Feature Selection Ensemble

Qiang Shen, Ren Diao and Pan Su

Aberystwyth University, Aberystwyth, UK
{qqs, rrd09, pas23}@aber.ac.uk

Abstract

Many strategies have been exploited for the task of feature selection, in an effort to identify more compact and better quality feature subsets. Such techniques typically involve the use of an individual feature significance evaluation, or a measurement of feature subset consistency, that work together with a search algorithm in order to determine a quality subset. Feature selection ensemble aims to combine the outputs of multiple feature selectors, thereby producing a more robust result for the subsequent classifier learning tasks. In this paper, three novel implementations of the feature selection ensemble concept are introduced, generalising the ensemble approach so that it can be used in conjunction with many subset evaluation techniques, and search algorithms. A recently developed heuristic algorithm: harmony search is employed to demonstrate the approaches. Results of experimental comparative studies are reported in order to highlight the benefits of the present work. The paper ends with a proposal to extend the application of feature selection ensemble to aiding the development of biped robots (inspired by the authors' involvement in the joint celebration of Olympic and the centenary of the birth of *Alan Turing*).

1 Introduction

The main aim of feature selection (*FS*) is to discover a minimal feature subset from a problem domain while retaining a suitably high accuracy in representing the original data [9]. Practical problems which arise when analysing data in real-world applications are often related to the number of features (so-called "curse-of-dimensionality" [2]), and the inability to identify and extract patterns or rules due to high inter-dependency amongst a large number of individual features. Human evaluation and subsequent pattern identification is also limited when considering such datasets [58]. Techniques to perform tasks such as text processing, data classification and systems control [32, 38, 46, 47] can benefit greatly from *FS*, once the noisy, irrelevant, redundant or misleading features are removed [22].

Given a dataset with N features, the task of *FS* can be seen as a search for an "optimal" feature subset through the competing 2^N candidate subsets. Optimality of subsets is subjective, depending on the problem at hand, and a subset that is selected as optimal using one particular evaluation function may not be equivalent to that of a subset selected by another. Various evaluation techniques have been developed in the literature to judge the quality of the discovered feature subsets. Several techniques rank the features based on certain importance measures, for example, information gain, chi-square analysis [62], symmetrical uncertainty measure [45], and the RELIEF algorithm [25]. This category also includes an approach that exploits the Good-Turing frequency estimation [43] that was originally developed by *Alan Turing* and his colleagues to aid the decryption of German communications during the Second World War. It works on the basis of estimating the number of times a certain feature would have occurred in a dataset if the dataset was perfectly representative of the problem domain.

Recent trends in developing feature selection methods focus on evaluating a given feature subset as a whole instead of measuring on an individual feature basis. This forms an alternative approach to the aforementioned. Popular methods include the fuzzy-rough feature selection [21, 23, 33], probabilistic consistency based feature selection [10], and correlation-based feature subset selection [16]. These techniques together with individual feature-based methods are often collectively classified as the filter based approach. Such an approach is usually used as a preprocessing step and is independent of any

learning algorithm that may be subsequently employed. Wrapper methods [19, 24] in contrast to the filter techniques are often used in conjunction with a learning or data mining algorithm, where the learning algorithm forms part of the feature validation process. The generalised wrapper algorithm is similar to the filter approach apart from the fact that a learning algorithm is employed in place of an evaluation metric as used in the strict filter methods. Note that hybrid algorithms [63] exist which attempt to combine the benefits provided by both types of approach.

Many of the existing mechanisms for feature selection follow the general principle of supervised learning, be they filter or wrapper based approaches. As such, they work by relying on identified correlations between class or decision labels and the underlying feature values [29]. However, in many real-world applications, the thorough interpretation of a large data may become infeasible and hence, the amount of labelled training samples is often limited. This makes unsupervised feature selection algorithms [4, 5, 30], and semi-unsupervised learning [31] techniques potentially beneficial and desirable [15]. The resulting techniques base their judgements on particular characteristics of data values, typically captured by entropy [8], data reliability [5] or locality preserving ability [61].

Independent of the learning mechanism, a common issue that all *FS* methods need to address is how they search for an "optimal" feature subset. To this end, an exhaustive method may be used, however it is often impractical for most datasets. Alternatively, hill-climbing based approaches have been exploited where features are added or removed one at a time until there is no further improvement to the current candidate solution. Unfortunately, these approaches may lead to the discovery of sub-optimal subsets, both in terms of the evaluation score and the subset size. Other algorithms therefore adopt random search or heuristic strategies in an attempt to avoid such short-comings. These include nature inspired heuristics such as genetic algorithms (*GA*) [57], genetic programming [40], and particle swarm optimisation (*PSO*) [54].

Harmony search (*HS*) [14, 27] is a relatively new meta-heuristic algorithm that mimics the improvisation process of music players. The *HS* algorithm has been very successful in a wide variety of engineering optimisation problems [13, 52] and machine learning tasks [35, 37]. It has demonstrated several advantages over traditional optimisation techniques. *HS* imposes only limited mathematical requirements and is not sensitive to its initial parameter settings. New potential solution vector is generated after considering all existing vectors. The base algorithm has been improved by methods that adapt its parameters during the search process [7, 34]. Taking advantages of the resulting powerful search methods, an *FS* algorithm based on *HS* has recently been developed [11]. Although the performance of this new development is promising, it merely contributes to the family of *FS* techniques as yet another single method that produces a single feature subset of features when presented with a training dataset. The performance of such techniques may vary significantly over different problem domains.

"Feature selection ensemble" (*FSE*) is an ensemble-based method that aims to construct a group of feature subsets, and then produce an aggregated result out of the group. In so doing, the performance variance of obtaining a single result from a single approach can be reduced. It is also intuitively appealing that the combination of multiple subsets may remove less important features, resulting in a compact, robust, and efficient solution. Ensembles of feature ranking techniques have been studied in the literature for the purpose of text classification [41] and software defect prediction [53], they work by combining the ranking scores or exploring the rank ordering of the features. Additionally, feature redundancy elimination has been achieved by the used of tree-based classifiers ensembles [50]. A number of terms similar to *FSE* have been introduced in the literature to represent a variety of different meanings, most of which refer to classifier ensembles built upon feature subsets (e.g. [42]).

In this paper, three novel approaches that implement the (*FSE*) concept are proposed. These include: 1) building ensembles using stochastic search algorithms, 2) generating diversity by partitioning the training data, and 3) constructing ensembles by mixing various different *FS* approaches. A preliminary, agreement threshold based approach for subset aggregation is also proposed, which may simulate

the popular "majority voting" scheme [48] often adopted by various ensemble approaches to classifier learning. The proposed methods are more flexible than the existing techniques, allowing feature subset evaluators to be used in conjunction with feature ranking. The stochastic search based, and the data partition based methods are able to spawn ensembles from just a single *FS* algorithm, which may potentially reduce the need to configure multiple base feature selectors.

The remainder of this paper is structured as follows. Section 2 describes how *FS* may be modelled as an optimisation task solvable by *HS*, and details the approaches developed to tackle such a problem. A feature evaluation metric that makes use of data reliability measures is introduced in section 3, which also serves to provide an overview of how unsupervised data analysis techniques can be employed to tackle *FS* tasks. The three proposed implementations of the *FSE* concept are explained in section 4, where illustrative flow charts and pseudo codes of the algorithms are provided to aid understanding. In addition, this section outlines a complexity analysis of the proposed implementations. Section 5 presents the experimentation carried out on real-world problem cases [1]. A discussion is also given in this section that attempts to empirically identify important characteristics of the presented methods. Finally, section 6 concludes the paper and proposes further research in the area. It also addresses a different application domain of *FS* from that of pattern classification, proposing the use of the *FSE* techniques to support the development of biped robots (which is inspired by the authors' involvement in the upcoming joint celebration of 2012 London Olympic and the centenary of the birth of *Alan Turing*).

2 Feature Selection with Harmony Search

2.1 Key Concepts

Harmony Search (*HS*) [27] mimics the improvisation process of musicians, during which, each musician plays a note for finding a best harmony all together. The basic concepts of *HS* and application of such concepts in performing optimisation are outlined below, together with an introduction to the dynamic parameter control involved in *HS*.

The key concepts of *HS* are musicians, notes, harmonies and harmony memory (HM). In most optimisation problems solvable by *HS*, the musicians are the decision variables of a certain function being optimised. The notes played by the musicians are the values each decision variable can take. The harmony contains the notes played by all musicians, or an emerging solution vector containing the values for each decision attribute. The harmony memory contains harmonies played by the musicians, or a storage place for potential solution vectors. A more concrete representation of harmony memory is a two dimensional matrix, where the rows contain harmonies (solution vectors) with the number of rows being predefined and bounded by the harmony memory size. Each column is dedicated to one musician, and the entire column stores all the notes played by the musician in all saved harmonies, referred to as the working note domain for each musician in this paper.

Harmony Search for *FS*(*HSFS*) [11] treats musicians as independent experts, and each musician can vote for one feature to be included in the feature subset when improvising a new harmony. The harmony is then the combined vote from all musicians, indicating which features are being nominated. The entire pool of original features forms the range of notes available to the musicians. Multiple musicians are allowed to choose the same feature, and they may opt to choose no feature at all. For example, the harmony $\{A,-,B,B,C,-\}$ translates into feature subset $\{A,B,C\}$, $-$ here represents a null note.

2.2 Iteration Steps

HS can be divided into two core phases, initialisation and iteration, as illustrated in Fig 1.

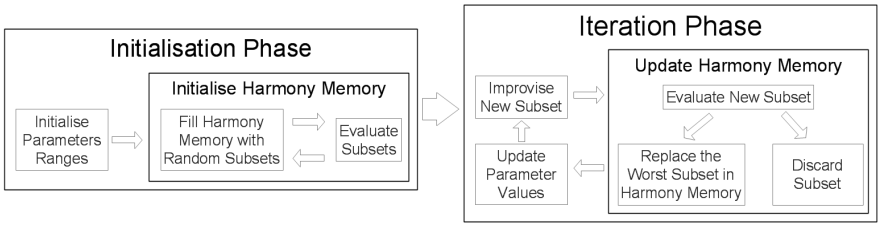


Figure 1: Illustration of Harmony Search

- **Initialise Problem Domain** The parameters of *HS* are assigned according to the problem, including: size of harmony memory, number of musicians, max iteration, and the harmony memory considering rate (HMCR). The harmony memory of size m is initialised by random generation. This provides each musician a working note domain of m values, which may include identical notes, and nulls. A new harmony is produced by each musician randomly choosing one feature from their note domain. The new harmony is then evaluated using the given cost function. It is used to replace the worst harmony in the harmony memory if a better score is achieved, or discarded otherwise.

- **Improvise New Harmony** A new value is chosen randomly by each musician out of their note domain, and together forms a new harmony. The HMCR parameter, ranging from 0 to 1, is the rate of choosing one value from the historical notes stored in the harmony memory. With $(1 - HMCR)$ set to be the rate of randomly selecting one value from the range of all possible notes of the corresponding variable. If HMCR is set low, the musicians will constantly explore other areas of the solution space, and a higher HMCR will restrict the musicians to historical choices. The other dynamic parameter: pitch adjustment rate (PAR) is not employed for the purpose of *FS*[11], because no general dependency exists between neighbouring features, where the original intention of PAR is to adjust to neighbouring values to refine solution quality.

- **Update Harmony Memory** If the new harmony is better than the worst harmony in the harmony memory, judged by the objective function, the new harmony is then included in harmony memory and the existing worst harmony is removed. The algorithm continues to iterate until the maximum number of iterations has been reached.

- **Parameter Control** To improve *HS* and eliminate the drawbacks lying with the use of fixed parameter values, a dynamic parameter adjustment scheme [11] was proposed to modify parameter values at run time. Parameters are gradually varied through a process of: initial solution space exploration, intermediate solution refinement, and fine tuning optimal solution towards termination.

3 Data Reliability Based Feature Selection

Data-oriented operators such as the dependent ordered weighted averaging (DOWA) utilise centralised data structures to generate reliable weights [59, 60] for aggregating information. An efficient nearest-neighbour-based method for the assessment of data reliability or relevance has been proposed [5] in which the local data structure that represents a strong agreement of consensus on information can be explored. This reliability measure is effective to discriminate the weight of different input arguments; and the local neighbouring context which has previously been realised as a closest cluster is replaced by a set of K nearest neighbours.

More formally, given a collection of data arguments $A = \{a_1, \cdots, a_L\}$, let $N_{a_i}^K$ be a set of K nearest neighbours of an argument a_i, where $N_{a_i}^K \subset A$, $n_j \in N_{a_i}^K$, $n_j \neq a_i$, $j = 1, \cdots, K$. The reliability measure $R_{a_i}^K \in [0, 1]$, $i = 1, \cdots, L$ can be computed such that:

$$R_{a_i}^K = 1 - \frac{D_{a_i}^K}{D_{\max}}, \quad D_{a_i}^K = \frac{1}{K} \sum_{\forall n_j \in N_{a_i}^K} |a_i - n_j| \tag{1}$$

where $D_{\max} = \max_{a_p, a_q \in A, a_p \neq a_q} |a_p - a_q|$.

The nearest-neighbour-based method presents two main advantages. First, the otherwise high computational cost required by conventional approaches to cluster-based measuring of data reliability is reduced. Both time and space complexity decrease, from $O(L^3)$ to $O(L^2)$ and $O(L^2)$ to $O(L)$, respectively. Second, the nature of the distributed approach to clustering is not only preserved but also reinforced such that arguments very far from the global centre can be considered reliable if they are close to members of their local neighbour sets. Figure 2 illustrates this approach where arguments a_1 and a_2 are considered reliable given their local neighbour sets N_{a_1} and N_{a_2}. This technique can be

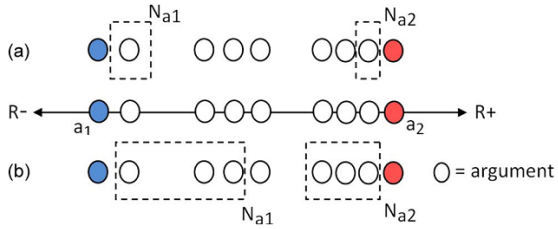


Figure 2: Different local neighbouring sets N_{a_1} and N_{a_2}, (a) $K = 1$ and (b) $K = 3$.

applied to perform unsupervised feature selection. In particular, the reliability measure can be regarded as the discriminant factor to justifying the relevance of each data feature. Its result reflects the intuition that a feature is considered reliable (or relevant) if its values are tightly grouped together (i.e., possessing a rigid value pattern). In essence, with a data set of N samples $X = \{x_1, \cdots, x_N\}$, and M features $F = \{f_1, \cdots, f_M\}$, the reliability FR_r of feature $f_r, r = 1, \cdots, M$, can be determined by estimating the accumulative reliability measures generated for each of its value f_{ir}, $i = 1, \cdots, N$. The computation process for this involves the following two steps:

Step 1. Acquire the reliability measure $R_{f_{ir}}^K$ of each feature value f_{ir}, $i = 1, \cdots, N$ according to Eqn. 1, using the set of K nearest neighbours.

Step 2. Calculate the accumulative reliability FR_r of feature f_r, $r = 1, \cdots, M$, by combining the reliability measures of all its values, i.e., $FR_r = \sum_{i=1}^{N} R_{f_{ir}}^K$.

From this, the original features can be ranked in accordance with their reliability degrees. The higher the reliability is, the more relevant the feature becomes. Similar to the work of [17, 18, 44], a simple threshold-based feature selection method can then be established as follows: A feature $f_r \in F$, $r = 1, \cdots, M$, is selected only if its corresponding reliability FR_r exceeds a given threshold. Such a discriminating limit may be subjectively provided. However, a predefined threshold may not be effective for a variety of data with different characteristics. It is better to learn this from the underlying data set. Empirically, the threshold can be set as the average reliability of all features $FR_{average} = \frac{1}{M} \sum_{r=1}^{M} FR_r$, over the training data available.

Summarising the above, a heuristic selection procedure as shown in Fig. 3 can be employed to justify the content of the reduced feature set $B \subseteq F$, where B is first initialised to the full feature set F, and a feature f_r is dropped from B if $FR_r < FR_{average}$.

F, the original feature set, $F = (f_1, \cdots, f_M)$;
f_i, the data feature, $i = 1, \cdots, M$;
B, the reduced feature set;
FR_i, the reliability of feature $i, i = 1, \cdots, M$;
$FR = \{FR_i, \cdots, FR_M\}$, the set of feature reliability;
$FR_{average}$, the average reliability of all features;

(1) $B \leftarrow F$
(2) $FR_{average} = \frac{1}{M} \sum_{i=1}^{M} FR_i$
(3) **for** $f_i \in F$
(4) **if** $FR_i < FR_{average}$
(5) $B \leftarrow B - f_i$
(6) **return** B

Figure 3: Pseudo Code of ReduceFeatureSet(F,FR)

4 Feature Selection Ensemble

In this section, the proposed implementations of the *FSE* concept are specified, with the aid of illustrative flow charts and pseudo codes. In the context of *FS*, an *information system* is a couple (X, F), where $X = \{x_1, \cdots, x_N\}$ and $F = \{f_1, \cdots, f_M\}$ are finite, non-empty sets of objects and features, respectively. Features can be either *qualitative* (discrete-valued) or *quantitative* (real-valued). Here, a feature subset $B \subseteq F$ is represented by a binary string b of length M, $b_i = 1$ if $f_i \in B$, $b_i = 0$ otherwise. An *FSE* can therefore be represented by a set of such binary strings, $E = \{b_1, \cdots, b_K\}$, where K denotes the size of the ensemble. The finally selected feature subset by the *FSE* is the outcome of aggregating the elements of E, which is denoted by b^* hereafter. The general notations used in the pseudo codes are provided in Table. 4.

Table 1: Notations Used in Pseudo Codes

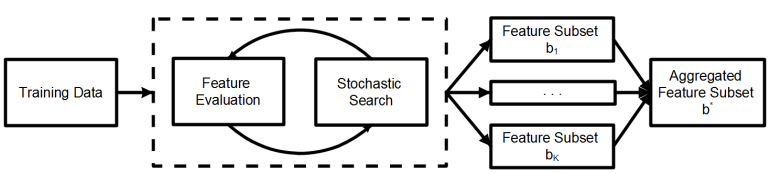
HS	the stochastic search algorithm	S	the search algorithm
$eval$	the feature selection algorithm	X	the set of training objects
b	a feature subset	b^*	ensemble output
E	the feature selection ensemble	K	the desired ensemble size
$P = \{p_1, \cdots, p_K\}$	the set of data partitions	$rand$	the pseudo random generator
$\{eval_1, \cdots, eval_Y\}$	the set of Y feature evaluators	$i = 1, \cdots, Y$	index of the feature evaluator

4.1 Ensemble Construction

4.1.1 Single Algorithm with Stochastic Search

Many of the existing nature-inspired heuristics, such as *GA*, *PSO*, and *HS*, share many commonalities, most notably the ability to generate multiple, good quality solutions. However, the search results obtained by them, even with an identical subset evaluation method, can be different. Sometimes, such differences may be rather distinct, even when the selection process is performed on the same training data. Thus, an *FSE* can be constructed.

As illustrated in Fig. 4, the stochastic algorithm searches for feature subsets until the targeted number of subsets K is satisfied. This implementation is very simple in concept, requiring only one evaluator and one search technique, therefore the effort spent in configuring the necessary components is minimal. However, for datasets with fewer features, the number of "optimal" subsets may be generally small, as compared to larger, more complex datasets. Thus, the diversity within the *FSE* may also be low. Furthermore, evaluators that rely on feature ranking are not applicable to this implementation, as stochastic search methods require the evaluation to be performed on the discovered subset as a whole, rather than selecting top most features from an ordered list.

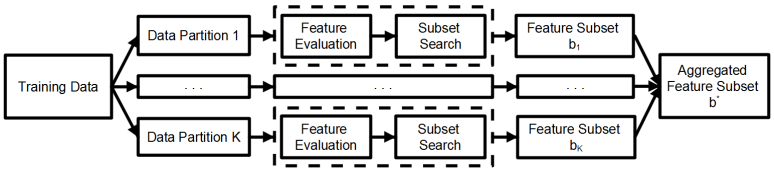


(1) **while** ($|E| < K$)
(2) $b := HS(X, eval)$
(3) $E.add(b)$
(4) $b^* = aggregate(E)$
(5) **return** b^*

Figure 4: Flow Chart and Pseudo Code for Single Algorithm with Stochastic Search

4.1.2 Single Algorithm with Partitioned Training Data

An alternative approach for creating a diverse *FSE* is to use data partitioning, where the training data is divided into a number of different chunks, and *FS* is then carried out on each individual partition. This is illustrated in Fig. 5. As the training instances employed by different *FS* algorithms are different, it



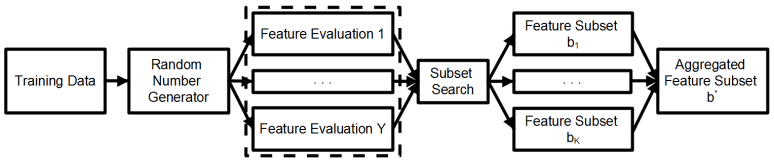
(1) $P := stratify(X, K)$
(2) **for** ($i = 1$ **to** K)
(3) $b := S(p_i, eval)$
(4) $E.add(b)$
(5) $b^* = aggregate(E)$
(6) **return** b^*

Figure 5: Flow Chart and Pseudo Code for Single Algorithm with Partitioned Training Data

is expected that various features subsets may be found by these algorithms. In order to maintain class balance, and to ensure minority classes are sufficiently represented in each data partition, techniques such as the stratified cross validation [39] may be employed. Of course, this approach to implementing *FSE* may be less effective for datasets with limited training objects, since most *FS* evaluators require a sufficient amount of data objects in order to determine the meaningful features. As a result, the number of data partitions is often restricted, which then puts constraint on the ensemble size K.

4.1.3 Mixture of Algorithms

By employing multiple *FS* algorithms, the ensemble diversity can be naturally obtained from the differences in opinions reached by the evaluators themselves. The ensemble construction process may be further randomised by the use of a pseudo random generator, as illustrated in Fig. 6, so that the available *FS* algorithms are randomly selected when forming the ensemble. This randomised approach may be



(1) **while** ($|E| < K$)
(2) $i := rand(Y)$
(3) $b := S(X, eval_i)$
(4) $E.add(b)$
(5) $b^* = aggregate(E)$
(6) **return** b^*

Figure 6: Flow Chart and Pseudo Code for Mixture of Algorithms

beneficial when the available feature selectors are fewer than the desired number of ensemble components, where certain selectors are expected to be used multiple times. Although many problems may

require such applications, due to the high diversity in the underlying *FS* components, their complexity and integration may affect the overall run-time efficiency. Also, as multiple evaluators and search algorithms are being used simultaneously, finding an optimal parameter settings for the ensemble may become challenging.

4.2 Decision Aggregation

One of the commonly used approaches for dealing with classifier ensembles is majority voting, where the most agreed class label is selected as the final ensemble prediction. Similarly, a majority voting scheme with threshold may be adopted for *FSE*. Using the notations introduced earlier, for a given ensemble E, the decisions of the ensemble components can be organised in a $K \times M$ boolean decision matrix D, where K is the size of the ensemble, and M is the total number of features. In this representation, the horizontal row D_i denotes the feature subset b_i, and the binary cell value D_{ij} indicates whether $f_j \in b_i$.

Borrowing the terminology of ensemble classifier learning, the ensemble agreement γ_j for the feature f_j can therefore be calculated by: $\gamma_j = \frac{\Sigma D_{ij}}{K}$. A agreement threshold α, $0 < \alpha \leq 1$, can then be defined to control the number of features being included in the final result b^*, such that: $b^*_j = 1, \;\; if \; \gamma_j > \alpha$. From this, the common majority (more than half) vote can be assimilated by setting $\alpha = 0.5$. The value α may be adjusted according to the problem at hand, if the amount of agreement is very high (which also indicates poor ensemble diversity), a higher α value can be used to control the size of the resultant feature subset. Alternatively, if a highly diverse *FSE* is obtained, there may exist no feature with $\gamma_j > 0.5$, to combat this, it may be necessary to employ a lowered α value.

4.3 Complexity Analysis

Preliminary complexity analysis has been performed on the ensemble construction approaches, and the aggregation method. As the ensemble procedure depends largely on the training (O_t), solution search (O_s), and evaluation (O_e) complexity of the employed feature evaluators, the overall complexity of an *FSE* is also relative to O_t, O_s, and O_e. For a given feature evaluator, using *HS* as an example, the complexity of the subset search process $O_s = O_e \times I_{\max}$ depends on O_e and the maximum number of iteration $I_{\max}$: The total complexity of training and obtaining the solution for a single feature selector is therefore $O_t + O_s$.

For ensembles constructed using a stochastic search method, the training complexity is O_t, as only a single algorithm is involved which needs training only once. The ensemble search complexity is $O_s \times K$, where K is the ensemble size. The total complexity is therefore $O_t + O_s \times K$. For data-partition based ensembles, the evaluators need to be re-trained for every data partition, resulting in a training complexity of $O_t \times K$ for these components, whilst having the same $Os \times K$ search complexity as stochastic ensembles. The total complexity is then $(O_t + O_s) \times K$. For ensembles generated from a mixture of algorithms, the training complexity is based on the number of available evaluators $\sum_{i=1}^{Y} O_{t_i}$, where Y is the number of evaluators. The search complexity is $\sum_{i=1}^{K} O_{s_i}$, where:

$$O_{s_i} = \begin{cases} O_{e_i} \times I_{\max} & \text{for subset evaluators} \\ O(N) & \text{for feature rankers} \end{cases} \tag{2}$$

and N is the number of features. The feature ranking approaches simply pick out the best features at $O(N)$ complexity, while subset evaluators need to perform a search on the solution space. The final complexity of the mixture approach is therefore $\sum_{i=1}^{Y} O_{t_i} + \sum_{i=1}^{K} O_{s_i}$. For decision aggregation, $O(N \times K)$ is required for computing ensemble agreement, while the features above threshold can be found with no extra cost.

In summary, the proposed ensemble structures are simple and efficient, imposing a worst case $O(K)$ complexity, linear to the ensemble size. This may be further improved by attempting to integrate the process of ensemble construction with the search process, so that O_s can be reduced.

5 Experimentation and Discussion

5.1 Experiment Rationale and Setup

The classification algorithms adopted in the experiments include the decision tree based *C4.5* algorithm [56], the rule based *Ripper* algorithm [56], and the vaguely quantified fuzzy-rough nearest neighbour [20], covering rather different underlying techniques. With such use of the various classifiers, a more comprehensive understanding of the resulting feature subset quality can be reached.

A number of subset evaluators are used in the experiments, including the data reliability [5] based feature selection (*DRFS*) that was introduced in section 3, the correlation-based feature subset selection (*CFS*) [16], the probabilistic consistency based feature selection (*PCFS*) [10], and the subset evaluation method based on fuzzy-rough set theory (*FRFS*) [23]. A number of feature ranking based methods are also employed in the mixture of algorithms implementation, which will be introduced in detail in its dedicated section (5.2.3).

The classification outcomes of the three proposed *FSE* implementations are compared against the averaged performance of the ensemble component feature selectors in section 5.2. The purpose is to determine whether the ensemble methods present advantages over single feature selectors in terms of classification accuracy, and subset size. The classification accuracies using the original datasets without feature selection are also included. Comparative studies between the three *FSE* implementations are further made in section 5.3, where the performance of the ensembles are averaged across different classifiers, thereby providing a higher level view of the characteristics of these approaches.

In total 12 real-valued *UCI* benchmark datasets [1] are used to demonstrate the capabilities of the approaches, a number of which are reasonably high in dimension and hence, present challenges to feature selection. A summary of the characteristics of these datasets is given in Table 5.1, and the parameter settings employed in the experiments are: memory size $= 10 - 20$, max iteration = 1000, HMCR$= 0.5 - 1$. The ensemble size is set to 10. Stratified 10-fold cross-validation (*10-FCV*) is

Table 2: Dataset Properties

Dataset	arrhythmia	cleveland	glass	heart	ionosphere	libras	olitos	ozone	secom	sonar	water3	wine
Features	280	14	10	14	35	91	26	73	591	61	39	14
Instances	452	297	214	270	230	360	120	2534	1567	208	390	178
Decisions	16	5	6	2	2	15	4	2	2	2	3	3

employed for data validation. In *10-FCV*, a given dataset is partitioned into 10 subsets. Of these 10 subsets, 9 subsets are used to perform a training fold, where feature selection algorithms are used to select the feature subsets. A single subset is retained as the testing data, so that the performance of a classifier learner is checked while using the selected feature subsets. This cross-validation process is then repeated 10 times (the number of folds). The advantage of *10-FCV* over random sub-sampling is that all objects are used for both training and testing, and each object is used for testing only once per fold. The stratification of the data prior to its division into different folds ensures that each class label has equal representation in all folds (as far as possible), thereby helping to alleviate bias/variance problems [3]. In the experiment, *10-FCV* is performed 5 times in order to lessen the impact of random factors within the heuristic algorithms, these 10×5 sets of evaluations are then aggregated to produce the final experimental outcomes.

5.2 Classification Results

5.2.1 Single Algorithm with Stochastic Search

Table 3: Classification Accuracy Result Comparison[1,2] of the Stochastic Search Implementation

FRFS

	b^* Accuracy				b_i Avg Accuracy				Full Accuracy			
Dataset	C4.5	Ripper	VQNN	b^* Size	C4.5	Ripper	VQNN	b_i Avg Size	C4.5	Ripper	VQNN	Full Size
cleveland	**53.61%**	**54.43%**	51.80%	7	53.12%	53.54%	51.95%	7	51.85%	53.89%	53.91%	14
glass	66.30%	67.77%	64.46%	6	66.30%	67.75%	64.46%	6	67.71%	64.96%	66.75%	10
heart	77.89%	74.15%	**76.89%**	**6**	77.99%	74.59%	76.50%	7	78.52%	79.26%	76.30%	14
ionosphere	**87.13%**	**87.83%**	**83.48%**	10.2	86.17%	86.50%	81.05%	9	85.65%	83.48%	83.91%	35
olitos	**62.83%**	**61.33%**	**62.33%**	6.2	60.25%	60.37%	61.55%	6	58.33%	69.17%	74.17%	26
sonar	72.77%	**74.69%**	**78.45%**	18.4	72.64%	73.04%	77.73%	16.2	72.62%	76.95%	76.50%	61
water3	78.46%	**80.72%**	79.13%	10.2	78.18%	79.51%	78.92%	9	79.74%	82.56%	82.31%	39
wine	89.15%	**89.56%**	**92.10%**	4.4	89.52%	88.41%	91.26%	4.6	93.82%	88.79%	94.38%	14

CFS

	b^* Accuracy				b_i Avg Accuracy				Full Accuracy			
Dataset	C4.5	Ripper	VQNN	b^* Size	C4.5	Ripper	VQNN	b_i Avg Size	C4.5	Ripper	VQNN	Full Size
arrhythmia	**67.44%**	70.46%	**64.48%**	**145**	66.54%	70.88%	63.18%	152.4	65.06%	70.02%	61.72%	280
cleveland	55.54%	54.89%	54.59%	6	55.67%	54.91%	54.38%	6	51.85%	53.89%	53.91%	14
glass	73.35%	67.77%	70.43%	6	73.35%	67.77%	70.43%	6	67.71%	64.96%	66.75%	10
heart	81.11%	81.85%	76.59%	6	81.04%	81.95%	76.74%	6	78.52%	79.26%	76.30%	14
ionosphere	84.78%	**87.04%**	81.57%	**12**	84.87%	86.43%	81.84%	12.8	85.65%	83.48%	83.91%	35
libras	**71.54%**	56.06%	**69.83%**	49	71.11%	56.06%	69.48%	50	70.28%	54.56%	71.11%	91
olitos	61.00%	66.17%	77.33%	13.8	60.93%	66.77%	77.72%	14	58.33%	69.17%	74.17%	26
ozone	93.34%	93.27%	93.69%	**33**	93.34%	93.25%	93.69%	35	92.62%	93.17%	93.69%	73
secom	90.50%	**92.56%**	93.36%	**273**	90.18%	92.49%	93.36%	328.4	88.96%	92.79%	93.36%	591
sonar	**74.10%**	**78.88%**	**81.76%**	**20**	73.81%	76.01%	80.11%	23.6	72.62%	76.95%	76.50%	61
water3	**83.54%**	82.15%	**86.97%**	**12**	82.88%	82.47%	85.95%	13.8	79.74%	82.56%	82.31%	39
wine	93.82%	90.42%	95.52%	8	93.82%	90.32%	95.49%	8	93.82%	88.79%	94.38%	14

PCFS

	b^* Accuracy				b_i Avg Accuracy				Full Accuracy			
Dataset	C4.5	Ripper	VQNN	b^* Size	C4.5	Ripper	VQNN	b_i Avg Size	C4.5	Ripper	VQNN	Full Size
arrhythmia	**66.86%**	70.24%	**62.82%**	**135**	66.10%	70.14%	62.06%	140	65.06%	70.02%	61.72%	280
cleveland	56.20%	54.89%	51.87%	7	56.14%	54.87%	51.89%	7	51.85%	53.89%	53.91%	14
glass	68.59%	64.34%	72.24%	6	68.44%	64.44%	72.18%	6	67.71%	64.96%	66.75%	10
heart	77.41%	80.74%	76.30%	9	77.41%	80.74%	76.29%	9	78.52%	79.26%	76.30%	14
ionosphere	**87.04%**	**85.39%**	80.61%	9.8	84.95%	85.02%	80.83%	10	85.65%	83.48%	83.91%	35
libras	67.22%	55.86%	69.11%	**33**	67.81%	55.61%	69.29%	44.6	70.28%	54.56%	71.11%	91
olitos	62.67%	66.67%	**76.33%**	8.8	63.27%	67.37%	75.42%	9	58.33%	69.17%	74.17%	26
ozone	93.06%	93.13%	93.69%	**25**	93.04%	93.03%	93.69%	28	92.62%	93.17%	93.69%	73
secom	**90.47%**	92.57%	93.36%	**285**	90.29%	92.46%	93.36%	321.8	88.96%	92.79%	93.36%	591
sonar	**74.54%**	**77.23%**	**80.80%**	**17.2**	73.24%	74.43%	79.27%	20.2	72.62%	76.95%	76.50%	61
water3	**82.97%**	81.38%	**86.26%**	**10**	82.36%	81.92%	85.02%	12	79.74%	82.56%	82.31%	39
wine	93.08%	**91.71%**	92.95%	**3.4**	93.22%	90.99%	93.05%	4	93.82%	88.79%	94.38%	14

[1] Compared against the averaged ensemble accuracy, and full dataset accuracy using various classifiers.
[2] Bold figures indicate statistically significant improvements over averaged ensemble performance.

The classification results are presented in Table 5.2.1. Paired t-test has been carried out to judge the statistical significance of the findings, the figures highlighted in bold indicate superior results in comparison to the averaged performance of single feature selectors (ensemble components). As explained previously in section 4.1.1, only the evaluators that judge the quality of a feature subset as a whole (such as *FRFS*, *CFS*, and *PCFS*) can be used in the stochastic implementation. Because the source of diversity arises from the randomised search results, a feature ranking based evaluator will always result in the same feature subset across different runs.

For the ensembles constructed using the *FRFS* evaluator, significant improvements in terms of classification accuracy are reported for all datasets except the dataset `glass`, where the accuracy stays the

same, possibly due to the low diversity. However, the ensemble aggregation also results in enlarged feature subsets, although this may be considered as a worthy sacrifice for the improvements on classification accuracy. For the ensembles constructed using *CFS*, the most significant improvement can be identified in the dataset `sonar` of 61 features. The ensemble manages to reduce the subset size, while increasing the classification performance for all tested classifiers. Similar improvements can be seen in the dataset `water3`, where both *C4.5* and *VQNN* have increased accuracy. For the ensembles constructed using *PCFS*, classification improvements are most noticeable for the datasets `ionosphere`, `sonar`, and `water3`. For dataset `wine`, the ensemble is able to reduce the averaged number of features in the subset down to 3.4, from 14, while maintaining comparable and better classification results.

For the datasets `cleveland`, `glass`, and `heart` that contain fewer features, both the *CFS* and *PCFS* ensembles result in no improvement in either classification accuracy or subset size. This agrees with the original assumption that the stochastic implementation is less suitable in dealing with such datasets. On the other hand, for the more complex datasets (most notably the `arrhythmia` dataset), the ensemble output presents higher classification accuracy and lower feature subset size.

5.2.2 Single Algorithm with Partitioned Training Data

In this experiment, the newly introduced *DRFS* evaluator is used to demonstrate unsupervised, feature ranking based *FS* performance, the previously used *CFS* and *PCFS* evaluators are also included. For each dataset, the original training data is divided into $K = 10$ partitions to produce the desired number of ensembles, subsets are then selected using the divided data.

For the ensembles constructed using *DRFS*, no major accuracy improvements are seen, expect for the datasets `olitos` and `sonar`. This may be expected because *DRFS* is an unsupervised approach that is typically used in clustering tasks and is generally difficult to compete against supervised methods [4]. Nevertheless, its view of feature importance, when tested in a supervised manner, still show reasonable, and for several datasets, competitive performance. Additionally, decrease in feature subset size is reported for the datasets `ionosphere` and `water3` when it is used.

For the ensembles constructed using *CFS* and *PCFS*, the most evident improvement is reflected by the accuracy increase of the *VQNN* classifier, in 5/12 (for *CFS*), and 7/12 (for *PCFS*) datasets. A reduction in terms of subset size can also be observed across multiple datasets, whilst the most significant reduction (of approximately 300 features) is reflected by the `secom` dataset. Note that for datasets with much less training instances, such as `olitos` with 120 objects, the partition based approach does not seem to bring forward as much benefits as it is applied to the other, larger datasets.

5.2.3 Mixture of Algorithms

For this experiment, a number of individual feature evaluators are considered, including several feature ranking approaches including *DRFS* [4], information gain, chi-squared [62], RELIEF [25], and symmetrical uncertainty [45], in conjunction with several feature subset evaluators such as *FRFS*, *CFS*, and *PCFS*. Together, a mixture of 8 different evaluation methods are employed (with their details omitted). Since the desired ensemble size is 10, a pseudo random generator as described in section 4.1.3 is used to 'create' the remaining ensemble components required.

The comparison on classification performance of the classifiers that utilise the subsets produced by the ensembles is given in Table 5.2.3. The most interesting results are achieved for the datasets `arrhythmia`, `ionosphere`, and `water3`, where all three tested classifiers have an improved performance. The overall accuracy of *C4.5* is improved for 6 out of 12 datasets, as compared to that of *VQNN* (4/12 datasets) and *Ripper* (3/12 datasets). Note that the ensembles of mixed algorithms out perform the two single algorithm based implementations in several occasions, but the size of the selected

Table 4: Classification Accuracy Result Comparison[1,2] of the Data Partition Implementation

DRFS

	b^* Accuracy				b_i Avg Accuracy				Full Accuracy			
Dataset	C4.5	Ripper	VQNN	b^* Size	C4.5	Ripper	VQNN	b_i Avg Size	C4.5	Ripper	VQNN	Full Size
cleveland	55.23%	55.66%	51.69%	8	55.38%	55.70%	51.91%	7	51.85%	50.48%	53.91%	14
glass	60.32%	61.06%	61.81%	3.8	62.47%	62.18%	61.52%	3	67.71%	69.18%	66.75%	10
heart	82.59%	80.00%	76.96%	7	82.31%	80.24%	77.36%	7	78.52%	80.37%	76.30%	14
ionosphere	85.91%	86.66%	77.65%	**11**	85.54%	86.63%	78.97%	12	85.65%	87.39%	83.91%	35
olitos	63.67%	62.87%	**64.00%**	9.8	63.52%	62.92%	63.42%	9.8	58.33%	69.17%	74.17%	26
sonar	**76.71%**	73.37%	**76.72%**	34.2	74.92%	74.23%	76.16%	33.6	72.62%	75.90%	76.50%	61
water3	80.15%	81.64%	**82.67%**	**15.2**	80.15%	80.70%	82.42%	16	79.74%	80.77%	82.31%	39
wine	84.94%	84.56%	83.01%	4.8	84.53%	85.70%	84.56%	4.8	93.82%	95.52%	94.38%	14

CFS

	b^* Accuracy				b_i Avg Accuracy				Full Accuracy			
Dataset	C4.5	Ripper	VQNN	b^* Size	C4.5	Ripper	VQNN	b_i Avg Size	C4.5	Ripper	VQNN	Full Size
arrhythmia	67.56%	70.14%	**64.60%**	**131.2**	67.52%	70.03%	64.00%	152.4	65.06%	70.02%	61.72%	280
cleveland	55.74%	55.57%	**55.19%**	6.2	56.38%	55.39%	54.56%	6	51.85%	54.88%	53.91%	14
glass	**72.22%**	**69.09%**	69.15%	6	71.27%	68.66%	69.12%	5.8	67.71%	69.22%	66.75%	10
heart	81.04%	80.59%	77.19%	6	81.10%	80.35%	77.25%	6	78.52%	78.52%	76.30%	14
ionosphere	85.65%	85.91%	81.91%	**12**	85.28%	86.14%	81.93%	13	85.65%	85.48%	83.91%	35
libras	72.00%	55.78%	**69.72%**	49	72.11%	55.56%	69.28%	49.6	70.28%	54.56%	71.11%	91
olitos	62.83%	65.00%	**79.67%**	14	62.97%	66.82%	77.33%	13.8	58.33%	68.50%	74.17%	26
ozone	**93.48%**	93.32%	93.69%	**32.6**	93.16%	93.22%	93.69%	34.8	92.62%	93.17%	93.69%	73
secom	**90.37%**	92.49%	93.36%	**293.4**	90.09%	92.56%	93.36%	326.2	88.96%	92.72%	93.36%	591
sonar	**72.78%**	**75.53%**	77.92%	**17.4**	72.81%	74.71%	78.66%	23	72.62%	76.47%	76.50%	61
water3	**83.28%**	**82.97%**	**86.92%**	**11.8**	82.53%	82.31%	85.66%	14	79.74%	82.05%	82.31%	39
wine	94.05%	91.70%	94.27%	7.8	94.13%	92.64%	95.11%	7	93.82%	93.15%	94.38%	14

PCFS

	b^* Accuracy				b_i Avg Accuracy				Full Accuracy			
Dataset	C4.5	Ripper	VQNN	b^* Size	C4.5	Ripper	VQNN	b_i Avg Size	C4.5	Ripper	VQNN	Full Size
arrhythmia	67.00%	70.33%	**63.54%**	**135.6**	66.85%	70.46%	62.05%	140.4	65.06%	70.02%	61.72%	280
cleveland	56.74%	55.55%	**53.05%**	**6.2**	56.50%	55.39%	52.52%	7	51.85%	54.62%	53.91%	14
glass	69.82%	**68.13%**	**71.70%**	6	69.36%	67.14%	71.30%	5	67.71%	66.00%	66.75%	10
heart	77.78%	79.11%	**77.11%**	9	77.94%	79.12%	76.51%	8	78.52%	80.59%	76.30%	14
ionosphere	**85.91%**	**85.57%**	80.17%	10.4	85.23%	84.67%	80.97%	10	85.65%	85.48%	83.91%	35
libras	**68.78%**	55.39%	68.91%	**31.8**	67.89%	55.55%	68.86%	34.6	70.28%	54.56%	71.11%	91
olitos	63.00%	65.67%	73.83%	**8.2**	63.50%	65.68%	73.32%	9	58.33%	69.00%	74.17%	26
ozone	93.11%	**93.45%**	93.69%	**26**	93.04%	92.99%	93.69%	27.8	92.62%	93.17%	93.69%	73
secom	**90.32%**	92.55%	93.36%	**285.4**	90.02%	92.46%	93.36%	321	88.96%	92.72%	93.36%	591
sonar	72.47%	**75.30%**	**86.26%**	**17.8**	72.83%	74.44%	85.20%	20	72.62%	77.13%	76.50%	61
water3	**82.31%**	82.72%	**86.26%**	**10.2**	81.83%	82.43%	85.02%	12	79.74%	81.18%	82.31%	39
wine	93.65%	90.94%	**94.05%**	**3.4**	93.49%	91.14%	93.83%	4	93.82%	93.40%	94.38%	14

[1] Compared against the averaged ensemble accuracy, and full dataset accuracy using various classifiers.
[2] Bold figures indicate statistically significant improvements over averaged ensemble performance.

subsets are also larger. This may be indicative that better quality feature subsets are selected by the ensemble approach.

5.3 Comparison Between the Three Implementations

A graphical view of the classification results is shown in Fig. 7, detailing the average performance and spread of the three *FSE* implementations, against the classification models built using the original, full feature datasets. Note that the graphs do not represent a single distribution, but the results obtained over different (base) classifiers that are used for each of the three types of implementation and also, over multiple ensemble subsets that are provided by the *CFS* and *PCFS* evaluators for the stochastic and partition based implementations. In Table 6, a more detailed comparison has been given in terms of the average performance of the reported approaches. The use of averaged results is in order to give a fair

Table 5: Classification Accuracy Result Comparison[1,2] of the Mixture of Algorithms Implementation

	b^* Accuracy				b_i Avg Accuracy				Full Accuracy			
Dataset	C4.5	Ripper	VQNN	b^* Size	C4.5	Ripper	VQNN	b_i Avg Size	C4.5	Ripper	VQNN	Full Size
arrhythmia	**68.64%**	**70.46%**	**64.78%**	**132.4**	68.01%	69.98%	63.61%	142.4	65.06%	70.28%	61.72%	280
cleveland	56.75%	55.42%	52.16%	7.8	56.27%	55.12%	52.60%	7	51.85%	53.89%	53.91%	14
glass	**70.32%**	63.65%	65.87%	6	69.58%	65.10%	67.54%	6	67.71%	64.96%	66.75%	10
heart	80.22%	78.74%	76.59%	8	80.40%	79.08%	76.73%	7.4	78.52%	79.26%	76.30%	14
ionosphere	**86.26%**	**86.52%**	**84.17%**	**14.6**	85.68%	85.65%	82.77%	15.6	85.65%	83.48%	83.91%	35
libras	68.48%	53.56%	64.33%	44.4	68.44%	53.54%	65.31%	44.2	70.28%	54.56%	71.11%	91
olitos	62.67%	67.33%	75.33%	13.2	63.00%	68.90%	76.37%	13	58.33%	69.17%	74.17%	26
ozone	**92.74%**	93.01%	93.69%	34.8	92.82%	93.00%	93.69%	35	92.62%	93.05%	93.69%	73
secom	89.85%	92.48%	93.36%	**282**	90.02%	92.49%	93.36%	306.2	88.96%	92.79%	93.36%	591
sonar	76.11%	77.03%	**82.52%**	**26.2**	75.95%	77.31%	80.42%	27.6	72.62%	76.95%	76.50%	61
water3	**83.49%**	**82.67%**	**85.79%**	17.2	82.86%	82.13%	85.30%	17.2	79.74%	82.56%	82.31%	39
wine	**94.93%**	94.16%	94.46%	7.6	94.23%	93.44%	94.46%	7.2	93.82%	88.79%	94.38%	14

[1] Compared against the averaged ensemble accuracy, and full dataset accuracy using various classifiers.
[2] Bold figures indicate statistically significant improvements over averaged ensemble performance.

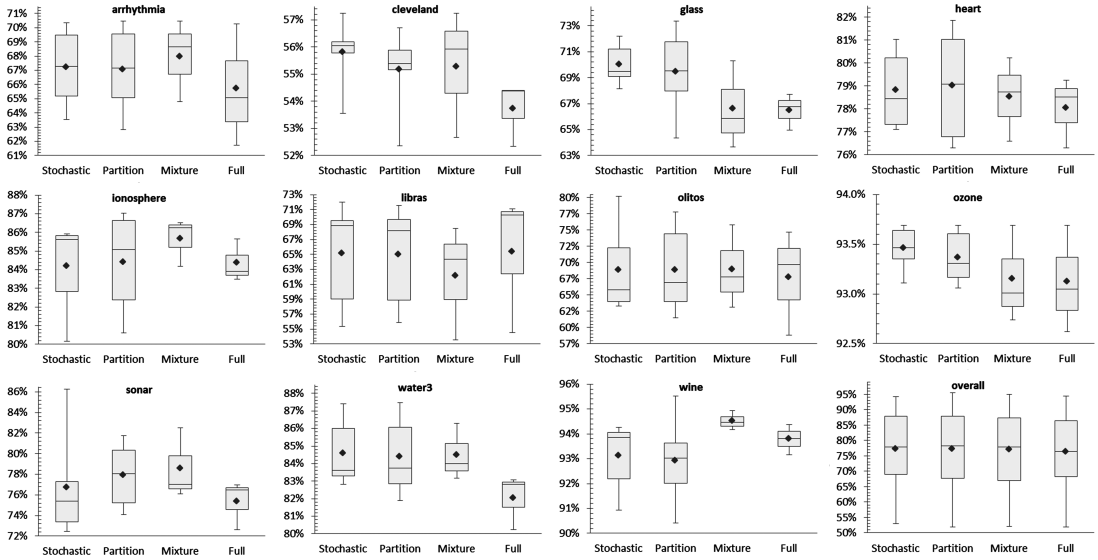


Figure 7: Comparison of average classification accuracies (solid dots) and spreads of the three *FSE* implementations for each dataset

comparison of the performance differences between the various implementations.

From these figures, it can be observed that the data partition based approach generally have a larger spread than the stochastic approach, other than a few exceptions where the stochastic implementation scores a very high maximum accuracy, such as `olitos` and `sonar`. It can be seen from this table that the stochastic search implementation leads in terms of overall classification accuracy, achieving best scores in 6/12 cases; whilst the mixture of algorithms implementation obtains best performance in 5/12 cases. The data-partition based implementation scores the highest only for the `heart` dataset. However, its accuracy is still very competitive for the other datasets, with an overall accuracy of 77.33% and a mere 0.09% difference from the average score of the stochastic search. One conclusion that may be drawn from these results is that the mixture of algorithms implementation appears to work best on datasets with the least number of training objects, such as datasets `olitos` (120), `wine` (178), `sonar` (208).

Table 6: Comparison[1] of Averaged[2] Classification Results of *FSE* implementations

	Stochastic		Partition		Mixture		Full			
Dataset	Acc	Size	Acc	Size	Acc	Size	Acc	Size	Instances	Better with *FSE*
arrhythmia	67.20%	133.4	67.05%	140	**67.96%**	**132.4**	65.60%	280	452	✓
cleveland	**55.31%**	**6.2**	54.66%	6.5	54.78%	7.8	53.46%	14	297	✓
glass	**70.02%**	**6**	69.45%	**6**	66.61%	**6**	66.82%	10	214	✓
heart	78.80%	**7.5**	**79.00%**	**7.5**	78.52%	8	78.47%	14	270	✓
ionosphere	84.19%	11.2	84.41%	**10.9**	**85.65%**	14.6	85.01%	35	230	✓
libras	**65.10%**	**40.4**	64.94%	41	62.12%	44.4	65.32%	91	360	
olitos	68.33%	**11.1**	68.36%	11.3	**68.44%**	13.2	67.17%	26	120	✓
ozone	**93.46%**	29.3	93.36%	**29**	93.15%	34.8	93.16%	73	2534	✓
secom	**92.72%**	**279**	92.08%	289.4	91.90%	282	91.68%	591	1567	✓
sonar	76.71%	**17.6**	77.89%	18.6	**78.55%**	26.2	75.42%	61	208	✓
water3	**84.08%**	**11**	83.88%	**11**	83.98%	17.2	81.08%	39	390	✓
wine	93.11%	**5.6**	92.92%	5.7	**94.52%**	7.6	93.87%	14	178	✓
overall	**77.42%**	46.52	77.33%	48.08	77.18%	49.52	76.42%	104.00	-	✓

[1] Bold figures indicate superior performance, ticked rows indicate the ensembles out perform accuracy obtained using full features.
[2] Averaged across multiple subset evaluators and all classifiers.

Further analysis into the implementations' detailed characteristics remains active research, in the hope that more behaviour patterns can be discovered in order to optimise the ensemble structure.

In terms of the size of a selected feature subset, the stochastic search implementation clearly shows to be the best, leading in 9/12 datasets (including tied cases), whilst the mixture of algorithms results in largest subsets overall. Note that for all the datasets tested, except `libras`, the use of *FSE* leads to the improvement on the classifiers accuracy, while the number of features required to perform the classification is also much reduced. This reflects that as a novel filter-based approach, *FSE* offers a beneficial pre-processing step for the purpose of classification.

6 Conclusion

This paper has introduced three distinctive techniques in an effort to implement feature selection ensemble (*FSE*), where the outcomes from multiple, different feature selection results are integrated together, for the purpose of producing an aggregated feature subset that helps to perform the subsequent classification tasks. The key advantage of *FSE* is that the performance of the feature selection procedure is no longer depended upon one selected subset, making this technique potentially more flexible and robust in dealing with high dimensional and large datasets. For such datasets, multiple feature subsets with equally highest attained scores may be discovered when judged by one single feature evaluator, but not all may perform equally well in terms of classification. Two of the proposed implementations, the stochastic search based and the data partition based, require the use of a single subset evaluation algorithm; whilst the mixture of algorithms approach aims to produce the ensemble from distinctive component feature selection methods.

Experimental comparative studies demonstrate that *FSE* significantly improves over single *FS* results. Indeed, all three implementations show strength in dealing with almost all datasets tested, generally resulting in an increase in classification accuracy, when compared against the classification models built using the original, full feature datasets or feature subsets returned by component selectors. In particular, the stochastic search based approach appears to perform better than the rest, which may have benefited from the quality search results ensured by *HS*. In depth analysis of the experimental findings, as well as the employment of higher dimensional, larger sized datasets are necessary to better reveal the characteristics of the proposed implementations.

Although promising, much can be done to further improve the potential of the present work. For

example, currently, the size of ensemble needs to be predefined, instead of being self-adaptive. The ensemble should be able to "recruit" or "fire" feature selectors according to the complexity of the data. An extreme case for this would be the situation where the dataset contains only one optimal feature subset. Such a case can be easily handled by a single evaluator, thereby eliminating the necessity of using *FSE* (equivalently, shrinking the ensemble size to one). Additionally, it would be useful to investigate the combination of different ensemble implementations, realising ensemble of ensembles, where certain components may also be dynamically modified during the feature selection process. Furthermore, *FSE* shares many similarities with classifier ensembles [12], such as the importance of ensemble diversity [6, 49] and decision aggregation [48]. Methods developed for classifier ensembles may also be adopted to handle *FSE* problems.

Last, but not least, it would be very important to examine how *FSE* may be applied to support tasks other than classification, such as intelligent robotics and systems control. Of particular interest to the authors is the potential application of *FSE* to the development of biped robots. The significant advantage of biped robots is that they allow locomotion in natural terrain inaccessible to conventional vehicles. Although the stable control of biped robots is much more challenging than that of multi-legged robots, they have specific merits compared with the latter. For example, they can operate in human environments more efficiently than other legged robots. Their particular footprint and aspect ratio means they can also help or replace humans, even in difficult or dangerous tasks. The ability of a humanoid robot carrying out a certain action with her hands while moving is of significant impact in almost all aspects of life, be they engineering, medical, educational or social (– imagine a robot carrying an Olympic torch while running).

There are many problems that have to be overcome before biped robots can be deployed in a natural environment, however. For instance, simultaneous mapping and localisation has been recognised to be a very important task for building such robots. Apart from the direct use of raw data or simple features as geometric representations, recent techniques have tried to utilise different representations that capture more context information, permitting an additional cognitive and reasoning mapping. Also, to help vision-based robot positioning [55] and activity recognition [51] in the working environment, rich and often non-independent features are necessary to be initially computed from sensory data, without prior knowledge of which features would be critical to the problem at hand. This means that a large number of features may result though not all are essential [26, 36]. Besides, the large amount of features generated puts high computational demands on the robot control process [28]. Feature selection techniques can be applied to address all these issues, pruning down the redundant, unessential features [22]. Thus, it is of natural appeal to apply *FSE* to aiding in the development of biped robots.

As the concluding remark, it is interesting to note that the representative of the authors of this paper is very much honoured to have been selected to carry the Olympic torch in memory of *Alan Turing*, for the 2012 London Olympic torch relay. May future humanoid robots be able to participate in Olympic torch relays, carrying the Olympic flame in celebration of *Alan Turing's* life and scientific impact!

References

[1] A. Asuncion and D. J. Newman. UCI machine learning repository, 2007.

[2] Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

[3] Yoshua Bengio and Yves Grandvalet. No Unbiased Estimator of the Variance of K-Fold Cross-Validation. *Journal of Machine Learning Research*, 5:1089–1105, September 2004.

[4] Tossapon Boongoen, Changjing Shang, Natthakan Iam-on, and Qiang Shen. Extending data reliability measure to a filter approach for soft subspace clustering. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(6):1705–1714, 2011.

[5] Tossapon Boongoen and Qiang Shen. Nearest-neighbor guided evaluation of data reliability and its applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(6):1622–1633, December 2010.

[6] Pádraig Cunningham and John Carney. Diversity versus quality in classification ensembles based on feature selection. In *In 11th European Conference on Machine Learning*, pages 109–116. Springer, 2000.

[7] Swagatam Das, Arpan Mukhopadhyay, Anwit Roy, Ajith Abraham, and Bijaya K. Panigrahi. Exploratory Power of the Harmony Search Algorithm: Analysis and Improvements for Global Numerical Optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(1):89–106, February 2011.

[8] Manoranjan Dash, Kiseok Choi, Peter Scheuermann, and Huan Liu. Feature selection for clustering - a filter solution. In *Proceedings of the Second International Conference on Data Mining*, pages 115–122, 2002.

[9] Manoranjan Dash and Huan Liu. Feature selection for classification. *Intelligent Data Analysis*, 1:131–156, 1997.

[10] Manoranjan Dash and Huan Liu. Consistency-based search in feature selection. *Artif. Intell.*, 151(1-2):155–176, 2003.

[11] Ren Diao and Qiang Shen. Feature selection with harmony search. *Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*. (to appear).

[12] Ren Diao and Qiang Shen. Fuzzy-rough classifier ensemble selection. In *Proceedings of IEEE International Conference on Fuzzy Systems*, pages 1516–1522, 2011.

[13] M. Fesanghary, M. Mahdavi, M. Minary-Jolandan, and Y. Alizadeh. Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems. *Computer Methods in Applied Mechanics and Engineering*, 197(33-40):3080–3091, 2008.

[14] Zong Woo Geem, editor. *Recent Advances In Harmony Search Algorithm*, volume 270 of *Studies in Computational Intelligence*. Springer, 2010.

[15] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, March 2003.

[16] M. A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1998.

[17] Julia Handl and Joshua Knowles. Feature Subset Selection in Unsupervised Learning via Multiobjective Optimization.

[18] Yi Hong, Sam Kwong, Yuchou Chang, and Qingsheng Ren. Consensus unsupervised feature ranking from multiple views. *Pattern Recognition Letters*, 29(5):595–602, 2008.

[19] Chun-Nan Hsu, Hung-Ju Huang, and Dietrich Schuschel. The ANNIGMA-wrapper approach to fast feature selection for neural nets. *Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 32(2):207–212, 2002.

[20] Richard Jensen and Chris Cornelis. Fuzzy-rough nearest neighbour classification. In *Transactions on Rough Sets XIII*, volume 6499 of *Lecture Notes in Computer Science*, pages 56–72. Springer Berlin / Heidelberg, 2011.

[21] Richard Jensen and Qiang Shen. *Computational Intelligence and Feature Selection: Rough and Fuzzy Approaches*. Wiley-IEEE Press, 2008.

[22] Richard Jensen and Qiang Shen. Are more features better? a response to attributes reduction using fuzzy rough sets. *IEEE T. Fuzzy Systems*, 17(6):1456–1458, 2009.

[23] Richard Jensen and Qiang Shen. New approaches to fuzzy-rough feature selection. *Trans. Fuz Sys.*, 17(4):824–838, August 2009.

[24] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1):273–324, 1997.

[25] Igor Kononenko, Edvard Simec, and Marko Robnik-Sikonja. Overcoming the myopia of inductive learning algorithms with relieff. *Applied Intelligence*, 7:39–55, 1997.

[26] B.J.A Kröse, N. Vlassis, R. Bunschoten, and Y. Motomura. A probabilistic model for appearance-based robot localization. In *In First European Symposium on Ambience Intelligence (EUSAI)*, pages 264–274. Springer,

2000.

[27] Kang S. Lee and Zong W. Geem. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering*, 194(36-38):3902–3933, September 2005.

[28] X. Li and L.E. Parker. Design and performance improvements for fault detection in tightly-coupled multi-robot team tasks. In *Proceedings of IEEE International Conference on Robotics and Automation*, 2009.

[29] Huan Liu and Hiroshi Motoda. *Computational Methods of Feature Selection (Chapman & Hall/Crc Data Mining and Knowledge Discovery Series)*. Chapman & Hall/CRC, 2007.

[30] Neil Mac Parthaláin and Richard Jensen. Measures for unsupervised fuzzy-rough feature selection. volume 7, pages 249–259, Amsterdam, The Netherlands, The Netherlands, December 2010. IOS Press.

[31] Neil Mac Parthaláin and Richard Jensen. Fuzzy-rough set based semi-supervised learning. In *IEEE International Conference on Fuzzy Systems*, pages 2465–2472, 2011.

[32] Neil Mac Parthaláin, Richard Jensen, Qiang Shen, and Reyer Zwiggelaar. Fuzzy-rough approaches for mammographic risk analysis. *Intell. Data Anal.*, 14(2):225–244, April 2010.

[33] Neil Mac Parthaláin, Qiang Shen, and Richard Jensen. A distance measure approach to exploring the rough set boundary region for attribute reduction. *Trans. on Knowl. and Data Eng.*, 22(3):305–317, March 2010.

[34] M. Mahdavi, M. Fesanghary, and E. Damangir. An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation*, 188(2):1567–1579, May 2007.

[35] Mehrdad Mahdavi, Morteza Haghir Chehreghani, Hassan Abolhassani, and Rana Forsati. Novel meta-heuristic algorithms for clustering web documents. *Applied Mathematics and Computation*, 201(1-2):441–451, 2008.

[36] A. Marán-Hernández, R. Méndez-Rodríguez, and F.M. Montes-González. Significant feature selection in range scan data for geometrical mobile robot mapping. In *Proceedings of the 5th International Symposium on Robotics and Automation*, 2006.

[37] M. Hadi Mashinchi, Mehmet A. Orgun, M. Mashinchi, and Witold Pedrycz. A tabu-harmony search-based approach to fuzzy linear regression. *IEEE T. Fuzzy Systems*, 19(3):432–448, 2011.

[38] Hahn ming Lee, Chih ming Chen, Jyh ming Chen, and Yu lu Jou. An efficient fuzzy classifier with feature selection based on fuzzy entropy. *Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 31:426–432, 2001.

[39] T. Mitchell. *Machine Learning (Mcgraw-Hill International Edit)*. McGraw-Hill Education (ISE Editions), 1st edition, October 1997.

[40] D. P. Muni, N. R. Pal, and J. Das. Genetic programming for simultaneous feature selection and classifier design. 36(1):106–117, 2006.

[41] J. Scott Olsson. Combining feature selectors for text classification. In *Proc. the 15th ACM international conference on Information and knowledge management*, pages 798–799, 2006.

[42] David W. Opitz. Feature selection for ensembles. In *Proceedings of 16th National Conference on Artificial Intelligence (AAAI*, pages 379–384. Press, 1999.

[43] Alon Orlitsky, Narayana P. Santhanam, and Junan Zhang. Always good turing: Asymptotically optimal probability estimation. *Science*, 302(5644):427–431, 2003.

[44] Jose Manuel Peña, Jose Antonio Lozano, Pedro Larrañaga, and Iñaki Inza. Dimensionality reduction in unsupervised learning of conditional gaussian networks. *Trans. Pattern Anal. Mach. Intell.*, 23(6):590–603, June 2001.

[45] S. Senthamarai Kannan and N. Ramaraj. A novel hybrid feature selection via symmetrical uncertainty ranking based local memetic search algorithm. *Know.-Based Syst.*, 23(6):580–585, August 2010.

[46] Changjing Shang, Dave Barnes, and Qiang Shen. Facilitating efficient mars terrain image classification with fuzzy-rough feature selection. *Int. J. Hybrid Intell. Syst.*, 8(1):3–13, January 2011.

[47] Qiang Shen and Richard Jensen. Selecting informative features with fuzzy-rough sets and its application for complex systems monitoring. *Pattern Recognition*, 37(7):1351–1363, 2004.

[48] Vicenç Torra and Yasuo Narukawa. *Modeling Decisions: Information Fusion and Aggregation Operators*.

Springer, 2007.

[49] Alexey Tsymbal, Mykola Pechenizkiy, and Pádraig Cunningham. Diversity in search strategies for ensemble feature selection. *Information Fusion*, 6(1):83–98, 2005.

[50] Eugene Tuv, Alexander Borisov, George Runger, and Kari Torkkola. Feature Selection with Ensembles, Artificial Variables, and Redundancy Elimination. *J. Mach. Learn. Res.*, 10:1341–1366, December 2009.

[51] Douglas L. Vail and Manuela M. Veloso. Feature selection for activity recognition in multi-robot domains. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 3*, pages 1415–1420, 2008.

[52] A. Vasebi, M. Fesanghary, and S. M. T. Bathaee. Combined heat and power economic dispatch by harmony search algorithm. *International Journal of Electrical Power & Energy Systems*, 29(10):713–719, December 2007.

[53] Huanjing Wang, Taghi M. Khoshgoftaar, and Amri Napolitano. A comparative study of ensemble feature selection techniques for software defect prediction. In *Proceedings of the 2010 Ninth International Conference on Machine Learning and Applications*, pages 135–140, 2010.

[54] Xiangyang Wang, Jie Yang, Xiaolong Teng, Weijun Xia, and Richard Jensen. Feature selection based on rough sets and particle swarm optimization. *Pattern Recogn. Lett.*, 28(4):459–471, March 2007.

[55] Gordon Wells and Carme Torras. Assessing image features for visionbased robot positioning. *Journal of Intelligent and Robotic Systems*, pages 95–118, 2001.

[56] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[57] Jakub Wróblewski. Ensembles of classifiers based on approximate reducts. *Fundam. Inf.*, 47(3-4):351–360, October 2001.

[58] Eric P. Xing, Michael I. Jordan, and Richard M. Karp. Feature selection for high-dimensional genomic microarray data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 601–608. Morgan Kaufmann, 2001.

[59] Zeshui Xu. An overview of methods for determining owa weights: Research articles. *Int. J. Intell. Syst.*, 20(8):843–865, August 2005.

[60] Zeshui Xu. Dependent owa operators. In *Proceedings of the Third international conference on Modeling Decisions for Artificial Intelligence*, MDAI'06, pages 172–178, Berlin, Heidelberg, 2006. Springer-Verlag.

[61] Jidong Zhao, Ke Lu, and Xiaofei He. Locality sensitive semi-supervised feature selection. *Neurocomput.*, 71(10-12):1842–1849, June 2008.

[62] Zhaohui Zheng, Xiaoyun Wu, and Rohini Srihari. Feature selection for text categorization on imbalanced data. *SIGKDD Explor. Newsl.*, 6(1):80–89, June 2004.

[63] Zexuan Zhu, Yew-Soon Ong, and Manoranjan Dash. Wrapper-filter feature selection algorithm using a memetic framework. *Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(1):70–76, 2007.