataset, the curve is much more pronounced in the results for $RSMOPSO_2$ because of the presence of solutions further from the centre. This can also be seen in the Dermatology dataset.

Table 4.2. presents the peaks in the datasets from the perspective of classification accuracy, the error is proportion of incorrectly classified instances. $RSMOPSO_2$ found superior subsets in the Chess, Dermatology, Soybean and Waveform datasets. $RSMOPSO_1$ found the



(a) Chess (b) Derm (c) Lymph

(d) Spect (e) Soybean (f) Waveform

Figure 4.6: $RSMOPSO_2$ with $\alpha = 0.5$

best set in the Lymph dataset. The two approaches achieved the same accuracy on the Spect dataset, but $RSMOPSO_1$ used fewer features. Often, the peaks in the $RSMOPSO_1$ results used fewer features than the peaks in $RSMOPSO_2$. However, given that there exists a set of potential results, $RSMOPSO_2$ can adjust the number of features down and select a different result to match the required

Overall, $RSMOPSO_2$ appears to be the better algorithm. A wider range of results are found, stretching out the Pareto front. In most cases $RSMOPSO_2$ also found better performing feature sets, giving additional options in terms of the trade off between classification performance and efficiency.

4.3.6 Comparison with Single-Objective Approach

To compare the single and multi-objective approaches the average performance of the solutions found by the single objective search and where it places in the Pareto front can be considered. In all cases the average performance of the single objective approach did not place *ahead* of the Pareto front returned by the MOPSO searches. Consider the Dermatology dataset, $RSPSO_1$ achieved an average performance of 0.935 with 21 features when $\alpha = 1.0$. This is an error rate of 0.065, so were this a result in a MOPSO search it would have been plotted at $(0.065, 21)$. When this is compared to Figure 4.1. it is much higher on the y-axis than any other of the other solutions. The performance is also lower than the best performing subset, as such it would not have dominated any of the solutions in the MOPSO Pareto front. This trend is evidenced across all datasets.

The multi-objective approach gives the ability to select how much importance should be given to the number of features versus the classification performance. Given that the average result found in the single objective approach would not place ahead of the Pareto

Dataset	α	numFeatures	Error
Chess	-	36	0.121
	1.0	11	0.049
	0.75	12	0.053
	0.5	18	0.054
Derm	-	33	0.041
	1.0	10	0.041
	0.75	8	0.025
	0.5	8	0.057
Lymph	-	18	0.122
	1.0	4	0.204
	0.75	3	0.204
	0.5	4	0.143
Spect	-	22	0.236
	1.0	10	0.124
	0.75	10	0.112
	0.5	10	0.124
Soy	-	35	0.097
	1.0	8	0.128
	0.75	10	0.154
	0.5	9	0.163
Wave	-	40	0.203
	1.0	7	0.236
	0.75	7	0.236
	0.5	7	0.247

Dataset	α	numFeatures	Error
Chess	-	36	0.121
	1.0	15	0.051
	0.75	18	0.051
	0.5	13	0.046
Derm	-	33	0.041
	1.0	9	0.008
	0.75	12	0.016
	0.5	12	0.016
Lymph	-	18	0.122
	1.0	6	0.163
	0.75	6	0.163
	0.5	3	0.184
Spect	-	22	0.236
	1.0	6	0.124
	0.75	12	0.112
	0.5	7	0.146
Soy	-	35	0.097
	1.0	16	0.110
	0.75	14	0.106
	0.5	17	0.115
Wave	-	40	0.203
	1.0	9	0.214
	0.75	9	0.211
	0.5	7	0.215

Table 4.2: Best Performing Subsets on $RSMOPSO_1$ (left) and $RSMOPSO_2$ (right)

front found in the multi-objective approach, it is clear the multi-objective approach is a better option. Multi-objective optimisation claims to be better than equivalent single objective approaches for 2 reasons.

- A set of results is returned.
- The performance across the objectives is superior to the single objective approach.

For two objectives there are then three things to consider. First, the multi-objective approach does return a set of possible solutions, allowing for better choice. Second, the classification performance as one of the objectives. The peak along the Pareto front is often better than the results found in the single objective search. Third, the number of features selected as one of the objectives. The multi-objective approach often achieved the highest performance in the fewer features relative to the single objective approach.

4.4 Chapter Summary

This chapter presents two multi-objective algorithms, $RSMOPSO_1$ and $RSMOPSO_2$, based on particle swarm optimisation and rough set theory. Each algorithm attempts to optimise two objectives by finding the Pareto front, a set of solutions that are non-dominated by each other. The first objective of both algorithms is a rough set measure, estimating the goodness of a set of features using rough set theory. The second objective measures the number of features selected by a potential solutions. $RSMOPSO_1$ uses the number of features as the objective, while $RSMOPSO_2$ uses the number of equivalence classes the features break the training set into. The multi-objective features are developed from the single objective algorithms, $RSPSO_1$ and $RSPSO_2$.

$RSMOPSO_2$ finds a larger number of solutions than $RSMOPSO_1$. The second objective of $RSMOPSO_2$ can take more values than the second objective in $RSMOPSO_1$ as two sets of features of equal size can describe a different number of equivalence classes. One of the major strengths of multi-objective optimisation is that the number of solutions it returns so $RSMOPSO_2$ could be considered stronger. The shape of the returned results also indicates the the second objective is a good estimation of the number of features, as it shows the same shape as $RSMOPSO_1$ which actually uses the number of features. Relaxing the value for α can be used to further spread the Pareto front. Relaxing α gives additional space to search, since new subsets can score highly. Relaxing α too far did not always have favourable results.

The multi-objective approach performs well when compared with the single objective, the choice of solutions give a greater ability to trade off between efficiency and performance. The best performing solution is capable of outperforming the single objective approaches and does so in fewer features. Overall, the multi-objective approach can be used to greater effect than the single-objective approach.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This project aimed to use rough set theory and particle swarm optimisation as a feature selection technique. The algorithms presented aimed to minimise the number of features used in analysis and improve the classification performance. Two approaches were successfully developed and tested, single objective particle swarm optimisation and multi-objective particle swarm optimisation. The approaches made use of probabilistic rough set theory as an evaluation criteria.

5.1.1 Single Objective Particle Swarm Optimisation

Two algorithms were presented, $RSPSO_1$ and $RSPSO_2$, with the aim of solving the problem of feature selection. The algorithms linearly combine a rough set measure of the fitness of a set of features with a measure designed to minimise the number of selected features. The rough set measure itself gave no consideration to the size of the subset, hence the need for the additional components. $RSPSO_1$ added a direct count of the number of features, while $RSPSO_2$ made use of rough set theory and measured the size of the equivalence classes.

Results showed that the new algorithms could compete with traditional methods, and improve the classification performance in some datasets. The additional components were successful in reducing the number of features beyond the sets returned by the simple rough set measure. The performance would sometimes drop along with the reduction of features, in many cases the trade off between efficiency and performance might make this beneficial. It could be claimed that as an additional feature reduction technique $RSPSO_2$ was superior. $RSPSO_1$ often lead to a large drop in performance along with a large drop in the number of features. The change in $RSPSO_2$ was often less drastic, making it more suited as a feature reduction technique.

Using probabilistic rough set theory was also found to be beneficial. By relaxing the criteria for entry into the upper and lower bounds, patterns could be extracted in fewer features improving the feature reduction aspect further. In some cases, reducing the value for α could also lead to an improvement in performance. In 30 runs the average performance was not always better than traditional methods, but the PSO search did often successfully find better sets at least once. The traditional methods were not finding the optimal subsets, where PSO could. Some of the alternate search methods for particle swarm optimisation, as presented in the related work section of the background, could make this search more reliable.

5.1.2 Multi-Objective Particle Swarm Optimisation

Once again two algorithms were presented, $RSMOPSO_1$ and $RSMOPSO_2$, based on $RSPSO_1$ and $RSPSO_2$ respectively. As with the single objective approach, $RSMOPSO_1$ used a direct count of the number of features while $RSMOPSO_2$ used the equivalence classes as the second objective. These algorithms were built on the OMOPSO search technique, using a set of non-dominated solutions as a guide and returning this set upon completion. The returned set can be used to fine tune requirements. If a drop in classification performance is acceptable, the set can be searched for a different solution using fewer features.

When compared with the single objective approach, the multi-objective approach could find smaller sets of features with better performance. The set of solutions would maximise the classification performance without needing such a large number of features. These solutions were competing with the larger sets of features in terms of performance. $RSMOPSO_2$ often found a larger number of results, as the second objective had a larger range of possible values, which gave $RSMOPSO_2$ the potential to find more solutions, an advantage in multi-objective searches. Again the effects of probabilistic rough set theory were tested, relaxing α gave a wider spread of results along the pareto front but again reducing it too far could negatively affect the performance.

5.2 Future Work

This section presents some potential future work using rough set theory and particle swarm optimisation.

- The datasets used to test the algorithms are reasonably small. The largest, Statlog, uses 6435 while Waveform has the most features at 40. The results could be further extended using larger datasets. The UCI repository contains the Connect4 dataset, a discretely valued dataset with 67557 instances and 42 features. There is the option of using discretization algorithms, as applied to Waveform and Statlog, on larger continuous datasets. The conclusions could be made more concrete with testing on larger datasets.

- Continuously valued datasets are far more common that discretely valued datasets. Dataset generation techniques would allow for additional large discretely valued datasets to be created and tested.

- The current algorithms require the value for α to be set. As has been demonstrated the correct value for α can have a positive effect on performance. Decision-theoretic rough set models have been presented [50] which can predict the required value for α through Bayesian decision procedure. This could simplify the algorithm and improve it by removing one of the required parameters.

- Feature selection is only one technique to overcome some of the problems seen in classification tasks. *Feature construction* is a second approach [43]. Feature construction attempts to build new high level features from the provided low level features. The approach is useful for when the existing features have been poorly chosen. Feature construction has been well researched using genetic programming [35, 34] but from a particle swarm optimisation perspective it has yet to be tested.

Bibliography

[1] ABDI, H., AND WILLIAMS, L. J. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics 2*, 4 (2010), 433–459.

[2] CARDIE, C. Using decision trees to improve case-based learning. In *Proceedings of the Tenth International Conference on Machine Learning (ICML)* (1993), pp. 25–32.

[3] CHAKRABORTY, B. Genetic algorithm with fuzzy fitness function for feature selection. In *IEEE International Symposium on Industrial Electronics (ISIE'02)* (2002), vol. 1, pp. 315–319.

[4] CHAKRABORTY, B. Feature subset selection by particle swarm optimization with fuzzy fitness function. In *3rd International Conference on Intelligent System and Knowledge Engineering (ISKE'08)* (2008), vol. 1, pp. 1038–1042.

[5] CHUANG, L. Y., CHANG, H. W., TU, C. J., AND YANG, C. H. Improved binary pso for feature selection using gene expression data. *Computational Biology and Chemistry 32*, 29 (2008), 29– 38.

[6] CHUANG, L. Y., TSAI, S. W., AND YANG, C. H. Improved binary particle swarm optimization using catfish effect for feature selection. *Expert Systems with Applications 38* (2011), 12699–12707.

[7] CRAMER, N. L. A representation for the adaptive generation of simple sequential programs. In *Proceedings of the 1st International Conference on Genetic Algorithms* (Hillsdale, NJ, USA, 1985), L. Erlbaum Associates Inc., pp. 183–187.

[8] DASH, M., AND LIU, H. Feature selection for classification. *Intelligent Data Analysis 1*, 4 (1997), 131–156.

[9] DORIGO, M., AND STÜTZLE, T. *Ant Colony Optimization*. Bradford Company, Scituate, MA, USA, 2004.

[10] ESTÉVEZ, P. A., TESMER, M., PEREZ, C. A., AND ZURADA, J. M. Normalized mutual information feature selection. *Trans. Neur. Netw. 20*, 2 (Feb. 2009), 189–201.

[11] GHEYAS, I. A., AND SMITH, L. S. Feature subset selection in large dimensionality domains. *Pattern Recognition 43*, 1 (2010), 5–13.

[12] GOLDBERG, AND HOLLAND. Genetic algorithms and machine learning. *MACHLEARN: Machine Learning 3* (1988).

[13] GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA, 1989.

[14] GUYON, I., AND ELISSEEFF, A. An introduction to variable and feature selection. *The Journal of Machine Learning Research 3* (2003), 1157–1182.

[15] HAMDANI, T. M., WON, J.-M., ALIMI, A. M., AND KARRAY, F. Multi-objective feature selection with NSGA II. In *8th International Conference on Adaptive and Natural Computing Algorithms (ICANNGA'07) Part I* (2007), vol. 4431, pp. 240–247.

[16] HOLLAND, J. H. *Adaption in Natural and Artificial Systems*. University of Michigan Press, 1975.

[17] KENNEDY, J., AND EBERHART, R. Particle swarm optimization. In *IEEE International Conference on Neural Networks* (1995), vol. 4, pp. 1942–1948.

[18] KENNEDY, J., AND EBERHART, R. A discrete binary version of the particle swarm algorithm. In *IEEE International Conference on Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation.* (1997), vol. 5, pp. 4104–4108.

[19] KOHAVI, R., AND JOHN, G. H. Wrappers for feature subset selection. *Artif. Intell. 97*, 1-2 (Dec. 1997), 273–324.

[20] KOTSIANTIS, S. B., ZAHARAKIS, I. D., AND PINTELAS, P. E. Machine learning :a review of classification and combining techniques. *Artif. Intell. Rev. 26*, 3 (Nov. 2006), 159–190.

[21] KOZA, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

[22] KOZA, J. R. *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge Massachusetts, 1994.

[23] KOZA, J. R. Introduction to genetic programming. In *Annual conference on Genetic and evolutionary computation (GECCO'07)* (2007), vol. 5, pp. 3323–3365.

[24] KWAK, N., AND CHOI, C.-H. Input feature selection by mutual information based on parzen window. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 24*, 12 (dec 2002), 1667 – 1671.

[25] LIN, S. W., YING, K. C., CHEN, S. C., AND 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.

[26] LIU, Y., WANG, G., CHEN, H., AND DONG, H. An improved particle swarm optimization for feature selection. *Journal of Bionic Engineering 8*, 2 (2011), 191–200.

[27] MARILL, T., AND GREEN, D. On the effectiveness of receptors in recognition systems. *IEEE Transactions on Information Theory 9*, 1 (1963), 11–17.

[28] MING, H. A rough set based hybrid method to feature selection. In *International Symposium on Knowledge Acquisition and Modeling (KAM '08)* (2008), pp. 585–588.

[29] MITCHELL, T. M. *Machine Learning*, 1 ed. McGraw-Hill, Inc., New York, NY, USA, 1997.

[30] MOHEMMED, A., ZHANG, M., AND JOHNSTON, M. Particle swarm optimization based adaboost for face detection. In *IEEE Congress on Evolutionary Computation (CEC'09)* (2009), pp. 2494–2501.

[31] MUNI, D., PAL, N., AND DAS, J. Genetic programming for simultaneous feature selection and classifier design. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 36*, 1 (2006), 106–117.

[32] NESHATIAN, K., AND ZHANG, M. Dimensionality reduction in face detection: A genetic programming approach. In *24th International Conference Image and Vision Computing New Zealand (IVCNZ'09)* (2009), pp. 391–396.

[33] NESHATIAN, K., AND ZHANG, M. Genetic programming for feature subset ranking in binary classification problems. In *European Conference on Genetic Programming* (2009), pp. 121–132.

[34] NESHATIAN, K., ZHANG, M., AND ANDREAE, P. A filter approach to multiple feature construction for symbolic learning classifiers using genetic programming. *Evolutionary Computation, IEEE Transactions on PP*, 99 (2012), 1.

[35] NESHATIAN, K., ZHANG, M., AND JOHNSTON, M. Feature construction and dimension reduction using genetic programming. In *Proceedings of the 20th Australian joint conference on Advances in artificial intelligence* (Berlin, Heidelberg, 2007), AI'07, Springer-Verlag, pp. 160–170.

[36] PAWLAK, Z. Rough sets. *International Journal of Parallel Programming 11* (1982), 341–356. 10.1007/BF01001956.

[37] PUDIL, P., NOVOVICOVA, J., AND KITTLER, J. V. Floating search methods in feature selection. *Pattern Recognition Letters 15*, 11 (1994), 1119–1125.

[38] REYES-SIERRA, M., AND COELLO, C. A. C. Multi-objective particle swarm optimizers: A survey of the state-of-the-art. *INTERNATIONAL JOURNAL OF COMPUTATIONAL INTELLIGENCE RESEARCH 2*, 3 (2006), 287–308.

[39] RUMELHART, D. E., HINTON, G. E., AND WILLIAMS, R. J. Neurocomputing: foundations of research. MIT Press, Cambridge, MA, USA, 1988, ch. Learning representations by back-propagating errors, pp. 696–699.

[40] RUSSELL, S. J., NORVIG, P., CANDY, J. F., MALIK, J. M., AND EDWARDS, D. D. *Artificial intelligence: a modern approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[41] SHANNON, C. E., AND WEAVER, W. *A Mathematical Theory of Communication*. University of Illinois Press, Champaign, IL, USA, 1963.

[42] SHI, Y., AND EBERHART, R. A modified particle swarm optimizer. In *IEEE International Conference on Evolutionary Computation (CEC'98)* (1998), pp. 69–73.

[43] SHUNG YANG, D., RENDELL, L., AND BLIX, G. A scheme for feature construction and a comparison of empirical methods. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence* (1991), Morgan Kaufmann, pp. 699–704.

[44] SIERRA, M. R., AND COELLO COELLO, C. A. Improving pso-based multi-objective optimization using crowding, mutation and ∈-dominance. In *Proceedings of the Third international conference on Evolutionary Multi-Criterion Optimization* (Berlin, Heidelberg, 2005), EMO'05, Springer-Verlag, pp. 505–519.

[45] STEARNS, S. On selecting features for pattern classifier. In *Proceedings of the 3rd International Conference on Pattern Recognition* (Coronado, CA, 1976), pp. 71–75.

[46] UNLER, A., AND MURAT, A. A discrete particle swarm optimization method for feature selection in binary classification problems. *European Journal of Operational Research 206*, 3 (2010), 528–539.

[47] WANG, X., YANG, J., TENG, X., XIA, W., AND JENSEN, R. Feature selection based on rough sets and particle swarm optimization. *Pattern Recognition Letters 28*, 4 (2007), 459–471.

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

[49] YANG, C. S., CHUANG, L. Y., KE, C. H., AND YANG, C. H. Boolean binary particle swarm optimization for feature selection. In *IEEE Congress on Evolutionary Computation (CEC'08)* (2008), pp. 2093–2098.

[50] YAO, Y. Decision-theoretic rough set models. In *Proceedings of the 2nd international conference on Rough sets and knowledge technology* (Berlin, Heidelberg, 2007), RSKT'07, Springer-Verlag, pp. 1–12.

[51] YAO, Y. Probabilistic rough set approximations. *Int. J. Approx. Reasoning 49*, 2 (Oct. 2008), 255–271.

[52] YAO, Y., AND ZHAO, Y. Attribute reduction in decision-theoretic rough set models. *Information Sciences 178*, 17 (2008), 3356 – 3373.

[53] YU, L., AND LIU, H. Efficient feature selection via analysis of relevance and redundancy. *J. Mach. Learn. Res. 5* (Dec. 2004), 1205–1224.

[54] YUAN, H., TSENG, S.-S., GANGSHAN, W., AND FUYAN, Z. A two-phase feature selection method using both filter and wrapper. In *Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on* (1999), vol. 2, pp. 132 –136 vol.2.

[55] YUSTA, S. C. Different metaheuristic strategies to solve the feature selection problem. *Pattern Recognition Letters 30* (2009), 525–534.

Appendix A

Summer Research

My summer research focused on investigating the applicability of Information Gain and Mutual Information based on Shannon's entropy as a filter based feature selection result. The following paper was accepted in IEEE 2012 Congress on Evolutionary Computation.

Liam Cervante, Bing Xue, Lin Shang and Mengjie Zhang. "Binary Particle Swarm Optimisation for Feature Selection: A Filter Based Approach". *Proceedings of 2012 IEEE Congress on Evolutionary Computation*. IEEE Press. Brisbane, Australia, June 2012.

The goals of the paper are highlighted as follows.

- Develop a filter feature selection algorithm based on PSO and the mutual information between pairs of features
- Develop a filter feature selection algorithm based on PSO and the group entroy of sets of features

The research undertaken over summer differs from the research presented in this report as the summer research focused on a different measure of fitness, information gain and entropy not rough set theory. The goals are different from the goals presented in the introduction.