Noname manuscript No.
(will be inserted by the editor)

An enhanced XCS rule discovery module using feature ranking

Mani Abedini · Michael Kirley

the date of receipt and acceptance should be inserted later

Abstract XCS is a genetics-based machine learning model that combines reinforcement learning with evolutionary algorithms to evolve a population of classifiers in the form of condition-action rules. Like many other machine learning algorithms, XCS is less effective on high-dimensional data sets. In this paper, we describe a new guided rule discovery mechanisms for XCS, inspired by feature selection techniques commonly used in machine learning. In our approach, feature quality information is used to bias the evolutionary operators. A comprehensive set of experiments is used to investigate how the number of features used to bias the evolutionary operators, population size, and feature ranking technique, affect model performance. Numerical simulations have shown that our guided rule discovery mechanism improves the performance of XCS in terms of accuracy, execution time and more generally in terms of classifier diversity in the population, especially for high-dimensional classification problems. We present a detailed discussion of the effects of model parameters and recommend settings for large scale problems.

Keywords Learning Classifier Systems, Genetics-Based Machine Learning, XCS, Feature Ranking, High-dimensional Classification, Microarray Gene Expression Profiles

Department of Computing and Information Systems
The University of Melbourne, Australia
E-mail: mabedini@studnet.unimelb.edu.au , mkirley@unimelb.edu.au

1 Introduction

Classification problems arise frequently in many areas of science and engineering. Examples include: disease classification based on gene expression profiles in bioinformatics; document classification in information retrieval; image recognition; and fraud detection [32]. The goal of any classification algorithm, is to build a model that captures the intrinsic associations between the class type and the features (attributes) in an attempt to match each input value to one of a given set of class labels [28].

Learning classifier systems (LCS) are a type of genetics-based machine learning (GBML) algorithm for rule induction, which have been applied to a large variety of classification problems [14,23,27]. LCS combine reinforcement learning with evolutionary computing and other heuristics to produce an adaptive system that learns to solve a particular problem. Of the Michigan-style GBML methods, XCS [13,34,35] is perhaps the most well-known architecture. Each individual in the XCS population encapsulates a single rule (*condition-action-prediction*) with an associated fitness value. Through an iterative learning process, the population of classifiers evolves. A key step in this iterative process is the rule discovery component that creates new classifiers to be added to the bounded population pool.

Over the last few years, the analysis of high-dimensional data sets has been the subject of numerous publications in statistics, machine learning, and bioinformatics. Consider a prototypical high-dimensional data set, such as a microarrray gene expression data set, that has several thousands genes (features) but only a small number of samples. Typically many of the features are irrelevant to the classification task [37]. Given the high-dimensional search space, building effective classification models is a non-trival task for microarrray gene expression data sets.

Standard XCS implementations, and many other machine learning classification algorithms, are typically less effective on data sets characterized by a high-dimensional space — *the curse of dimensionality*. It is difficult to effectively explore the solution space and build an appropriate classification model. There has only been a small number of studies investigating XCS for high-dimensional classification tasks that have appeared in the literature. Representative examples include a study of large multiplexer problems [4] and more generally, work examining the scalability of XCS [31]. In contrast, Pittsburg-style GBML implementations have been used to classify large-scale bioinformatics data sets (e.g. [8,9]). In this approach, each individual in the evolving population represents a complete classifi-

cation rule. As such, the evolved rules for the high-dimensional data sets are typically very large. Consequently, specialized evolutionary operators are required to guide the learning process.

In this paper, a new guided rule discovery mechanisms is proposed for XCS for high-dimensional classification problems[1]. Our model, called GRD-XCS, is inspired by feature selection techniques commonly used in machine learning. Typically, filtering techniques assess the relevance of features in the data set. A subset of the "more important" features is then presented as input to the classification algorithm while the "less important" features are ignored. However, in our model the filtering process is used to build a probability distribution that biases the evolutionary operators encapsulated in the rule discovery component of XCS. This probability distribution can be thought of as a mask that biases the uniform crossover and mutation operators. This flexible approach is scalable, and the enhanced XCS can be used to tackle high-dimensional classification tasks without reducing the dimensionality of the data set.

The remainder of this paper is organized as follows. In section 2, the XCS model is described and an overview of the role of feature ranking in machine learning is presented. Related work focussed on GMBL evolutionary operators is presented in section 3. The GRD-XCS model is presented in section 4. In section 5, the experimental methodology is described and parameters listed. Experiment results are presented in section 6. Section 7 describes the effects of alternative feature ranking techniques in the GRD-XCS model. Finally, in section 8 we summarize the findings of this study and recommend parameter settings for complex, large-scale classification problems. Avenues for future work are also presented.

2 Background

2.1 XCS: The eXtended classifier system

In this subsection, we provide a brief overview of the functionality of the XCS model. A more detailed discussion of GBML classifier systems can be found in [14, 23, 27].

As an instance of a GBML algorithm, XCS represents the knowledge extracted from a given problem as a population of evolving *condition-action-prediction* rules [34]. At each

[1] GRD-XCS was introduced in [5]. This paper is a revised and a substantially extended version of that paper.

Algorithm 1 High level overview of XCS

Require: Input data: σ, Population:[Δ]
  repeat
    $\sigma \leftarrow env$
M $\leftarrow GetMatchSet(\sigma,[\Delta])$
PA $\leftarrow CreatePredictionArray([M])$
    $act \leftarrow SelectAnAction([PA])$
A $\leftarrow CreateActionSet([M], act)$
    $R \leftarrow ExecutingActionOnENV(act)$
A $\leftarrow UpdateSet([A], R)$
Δ $\leftarrow RuleDiscovery([A], [\Delta])$
  until certain number of iterations

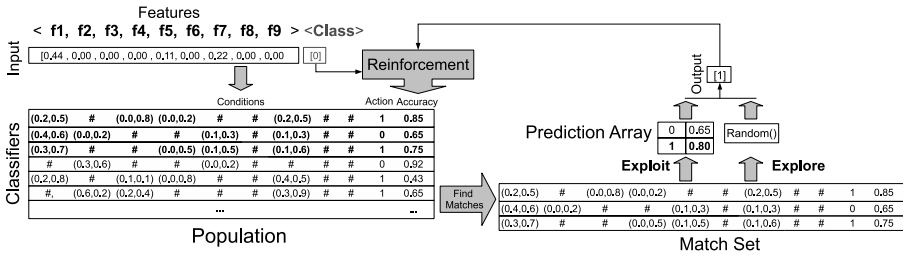


Fig. 1: XCS model overview. The condition segment of the classifier consists of a vector of features, each encoded using real or binary values. The output signal (prediction class) is a binary value in this case. The classifier's fitness value is proportional to the accuracy of the prediction of the reward. See text for further explanation.

iteration of the model, these rules are evaluated. Reinforcement learning and a search algorithm are then used to evolve the population of classifiers (rules). The three key components of the model are the *representation scheme* used to encode the population of classifiers, the *credit assignment* mechanism and the *rule discovery*. These components are described below.

Each individual in the XCS population is a classifier that maps input vectors to output signals (or class labels). A suitable representation scheme is required for this mapping. Originally, each feature in the vector (condition part of the rule) was encoded using a ternary alphabet {0,1,#}, where # represents a "don't care" feature. Wilson [35] proposed an alternative real-value encoding style for the feature vector that is now widely used. The predicted class label (condition part of the rule) for a binary problem can be encoded using a binary alphabet {0,1}. For multi-class problems, larger alphabets can be used.

Given the population sizes typically used in GBML, it is reasonable to expect that there will be many copies of the same classifier in the population at any given point in time. Consequently, the term *macro classifier* is used to describe unique instances of classifiers in the population. Each macro classifier has a *numerosity* value that counts the number of identical macro classifier instances in the population. As such, the number of macro classifiers in the population is a measure of population diversity. From a programming and implementation point of view, key processing steps are performed using the macro classifiers, which significantly improves the execution time of the model.

At each time step, the classifier system receives a problem instance in the form of a vector of features. A decision, that is, an action to be performed next based on the input vector, must be determined. A *match set* $[M]$ is created consisting of rules (classifiers) that can be "triggered" by the given data instance. A covering operator is used to create new matching classifiers when $[M]$ is empty. A prediction array $[PA]$ is calculated for $[M]$ that contains an estimation of the corresponding rewards for each of the possible actions. Based on the values in the prediction array, an action act is selected. Classifiers that support the predicted action make up the *Action Set* $[A]$ (see algorithm 1).

In response to act, the reinforcement mechanism is invoked and the prediction p, prediction error ϵ, accuracy k, and fitness F of the classifiers are updated. The corresponding numerical reward is distributed to the rules accountable for it, thus improving the estimates of the action values. The prediction and prediction error are updated as follows:

$$p \leftarrow p + \beta(R - p) \qquad and \qquad \epsilon \leftarrow \epsilon + \beta(|R - p| - \epsilon)$$

where β is the learning rate ($0 < \beta < 1$). The classifier accuracy is calculated from the following equations:

$$k = \begin{cases} 1 & \text{if } \epsilon < \epsilon_0 \\ \alpha(\frac{\epsilon}{\epsilon_0})^{-\nu} & \text{otherwise} \end{cases} \qquad and \qquad k^{'} = \frac{k}{\sum\limits_{x \in [A]} k_x}$$

Finally, the classifier fitness F is updated using the relative accuracy value:

$$F \leftarrow F + \beta(k^{'} - F)$$

In XCS, the classifier's fitness value is updated based on the accuracy of the actual reward prediction. This accuracy-based fitness provides a mechanism for XCS to build a complete action map of the input space.

Figure 1 provides a high-level overview of the classification process using XCS. In this binary classification example, the data features have real-values. Here, we show one cycle of the "classification" steps: creating the matched set [M], prediction set [PA] and inferring the predicted label act.

The rule discovery module of XCS plays a very important role. Typically, a genetic algorithm is responsible for improving the population of classifiers. During the evolutionary learning process, fitness-proportionate selection is applied to [A]. Standard evolutionary operators, *uniform crossover* and *mutation*, are then applied to the selected individuals. In addition, a second mutation-style operator — the *don't care* operator — is used to randomly modify a condition part of a classifier to the "don't care" value #. The newly created offspring (classifiers) are then added to the bounded population. A form of niching is then used to determine if the offspring survive in the population and/or which of the old members of the population are to be deleted to make room for the new classifiers (offspring). A subsumption mechanism combines similar classifiers and a randomized deletion mechanism removes from the population classifiers with a low fitness.

2.2 Feature ranking

Our novel extension of the standard XCS framework is based on feature ranking. Below we describe feature ranking categories typically used in machine learning. Specific techniques used in this study are discussed in more detail in sections 4 and 7.

Feature (or attribute) selection is a term that describes a range of machine learning and/or statistical techniques used to select a subset of relevant features when building robust learning models. Based on a nominated metric, the ranking of particular features provides valuable information about the relative "importance" of the features. It is then assumed that the "top ranked" features are more likely to be helpful for the classification task, rather than lower ranked features [19]. As such, feature selection / ranking raises the possibility of reducing the negative effect of the irrelevant features on a given learning task, potentially speeding up the learning process significantly.

Feature ranking methods may be categorized into three groups [19]: *Filter*, *Wrapper* and *Embedded* methods:

- *Filter* methods are independent of the learning process. They typically use statistical information to rank features. Common evaluation metrics include:
  - *Distance measure*: based on the geometrical (or Euclidean) distance between feature vectors.
  - *Information measure*: based on entropy. Entropy is a measure of information contents and allows finding features which carry valuable information.
  - *Dependency measure*: considers the dependency between a feature and the class labels.
  - *Inconsistency measure*: evaluate features with a matched pattern, but different class labels.
- *Wrapper* methods represent a class of models that are "wrapped around" a classifier. They provide a feature set to work with and collect valuable feedback related to the learning process. These methods measure the *classification error* and attempt to correct the feature set. The most commonly used wrapper methods are: *Sequential Forward Selection* and *Sequential Backward Selection*.
- *Embedded* methods integrate the learning algorithm with the feature ranking step. In this model, both algorithms work tightly together to improve performance. For instance, SVM/RFE uses Support Vector Machine (SVM) ranking feature weights to eliminate the least important feature each at each iteration of the model [16].

It is important to note, that our model uses "feature quality" information to bias the rule discovery component of XCS (detail in section 4). In other words, our model uses a feature ranking method to improve the accuracy and speed of the learning, but does not remove any features from the actual classification process.

3 Related work

It is well documented in the evolutionary computation literature that the implementation of the genetic operators can influence the trajectory of the evolving population. However, there has been a paucity of studies focussed specifically on the impact of selected evolutionary operator implementations in LCS. We briefly discuss some of the key studies related to XCS/LCS below.

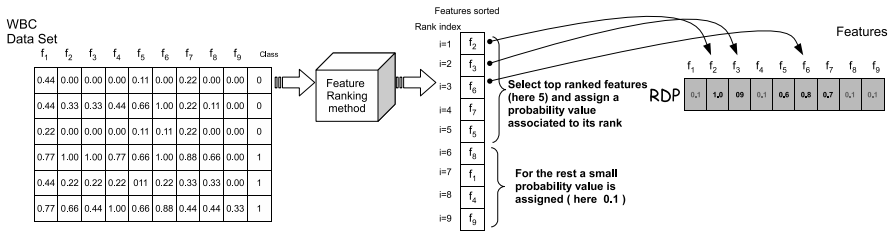


Fig. 2: Information Gain is used to rank the features. The top Ω features (in this example $\Omega = 5$) are selected and allocated relatively large probability values $\in [\gamma, 1]$. The RDP vector maintains these values. The highest ranked feature value is set to 1.0. Other features receive smaller values relative to their rank (in this example γ =0.5). Features that are not selected based on information gain are assigned very small probability values (in this example ξ = 0.1).

In one of the first studies focussed on the rule discovery component specifically for XCS, Butz et al. [11] have shown that uniform crossover can ensure successful learning in many tasks. In subsequent work, Butz et al. [12] introduced an informed crossover operator, which extended the usual uniform operator such that exchanges of effective building blocks occurred. This approach helped to avoid the over-generalization phenomena inherent in XCS [23]. In other work, Bacardit et al. [7] customized the GAssist crossover operator to switch between the standard crossover or a new simple crossover, SX. The SX operator uses a heuristic selection approach to take a minimum number of rules from the parents (more than two), which can obtain maximum accuracy. Morales-Ortigosa et al. [25] have also proposed a new XCS crossover operator, BLX, which allowed for the creation of multiple offspring with a diversity parameter to control differences between offspring and parents. In a more comprehensive overview paper, Morales-Ortigosa et al. [26] presented a systematic experimental analysis of the rule discovery component in LCS. Subsequently, they developed crossover operators to enhance the discovery component based on evolution strategies with significant performance improvements.

Other work focussed on biased evolutionary operators in LCS include the work of Jos-Revuelta [21], who introduced a hybridized GA - Tabu Search (GA-TS) method that employed modified mutation and crossover operators. Here, the operator probabilities were tuned by analyzing all the fitness values of individuals during the evolution process. Wang et al. [33] used *information gain* as part of the fitness function in a GA. They reported improved results when comparing their model to other machine learning algorithms. Recently, Huerta et al. [10] combined *linear discriminant analysis* with a GA to evaluate the fitness of individuals and associated discriminate coefficients for crossover and mutation operators.

Moore et al. [24] argued that the biasing of the initial population, based on expert knowledge preprocessing, should lead to improved the performance of the evolutionary based model. In their approach, a statistical method, *Tuned ReliefF*, was used to determine the dependencies between features to seed the initial population. A modified fitness function and a new guided mutation operator based on features dependency was also introduced, leading to significantly improved performance.

4 Model

The motivation behind the design and development of the GRD-XCS was to improve classifier performance especially for high-dimensional classification problems. Our goal was to make the overall task computationally faster, without degrading accuracy. To meet this goal, GRD-XCS introduces a probabilistically guided rule discovery mechanism for XCS.

In GRD-XCS, information gathered from a nominated feature ranking method is used to build a probability model that biases the evolutionary mechanism of the XCS. The feature ranking probability distribution values are recorded in a Rule Discovery Probability vector (RDP). Each value of the ***RDP*** vector ($\in [0, 1.0]$) is associated with a corresponding feature. The RDP vector is then used to bias the feature-wise uniform *crossover*, *mutation*, and *don't care* operators, which are part of the XCS rule discovery component. The GRD-XCS framework is not restricted to one specific feature ranking method. However, to clarify the functionality of our model, and to illustrate how RDP vector values can be calculated, we will use Information Gain (IG) as the feature ranking method initially. In section 7, the effectiveness of a range of alternative feature selection techniques will be examined.

The IG measure is defined as *entropy* reduction [20]:

$$IG = H(C) - H(C|f_i) \tag{1}$$

where H represent entropy, $F = \{f_0, f_1, ...f_i, ..., f_n\}$ is the feature set, and C the classes in this context. Entropy is a measure to quantify the information content, it is calculated using the formula:

$$H(C) = \sum_{j \in C} p_j \log_2 p_j \tag{2}$$

where p_j is the probability of having j in C, and the conditional entropy is calculated as:

$$H(C|f_i) = \sum_{j \in C} p_j \log_2 H(C|f_i = j) \tag{3}$$

The actual values in the RDP vector are calculated based on the IG values as described below:

$$RDP_i = \begin{cases} \frac{1-\gamma}{\Omega} \times (\Omega - i) + \gamma & \text{if } i \leq \Omega \\ \\ \xi & \text{otherwise} \end{cases} \tag{4}$$

where i represents the rank index in ascending order for the selected top ranked features Ω. The probability values associated with the other features are given a very low value ξ. Thus, all features have a chance to participate in the rule discovery process. However, the Ω-top ranked features have a greater chance of being selected (see Figure 2).

GRD-XCS uses the probability values recorded in the RDP vector in the pre-processing phase to bias the evolutionary operators used in the rule discovery phase of XCS. The modified algorithms describing the *crossover*, *mutation* and *don't care* operators in GRD-XCS are very similar to standard XCS operators:

- GRD-XCS *crossover* operator: The crossover operator is a hybrid uniform/n-point function. An additional check of each feature is carried out before exchange of genetic material. If $rand() < RDP[i]$ then feature i is swapped between the selected parents.
- GRD-XCS *mutation* operator: Uses the RDP vector to determine if feature i is to undergo a mutation, if the feature was randomly selected to be mutated.
- GRD-XCS *don't care* operator: In this special mutation operator, the values in the RDP vector are used in the reverse order. That is, if the feature i has been selected to be mutated and $rand() < (1 - RDP[i])$, then feature i is changed to # ("don't care").

The application of the RDP vector reduces the crossover and mutation probabilities for "uninformative" features. However, it increases the "don't care" operator probability for the same feature. Therefore, the more informative features (based on the IG measure in this case) should appear in rules more often than the "uninformative" ones.

Table 1: Data set details

Data Set	#Instances	#Features	Cross Validation	Reference
Low-dimensional data sets (UCI examples)				
Pima	768	8	10	[2]
WBC	699	9	10	[2]
Hepatits	155	19	10	[2]
Parkinson	197	23	10	[2]
High-dimensional data sets (Microarray DNA gene expression)				
Breast cancer	22	3226	3	[18]
Colon cancer	62	2000	10	[6]
Leukemia cancer	72	7129	10	[15]
Prostate cancer	136	12600	10	[30]

5 Experiments

We have conducted a series of independent experiments to verify if the guided rule discovery mechanism for XCS was able to evolve accurate classifiers. In particular, we wished to establish if our proposed model had statistically significantly improved accuracy values when compared to the standard XCS across a suite of benchmark classification problems. Detailed analyses of the effects of GRD-XCS parameter values and alternative feature ranking techniques were performed to evaluate the efficacy of the model.

5.1 Data sets

Eight different data sets — four low-dimensional data sets and four high-dimensional data sets — have been used in the experiments. The details of these data sets are reported in Table 1. The low dimensional data sets were selected from the UCI [2] machine learning repository. They are benchmark data sets and provide a base line to compare our model with other machine learning methods.

Four DNA microarray gene expression data sets were selected to represent the high-dimensional data sets. Gene expression profiles provide important insights into, and further our understanding of, biological processes. They are key tools used in medical diagnosis, treatment, and drug design [36]. From a clinical perspective, the classification of gene expression is an important problem and a very active research area. DNA microarray technology has advanced a great deal in recent years. It is possible to simultaneously measure the expression levels of thousands of genes under particular experimental environments and conditions [37]. However, the number of samples tends to be a much smaller than the number of genes (features). Consequently, the high dimensionality of a given data set poses many

statistical and analytical challenges, often degrading the generality of the machine learning methods.

5.2 Parameters

Two different parameter types must be considered in GRD-XCS. These are parameters specific to the underlying XCS model and parameters specific to the guided-rule discovery component of GRD-XCS.

For the base-XCS parameters in our model, we have used default values recommended in [13]. For the UCI low-dimensional data sets, the population size *pop_size* was set to 2000 individuals. For the gene expression high-dimensional data sets, a range of alternative population sizes were investigated (*pop_size*=500, 1000, 2000, 5000).

When calculating the *RDP* vector values, we have used IG for all experiments described in section 6. Alternative feature selection techniques are used in section 7. For the low dimensional data sets, all features were considered. That is, the number of top ranked features Ω was equal to the number of features in the data set. For the high dimensional data sets, a range of alternative top ranked features sizes were investigated ($\Omega = 32, 64, 128, 256$). In Equation 4, $\gamma = 0.5$ and $\xi = 0.1$.

In all experiments the number of iterations was set to 5000.

5.3 Implementation

GRD-XCS was implemented in C, based on the XCS code available from [1]. The WEKA package (version 3.6.1) [3] was used for feature ranking. WEKA implementations for the alternative machine learning algorithms listed in section 6.4 were also used.

All experiments were performed on the *VPAC*[2] Tango Cluster server. Tango has 111 computing nodes. Each node is equipped with two 2.3 GHz AMD based quad core Opteron processors, 32GB of RAM and four 320GB hard drives. Tango's operating system is the Linux distribution CentOS (version 5).

Statistical analysis was performed using the IBM SPSS Statistics (version 19) software.

[2] Victorian Partnership for Advanced Computing: www.vpac.org

5.4 Evaluation

For each scenario (parameter values—data set combination), we performed N-fold cross validation experiments over 100 trials (see Table 1). The average accuracy values for specific parameter combinations have been reported using the Area Under the ROC Curve – the AUC value. The ROC curve is a graphical way to depict the tradeoff between the *True Positive rate* (TPR) on the Y axis and the *False Positive rate* (FPR) on the X axis. The AUC values obtained from the ROC graphs allow for easy comparison between two or more plots. Larger AUC values represent higher overall accuracy.

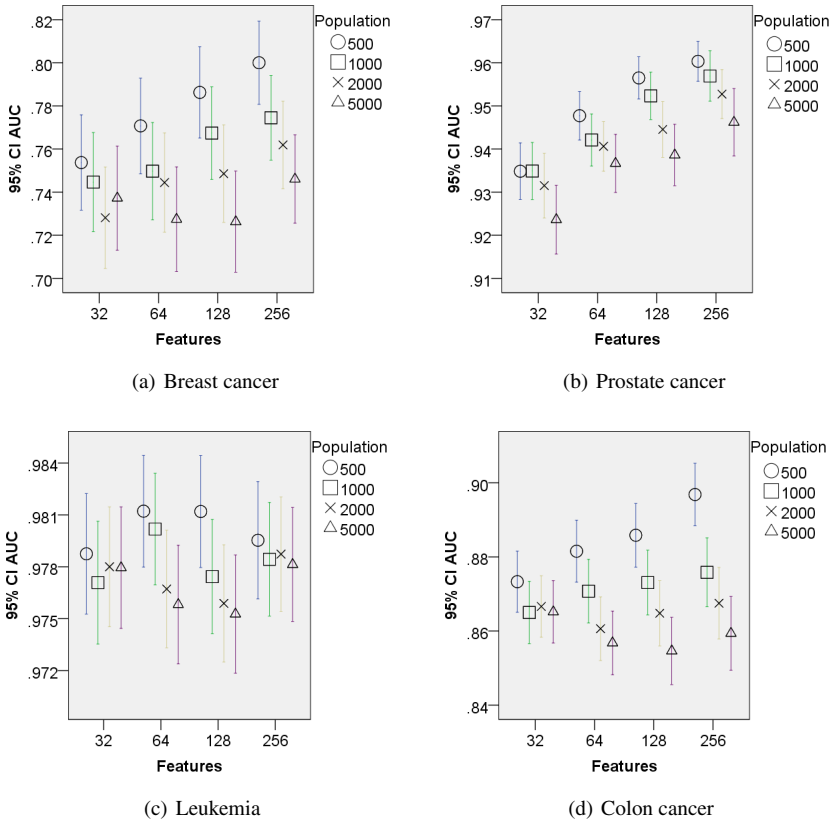
Appropriate statistical analyses using paired t-test and/or analysis of variance (ANOVA) were conducted to determine whether there were statistically significant differences between particular scenarios in terms of both accuracy and execution times. Scatter plots of the observed and fitted values and Q-Q plots were used to verify normality assumptions.

6 Results

The experimental results using the IG feature ranking technique have been divided up into four sections to highlight key attributes of the GRD-XCS model. In section 6.1, an analysis of the accuracy (AUC) values is presented for different population sizes and Ω values. In section 6.2, we analyze execution times for different population sizes and Ω values. In section 6.3, the population diversity, measured in terms of the number and length of macro classifiers in the final population, is examined. Finally, in section 6.4, we compare the performance of GRD-XCS with other well-known machine learning classifiers.

6.1 Accuracy analysis

Figure 3 plots the mean of the AUC values with 95% confidence interval for the GRD-XCS model on the four high-dimensional data sets for various combinations of *pop_size* and Ω. It is difficult to draw firm conclusions directly from the plots. For both the Breast cancer and Prostate cancer data sets, an increase in the Ω value results in improved AUC values. However, the trend is not as pronounced for the other high-dimensional data sets. Interestingly, the smaller population size of 500 typically produced the higher AUC values.



(a) Breast cancer (b) Prostate cancer

(c) Leukemia (d) Colon cancer

Fig. 3: AUC results for GRD-XCS.

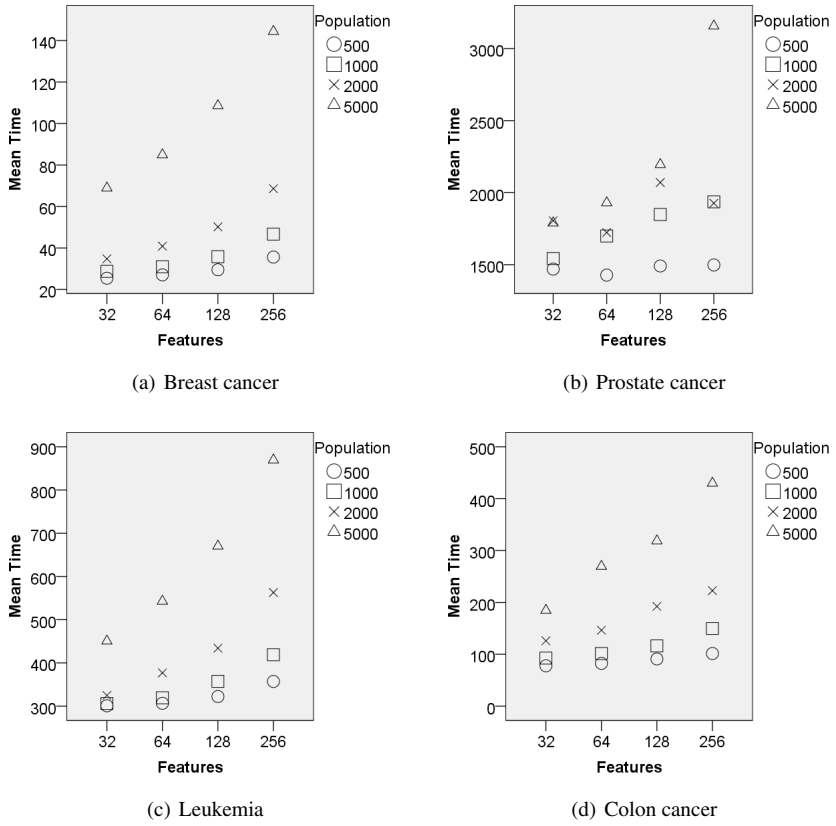
Table 2: ANOVA test results: Ω value is fixed — *pop_size* variable. For example for the Breast cancer data set, when $\Omega = 32$ there is no significant difference ($p = 0.465$) between population sizes. In contrast, when $\Omega = 128$ the results are significantly different ($p = 0.001$).

	Top ranked features (Ω)			
Data set	**32**	**64**	**128**	**256**
Breast cancer	0.465	0.073	0.001	0.002
Prostate cancer	0.102	0.075	<0.001	0.011
Leukemia	0.932	0.067	0.062	0.944
Colon cancer	0.465	<0.001	<0.001	<0.001

In Tables 2 and 3, the result of statistical analysis using ANOVA tests are listed ($F_{3,1196}$ for Breast cancer and $F_{3,3996}$ for other data sets). The null hypothesis tested was that there was no significant difference in accuracy values between particular configurations for *pop_size*= 500, 1000, 2000, 5000 and $\Omega = 32, 64, 128, 256$ when one of the *pop_size* or the Ω values was fixed and the other parameter varied. An inspection of the values in Table 2 show

Table 3: ANOVA test results: *pop_size* is fixed — Ω value variable. For example for the Breast cancer data set, when *pop_size*=500 there is a significant difference ($p = 0.017$) between the Ω values. In contrast, when *pop_size*=2000 there is no significant difference ($p = 0.214$).

	Population Size (*pop_size*)			
Data Set	**500**	**1000**	**2000**	**5000**
Breast cancer	0.017	0.176	0.214	0.600
Prostate cancer	<0.001	<0.001	<0.001	0.001
Leukemia	0.664	0.573	0.735	0.885
Colon cancer	0.001	0.363	0.711	0.426

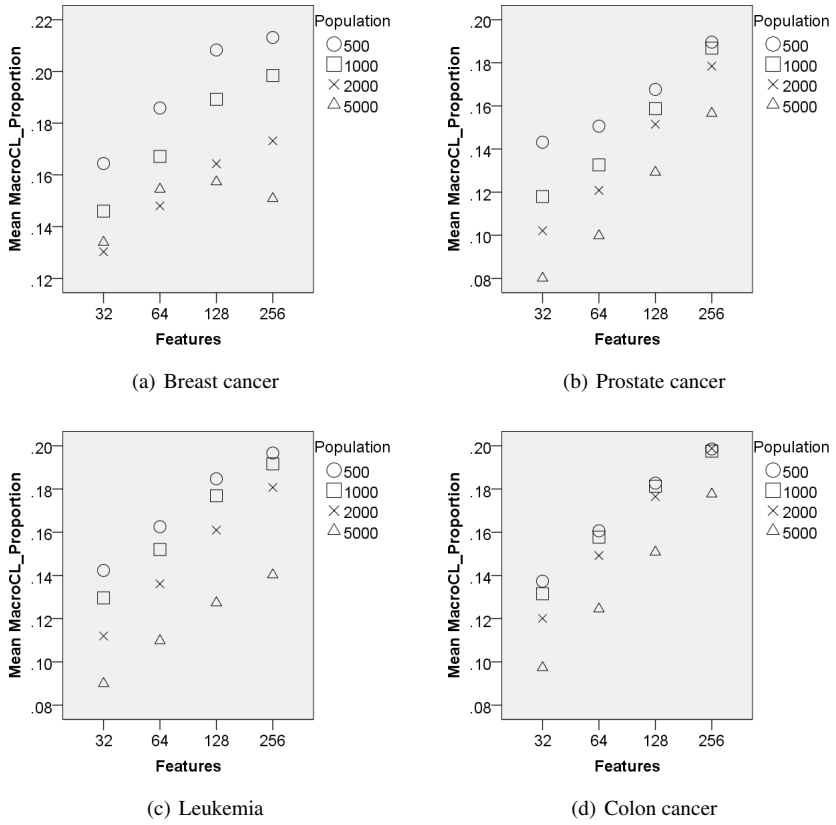


(a) Breast cancer (b) Prostate cancer

(c) Leukemia (d) Colon cancer

Fig. 4: Execution time results for GRD-XCS.

that when the population size was varied, significant differences existed between the accuracy values recorded when $\Omega = 128$ and higher for three of the data sets. Table 3 shows that significant differences existed between the accuracy values recorded for the small population size (500) for three of the data sets when Ω was varied. A Tukey HSD post-hoc comparison test suggests that for small changes in the configuration (either *pop_size* or Ω), the accuracy



(a) Breast cancer (b) Prostate cancer

(c) Leukemia (d) Colon cancer

Fig. 5: The proportion of macro classifier in the final population.

levels of the GRD-XCS are not significantly different. A linear regression analysis reveals that the Ω parameter has a positive coefficient and the population size has a negative coefficient. Here, the Ω coefficient is 10 times larger than the other coefficient. This suggests that smaller population sizes with relatively larger Ω values should produce the best results. This observation is consistent with the plots in Figure 3.

It is interesting to note that there were no significant differences between configuration examined for the Leukemia dataset. In the following sections, we will see that the Leukemia data set is relatively easy to classify, not only by GRD-XCS but also with other machine learning methods. Therefore, changing the GRD-XCS configuration does not have any significant effect on the accuracy values obtained.

6.2 Execution time analysis

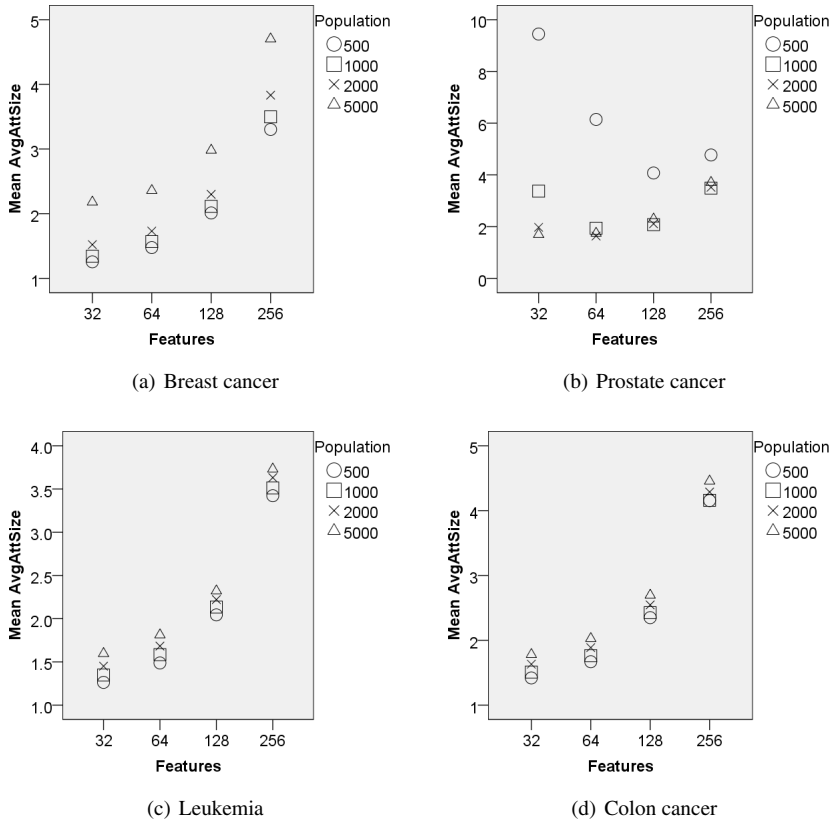
The execution time of a GBML classifier system is a performance metric that depends on many factors, including the hardware specification, the number of records in the data set, and the number of features in the data set. Here, we restrict our analysis to a comparison of different GRD-XCS parameters (population size and Ω) for each of the data sets using the same hardware.

Figure 4 shows the average GRD-XCS model execution time over 100 trials for each of the configurations examined. Error bars have been omitted for clarity. As expected, an increase in the value of Ω and/or population size slows the learning process down. This may be attributed to increases in the execution time of the *GetMatchSet()* function and the number of operations performed as part of the rule discovery component (that is, as Ω increases in value more features are involved in the crossover and mutation operators). The large population size means that there are more classifiers in the population to be processed leading to an increase in execution time.

Statistical tests (ANOVA tests and Tukey HSD post-hoc comparisons) show that execution time of all configuration are significantly different ($p < 0.001$). The regression analysis for execution time indicates that the Ω coefficient is 10 times larger than the population size coefficient. If the Ω values are not restricted to relatively small values, execution time grows very fast. Based on this observation, and the accuracy results listed in section 6.1, a Ω of 128 appears to be a good choice for most high-dimensional data sets.

6.3 Population diversity analysis

Population diversity is often seen as an important indicator of evolutionary algorithm performance. Finding the right balance between population diversity and convergence speed is an on-going research challenge when confronted with complex search and optimization problems. For GRD-XCS, population diversity can be measure by the number of unique classifiers — the macro classifiers — in the population. Figure 5 plots the proportion of macro classifiers in the final population averaged across 100 trials for different scenarios. Errors bars have been omitted for clarity. As expected, as the value of Ω increase the proportion of macro classifiers, and thus population diversity, increases. Population diversity tends to decrease based on this metric for larger population sizes for fixed Ω values.



(a) Breast cancer (b) Prostate cancer

(c) Leukemia (d) Colon cancer

Fig. 6: The average length of the macro classifier in the final population.

Given the role of the *don't care* (#) feature encoding in XCS, the length of the macro classifiers provides further insights into algorithm performance. Figure 6 plots the mean length (labelled as Mean Avg Att Size) of macro classifiers in the final population averaged across 100 trials for different scenarios. As the value of Ω increases, the mean length of the macro classifiers increases. Generally, an increase in the population size results in the evolution of small macro classifiers, although this finding is not always statistically significant across the high-dimensional data sets.

One observation that can be made based on the plots is that an increase the length of the macro classifiers does not necessarily improve the accuracy, but it certainly will increase the execution time.

6.4 Comparing GRD-XCS with other machine learning techniques

In this subsection, the performance of the GRD-XCS model in terms of accuracy is compared with a number of well established machine learning methods. The WEKA toolkit with default implementation was used to generate results for these methods. The results for the GRD-XCS were generated using a *pop_size*=500 and $\Omega = 128$. All other parameters were default values listed in section 5.2

Tables 4 and 5 list accuracy results for the low-dimensional and high-dimensional respectively. The bold value in each column indicates the highest mean accuracy value over all trials. The †symbol indicates that the result for the classifier listed in the row was significantly better than the GRD-XCS result based on a paired t-test ($p < 0.05$).

For the low-dimensional data sets, the results show that GRD-XCS is more accurate than standard XCS, although this difference was not always statistically significant. However, it is a different story when GRD-XCS is compared to other machine learning methods. The performance of GRD-XCS was best only for one data set, the Parkinson data set.

In contrast, for the high-dimensional data sets the results for GRD-XCS were significantly better than the other machine learning methods based on paired t-tests. A direct comparison between GRD-XCS and the standard XCS clearly illustrates that the guided rule discovery mechanism leads to improved performance.

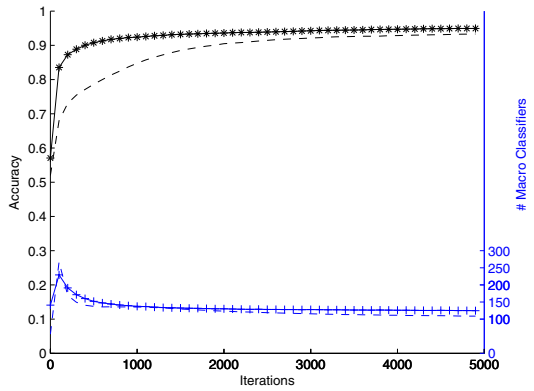
To further explore the efficacy of the our guided rule discovery enhancements, Figure 7 plots time series performance values. Here, the overall accuracy and the number of macro

Table 4: AUC results for low-dimensional data sets.

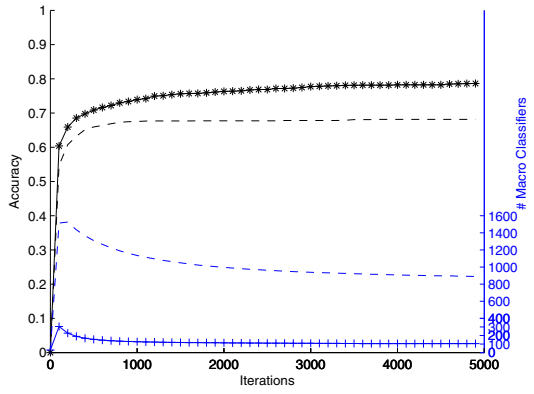
Classifier	Pima	WBC	Hepatit	Parkinson
j48	0.75 ± 0.01 †	0.94 ± 0.01	0.60± 0.04	0.78 ± 0.03
SVM	0.71 ± 0.01 †	0.96± 0.01	0.75 ± 0.02	0.75 ± 0.01
Naive Bayes Classifier	**0.81 ± 0.01** †	**0.98 ± 0.01** †	**0.84± 0.01** †	0.85 ± 0.01
NBTree	0.80 ± 0.01 †	**0.98 ± 0.01** †	0.76± 0.03	0.88 ± 0.02
One Rule	0.65 ± 0.01	0.90 ± 0.01	0.56± 0.02	0.77 ± 0.01
Random Forest	0.79 ± 0.01 †	**0.98 ± 0.01** †	0.81± 0.02 †	**0.94 ± 0.01** †
XCS	0.70± 0.03	**0.98 ± 0.01** †	0.81 ± 0.11 †	0.93 ± 0.08
GRD-XCS	0.72± 0.03	**0.98 ± 0.01**	0.82 ± 0.13	**0.94 ± 0.07**

Table 5: AUC results for high-dimensional data sets.

Classifier	Breast Cancer	Colon Cancer	Leukemia	Prostate Cancer
j48	0.43 ± 0.09	0.76 ± 0.04	0.79 ± 0.03	0.79 ± 0.02
SVM	0.63 ± 0.06	0.81 ± 0.03	0.97 ± 0.01	0.91 ± 0.01
Naive Bayes	0.55 ± 0.02	0.64 ± 0.02	0.98 ± 0.01 †	0.58 ± 0.01
NBTree	0.66 ± 0.03	0.75 ± 0.05	0.97 ± 0.01	0.90 ± 0.01
One Rule	0.42 ± 0.05	0.66 ± 0.04	0.82 ± 0.02	0.81 ± 0.02
Random Forest	0.67 ± 0.09	0.82 ± 0.03	0.92 ± 0.02	0.88 ± 0.01
XCS	0.66 ± 0.12	0.74 ± 0.18	0.93 ± 0.11	0.83 ± 0.09
GRD-XCS	**0.79 ± 0.19**	**0.89 ± 0.14**	0.98 ± 0.01	**0.96 ± 0.05**



(a) Parkinson



(b) Breast cancer

Fig. 7: (Colour on line). Time series performance plots for accuracy (left y-axis, black lines) and the number of macro classifiers (right y-axis, blue lines) for the (a) Parkinson (low-dimensional) and (b) Breast cancer (high-dimensional) data sets. Results for the base line XCS model (dotted line) and the GRD-XCS model (solid line) are provided.

classifiers in the evolving population for both the GRD-XCS and the standard XCS for representative low-dimensional and high-dimensional data sets are provided. Space constraints preclude the inclusion of plots for all data sets. However, the general trends for other data sets are qualitatively similar. There is a correlation between the accuracy of the model and the number of macro-classifiers in the population for high-dimensional classification problems examined. As expected, the number of unique classifiers (individuals) in the population for both XCS and GRD-XCS decreases over time. However, GRD-XCS typically maintains a smaller number of macro classifiers.

7 Investigating the effects of alternative feature ranking methods

The results in section 6 were generated using Information Gain (IG) as the feature ranking method for the guided rule discovery component in our model. However, as indicated in section 4, the IG feature ranking method can be replaced by any alternative method that is able to provide a relative ranking of the "importance" of features and thus be used to build the *RDP* vector. In this section, we test the hypothesis that there is no significant difference in performance, measured in terms of classifier accuracy, between well-known feature ranking techniques when used to build the *RDP* vector in GRD-XCS.

In the following subsection, we present a brief description of the alternative feature ranking methods used to build the *RDP*. This is followed by a description of the experiments and results.

7.1 Feature ranking methods

The five additional feature rankings methods to be compared are:

- *Gain Ratio*: The Gain Ratio is an extension of the Information Gain metric [19]. It is defined as:

$$GR(C) = \frac{IG(C)}{SI(C)} \tag{5}$$

  where the split information $SI(C)$ is a normalized entropy measure, calculated by splitting the data sets (S) into v partitions. Here, v represents the distinct values for feature f_i, and S_i contains all samples of S if $f_i = a_j$. Split information is defined as:

$$SI(C) = \sum_{j=1}^{v} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|} \tag{6}$$

- *ReliefF*: This method is an extension of the popular *Relief* feature selection method. The basic idea in both algorithms is to adjust a weight vector for features, thereby selecting random sample points and computing their nearest neighbors. More weight is given to features that discriminate samples from neighbors of different class [22].
- *Correlation based* feature selection (CFS): CFS consider both the relation between features and class, and inter-correlation between features. The goal is to find a subset of features that are highly correlated with the predict the class label, as well as highly uncorrelated to each others [17].

- *Support Vector Machine* feature selection: In this approach, an SVM model is trained. Then in an iterative fashion, features associated with small weights are removed. The SVM is trained again with the remaining features. This process continues until all feature have been processed. A backward collecting stage is then used to create the rank list [29].
- *One Rule* is a very simple, yet accurate, classification method. This method is based on creating one rule for each feature. Then, the error rates of each rule can be used for ranking the features [19].

7.2 Experiments and result

Experiments were conducted to compare the accuracy values found by GRD-XCS when each of the alternative feature ranking methods was used. The GRD-XCS parameter values used in the experiments were based on the recommendation from section 6: *pop_size*=500 and $\Omega = 128$. The remaining GRD-XCS parameters were set to default values (see section 5.2). The four high-dimensional microarray gene expression data sets were used for comparisons purposes.

Detailed statistical tests were carried out to determine if there were significant differences between the scenarios considered. Table 6 lists the accuracy values obtained averaged over 100 trials for each of the feature ranking methods. The results of the ANOVA tests indicate that feature ranking method used has a significant impact on the accuracy values found using GRD-XCS for the Breast cancer, Prostate cancer and Colon cancer data sets ($F_{5,1794} = 37.113, p < 0.0001$; $F_{5,5994} = 9.077, p < 0.0001$; $F_{5,5994} = 9.608, p < 0001$ respectively). However, the results from the Leukemia data set ($F_{5,1794} = 1.079, p < 0.370$) were not significantly different. Given the high accuracy levels obtained for the Leukemia data set, simply by using a feature selection method it is reasonable to expect an improvement in performance.

Tukey HSD post-hoc comparison tests were performed for each scenario. For the Breast cancer data set, IG, CFS, ReliefF, and SVM have similar performance levels ($p < 0.05$). The results also indicate that ReliefF, SVM, and Gain Ratio have similar performance levels ($p < 0.05$). For the Prostate cancer data set, the only feature ranking technique that was statistically significantly different to the other techniques was the CFS method. The results of the test for Colon data sets suggests that the six different feature ranking methods can be categorized into three subsets: CFS and SVM as the first subset; SVM, Information Gain,

Table 6: AUC feature ranking methods results.

Data set	method	Mean of accuracy	$\%95CI$	$Rank$
Breast Cancer	Information Gain	0.79	[0.77 0.81]	1
	Gain Ratio	0.74	[0.72 0.76]	5
	ReliefF	0.77	[0.75 0.79]	3
	Correlation Based	0.78	[0.76 0.80]	2
	SVM	0.76	[0.74 0.78]	4
	One Rule	0.63	[0.61 0.64]	6
Prostate Cancer	Information Gain	0.96	[0.95 0.96]	=2
	Gain Ratio	0.95	[0.95 0.96]	3
	ReliefF	0.97	[0.96 0.97]	=1
	Correlation Based	0.93	[0.92 0.95]	4
	SVM	0.97	[0.96 0.98]	=1
	One Rule	0.96	[0.96 0.97]	=2
Leukemia	Information Gain	0.98	[0.98 0.98]	=1
	Gain Ratio	0.98	[0.98 0.98]	=1
	ReliefF	0.98	[0.98 0.99]	=1
	Correlation Based	0.98	[0.98 0.99]	=1
	SVM	0.98	[0.97 0.99]	=1
	One Rule	0.98	[0.97 0.98]	=1
Colon Cancer	Information Gain	0.89	[0.88 0.89]	=2
	Gain Ratio	0.89	[0.89 0.90]	=2
	ReliefF	0.89	[0.88 0.90]	=2
	Correlation Based	0.86	[0.85 0.87]	4
	SVM	0.87	[0.86 0.88]	3
	One Rule	0.90	[0.89 0.91]	1

and ReliefF as the second subset and Information Gain, ReliefF, Gain Ratio, and One Rule as the third subset.

This detailed statistical analysis suggests that the effectiveness of the feature ranking method within the GRD-XCS rule discovery component is somewhat dependent on the underlying characteristics of the data set. There is no one specific feature ranking method that always provides the best result. The use of any feature ranking method generally leads to an improvement in the performance of GRD-XCS when compared to the base-line XCS model. In Table 6, each of the feature selection techniques has been allocated a relative rank for each data set based on the 95% confidence interval values. Across all data sets, IG has the highest overall rank (1.5), followed by ReliefF, SVM, One Rule, Gain Ratio, and CFS, respectively. A reasonable conclusion, based on this analysis, is that IG is an effective feature ranking method for GRD-XCS. IG is a simple, informative approach that can be used to create the probability model (the RDP vector) for biasing the evolutionary operators in XCS.

8 Conclusion

In this paper, we have introduced a novel guided rule discovery component for XCS specifically designed to tackle high-dimensional classification problems. Here, a filtering or feature

ranking process is applied to build a probabilistic model. This probability distribution is then used to bias the evolutionary operators in the underlying XCS model.

Comprehensive numerical simulations have shown that our guided rule discovery mechanism improves the performance of XCS in terms of accuracy, execution time and more generally in terms of classifier diversity in the population. Detailed statistical analysis of the results suggests that relatively small population sizes coupled with an increasing number of top ranked features (Ω value) generally leads to high accuracy values. However, increased execution time is a direct negative effect of increasing the number of top features used in the *RDP* vector and/or the population size. A comparative study of alternative feature selection methods suggests that the use of any feature selection technique, given the right parameter values, leads to an improvement in performance. Clearly the quality of the information extracted using any feature ranking method is correlated with the underlying characteristics of the data set. However, the use of any additional information to bias the rule discovery evolutionary operators has been shown to be useful. We conclude that the use of IG as the feature selection method, with relatively small population sizes and a small number of top ranked features in comparison to the number of features in the data set, generally leads to the best accuracy values given constraints on computational time.

There are many avenues to explore in future work. The feature quality information extracted from a feature ranking method could be further refined. For example, a correlation detection method could be used to modify the *RDP* values for correlated features. In addition, specific domain knowledge (such as the identification of highly suspicious genes in microarray gene expression data sets) could provide an alternative feature quality information for biasing the evolutionary operators. There is also scope to investigate ways to reduce execution time based on parallel deployment of the GRD-XCS model.

References

1. The XCS source code in C is freely available on Illinois Genetic Algorithms Laboratory (IlliGAL) web site: http://illigal.org/category/source-code/.
2. UCI Machine Learning Repository: The Center for Machine Learning and Intelligent Systems at the University of California, Irvine. http://archive.ics.uci.edu/ml/.
3. Weka 3, is an open source data mining tool (in java), with a collection of machine learning algorithms developed by Machine Learning Group at University of Waikato. http://www.cs.waikato.ac.nz/ml/weka/.
4. M. Abedini and M. Kirley. A multiple population XCS: Evolving condition-action rules based on feature space partitions. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1 –8, july 2010.

5. M. Abedini and M. Kirley. Guided rule discovery in XCS for High-dimensional Classification Problems. In *Proceedings of 24th Australasian Artificial Intelligence Conference*, volume 7106 of *Lecture Notes in Artificial Intelligence*, 2011.
6. U. Alon, N. Barkai, D. A. Notterman, K. Gishdagger, S. Ybarradagger, D. Mackdagger, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. of the National Academy of Sciences of the USA*, 96:6745–6750, 1999.
7. J. Bacardit and N. Krasnogor. Smart crossover operator with multiple parents for a Pittsburgh learning classifier system. In *Proceedings of the 8th conference on GECCO*, pages 1441–1448. ACM, 2006.
8. J. Bacardit, M. Stout, J. D. Hirst, K. Sastry, X. Llorà, and N. Krasnogor. Automated alphabet reduction method with evolutionary algorithms for protein structure prediction. In D. Thierens, H.-G. Beyer, J. Bongard, J. Branke, J. A. Clark, D. Cliff, C. B. Congdon, K. Deb, B. Doerr, T. Kovacs, S. Kumar, J. F. Miller, J. Moore, F. Neumann, M. Pelikan, R. Poli, K. Sastry, K. O. Stanley, T. Stutzle, R. A. Watson, and I. Wegener, editors, *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, volume 1, pages 346–353, London, 7-11 July 2007. ACM Press.
9. J. Bacardit, M. Stout, N. Krasnogor, J. D. Hirst, and J. Blazewicz. Coordination number prediction using learning classifier systems: performance and interpretability. In M. Cattolico, editor, *Genetic and Evolutionary Computation Conference, GECCO 2006, Proceedings, Seattle, Washington, USA, July 8-12, 2006*, pages 247–254. ACM, 2006.
10. E. Bonilla Huerta, J. Hernandez Hernandez, and L. Hernandez Montiel. A new combined filter-wrapper framework for gene subset selection with specialized genetic operators. In *Advances in Pattern Recognition*, volume 6256 of *Lecture Notes in Computer Science*, pages 250–259. Springer Berlin / Heidelberg, 2010.
11. M. V. Butz, D. E. Goldberg, and K. Tharakunnel. Analysis and improvement of fitness exploitation in XCS: bounding models, tournament selection, and bilateral accuracy. *Evol. Comput.*, 11:239–277, September 2003.
12. M. V. Butz, M. Pelikan, X. Llorà, and D. E. Goldberg. Automated global structure extraction for effective local building block processing in XCS. *Evol. Comput.*, 14:345–380, September 2006.
13. M. V. Butz and S. W. Wilson. An Algorithmic Description of XCS. In *Advances in Learning Classifier Systems*, volume 1996/2001 of *Lecture Notes in Computer Science*, pages 267–274. Springer Berlin / Heidelberg, 2001.
14. A. Fernandndez, S. Garcianda, J. Luengo, E. Bernado-Mansilla, and F. Herrera. Genetics-based machine learning for rule induction: State of the art, taxonomy, and comparative study. *Evolutionary Computation, IEEE Transactions on*, 14(6):913 –941, 2010.
15. T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, and C. D. Bloomfield. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
16. I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Mach. Learn.*, 46:389–422, March 2002.
17. M. A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1998.

18. I. Hedenfalk, D. Duggan, Y. Chen, M. Radmacher, M. Bittner, R. Simon, P. Meltzer, B. Gusterson, M. Esteller, O. P. Kallioniemi, B. Wilfond, A. Borg, and J. Trent. Gene-Expression profiles in hereditary breast cancer. *N Engl J Med*, 344(8):539–548, February 2001.
19. E. F. Ian H. Witten. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, 2 edition edition, 2005.
20. M. N. Isabelle Guyon, Steve Gunn and L. Zadeh, editors. *Feature Extraction, Foundations and Applications*. Springer, 2006.
21. L. M. S. Jose-Revuelta. *A Hybrid GA-TS Technique with Dynamic Operators and its Application to Channel Equalization and Fiber Tracking*. I-Tech Education and Publishing, 2008.
22. I. Kononenko. Estimating attributes: Analysis and extensions of relief. In F. Bergadano and L. D. Raedt, editors, *Machine Learning: ECML-94, European Conference on Machine Learning, Catania, Italy, April 6-8, 1994, Proceedings*, volume 784 of *Lecture Notes in Computer Science*, pages 171–182. Springer, 1994.
23. P. L. Lanzi. A Study of the Generalization Capabilities of XCS. In T. Bäck, editor, *Proceedings of the 7th International Conference on Genetic Algorithms*, pages 418–425. Morgan Kaufmann, 1997.
24. J. H. Moore and B. C. White. Exploiting expert knowledge in genetic programming for genome-wide genetic analysis. In *PPSN*, volume 4193 of *Lecture Notes in Computer Science*, pages 969–977. Springer, 2006.
25. S. Morales-Ortigosa, A. Orriols-Puig, and E. Bernadó-Mansilla. New Crossover Operator for Evolutionary Rule Discovery in XCS. In *8th International Conference on Hybrid Intelligent Systems*, pages 867–872. IEEE Computer Society, 2008.
26. S. Morales-Ortigosa, A. Orriols-Puig, and E. Bernadó-Mansilla. Analysis and improvement of the genetic discovery component of XCS. *International Joint Conference on Hybrid Intelligent Systems*, 6:81–95, April 2009.
27. A. Orriols-Puig, J. Casillas, and E. Bernadó-Mansilla. Genetic-based machine learning systems are competitive for pattern recognition. *Evolutionary Intelligence*, 1:209–232, 2008. 10.1007/s12065-008-0013-9.
28. M. S. V. K. Pang-Ning Tan. *Introduction to Data Mining*. Addison-Wesley Longman Publishing Co., Inc., 2006.
29. J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schoelkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.
30. D. Singh, P. G. Febbo, K. Ross, D. G. Jackson, J. Manola, C. Ladd, P. Tamayo, and A. A. Renshaw. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1:203–209, 2002.
31. P. O. Stalph, M. V. Butz, D. E. Goldberg, and X. Llorà. On the scalability of xcs(f). In F. Rothlauf, editor, *GECCO*, pages 1315–1322. ACM, 2009.
32. S. Sumathi and S. N. Sivanandam. *Introduction to Data Mining and its Applications*, volume 29 of *Studies in Computational Intelligence*. Springer, 2006.
33. P. Wang, T. Weise, and R. Chiong. Novel evolutionary algorithms for supervised classification problems: an experimental study. *Evolutionary Intelligence*, 4(1):3–16, 2011.

34. S. W. Wilson. Classifier Fitness Based on Accuracy. *Evolutionary Computation*, 3(2):149–175, 1995. http://prediction-dynamics.com/.
35. S. W. Wilson. Get Real! XCS with Continuous-Valued Inputs. In P. L. Lanzi, W. Stolzmann, and S. W. Wilson, editors, *Learning Classifier Systems, From Foundations to Applications*, volume 1813 of *Lecture Notes in Computer Science*, pages 209–222. Springer, 1999.
36. F.-X. Wu, W. Zhang, and A. Kusalik. On Determination of Minimum Sample Size for Discovery of Temporal Gene Expression Patterns. In *First International Multi-Symposiums on Computer and Computational Sciences*, pages 96–103, 2006.
37. Y. Zhang and J. C. Rajapakse. *Machine Learning in Bioinformatics*. Wiley Series in Bioinformatics. 1'st edition, 2008.

Author/s:
Abedini, M;Kirley, M

Title:
An enhanced XCS rule discovery module using feature ranking

Date:
2013-06

Citation:
Abedini, M. & Kirley, M. (2013). An enhanced XCS rule discovery module using feature ranking. INTERNATIONAL JOURNAL OF MACHINE LEARNING AND CYBERNETICS, 4 (3), pp.173-187. https://doi.org/10.1007/s13042-012-0085-9.

Persistent Link:
http://hdl.handle.net/11343/283265