International Journal of Computational Intelligence and Applications

A Comprehensive Comparison on Evolutionary Feature Selection Approaches to Classification

Bing Xue*, Mengjie Zhang, and Will N. Browne
School of Engineering and Computer Science, Victoria University of Wellington, PO Box 600, Wellington 6140, New Zealand

Feature selection is an important data preprocessing step in machine learning and data mining, such as classification tasks. Research on feature selection has been extensively conducted for more than fifty years and different types of approaches have been proposed, which include wrapper approaches or filter approaches, and single objective approaches or multi-objective approaches. However, the advantages and disadvantages of such approaches have not been thoroughly investigated. This paper provides a comprehensive study on comparing different types of feature selection approaches, specifically including comparisons on the classification performance and computational time of wrappers and filters, generality of wrapper approaches, and comparisons on single objective and multi-objective approaches. Particle swarm optimisation based approaches, which include different types of methods, are used as typical examples to conduct this research. A total of 10 different feature selection methods and over 7000 experiments are involved. The results show that filters are usually faster than wrappers, but wrappers using a simple classification algorithm can be faster than filters. Wrapper often achieve better classification performance than filters. Feature subsets obtained from wrappers can be general to other classification algorithms. Meanwhile, multi-objective approaches are generally better choices than single objective algorithms. The findings are not only useful for researchers to develop new approaches to addressing new challenges in feature selection, but also useful for real-world decision makers to choose a specific feature selection method according to their own requirements.

Keywords: Feature Selection; Wrapper and Filer Approaches; Single Objective and Multi-objective Optimisation; Particle Swarm Optimisation; Classification;

1. Introduction

Classification tasks often have a large number of features, which include not only relevant features, but also redundant and irrelevant features. They increase the difficulty of classification as the size of the search spaces grows, known as "the curse of dimensionality" [1]. The redundant or irrelevant features may also deteriorate the classification performance [2]. Feature selection is proposed to select a small subset of relevant features to reduce the dimensionality of the data and maintain or increase the classification performance [1,2].

*Corresponding author. Email: Bing.Xue@ecs.vuw.ac.nz

Feature selection has been widely investigated in the data mining and machine learning communities. A large number of feature selection algorithms have been proposed [1,2]. A feature selection approach typically needs a search technique to find the optimal feature subsets and an evaluation criterion to assess the quality of the obtained feature subsets to guide the search. Based on the evaluation criterion, feature selection algorithms can be generally grouped into two categories: wrapper and filter approaches [3]. Feature selection has two main conflicting objectives, which are to maximise the classification accuracy and minimise the number of selected features. Both single objective approaches and multi-objective approaches have been proposed. Therefore, there are four main types of feature selection approaches, which are single objective wrapper methods, multi-objective wrapper methods, single objective filter methods, and multi-objective filter methods.

A major challenge in feature selection is the large search space, where the total number of possible solutions is 2^n for a dataset containing n features. Evolutionary computation (EC) techniques have recently gained much attention to address feature selection problems due to their powerful search ability. To our best knowledge, all the existing multi-objective feature selection approaches are based on EC techniques because their population based search mechanism is particularly suitable for multi-objective optimisation. Particle swarm optimisation (PSO) [4,5] is one of the most popular EC techniques and is considered the most popular EC technique in addressing feature selection problems in the recent five years [6,7,8,9,10]. PSO has been used to propose all the four main types of feature selection approaches, i.e. single objective wrapper methods [11,12,6,7,8], multi-objective wrapper methods [13,14,15], single objective filter methods [16,17,18,19], and multi-objective filter methods [9,10,20].

There are a large number of different feature selection algorithms, but comparisons across different types of approaches have not been investigated. There are still important open issues in feature selection, which can be seen as follows:

- Wrapper and filter approaches are argued to have their own advantages and disadvantages [21]. Wrappers can achieve better classification performance than filters, but filters are computationally less expensive and more general than wrappers. However, no thorough investigations have been made on how much difference there is between the two approaches in terms of the classification performance and the computational cost.
- Wrapper approaches employ a learning/classification algorithm during the feature selection process. Therefore, they are argued to be less general than filter approaches, i.e. the features selected by a wrapper algorithm can not achieve good performance when used with other classification algorithms [21]. However, no serious investigation has been conducted to test this statement.
- For single objective and multi-objective methods, some direct comparisons on the final results have been made in the literature [9,10], but observations and analyses through the search process have not been investigated.

The overall goal of this paper is to compare and contrast the four types of

feature selection algorithms to provide suggestions for future researchers and real-world users. There will be *three specific objectives*, each of which aims to address one of the *three open issues* listed above. PSO-based approaches are used as examples because of the extensive study in PSO for feature selection. This paper uses a total of 10 feature selection methods and over 7000 runs of experiments to thoroughly investigate these objectives. Note that the contribution of this work is not to develop a new PSO-based approach for feature selection, but to provide a comprehensive comparison and analysis across different types of feature selection algorithms and PSO-based approaches are used as a typical example.

2. Different Feature Selection Approaches

To investigate the above three objectives, 10 feature selection algorithms are used in the empirical study, which are briefly described in this section.

2.1. *Wrapper Approaches*

To investigate the first two objectives, four wrapper based single objective algorithms were used in the experiments. Multi-objective algorithms were not chosen since they produced multiple solutions in each run and hard to do comparisons. Four most commonly used classification algorithms in wrapper feature selection are chosen as representative examples in the four wrapper approaches. The four classification algorithms are support vector machine (SVM), decision tree (DT), K-nearest neighbours (KNN) with K=5, and Naïve Bayes (NB), and the four corresponding wrapper methods are *W-SVM, W-KNN, W-DT,* and *W-NB*.

In W-SVM, W-KNN, W-DT and W-NB, PSO was used to search for the optimal feature subset (s). The dimensionality of the search space equals to n, i.e. the total number of features in the dataset. Each particle represents a feature subset and its position is typically specified by an $n$dimensional vector, where each dimension corresponds to one feature in the dataset. Each element of the position vector is a real valued number between [0, 1], which represents a confidence level that the corresponding feature is selected. The feature subset is typically constructed by selecting each feature if the confidence level is above a predefined threshold. The fitness functions are to maximise the classification accuracy of the selected feature subsets.

To investigate the third objective, i.e. single objective versus multi-objective algorithms, two wrapper approaches are used as examples since the change of the classification performance is easier to observe in wrapper approaches than filter approaches during the search process. The representative single objective algorithm is *PSOIniPG* [22] and the representative multi-objective algorithm is *CMDPSOFS* [14]. PSOIniPG is chosen because PSOIniPG simultaneously handled the two objectives during the feature selection process and achieved better performance than the commonly used way of aggregating two objectives into a single objective function

[22]. The representation of each particle in PSOIniPG is the same as in W-SVM, W-KNN, W-DT and W-NB. In PSOIniPG [22], an initialisation strategy and a *pbest* and *gbest* updating mechanism were proposed in PSO to improve its performance on feature selection [22]. The initialisation strategy was inspired by two classical feature selection selection methods, forward selection and backward selection, to utilise their advantages and avoid their disadvantages. The *pbest* and *gbest* were updated not only according to the classification performance of the feature subsets, but also considered the number of features. CMDPSOFS [14] is the best-so-far multi-objective wrapper algorithm, where the two objectives are to maximise the classification accuracy and minimise the number of features. CMDPSOFS employed a multi-objective search mechanism in PSO and the representation of each particle in CMDPSOFS is the same as in PSOIniPG. CMDPSOFS employed two mutation operators, a crowding distance measure, and a dominance measure to improve the performance of multi-objective PSO for feature selection. More details can be found in[14].

2.2. *Filter Approaches*

To investigate the first objective, i.e. comparing wrapper algorithms with filter algorithms, four filter algorithms are used in this paper. They are based on two of the most popular theories used in feature selection, which are information theory and rough set theory. The four filter algorithms are *F-MI* [18] and *F-E* [18] based on information theory, and *F-RS* [19] and *F-PRS* [19] based on rough set theory. F-MI, F-E, F-RS and F-PRS used PSO as the search technique and the representation of each particle is the same as in the above wrapper methods. The fitness functions are described as follows.

2.2.1. *F-MI and F-E*

Both F-MI and F-E are based on information theory, where F-MI uses a mutual information based fitness function and F-E uses a entropy based fitness function. Entropy and mutual information in information theory are able to measure the information of random variables [23]. Due to page limit, only closely relevant equations are presented here and more details can be found in [18].

F-MI: The fitness function of F-MI is shown by Equation (1).

$$Fit_{MI} = \sum_{F_i \in S} I(F; c) - \sum_{F_i, F_j \in S} I(F_i; F_j) \tag{1}$$

where $\sum_{F_i \in S} I(F_i; C)$ represents the relevance between the selected feature subset S and the class label C, while $\sum_{F_i, F_j \in S} I(F_i, F_j)$ represents the redundancy within the feature subset S, where $I(F_i, F_j)$ measures the mutual information between two features, F_i and F_j, in S. Therefore, Equation (1) is a maximisation fitness function, which aims maximise the relevance and minimise the redundancy of the selected feature subset.

F-E: The fitness function of F-E is shown by Equation (2).

$$Fit_E = IG(C|S) - \frac{1}{|S|}\sum_{F_i \in S} IG(F|\{S/F_i\} \tag{2}$$

where $IG(C|S)$ evaluates the information gain of the class label C given information of the features in S, which shows the relevance between S and C. $\frac{1}{|S|}\sum_{F_i \in S} IG(F_i|\{S/F_i\}$ evaluates the redundancy contained in S by summing up the information gained for each $F_i \in S$ by giving S/F_i, where S/F_i means all the features in S except for feature F_i. Both relevance and redundancy involve calculating the information gain of a single feature given a set of features. Taking $IG(C|S)$ as an example,

$$\begin{aligned} IG(C|S) &= H(C) - H(C|S) \\ &= H(C) - (H(C \cup S) - H(S)) \\ &= H(C) + H(S) - H(C \cup S) \end{aligned}$$

where $H(C)$ means the entropy of the class label C and $H(S)$ is the joint entropy of all the features in S. If $S = \{F_1, F_2, F_3\}$ (F_1, F_2, and F_3 are the selected features), then

$$H(F_1, F_2, F_3) = -\sum_{f_1 \in F_1}\sum_{f_2 \in F_2}\sum_{f_3 \in F_3} p(f_1 f_2 f_3)\log_2 p(f_1 f_2 f_3).$$

where $p(f_1 f_2 f_3)$ shows the probability density function of F_1, F_2, and F_3.

2.2.2. *F-RS and F-PRS*

The fitness functions of F-RS and F-PRS are based on standard rough set theory and probabilistic rough set theory, respectively. In rough set theory, an information system can be denoted as $T = (U, A)$, where U is the universe of objects (i.e. instances) in the system and A is the set of attributes (i.e. features) that describe each object. Equivalence relation is a relation that partitions a set so that every element of the set is a member of one and only one cell of the partition. Based on all equivalence relations described by A, the equivalence class relation partitions of U is $U_1, U_2, U_3, ..., U_c$, where c is the number of classes that objects in U may belong to.

For any $S \subseteq A$ and $X \subseteq U$, the equivalence relation is defined as $IND(S) = \{(x, y) \in \mathbb{U}^2 | \forall a \in S, a(x) = a(y)\}$ an the equivalence classes of $IND(S)$ are denoted $[x]_S$, this means that $\forall y \in [x]_S$ (x, y) are indiscernible with regards to S. Based on equivalence class, rough set theory defines a lower approximation and an upper approximation of X based on the equivalent classes described by S [24]. The lower approximation $\underline{S}X$ is defined as $\underline{S}X = \{x \in U | [x]_S \subseteq X\}$ while the upper approximation $\overline{S}X$ is defined as $\overline{S}X = \{x \in U | [x]_S \cap X \neq \emptyset\}$. $\underline{S}X$ contains all the objects, which can be surely classified to the target set X. $\overline{S}X$ contains the objects, which probably belong to the target set X.

F-RS: When using rough set theory for feature selection, the datasets for a classification problem can be treated as an information system $T = (U, A)$, where all available attributes (features) can be considered as A in the rough set theory. Based on the equivalence described by A, U can be partitioned to $U_1, U_2, U_3, ..., U_c$, where c is the number of classes in the dataset. After feature selection, the obtained reduct (i.e. feature subset) can be considered as $S \in A$. Therefore, the fitness of S can be evaluated by how well S represents each target set in U, which is a class in the dataset. For $U_1 \in U$, let $\underline{S}U_1 = \{x \in U | [x]_S \subseteq U_1\}$ be the lower approximation of S according to U_1 if $[x]_S$ only contains instances in U_1. Let $\overline{S}U_1 = \{x \in U | [x]_S \cap U_1 \neq \emptyset\}$ be the upper approximation of S according to U_1 if $[x]_S$ contains at least one element not in U_1. Therefore, the purity of $[x]_S$ according to U_1 can be measured by $\frac{\underline{S}U_1}{\overline{S}U_1}$, which shows how well S represents the target set U_1. Therefore, how well S describes each target in U can be calculated by Equation (3), which is also the fitness function of F-RS.

$$Fit_{RS} = \frac{\sum_{U_i \in U} |SU_i|}{|\mathbb{U}|} \tag{3}$$

If a feature selection algorithm obtains a reduct with $Fit_R S = 1.0$, it means the reduct can completely separate each class from other classes in the dataset.

F-PRS: F-PRS is based on probabilistic rough set theory, which relaxes the definitions of the lower and upper approximations [25]. For the target set U_1, $\mu_S[x] = \frac{|[x]_S \cap U_1|}{|[x]_S|}$. $\mu_S[x]$ quantifies the proportion of $[x]_S$ is in U_1. $\underline{apr}_S U_1 = \{x | \mu_S[x] \geq \alpha\}$ defines the lower approximation of S according to U_1 rather than $\underline{S}U_1$. $[x]_S$ does not have to completely contained in U_1. α can be adjusted to restrict or relax the lower or upper approximations. When $\alpha = 1.0$, the definition of $\underline{apr}_S U_1$ is the same as $\underline{S}U_1$. The fitness function of F-PRS is shown by Equation (4), which essentially measures the number of instances that S correctly makes indistinguishable from others.

$$Fit_{PRS} = \frac{\sum_{x=1}^{n} |\underline{apr}_S X_i|}{|\mathbb{U}|} \tag{4}$$

Note that the fitness functions of F-MI and F-E combines the maximisation of the relevance and the minimisation of the redundancy into single objective functions. The fitness function of F-RS and F-PRS are to maximise the rough set and probabilistic rough set based relevance measures to maximise the classification accuracy. In filter approaches, the relevance measure is closely related to the classification accuracy while the redundancy is closely related to the number of features. One can see that the fitness functions of F-MI and F-E include both the relevance and redundancy, while the four wrapper algorithms, F-RS and F-PRS only include the classification accuracy or the relevance measure. The reason is that maximising the relevance measure in F-MI and F-E will include all the available features. By contrast, the measures used in F-RS and F-PRS have been shown to be able to remove

some redundancy because adding the minimisation of the number of features in the fitness function did not significantly reduce the size of the feature subset (detailed experimental results can be seen from the literature [19]). Furthermore, the fitness function using the classification accuracy in the wrappers is expected to automatically remove some redundancy because highly redundant feature subsets usually deteriorate the classification performance of a classifier.

3. Wrappers versus Filters

This section focuses mainly on comparing wrapper approaches with filter approaches in terms of the classification performance and the computational cost. The four wrapper methods *W-SVM, W-KNN, W-DT,* and *W-NB*, and the four filter methods, *F-MI*, *F-E*, *F-RS*, and *F-PRS*, are used as representative algorithms in the experiments.

3.1. *Experiments Configuration*

Experiments have been conducted on 12 datasets chosen from UCI machine learning repository [27] as representative examples. The details of the datasets can be seen from Table 1, which indicated by "3" in the last column. The 12 datasets include 10 discrete/categorical datasets because all the four filter algorithms only work on discrete features, and two continuous datasets (Hillvalley and Madelon), which were discretised and used as the representative examples of datasets with medium and large numbers of features (since the available discrete datasets include a relatively small number of features). The Lung dataset includes discrete data but not used here because it has a very small number of instances, which is not enough for filter measures to capture the characteristics of the data. In the experiments, all the instances in each dataset are randomly divided into two sets: 70% as the training set and 30% as the test set. The instances are selected so that the proportion of instances from different classes remains the same in both the training set and the test set. All the algorithms are firstly run on the training set to obtain a good feature subset.

Note that during the evolutionary feature selection (training) process, each evaluation in a wrapper approach involves a classification process that needs a training set and a test set [3]. Therefore, for all the wrapper algorithms, the training set is further split into a sub-training set and a sub-test set. The reason for this is to avoid feature selection bias [3,28]. Note that the results of W-SVM on the Madelon dataset is not available because it could not finish the running process within one week, which is shown as "N/A" in the figures. Two Java based machine learning libraries [29,30] were used to conduct the experiments. The parameters of PSO follow the common settings suggested by Clerc and Kennedy [31]. Since PSO is a stochastic search technique, the experiment of each algorithm on each dataset have been conducted for 40 independent runs. So in total, the results of 3840 (8*12*40) experiments are

Table 1: Datasets

Dataset	NO. of Features	NO. of Classes	NO. of Instances	Data Type	Used in Section
Lymphography (Lymph)	18	4	148	Categorical	3
Mushroom	22	2	5644	Categorical	3
Spect	22	2	267	Categorical	3
Leddisplay	24	10	1000	Categorical	3
Dermatology	34	6	366	Categorical	3
Soybean Large	35	19	307	Categorical	3
Chess	36	2	3196	Categorical	3
Statlog	36	6	6435	Categorical	3
Waveform	40	3	5000	Categorical	3
Splice	60	3	3190	Categorical	3
Lung Cancer (Lung)	56	3	32	Categorical	4, 5
Wine	13	3	178	Continuous	4, 5
Australian	14	2	690	Continuous	4, 5
Zoo	17	7	101	Continuous	4, 5
Vehicle	18	4	846	Continuous	4, 5
German	24	2	1000	Continuous	4, 5
Wisconsin Breast Cancer Diagnostic (WBCD)	30	2	569	Continuous	4, 5
Ionosphere (Ionosp)	34	2	351	Continuous	4, 5
Sonar	60	2	208	Continuous	4, 5
Movementlibras(MoveLib)	90	15	360	Continuous	4, 5
Hillvalley	100	2	606	Continuous	3, 4, 5
Musk Version1 (Musk1)	166	2	476	Continuous	4, 5
Madelon	500	2	2600	Continuous	3, 4, 5
Isolet5	617	26	1559	Continuous	4, 5

used here to compare these wrapper and filter algorithms.

3.2. *Computational Time*

Fig. 1 shows the computational time of the four wrapper algorithms and the four filter algorithms. Each chart in the figure corresponds to one of the twelve datasets used in the experiments. The numbers in the bracket shows the number of features and the number of instances included in the corresponding dataset.

3.2.1. *Computational Time of Filters*

F-MI. As can be seen from Fig. 1, the computational time used by F-MI is the shortest in all datasets. F-MI as a filter approach does not involve any classification process during the evolutionary feature selection process. The mutual information based fitness function of F-MI takes a very short time to calculate. Meanwhile, the calculation of the mutual information between each feature and the class labels, and the mutual information between each pair of features, only needs to be performed once for each dataset before the evolutionary feature selection process. During the evolutionary process, the calculation of the fitness only needs to refer to these values. Therefore, F-MI is fast for all the datasets used in the experiments.

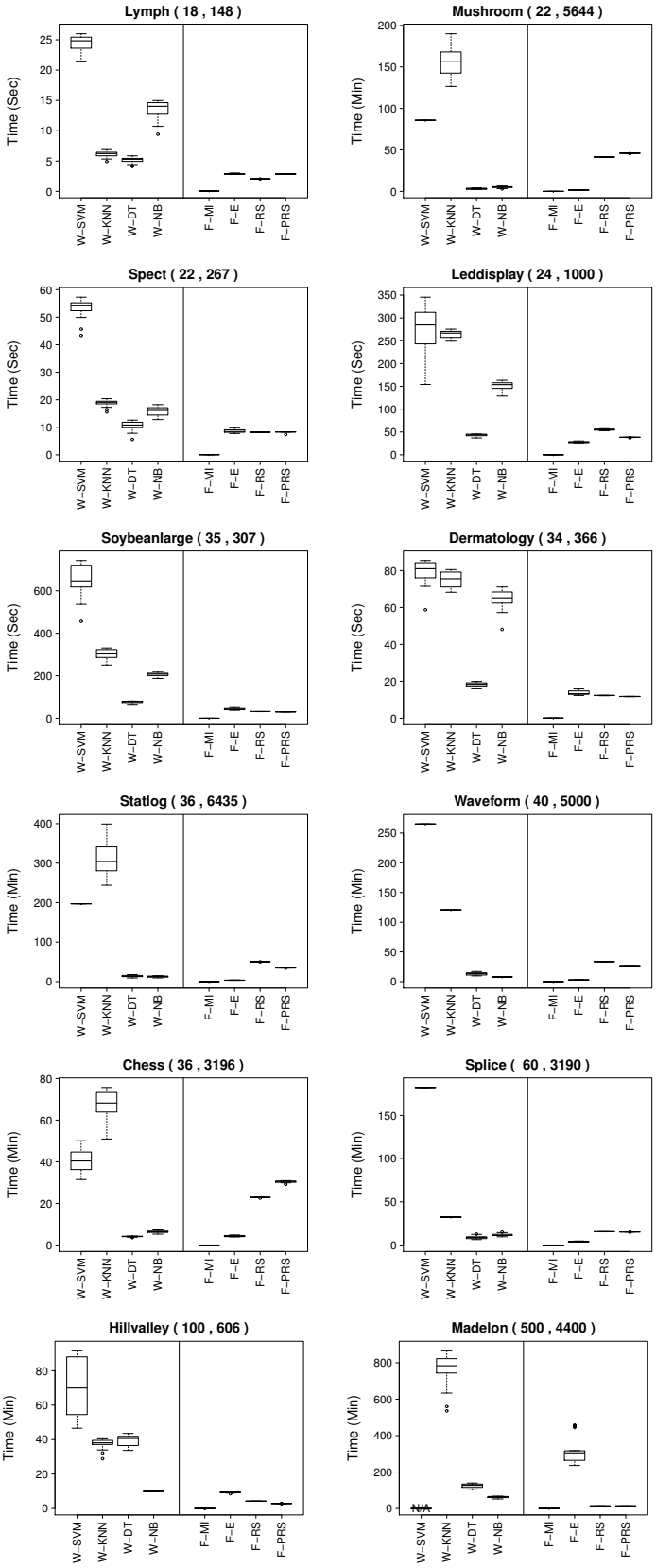

Fig. 1: Comparisons on Computational Time.

F-E. According to Fig. 1, the computational time used by F-E is longer than F-MI in all cases and longer than F-RS and F-PRS on datasets with a large number of features. The main reason is that the fitness function of F-E is more complex than the fitness function used in F-MI. The complexity of the fitness function in F-E increases rapidly along with the number of features. Therefore, on the datasets with a relatively small number of features, F-E is often faster than F-PR and F-PRS, but on the datasets with a large number of features, F-E is usually slower than F-RS and F-PRS.

F-RS and F-PRS. According to Fig. 1, there is no much difference between the time used by F-RS and F-PRS. The main reason is that the calculation of the standard rough set based measure in F-RS is similar to that of the probabilistic rough set based measure in F-PRS. The only difference is that probabilistic rough set (F-PRS) determines equivalent classes based on a threshold while the standard rough set (F-RS) does not have any threshold. F-PRS used a shorter time than F-RS on some datasets because F-PRS selected a smaller number of features than F-RS. The calculation of the measures in both F-RS and F-PRS are significantly influenced by the number of instances. Therefore, on the datasets with a relatively small number of instances, F-RS and F-PRS is faster than F-E, but on the datasets with a large number of instances, F-RS and F-PRS are slower than F-E.

3.2.2. *Computational Time of Wrappers*

W-SVM. According to Fig. 1, it can be seen that W-SVM used longer time than other methods in most cases. As a wrapper approach, each evaluation in W-SVM needs a training and testing classification process of a SVM to evaluate the goodness of the selected features. The SVM used here is the library LIBSVM developed by Chang and Li [32], which involves a number of iterations and a cache method during the training of LIBSVM. The time complexity of LIBSVM is between $O(n * m^2)$ and $O(n * m^3)$, where n is the number of features and m is the number of instances. The computational time depends on how efficiently the cache method is used (dataset dependent) and the number of iterations. There is no theoretical analysis on the number of iterations needed in LIBSVM. Empirically, the number of iterations may be higher than linear to the number of training instances [32]. Therefore, LIBSVM may take a very long time for large datasets. For the Madelon dataset with 500 features, W-SVM could not finish the evolutionary feature selection (training) process within one week. The possible reason is that on Madelon with a large number of features, the number of iterations needed for LIBSVM is huge and the cache method was not able to be used efficiently. Another reason for W-SVM using longer time than other wrapper algorithms is that W-SVM selected a larger number of features than the other algorithms. A large number of features need longer computational time for each evaluation than a smaller number of features.

W-KNN. In most cases, the computational time used by W-KNN is shorter than W-SVM, but longer than other algorithms, especially on the datasets with a large number of instances. The main reason is that in the classification process of KNN, each instance in the sub-testing set is compared with all the instances in the sub-training set to determine its class label. The time complexity for each testing instance is around $O(n * m)$ [33]. If there are q instances in the sub-testing set, the time complexity of KNN is $O(n * m * q)$. Therefore, the increase of the number of instances causes a significant increase in the computational time of W-KNN.

From Fig. 1, it can be observed that the computational time used by W-KNN is significantly influenced by the number of instances while that of W-SVM is significantly influenced by the number of features in the datasets. For example, the Mushroom and Spect datasets include the same number of features, but different numbers of instances. W-KNN spent a shorter time than W-SVM on the Spect dataset with a smaller number of instances, but spent a longer time than W-SVM on the Mushroom dataset with a larger number of instances. By contrast, the Chess and Splice datasets contain a similar number of instances, but different numbers of features. W-SVM spent a shorter time than W-KNN on the Chess dataset with a smaller number of features, but W-SVM spent a longer time than W-KNN on the Splice dataset with a larger number of features. On the Madelon dataset with the largest number of features, W-SVM even could not finish the feature selection process within one week.

W-DT and W-NB. According to Fig. 1, the computational time of W-DT and W-NB is shorter than that of W-SVM and W-KNN in almost all cases. The main reason is that the time complexity of is around $O(n^2 * m)$ [34] in DT and $O(n * m)$ in NB, which is less than that of KNN and SVM on these datasets. The number of features produces greater influence to W-DT than to W-NB in the computational time. Therefore, on the datasets with a large number of features, i.e. Hillvalley and Madelon, W-NB is faster than W-DT.

3.2.3. *Comparisons Between Filters and Wrappers*

According to Fig. 1, it can be seen that the wrapper algorithms using SVM and KNN spent longer time than all other algorithms. Wrappers using DT and NB used a similar or even shorter time than filter algorithms in some cases, such as on the Chess and Madelon. When the number of features increases, the computational time used by W-SVM, W-DT and F-E increased more than other algorithms. When the number of instances increases, the computational time used by W-KNN, F-RS and F-PRS increased more than other algorithms. Overall, F-MI (filter) is always the fastest algorithm regardless of the number of features and the number of instances. For wrapper algorithms, when the number of features is large, the fastest wrapper algorithm is W-NB. When the number of instances is small, one can choose the wrapper algorithm W-DT to perform a fast feature selection process.

3.3. *Classification Performance*

Fig. 2 shows the classification performance of the four wrapper algorithms and the four filter algorithms. In the wrapper approaches, the classification performance is evaluated by the internal classification algorithm used during the evolutionary feature selection process, e.g. the classification performance of the features selected by W-KNN is evaluated by KNN. Filter approaches do not involve any classification algorithm during the evolutionary feature selection process. All the classification algorithms can be used to evaluate the classification performance. SVM was used here because W-SVM using SVM achieved better performance than the other three wrapper algorithms and a slight bias is given to filter approaches. The performance of using other classification algorithms is often similar or worse than that of SVM, so are not presented here due the the limited space.

According to Fig. 2, the best classification performance is achieved by one of the wrapper approaches in almost all cases. The worst classification performance is produced by one of the filter approaches. For the four filter algorithms, the performance of F-MI and F-E are better than that of F-RS and F-PRS. F-E is slightly better than F-MI because the fitness function of F-E considers the selected features as a whole rather than each pair of features in F-MI. All the four wrapper algorithms achieved almost the same performance on the Leddisplay dataset, which is 100% and because this problem is relatively easy. W-SVM achieved better performance than the other three algorithms on 5 of the 11 datasets (except for Leddisplay). This number is 2 for W-KNN, 2 for W-DT, and 2 for W-NB. Clearly, these classification algorithms performed differently on different datasets, depending on characteristics of the algorithm and the dataset itself. The choice of the best classification algorithm for a certain type of data is beyond the scope of this paper, but the results here show that any of the four wrapper algorithms can be used to obtain feature subsets with reasonable good classification performance.

Overall, considering both the classification performance and the computational cost, if users have enough time, wrapper approaches W-SVM and W-KNN are good choices. If the demand of users is more on the computational time than the classification performance, it is better to use a filter algorithms, such as F-MI. Furthermore, if users need to avoid poor classification performance, fast wrapper algorithms, such as W-DT and W-NB, are good choices.

3.4. *Further Comparisons*

This section further investigates individual features selected by the eight different algorithms to test their consistency. Fig. 3 takes the Chess dataset as an example to show the number of appearances of each individual feature over the 40 independent runs in the eight algorithms, where each chart corresponds to one of the eight algorithms. In each chart, the vertical axis shows the frequency of the feature being selected, where 40 means that the feature was selected in all the 40 independent runs while 0 means the feature was never selected. The horizontal axis shows the

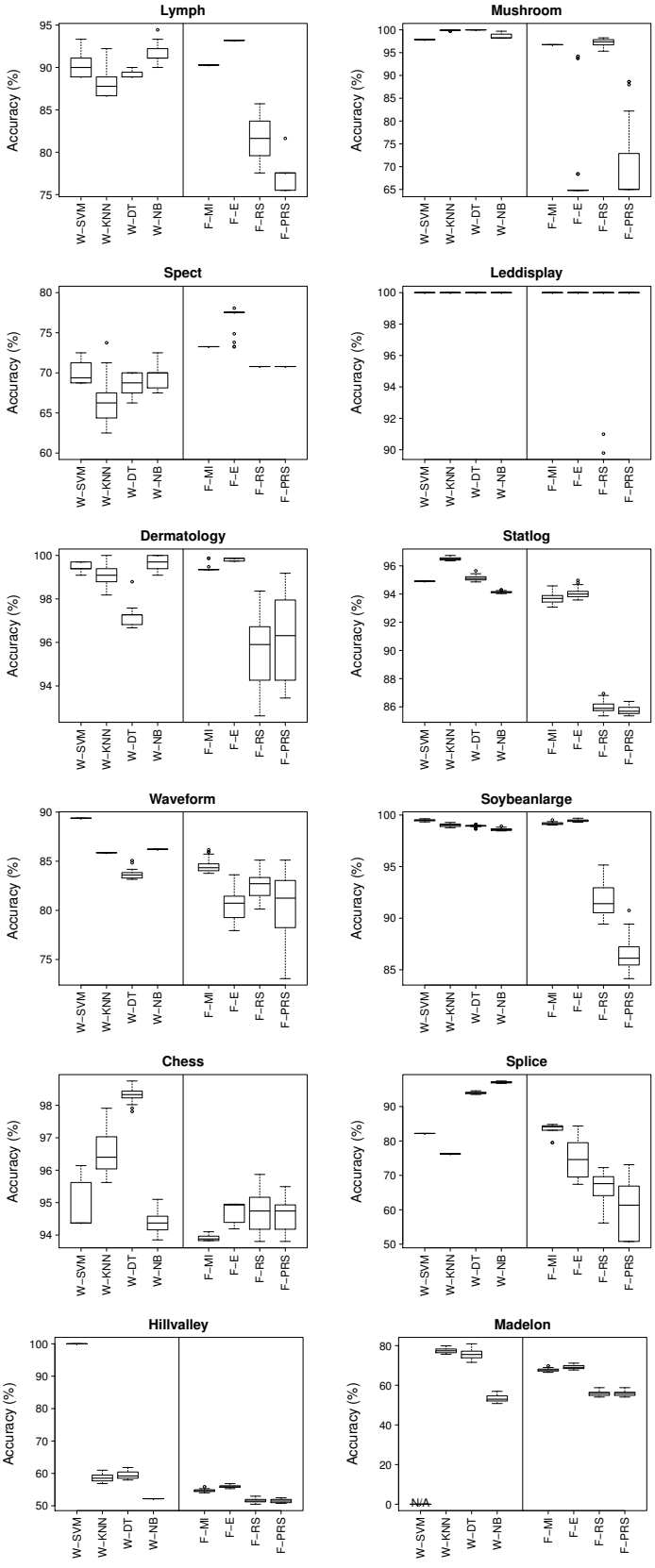

Fig. 2: Comparisons on Classification Performance.

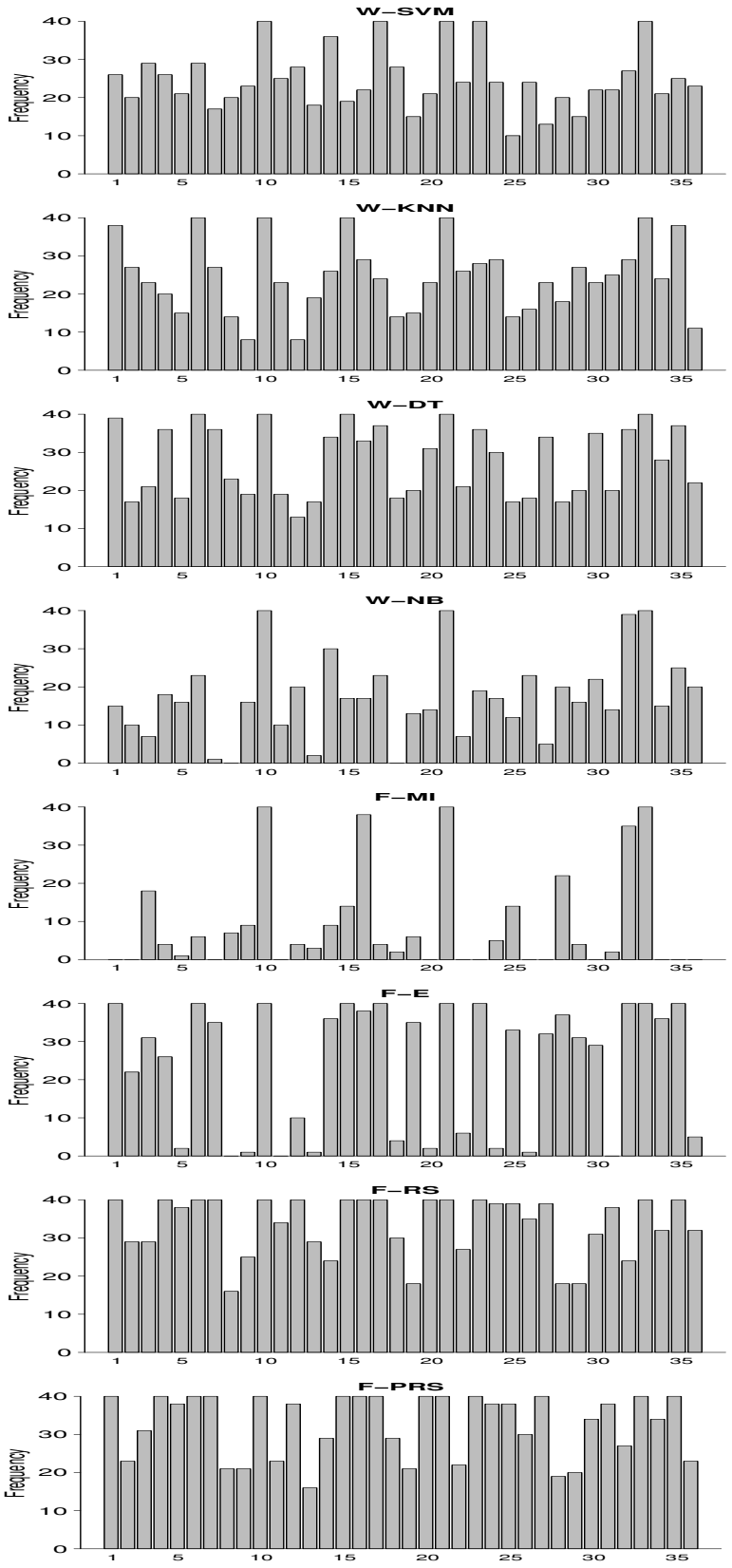

Fig. 3: Comparisons on the Selected Features in the Chess Dataset.

index of all the 36 features in the Chess dataset.

Fig. 3 shows that the eight algorithms selected different numbers of features. For example, W-NB and F-MI usually selected a relatively small number of features while F-RS and F-PRS usually selected a relatively large number of features. This is mainly because they used different criteria to evaluate the quality of the feature subsets, where the best feature subset for one criterion may not be the best feature subset for another criterion. Fig. 3 also shows that all the eight algorithms *consistently* selected Features 10, 21 and 33 on all their 40 independent runs. These three features may be the "core" features of the Chess dataset. The detailed results also reveal that they selected other different individual features. This is due mainly to the feature interaction problem, where some other different (complementary) features are needed to work together with such "core" features to optimise their fitness functions (based on different criteria).

The other datasets show a similar pattern to the Chess dataset, i.e. there are always some "core" features selected by all the eight algorithms and also some different complementary features selected by different algorithms. Further investigation of "core" features and feature interaction in different datasets requires domain knowledge, which forms part of the future work.

4. Generality of Wrappers

This section tests the generality of the four wrapper algorithms, i.e. W-SVM, W-KNN, W-DT, and W-NB. The classification performance of the features selected by each wrapper algorithm is tested using all the four classification algorithms, i.e. SVM, KNN, DT and NB. 14 datasets are used here in the experiments in this section, which are different from the ones in Section 3 because the wrapper algorithms do not require the data to be discrete values and a variety of datasets can used in this section to further and better test the algorithms. The detailed parameter settings are the same as in Section 3.1. Each algorithm has been conducted for 40 independent runs on each dataset, which are in total 2240 (4*14*40) experiments producing 2240 feature subsets. Then each of the 2240 feature subsets were tested by the four classification algorithms resulting in 8960 (2240*4) classification accuracies, which are used to test the generality of wrappers.

The experimental results are shown in Table 2. The feature subsets selected by each of the four wrappers (W-KNN, W-NB, W-DT, and W-SVM) are tested using all the four classification algorithms. The classification performance of KNN (or NB, DT, SVM) using the selected features are then compared with KNN (or NB, DT, SVM) using all features. In order to make the results easy to observe, the detailed classification performance are not presented here. Only the results of the statistical significance tests between the classification performance of the *selected* features and that of *all* features are shown in the table. The pairwise Student's T-test with a significance level of 0.05 (or confidence interval of 95%) was performed here. Note that a non-parametrical significance test named Wilcoxon test was also

used here and the results are not shown here since they are almost identical to that of T-test. In the table, "+" (or "−") means one classification algorithm (e.g KNN) using the selected feature subsets achieved significantly better (or worse) performance than using all features. "=" means there is no significant different between the results. The last row summaries the total number of datasets where the classification performance of a classification algorithm is similar or significantly better than using all features. The results of W-SVM on Madelon are not presented because the dataset is too big and the evolutionary feature selection process could not finish within one week.

According to Table 2, when using KNN during the evolutionary feature selection (training) process, KNN itself using the selected feature subsets increased or maintained the classification performance on 13 out of the 14 datasets. On 7 of the 14 datasets, the classification performance of NB, DT and SVM were maintained or improved over using all features. The results show that W-KNN benefits KNN itself more, but it also benefits the other classifiers. The possible reason is that KNN is very simple, which has a small probability to overfit the training data.

When using NB during the feature selection process, NB itself using the selected feature subsets increased or maintained the classification performance on 9 datasets. The classification performance of KNN, DT and SVM were maintained or increased over using all features in 8, 9 and 6 cases, respectively. The results show that W-NB benefits NB itself more than the others. Although the total number of "+" and "=" for DT is 9, the same as for NB, but 7 of these 9 cases is "=". This means the classification performance of DT using the feature selected by W-NB is similar to that of all features. The possible reason is that NB assumes the features are conditionally independent to each other and it is easy to select a group of individually good features but not complementary features. Such features usually contain most of the useful information in the original features, but may also have redundancy.

When using DT during the feature selection process, as can be seen from Table 2, the classification performance of DT, KNN, NB and SVM were the similar or increased over using all features on 7, 8, 8 and 7 datasets, respectively. When using SVM during the feature selection process, the classification performance of SVM, KNN, NB and DT were the same or increased over using all features on 7, 9, 6, and 7 datasets, respectively.

Overall, the results show that wrapper approaches can be reasonably general. Wrappers using a relatively simple classification algorithms, e.g. KNN and NB, can be general to different classification algorithms. Wrappers using a relatively complicated classification algorithm, e.g. DT and SVM, are less general than using KNN and NB. The possible reason is that the complicated classification process of DT and SVM may select features that are particularly suitable for themselves. The classification performance of SVM can be increased by features selected by W-KNN and W-NB, which is faster and even better classification performance than W-SVM. Meanwhile, SVM as a complicated algorithm using either all features or

Table 2: Results to Test the Generality of Wrappers on 14 Datasets

E.g. KNN using the selected features compared with that of KNN using all features.

	Feature Subsets from W-KNN				Feature Subsets from W-NB			
	KNN	NB	DT	SVM	NB	KNN	DT	SVM
Wine	+	-	+	+	+	+	=	+
Australian	+	+	-	+	+	+	+	+
Zoo	+	-	+	+	-	+	=	+
Vehicle	+	-	=	-	-	=	=	-
German	=	=	-	-	+	=	-	-
WBCD	+	-	=	+	+	-	+	+
Ionosphere	+	+	-	-	+	+	=	-
Lung	+	-	-	=	-	+	=	+
Sonar	+	-	=	-	-	-	-	-
Movementlibras	-	-	=	=	-	-	=	-
Hillvalley	+	=	-	-	=	=	-	-
Musk1	+	+	=	-	+	-	=	-
Madelon	+	=	-	+	+	-	-	+
Isolet5	+	=	-	-	+	-	-	-
Total NO. of "+"	12	3	2	5	8	5	2	6
Total NO. of "+" and "="	13	3	7	7	9	8	9	6

	Feature Subsets from W-DT				Feature Subsets from W-SVM			
	DT	KNN	NB	SVM	SVM	KNN	NB	DT
Wine	+	+	-	+	+	+	-	=
Australian	+	+	+	+	+	+	-	=
Zoo	+	+	-	=	+	+	-	=
Vehicle	+	-	-	-	-	-	-	-
German	-	=	=	-	-	+	+	-
WBCD	=	-	-	+	+	-	-	-
Ionosphere	-	+	+	-	=	+	=	-
Lung	=	+	-	+	=	+	-	=
Sonar	-	=	-	-	-	=	-	+
Movementlibras	-	-	-	=	-	-	=	=
Hillvalley	-	=	=	-	=	+	=	-
Musk1	+	-	+	-	-	-	+	+
Madelon	-	-	=	+				
Isolet5	-	-	=	-	-	=	=	-
Total NO. of "+"	5	5	3	5	4	7	2	2
Total NO. of "+" and "="	7	8	7	7	7	9	6	7

the selected features usually obtains high classification performance and the best classification performance is often achieved by SVM. Therefore, if users want to reduce the number of features but still achieve high classification performance, it is good to use W-KNN and W-NB to select features and use SVM for classification on the unseen test set.

Note that Table 2 aims to test the generality of wrappers, e.g. whether features selected by a wrapper algorithm can increase the performance of other classification algorithms. So, Table 2 only shows the comparisons between one classification algorithm using the selected features and using all features, but does not show the absolute classification accuracy since it is not the focus of this paper.

5. Single Objective versus Multi-Objective

Most of the existing feature selection algorithms are single objective approaches. A relatively small number of multi-objective. In our previous research [13,35,9,14,10,36], comparisons on the final solutions have shown that the multi-objective approaches can usually discover multiple and better solutions than the single objective algo-

Table 3: Computational Time (In minutes)

Method	Wine	Australian	Zoo	Vehicle	German	WBCD	Ionosp
PSOIniPG	0.21	2.57	0.07	6.02	9.37	2.07	0.71
CMDPSOFS	0.25	4	0.09	6.59	9.3	2.71	1.54
Method	Sonar	MoveLib	Lung	Hillvalley	Musk1	Madelon	Isolet5
PSOIniPG	0.37	1.88	0.02	14.6	7.22	651.88	247.12
CMDPSOFS	0.475	2.075	0.01	23.49	6.02	394.45	200.71

rithms, but there has not been serious comparisons between single objective and multi-objective algorithms by looking deeply the evolutionary process. *PSOIniPG* [22] as a typical single objective algorithm considering both the number of features and the classification accuracy, and *CMDPSOFS* [14] as the best-so-far multi-objective algorithms are used in the section.

CMDPSOFS and PSOIniPG used the same classification algorithm (i.e. KNN with K=5), population size and total number of evaluations in PSO. Other parameters can be seen in the literature [14,22]. Each algorithm has been conducted for 40 independent runs on each dataset, which are in total 1120 (2*14*40) experiments to compare single objective and multi-objective feature selection approaches.

5.1. *Computational Time*

Table 3 shows the average computational time used by PSOIniPG and CMDPSOFS for a single run, where PSOIniPG produces a single solution and CMDPSOFS produces a set of non-dominated solutions. As wrapper approaches, most of the computational time is spent on the evaluation of the selected feature subsets, which involves a classification process and the time is significantly influenced by the number of features in a certain dataset. From Table 3, it can be observed that on the datasets with a relatively small number of features, both PSOIniPG and CMDPSOFS completed the evolutionary training (feature selection) process within 25 minutes. CMDPSOFS often used a slightly longer time than PSOIniPG on such datasets. The main reason is that there is no significant difference between the evaluation time (depending on the number of selected features) used by PSOIniPG and CMDPSOFS. Meanwhile, CMDPSOFS involves additional procedures related to the multi-objective mechanism, which takes a slightly longer time than PSOIniPG. However, on the large datasets, i.e. Madelon and Isolet5, CMDPSOFS used a much shorter time than PSOIniPG. The main reason is that CMDPSOFS selected a much smaller number of features than PSOIniPG on these datasets, which needs a much shorter time for each evaluation during the evolutionary feature selection process.

5.2. *Evolutionary Process*

The number of features selected by PSOIniPG is larger than CMDPSOFS due mainly to its single objective updating mechanism. Although PSOIniPG considers both the classification performance and the number of features during the evolutionary search process, the classification performance is treated as the priority. When

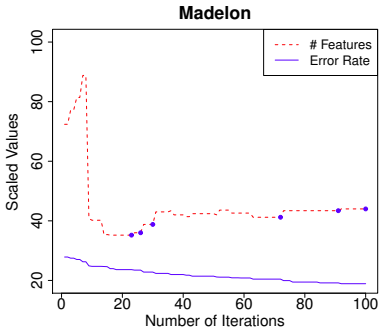

Fig. 4: Change of *gbest* in PSOIniPG during an evolutionary process.

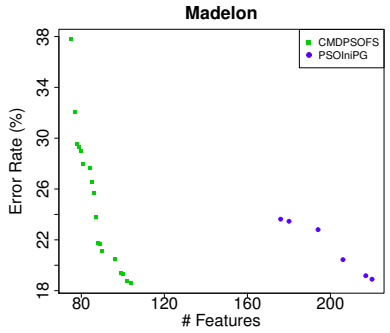

Fig. 5: The solutions obtained by CMDPSOFS and PSOIniPG.

available, PSOIniPG will search toward the regions in the solution space with a low classification error rate regardless of the number of features.

Fig. 4 shows the change of the classification error rate and the number of features selected by the best particle (*gbest*) in each iteration during a single evolutionary process of PSOIniPG on the Madelon dataset. The horizontal axis show the number of iterations from 1 to 100. Since the total number of features is 500, which is much larger than the error rate in [0, 100], the numbers of features are divided by 5 to scale the range to [0, 100] in the vertical axis.

According to Fig. 4, it can be seen that the error rate of *gbest* always became smaller and smaller since the classification performance is the fitness function (the priority). The number of features fluctuated because *gbest* was updated whenever the error rate became smaller, but the number of features might be larger. The blue points in the red line shows the solutions that are non-dominated to each other and dominate all other solutions of *gbest* during the evolutionary search process. Since the single objective mechanism only keeps one single solution, only the solution with the lowest classification error rate was returned by PSOIniPG. Note that PSOIniPG selected smaller feature subsets than other single objective algorithms (see the paper [12]) because PSOIniPG considers the number of features in the *pbest* and *gbest* updating procedure. For other single objective algorithms, there are more (non-dominated) solutions like the blue points found during the evolutionary search process, but none of them were reported by the algorithm.

Fig. 5 shows the non-dominated solutions found by CMDPSOFS and all the non-dominated solutions found by PSOIniPG (i.e. the blue points in Fig. 4). It can be seen that the solutions of CMDPSOFS have a smaller number of features and a lower classification error rate than that of PSOIniPG. This is due mainly to the multi-objective mechanism in CMDPSOFS, where the non-dominated solutions obtained during the evolutionary search process are kept as potential leaders (*gbest*) to guide the algorithm to search around to find better solutions with smaller numbers of features and higher classification performance. Furthermore, CMDPSOFS returns multiple solutions, which provide more choices than PSOIniPG.

6. Conclusions

This paper investigated three issues in feature selection. The first one is the comparisons on the classification performance and computational time of wrapper approaches and filter approaches. The second one is the generality of wrapper approaches. The third one is the discussions on the advantages of multi-objective feature selection algorithms over single objective algorithms.

Wrapper approaches are argued to be computationally more expensive and can achieve better classification performance than filter approaches. This paper shows that when a wrapper algorithm using NB or DT, it can be computationally cheaper than a filter algorithm with a complex measure, such as rough set based measures. Meanwhile, because of the interaction between features and a given classification algorithm, wrapper algorithms are often relatively good in terms of the classification performance, although not always better than filter approaches. Some filter methods, such as mutual information, are very fast and can usually achieve good performance.

Wrapper approaches are also argued to be lack of generality. This paper shows that this is not always the case, as wrappers using a simple classification algorithm, e.g. KNN and NB, during the feature selection process are often general to other classification algorithms. Wrappers using a relatively complicated classification algorithm, e.g. SVM, are usually not general to other algorithms because SVM involves a complicated classification process and the features selected are more specifically suitable to itself rather than other classification algorithms.

This paper also shows that the multi-objective mechanism is a more appropriate way than the single objective mechanism for feature selection tasks by directly observing the evolutionary process. Single objective algorithms only keep one single solution *gbest* to guide the search, which is more likely to become stuck in local optima. Multi-objective algorithms keep the non-dominated solutions found during the evolutionary search process, which are used as potential leaders to guide the algorithm to search around and find better solutions.

Overall, this paper shows that wrapper algorithms can be faster than (some) filter approaches and general to different classification algorithms. If users have a high demand on the computational time and also need to avoid poor classification performance, a filter algorithm (like F-MI) and a fast wrapper algorithm (W-DT or W-NB) are good choices. If users have a high demand on the classification performance, a fast classification algorithm (i.e. NB) can be used in a wrapper to select features, and the selected features can be used in SVM for classification. Meanwhile, a multi-objective algorithm is a better choice than a single objective algorithm in most cases.

References

1. H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17,

no. 4, pp. 491–502, 2005.
2. I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *The Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.
3. R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, pp. 273–324, 1997.
4. J. Kennedy and R. Eberhart, "Particle swarm optimization," in *IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.
5. Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *IEEE International Conference on Evolutionary Computation (CEC'98)*, pp. 69–73, 1998.
6. A. Unler and A. Murat, "A discrete particle swarm optimization method for feature selection in binary classification problems," *European Journal of Operational Research*, vol. 206, no. 3, pp. 528–539, 2010.
7. I. Babaoglu, O. Findik, and E. Ulker, "A comparison of feature selection models utilizing binary particle swarm optimization and genetic algorithm in determining coronary artery disease using support vector machine," *Expert Systems with Applications*, vol. 37, no. 4, pp. 3177 – 3183, 2010.
8. M. Mohamad, S. Omatu, S. Deris, and M. Yoshioka, "A modified binary particle swarm optimization for selecting the small subset of informative genes from gene expression data," *IEEE Transactions on Information Technology in Biomedicine*, vol. 15, no. 6, pp. 813–822, 2011.
9. B. Xue, L. Cervante, L. Shang, W. N. Browne, and M. Zhang, "A multi-objective particle swarm optimisation for filter based feature selection in classification problems," *Connection Science*, vol. 24, no. 2-3, pp. 91–116, 2012.
10. B. Xue, L. Cervante, L. Shang, W. N. Browne, and M. Zhang, "Binary PSO and rough set theory for feature selection: A multi-objective filter based approach," *International Journal of Computational Intelligence and Applications*, vol. 13, no. 02, p. 1450009, 2014.
11. B. Xue, M. Zhang, and W. N. Browne, "New fitness functions in binary particle swarm optimisation for feature selection," in *IEEE Congress on Evolutionary Computation (CEC'12)*, pp. 2145–2152, 2012.
12. B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms," *Applied Soft Computing*, vol. 18, no. 0, pp. 261–276, 2014.
13. B. Xue, M. Zhang, and W. N. Browne, "Multi-objective particle swarm optimisation (PSO) for feature selection," in *Genetic and Evolutionary Computation Conference (GECCO'12), Philadelphia, PA, USA*, pp. 81–88, ACM, 2012.
14. B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimization for feature selection in classification: A multi-objective approach," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1656–1671, 2013.
15. Y. Zhang, C. Xia, D. Gong, and X. Sun, "Multi-objective PSO algorithm for feature selection problems with unreliable data," in *Advances in Swarm Intelligence*, vol. 8794 of *Lecture Notes in Computer Science*, pp. 386–393, Springer International Publishing, 2014.
16. C. Bae, W.-C. Yeh, Y. Y. Chung, and S.-L. Liu, "Feature selection with intelligent dynamic swarm and rough set," *Expert Systems with Applications*, vol. 37, no. 10, pp. 7026 – 7032, 2010.
17. S. S. S. Ahmad and W. Pedrycz, "Feature and instance selection via cooperative PSO," in *IEEE International Conference on Systems, Man, and Cybernetics (SMC'11)*, pp. 2127–2132, 2011.
18. L. Cervante, B. Xue, M. Zhang, and L. Shang, "Binary particle swarm optimisation

for feature selection: A filter based approach," in *IEEE Congress on Evolutionary Computation (CEC'12)*, pp. 881–888, 2012.
19. L. Cervante, B. Xue, L. Shang, and M. Zhang, "A dimension reduction approach to classification based on particle swarm optimisation and rough set theory," in *25nd Australasian Joint Conference on Artificial Intelligence*, vol. 7691 of *Lecture Notes in Computer Science*, pp. 313–325, Springer, 2012.
20. M. A. Mandal M, "A graph-theoretic approach for identifying non-redundant and relevant gene markers from microarray data using multiobjective binary PSO," *PLoS ONE*, vol. 9, pp. 1–13, 2014.
21. M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, no. 4, pp. 131–156, 1997.
22. B. Xue, M. Zhang, and W. Browne, "Novel initialisation and updating mechanisms in PSO for feature selection in classification," in *Applications of Evolutionary Computation*, vol. 7835 of *Lecture Notes in Computer Science*, pp. 428–438, Springer Berlin Heidelberg, 2013.
23. C. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Urbana:The University of Illinois Press, 1949.
24. Z. Pawlak, "Rough sets," *International Journal of Parallel Programming*, vol. 11, pp. 341–356, 1982.
25. Y. Yao and Y. Zhao, "Attribute reduction in decision-theoretic rough set models," *Information Sciences*, vol. 178, no. 17, pp. 3356 – 3373, 2008.
26. H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.
27. K. Bache and M. Lichman, "Uci machine learning repository," 2013.
28. C. Ambroise and G. J. McLachlan, "Selection bias in gene extraction on the basis of microarray gene-expression data," *Proceedings of the National Academy of Sciences*, vol. 99, no. 10, pp. 6562–6566, 2002.
29. I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques (Second Edition)*. Morgan Kaufmann, 2005.
30. T. Abeel, Y. V. de Peer, and Y. Saeys, "Java-ml: A machine learning library," *Journal of Machine Learning Research*, vol. 10, pp. 931–934, 2009.
31. M. Clerc and J. Kennedy, "The particle swarm– explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.
32. C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, 2011.
33. C. Elkan, "Nearest neighbor classification," Technical Report http://cseweb.ucsd.edu/users/elkan/250Bwinter2011/, University of California, 2011.
34. J. Su and H. Zhang, "A fast decision tree learning algorithm," in *Proceedings of the 21st national conference on Artificial intelligence – Volume 1*, AAAI'06, pp. 500–505, AAAI Press, 2006.
35. B. Xue, L. Cervante, L. Shang, and M. Zhang, "A particle swarm optimisation based multi-objective filter approach to feature selection for classification," in *12th Pacific Rim International Conference on Artificial Intelligence (PRICAI 2012)*, vol. 7458 of *Lecture Notes in Computer Science*, pp. 673–685, Springer, 2012.
36. B. Xue, L. Cervante, L. Shang, W. N. Browne, and M. Zhang, "Multi-objective evolutionary algorithms for filter based feature selection in classification," *International Journal on Artificial Intelligence Tools*, vol. 22, no. 04, p. 1350024, 2013.