REUCS-CRG: Reduct based Ensemble of sUpervised Classifier System with Combinatorial Rule Generation for Data Mining

Essam Debie
School of Engineering and Information Technology
University of New South Wales
Australian Defence Force Academy, Canberra, Australia
E.Debie@adfa.edu.au

Kamran Shafi
School of Engineering and Information Technology
University of New South Wales
Australian Defence Force Academy, Canberra, Australia
K.Shafi@adfa.edu.au

Chris Lokan
School of Engineering and Information Technology
University of New South Wales
Australian Defence Force Academy, Canberra, Australia
C.Lokan@adfa.edu.au

ABSTRACT

This paper proposes REUCS-CRG, a Reduct-based Ensemble of sUpervised Learning Classifier Systems with Combinatorial Rule Generation, which is an extension to the classical sUpervised Classifier System (UCS). In REUCS-CRG we build a two-stage ensemble architecture to improve generalization in UCS. In the first-stage, rough set attribute reduction is used to generate a set of reducts with different attribute subspaces, and then a diverse subset of these reducts is selected to train an ensemble of base classifiers. New instances are sent to several UCS-CRGs for classification, which includes a combinatorial rule searching component based on differential evolution algorithm. In the second-stage, a fusion method is used to combine the classification results of individual UCS-CRGs into a final decision. Three combining method are used and their results are compared: simple majority voting, winner-takes-all, and median rule. Experiments on some benchmark data sets from the UCI repository have shown that REUCS-CRG has better performance and better generalization ability than the single UCS and other UCS extensions. It also produces comparable results with other supervised learning methods. The experiments did not show significant differences in the accuracy rates obtained by the three combination methods.

Categories and Subject Descriptors

I.2.6 [**Computing Methodologies**]: ARTIFICIAL INTELLIGENCE, Learning—*Concept learning*

Keywords

Learning Classifier System; Rough Set Theory; Ensemble Learning; Differential Evolution

GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.

1. INTRODUCTION

Learning Classifier Systems (LCSs) [Holland 1976] are online learning systems. The rules are usually represented in the traditional production system "IF state THEN action" form. The rule set is incrementally updated through the interaction with the environment and eventually improved by the action of evolutionary algorithms. One of the most successful LCSs is the sUpervised Classifier System (UCS) that has been introduced in [Bernado et al. 2003]. UCS's fitness is based on accuracy, computed as the percentage of correct classifications. This makes UCS explore the consistently correct classifiers and thus evolve only best action maps. LCSs in general and UCS in particular have shown competitiveness with respect to other widely-used machine learning techniques on data mining applications [Shafi et al. 2009, Bernado et al. 2003, Bacardit and Butz 2007]. Being evolutionary based, LCSs are inherently adaptive to the problem at hand. Moreover, LCSs are also competitive for mining large data sets due to their population-based architecture which permits their parallelization for use on supercomputing resources [Bull et al. 2007]. Last but not least, they can learn knowledge from imperfect data taken from the real environment.

Though several researches have shown that LCSs work well on data mining domain, there are still some problems that hinder its performance. For example, LCSs tend to overfit smaller data sets. Other problems include noisy data and the missing data which often takes place in real data sets. To mitigate the effects of these problems with LCS, we need to improve its performance in terms of generalization capabilities to avoid over-fitting and increase classification accuracy in general. It has been recently shown in the evolutionary computation literature that the implementation of the genetic operators can influence the flow of the evolving population [Bacardit and Krasnogor 2006, Morales-Ortigosa et al. 2008]. Butz et al. introduced a new crossover operator called informed crossover [Butz et al. 2006], which adapted the usual uniform operator such that exchanges of effective building blocks occurred. It was shown that this approach helped to avoid the over-generalization phenomenon inherent in XCS. Morales-Ortigosa et al. [Morales-Ortigosa et al. 2008] 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 subsequent work, they presented a systematic experimental analysis of the rule discovery component in LCS [Morales-Ortigosa et al. 2009]. Moreover, they developed crossover operators to enhance the discovery component based on evolutionary strategies with significant performance improvements.

In previous work [Debie et al. 2013a], the authors proposed a differential evolution (DE) based discovery component in UCS. Experiments conducted with three variations of the DE-based discovery component on synthetic and real data sets showed encouraging results in terms of generalization performance. The DE based discovery component led to more accurate solutions and sped up the search for maximally general and accurate rules.

On the other hand, ensemble method is one of the most interesting and successful learning systems with powerful generalization capabilities. Breiman [Breiman 2001] showed that generating multiple versions of a classifier and using them to get an aggregated classifier can improve the classification accuracy. Rough sets have been applied to a very wide variety of application domains with success, in particular to feature selection [Wang and Wang 2001]. Recently, the utility of rough set theory in constructing ensemble systems has been shown in the literature with success [Hu et al. 2007]. In previous work [Debie et al. 2013b] the authors proposed an explicit divide and conquer approach to evolve a robust ensemble of supervised learning classifier systems by detecting variable interactions that exist in a problem, and subsequently making use of this expert knowledge when partitioning the feature space. In this model, rough set attribute reduction was firstly applied on the training set, resulting in a set of reducts. Then, a subset of these reducts was chosen to train an ensemble of UCSs. Results on real data sets from the UCI repository showed generalization improvements over the traditional UCS.

In this paper, we propose a hybrid model, combining the strengths of both the directed DE-based rule searching and reduct based ensemble learning, in learning classifier systems. Theoretical and experimental research suggest that combining a set of accurate and diverse classifiers will lead to a powerful classification system [Hu et al. 2007]. Our hypothesis is that by employing rough set reducts it is possible to reduce the problem dimensionality and evolve robust classifiers. To add to that, a directed rule searching can speed up the search for maximally general and accurate classifiers and lead to more accurate solutions. Thus, overall ensemble performance is boosted by combining a set of accurate classifiers while diversity is also maintained by choosing variant reducts.

Our proposed model REUCS-CRG, a Reduct-based Ensemble of sUpervised Learning Classifier System with Combinatorial Rule Generation, is a two-stage ensemble architecture. In the first-stage, rough set attribute reduction is used to generate a set of reducts with different attribute subspaces, and then a diverse subset of these reducts is selected to train an ensemble of base classifiers. New instances are then sent to several UCS-CRGs for classification which includes a combinatorial rule searching component based on differential evolution algorithm. Then, the learnt knowledge (LK) from each learner is used to give a decision regarding testing samples. In the second-stage, a combining method is used to integrate the classification results of individual UCS-CRGs into a final decision. In this paper, we have experimented and compared results of three combining method namely: simple majority voting, winner-takes-all, and median rule. Our aim is to improve LCSs' performance in supervised learning problems with real-valued attributes. The performance of the proposed model is evaluated on nine real-world data sets taken from the UCI machine learning repository.

The remainder of this paper is organized as follows: Section 2 briefly describes the UCS algorithm. UCS with DE based discovery component algorithm is explained in Section 3, Section 4.2 introduces the construction of Reduct based UCS ensemble model. Experiments including experimental setup, data sets used, and results and discussions are presented in Section 5. Section 6 summarizes, concludes, and discusses future work.

2. DESCRIPTION OF UCS

Since the eXtended Classifier Systems (XCS) [Wilson 1995] was firstly introduced, a great amount of research has been conducted on accuracy-based LCSs, resulting in different LCSs with a core architecture inherited from XCS. One of the most prominent proposals is UCS, which inherits the main components of XCS, but specifies them for supervised learning tasks. UCS works as an on-line learner. For each input example x with its associated output, UCS forms the match set [M], which consists of all the classifiers in the population [P] with their condition matching x. The next steps depend on whether the system is in exploration (training) mode or exploitation (testing) mode. In exploration mode, the system creates the correct set [C] with all classifiers in [M] that advocate x, that is it has the same class as the example. If [C] is empty, the covering operator is triggered. It creates a new rule whose condition is generalized from x and which predicts its class. Then, the parameters of the all rules in [M] are updated depending on whether they predicted correctly. Eventually, a genetic algorithm is triggered on the correct set [C], creating two new classifiers by means of crossover and mutation. The offspring are introduced in the population, and other classifiers are deleted from the population if there is no room for the new rules. The combination of niched-based selection and population-based replacement is mainly responsible for the generalization pressure in UCS. In exploitation mode, each classifier in [M] emits a vote weighted by the fitness of the rule for the class it predicts. The most voted class is selected as the output.

3. DIFFERENTIAL EVOLUTION BASED RULE SEARCHING COMPONENT

The purpose of this section is to introduce a DE-based search component into UCS, with the aim of providing a more guided search toward maximally general and accurate classifiers.

3.1 Differential Evolution

Differential Evolution (DE) is a floating-point encoding evolutionary algorithm for global optimization over continuous spaces which can also work with discrete variables. Usually, a differential evolution algorithm is abbreviated as DE/x/y/z, where x denotes how the differential mutation base is chosen, y denotes the number of vector differences added to the base vector and z indicates the crossover

method. The most popular strategy, denoted by abbreviation DE/rand/1/bin, generates the point v by adding the weighted difference of two points and uses a binomial (uniform) crossover operator.

DE begins with a randomly initiated population of N D-dimensional real-valued vectors. Each vector, also known as genome/chromosome, forms a candidate solution to the multidimensional problem. Subsequent generations in DE are denoted by $g = 0, 1, ..., G$. At any generation g, the i^{th} vector of the population is represented as follows:

$$\vec{X}_{i,g} = [x_{1,i,g}, x_{2,i,g}, ..., x_{D,i,g}] \quad (1)$$

where D is the dimensionality of the problem.

Mutation with Difference Vectors.

In the DE literature, a parent vector from the current generation is called a target vector, a mutant vector obtained through the differential mutation operation is known as a donor vector, and finally an offspring formed by recombining the donor with the target vector is called a trial vector. In the DE/rand/1/bin form of DE, a donor vector is created for a target vector $X_{i,g}$ from the current population by sampling three other distinct vectors, say $\vec{X}_{r_1,g}$, $\vec{X}_{r_2,g}$, $\vec{X}_{r_3,g}$ randomly from the current population. The indices r_1, r_2, and r_3 are mutually exclusive integers randomly chosen from the range $[1, N]$, which are also different from the base (target) vector index i. These indices are randomly generated once for each mutant vector. Then the difference of any two of these three vectors is scaled by a scalar number F (selected within the the interval [0.4, 1] [Tusar and Filipic 2007]) and the scaled difference is added to the third one as follows:

$$\vec{V}_{i,g} = \vec{X}_{r_1,g} + F.\left(\vec{X}_{r_2,g} - \vec{X}_{r_3,g}\right) \quad (2)$$

Crossover.

To enhance the diversity of the population, a crossover operation comes into play after generating the donor vector through mutation. The donor vector $V_{j,i,g}$ exchanges its components with the target vector $\vec{X}_{i,g}$ under this operation to form the trial vector $\vec{U}_{i,g} = [u_{1,i,g}, u_{2,i,g}, ..., u_{D,i,g}]$.

Although various crossover methods are applicable, binomial (or uniform) crossover is widely used in differential evolution [Price et al. 2005].

In binomial crossover, the elements of the trial vector $U_{i,g}$ are chosen using the following formula:

$$\mathrm{U}_{j,i,g} = \begin{cases} V_{j,i,g} & \text{if } (rand_{i,j}\,[0,1] \;\leq\; Cr \text{ or } j \;=\; j_{rand}) \\ \mathrm{X}_{j,i,g} & \text{otherwise} \end{cases} \quad (3)$$

where $X_{i,g}$ is the target vector, $V_{i,g}$ is the donor vector, $U_{i,g}$ is the new offspring, $rand_{i,j}\,[0,1]$ is a uniformly distributed random number, j_{rand} is a random number chosen in the range $[1, D]$, and D is the number of problem length.

3.2 Combinatorial Rule Generation In LCS

There is a restriction on the population size to be at least four in most commonly known versions of DE. However, Combinatorial Sampling Differential Evolution (CSDE) algorithm [Iorio and Li 2011] has reduced this restriction to two individuals only, so that it has the capability to explore and find global optima using relatively smaller population sizes. In LCSs, on the other hand, population size is one of the critical factors that system performance depends on, especially in complex and high dimensional problems. In this section, we introduce a combinatorial rule searching component into UCS, with the aim of providing more guided search toward maximally general and accurate rules. In what follows, we first review the CSDE algorithm, then we introduce our model sUpervised Classifier System with Combinatorial Rule Generation (UCS-CRG).

3.2.1 Combinatorial Sampling Differential Evolution

In contrast with other DE variations, CSDE uses only two individuals X_i and X_r $(i \neq r)$ for a difference vector calculation. It does not have a crossover operator to enforce diversity while searching the problem space. Instead, two types of mutations are performed in this algorithm around the individual that is deemed to be better than the other; better here is described in terms of the fitness of solutions. The first type of mutation is called a C-sampling (correlated sampling) such that the vector difference and perturbation are in the same direction, around the better individual. For the purpose of explaining the operation of the algorithm suppose $X^{(i)}$ is better than $X^{(r)}$ in Figure 1. The point labeled by 4 and 5 corresponds to the point specified by Eqs. 4 and 5. Similarly, the point labeled by 6 and 7 correspond to the point specified by Eqs. 6 and 7. Both points 4,5 and 6,7 are correlated because they are in the same direction as the difference vectors and sampled with the same probability.

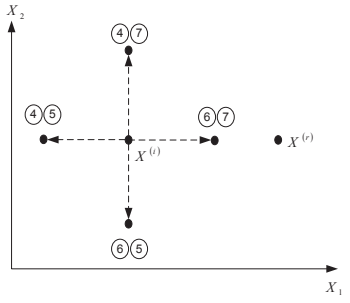

Figure 1: Offspring sampling in a two dimensional problems with CSDE

The second type of sampling is called UC-sampling (Uncorrelated sampling) and is labeled in Figure 1 by the points 4,7 and 6,5 which, respectively, correspond to the points generated by Eqs. 4 and 7 and Eqs. 6 and 5. Both of these samples are uncorrelated and it is this second type of sampling that contributes diversity to the search.

$$u_1^{(i)} = x_1^{(i)} + F\left(x_1^{(i)} - x_1^{(r)}\right) \quad (4)$$

$$u_2^{(i)} = x_2^{(i)} + F\left(x_2^{(i)} - x_2^{(r)}\right) \quad (5)$$

$$u_1^{(i)} = x_1^{(i)} + F\left(x_1^{(r)} - x_1^{(i)}\right) \quad (6)$$

$$u_2^{(i)} = x_2^{(i)} + F\left(x_2^{(r)} - x_2^{(i)}\right) \quad (7)$$

In CSDE, there are two pressures in the generation of offspring; exploitation results from the highly correlated rotationally invariant samples (C-samples) being generated, which rapidly drives the algorithm towards better solutions, and exploration occurs from the UC-sampling, which attempts to discover new and diverse points around the better individual. The general idea of this approach is to increase the diversity that DE is capable of generating using a relatively small population size.

3.2.2 sUpervised Classifier System with Combinatorial Rule Generation

In this model, which we refer to as UCS-CRG, the combinatorial rule searching algorithm is applied to $[C]_t$ by choosing two parent classifiers p_i, p_r $(i \neq r)$ using tournament selection. Mutation is then performed on the parent that is deemed to be better than the other in terms of macroclassifier fitness. Whether a correlated or uncorrelated rule generation occurs is determined probabilistically by a control parameter k. Choosing the best value for k along with a sensitivity analysis are explained in detail in [Iorio and Li 2011].

4. REDUCT BASED ENSEMBLE OF LEARNING CLASSIFIER SYSTEMS

4.1 Rough Set Theory

The idea behind Rough Set Theory is to approximate a given concept by two descriptive sets, called lower and upper approximations. The lower and upper approximations must be extracted from available training data. The main philosophy of rough set approach to concept approximation problem is based on minimizing the difference between upper and lower approximations (the boundary region).

The rough set concept can be defined quite generally by means of topological operations, interior and closure, called approximations. Data sets are usually given in the form of tables. A data set is a four-tuple $DT = (U, A, V, f)$, where U is the universe of objects x, cases, and A is a set of attributes. With every attribute $a \in A$ we associate a set V_a, of its values, called the domain of a. The union of all attributes' values is V. f is the information function $f : U \times A \rightarrow V$. The set of attributes is divided into condition attributes C and decision attributes D.

Any subset B of A determines a binary relation (indiscernibility relation) $IND(B) \subseteq U \times U$:

$$IND(B) = \{\langle x, y\rangle \in U \times U \mid \forall b \in B, f(x, b) = f(y, b)\} \quad (8)$$

The equivalence class induced by B for element x (denoted by $[x]_B$) is the subset of all elements $\in X$ that are equivalent to x according to B. It is referred to as an *elementary set*. The family of all equivalence classes induced by B, i.e. partitions determined by $IND(B)$, are denoted by $U/IND(B)$, or simply U/B.

Given a subset X of U, the concept X is approximated by two sets of elementary granules:

1. The lower approximation of a set X with respect to B is the set of all objects, which can be certainly classified as members of X with respect to B (are certainly X with respect to B).

$$\underline{B}\,(X) = \{[x]_B | [x]_B \subseteq X, x \in U\} \quad (9)$$

2. The upper approximation of a set X with respect to B is the set of all objects which can be possibly classified as X with respect to B (are possibly X with respect to B).

$$\bar{B}\,(X) = \{[x]_B | [x]_B \cap X \neq \emptyset, x \in U\} \quad (10)$$

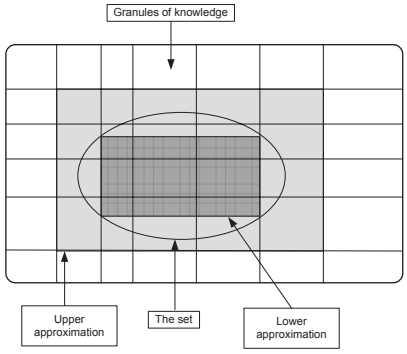

Figure 2: A graphical illustration of the set approximations.

A set X is called *crisp* (exact) with respect to B if and only if the lower approximation and upper approximation of X are the same. A set X is called *rough* (inexact) with respect to B otherwise. Assume C is the set of attributes and D is the decision in a given nonempty and finite universe U. C and D generate two partitions of the universe. The expression $POS_C(D)$, called a *positive region* of the partition U/D with respect to C, is the set of all elements of U that can be uniquely classified to blocks of the partition U/D by means of C. It is defined as follows:

$$POS_C\,(D) = \bigcup_{X \in U/D} \underline{C}\,(X) \quad (11)$$

Thus the concept of dependency of attributes is strictly connected with that of consistency of the decision table.

We will say that D *depends on* C in a *degree* k $(0 \leq k \leq 1)$, denoted $C \Rightarrow_k D$, where

$$k(C, D) = \frac{|POS_C(D)|}{|U|} \quad (12)$$

Given a decision table $DT = \langle U, C \cup D, V, f\rangle, B \subseteq C, b \in B$, we say attribute b is indispensable in B if $k_{(B-b)}(D) < k_B(D)$; otherwise, we say b is redundant. $B \subseteq C$ is independent if any b in B is indispensable. An attribute subset B is a reduct of the decision table if

1. $k_B(D) = k_C(D)$;
2. $\forall b \in B : k_B(D) > k_{B-b}(D)$

Thus a reduct is a set of attributes that preserves partition. It means that a reduct is a minimal subset of attributes that enables the same classification of elements of the universe as the whole set of attributes. In other words, attributes that do not belong to a reduct are superfluous with regard to classification of elements of the universe. Rough set theory admits that there exist multiple subsets of attributes which can keep the classifiability of the original data. They characterize the problem in different subspaces and therefore

capture different information of classification tasks. Reducts are considerably complementary to each other. Therefore, generalization power may be improved via combining a set of rough-set-based reducts.

4.2 REUCS-CRG

Our proposed system (REUCS-CRG) is a multiple classifier system, in which base classifiers are trained with attribute subsets generated by rough set attribute reduction. The structure of REUCS-CRG is shown in Fig 3. Given a training set, rough set attribute reduction is used to generate a set of reducts with different attributes, and then a diverse subset of these reducts is selected to train an ensemble of base classifiers. New instances are then sent to several UCS-CRGs for classification which includes a combinatorial rule searching component based on differential evolution algorithm. Then, in the second-stage, a combining method is used to fuse the classification results of individual UCS-CRGs into a final decision. From the perspective of rough set theory, reducts do not lose any essential information from the raw data. The idea is that classifiers trained with reducts will have a greater generalization power than those trained with either the whole set of attributes or randomly chosen subsets of attributes. Moreover, classifiers trained with different reducts are diverse, because they are trained in different attribute subspaces.

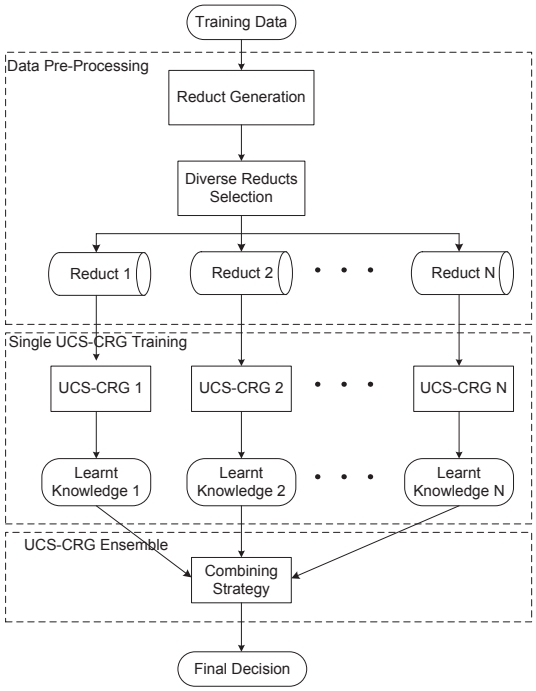

Figure 3: High level overview of the REUCS-CRG model. Each isolated sub-population evolves solutions based on a specific reduct using a separate UCS-CRG.

4.2.1 Generating Multiple Reducts

A number of methods for discovering reducts have already been proposed in the literature. The most popular methods are based on discernibility matrices, and information entropy. There are also approximation approaches to calculate reducts. Among these algorithms is the hitting set approach [Vinterbo and Øhrn 2000]. In the hitting set approach, non-empty elements of the discernibility matrix are chosen as elements of a multiset $\mathcal{L}$. The minimal hitting sets of $\mathcal{L}$ are exactly the reducts. Since finding minimal hitting sets is an NP-hard problem, GAs are used to find approximate hitting sets (reducts). Here we use the implementation of SAVGeneticReducer algorithm provided by the Rosetta toolkit [Komorowski and Ohrn 1997], which is based on the hitting sets approach, to generate multiple relative reducts.

We provide here a brief overview of the algorithm; for detailed information the reader is directed to [Vinterbo and Øhrn 2000]. Let O be a finite set of objects, and define a *attribute* a on O to be a function : $O \rightarrow V_a$ from O into an attribute value set V_a. Let $A \cup \{d\}$ be a set of distinct attributes on O such that the *decision attribute* d is not in A. An equivalence relation $\equiv_A$ on O can now be defined as follows:

$$x \equiv_A y \Leftrightarrow a(x) = a(y) \text{ for all a } \in A. \tag{13}$$

Then the *generalized decision attribute* d_A with respect to A is defined as

$$d_A(x) = \{d(y) | y \equiv_A x\}. \tag{14}$$

For $x, y \in O$ we further define

$$\mathcal{M}_A(x,y) = \begin{cases} \emptyset & \text{if } d_A(x) = d_A(y), \\ \{a \in A | a(x) \neq a(y)\} & \text{otherwise.} \end{cases} \tag{15}$$

Skowron and Rauszercalled this the *discernibility matrix* with respect to the decision attribute. Each entry contains the set of attributes that discern between two objects that are discerned by d_A. We can define a partition of O using $\mathcal{M}_A$. The equivalence classes of this partition can be defined as follows:

$$[x]_A = \{y | \mathcal{M}_A(x,y) = \emptyset\}. \tag{16}$$

A minimalization problem in *rough set theory* is to determine minimal sets of attributes that preserve a given partition of O. These minimal sets are called *reducts*. Taking non-empty elements of $\mathcal{M}_A$ to be the multiset $\mathcal{C}$, the minimal hitting sets of $\mathcal{C}$ are exactly the rough set reducts. An *r-approximate reduct* is defined to be an r-approximate hitting set of $\mathcal{C}$ constructed from $\mathcal{M}_A$. The algorithm fitness function f consists of two parts, and a weighed sum of both parts is taken: for each candidate solution B (reduct, or minimal hitting set) the following fitness function is defined:

$$\begin{aligned} f(B) &= (1-\alpha) \times \frac{cost(A) - cost(B)}{cost(A)} \\ &+ \alpha \times \min\left\{\mathcal{E}, \frac{|[S \in \mathcal{L} | \mathcal{S} \cap \mathcal{B} \neq \emptyset]|}{|\mathcal{L}|}\right\}. \end{aligned} \tag{17}$$

where α lies between 0 and 1, A is the set containing all the attributes, $\mathcal{L}$ is the set containing all elements S of the discernibility matrix, and the parameter $\mathcal{E}$ signifies a minimal value for the hitting fraction. Note that $\mathcal{E}$=1 implies proper minimal hitting sets. The first term rewards the shorter elements and the second tries to ensure that we reward sets that are hitting sets. The subsets B of A that are found through

the evolutionary search driven by the fitness function and that are good enough hitting sets (i.e., have a hitting fraction of at least $\mathcal{E}$) are collected in a keep list whose size is specified a priori. Different approaches can be used to define the *cost* function of an attribute subset; a trivial approach is to use the cardinality of candidate solutions $cost(B) =\mid B \mid$. Each reduct in the returned reduct set has a support count associated with it. The support count is a measure of the strength of the reduct.

Compared with random subspace method and attribute bagging, rough set attribute reduction presents a systematic method to get a set of attribute subsets that do not lose the distinguishing information in the original data. However, the former two methods randomly select attributes and the quality of the base classifiers is uncertain. Therefore, reduct-based ensembles have more opportunity to get good generalization than the random subspace method and the attribute bagging method.

4.2.2 Choosing Reducts for Ensemble Learning

Rough set attribute reduction results in hundreds or even thousands of reducts. Choosing the appropriate subset of reducts for ensemble learning is not a trivial task. In this section, we propose an algorithm that chooses N diverse reducts from the set of all reducts generated. The idea of the algorithm is to choose a first reduct randomly from the original list of reducts. Then, for each subsequent step choose a reduct which was not chosen before and is as diverse as possible to the list of chosen reducts so far. Diversity here is calculated as the inverse of the average similarity to other reducts, as shown in Equation 18:

$$SD_i = \frac{\frac{\sum_{j \in L} red_i \cap red_j}{red_i \cup red_j}}{L} \tag{18}$$

where red_i is the current reduct to calculate diversity for, L is the number of reducts chosen so far. After calculating the similarity degree for each available reduct we choose the one with the minimum amount of similarity $r_{best} \leftarrow \arg\min_{r \in RED} SD_i$.

4.2.3 Combining Methods

Three combination methods for determining the output of the ensemble have been investigated in REUCS-CRG. The first is majority voting. The output of the greatest number of individual UCS-CRG will be the output of the ensemble. The second is winner-takes-all. For each pattern of the testing set, the output of the ensemble is only decided by the individual REUCS-CRG whose output has the highest prediction weight. The third is median rule. The output of the ensemble is formed by a simple median of output of individual REUCS-CRG in the ensemble.

5. EXPERIMENTS

The objective of this section is to evaluate the generalization capabilities of our proposed system. A series of experiments was conducted to validate our approach. The underlying hypothesis was that classification using the REUCS-CRG model would lead to improved generalization for real-valued data sets.

We compared the performance of REUCS-CRG with the standard UCS; UCS with correlated rule generation (UCS-CRG); a reduct-based ensemble of UCS with correlated rule generation (REUCS). We also compared its performance with three other algorithms: Bagged C4.5, Boosted C4.5 and SVM.

We used a test suite of nine data sets from the University of California at Irvine (UCI) Machine Learning Repository. They are summarized in Table 1.

Table 1: Data Description — UCI Data Sets

Data set	No. Samples	Type of Attributes	No. Classes	No. Attributes
Breast Cancer	569	Real	2	30
Horse-colic	368	Nominal	2	22
Sonar	208	Real	2	60
Letter	20,000	Integer	26	16
Segment	2,310	Real	7	19
Soybean	683	Categorical	19	35
Hepatitis	155	Categorical, Integer, Real	2	19
Spectf	267	Integer	2	44
Wpbc	198	Real	2	33

5.1 Experimental Setup

All the data sets have been partitioned into two parts: a training and a testing set. The results shown are the average of a total of 10 runs. Student t-tests with a confidence interval of 95% is used to determine whether significant differences between the performance of standard UCS and the proposed model REUCS-CRG exist. The input data for the t-test is the test accuracy obtained in each of the 10 test sets that we have. We set UCS with standard values used in the literature for its configuration parameters as follows: $\beta = 0.2, \alpha = 0.1, \nu = 10, \theta_{GA} = 25, x = 0.8, \mu = 0.04, \theta_{del} = 20, \delta = 0.1, acc_0 = 0.999$. Tournament selection was applied. Subsumption was activated in the genetic algorithm with $\theta_{sub} = 20$, initial covering interval $r_0 = 0.6$, the population size $N = 6,400$. For UCS, we use two point crossover with $\chi = 0.8$, and bitwise mutation with $\mu = 0.04$. For UCS-CRG, we use mutation scaling factor $F = 0.8$ and crossover probability $Cr = 0.95$. Ensemble size is set to 10 classifiers for all data sets.

5.2 Results and Discussion

Table 2 shows the value of rough set attribute subset selection. It summarizes the attribute reduction and population size statistics for the nine data sets under consideration. For each data set, the original number of attributes is shown in the second column. The third column represents the average number of attributes used to train a single classifier in REUCS-CRG after attribute reduction, where the fourth column shows the percent of reduction achieved. In 5 out of the 9 data sets, the algorithm has identified more than 50% of the original attributes as superfluous and can be removed without affecting the system performance, while the minimum percent of reduction recorded in the Letter data set was 25%.

Table 3 shows the accuracy rates of different combining methods where the significantly best result for each data set is shown in bold. Except for Soybean and Wpbc data sets, comparison with the accuracy rates obtained by three

Table 2: Summary of attribute reduction statistics

Dataset	# Attributes	Avg # Attributes after reduction	% reduction
Breast Cancer	30	21.00	30.00
Horse-colic	22	8.50	61.36
Sonar	60	5.90	90.17
Letter	16	12.00	25.00
segment	19	8.10	57.37
soybean	35	19.40	44.57
Hepatitis	19	10.40	45.26
Spectf	44	8.20	81.36
Wpbc	33	9.60	70.91

Table 3: Accuracy rates of different combining methods

Data set	Majority voting	Winner-takes-all	Median
Breast Cancer	93.8 ± 0.9	94.5 ± 0.9	93.9 ± 0.8
Horse-Colic	86.0 ± 3.1	85.2 ± 1.5	87.3 ± 2.0
Sonar	66.6 ± 4.9	65.9 ± 5.7	65.9 ± 3.9
Letter	88.8 ± 0.4	87.3 ± 0.8	89.1 ± 0.7
Segment	97.7 ± 0.5	97.8 ± 0.5	97.9 ± 0.3
Soybean	59.7 ± 1.8	**74.0 ± 2.7**	59.2 ± 2.4
Hepatitis	84.1 ± 3.0	84.5 ± 2.2	81.8 ± 2.9
Spectf	81.8 ± 2.2	79.5 ± 3.3	80.8 ± 2.9
Wpbc	**87.9 ± 2.9**	84.6 ± 4.8	81.8 ± 4.4

combination methods shows similar performance on most of the problems. In Soybean, winner-takes-all outperformed median and majority voting. The justification for this better performance is that not all individuals are equally important. Because different individuals trained by different reducts, this gives an indication that not all reducts used with the Soybean data set were equally good.

Table 4 shows the results of the experiments conducted on nine real data sets studied in this paper. The best result for each data set is shown in bold. Significant results obtained by REUCS-CRG over standard UCS are marked by '*'.

In seven out of nine data sets, UCS-CRG performed better than or similar to the standard UCS. In four out of these seven data sets it significantly outperformed the standard UCS. Standard UCS outperformed UCS-CRG on Letter while it showed slightly better performance on the Wpbc data set. The mean reason for the poor performance of UCS-CRG my be traced back to the type of attributes of the problem since all the attributes of the Letter problem are integers.

It is also observable how the ensemble learning increased the generalization power of UCS as can be seen from the fourth column where the subspaces in which single classifiers were trained are defined by ten different reducts. REUCS model performed better than the standard UCS on eight data sets where the t-test showed significant accuracy differences in six of them.

The fifth column shows the fusion of ten classifiers using majority voting. The subspaces in which these classifiers were trained are exactly the same as those used by REUCS model and where combinatorial rule searching was used. As is shown, REUCS-CRG significantly outperformed standard UCS on eight data sets while the accuracy difference on the Hepatitis data set was not significant. In comparison to UCS-CRG, it is shown that ensemble learning with reducts has significantly improved the performance of classifiers with combinatorial rule generation in six out of nine data sets, where in the Breast-cancer, Soybean and Hepatitis data sts the accuracies were not statistically different. Compared to REUCS, the proposed model (REUCS-CRG) has shown similar or better improvements in all data sets with significantly better performance in four of them, while no significant differences were recorded on the remaining data sets. These results validate our hypothesis that combining a directed rule searching mechanism with ensemble learning can boost system performance on real-valued classification problems.

Comparisons to other non evolutionary algorithms are also shown in Table 4. The results of those algorithms were generated using the Weka package. Our proposed model showed competitive performance to these algorithms on most of the data sets under consideration. REUCS-CRG outperformed bagged and boosted decision trees on two data sets and was outperformed on three other data sets and produced statistically similar performance on the other data sets. An important note to be mentioned here is that the performance of the proposed model on domains with a high number of classes such as Letter and Soybean was significantly worse than boosted decision tree. One way to improve the performance is to increase the ensemble size; the number of single classifiers set to be more than the number of classes in the problem. Comparison with SVM showed that REUCS-CRG performed significantly better that SVM on three data sets while it showed significantly poorer performance on two data sets.

One drawback of the proposed model is that the run-time of the approach, from the creation of the reducts to the final classification, is long. Especially, it is much longer than that of the decision trees based approaches. However, obtaining improved classification with high interpretability (human readable rules) mitigates the additional time required.

6. CONCLUSION AND FUTURE WORK

In this paper, we have introduced REUCS-CRG, a two-stage ensemble structure of UCS in order to improve its generalization capabilities. Experiments with nine data sets from UCI showed that our model has achieved multiple objectives simultaneously. Firstly, it removed irrelevant attributes of the problem and reduced the dimensionality of the rule searching space. Secondly, it improved the discovery of new rules by adopting a directed searching mechanism based on differential evolution. Thirdly, classification accuracies has been significantly improved using reduct based ensemble learning. However, one drawback in the proposed model is that DE based rule discovery evolved relatively bigger population sizes to derive classifiers than the conventional algorithms. This is due to the principles of differential evolution which require creating many candidate solutions. Future work will consider this problem and propose solution for it. Comparison of the accuracy rates obtained by three combination methods did not show significant differences in the results obtained. Extended analysis on more real data

Table 4: Comparison of classification accuracies

Data set	UCS	UCS-CRG	REUCS	REUCS-CRG	Bagged C4.5	Boosted C4.5	SVM
Breast Cancer	79.3 ± 3.2	90.6 ± 4.2	90.7 ± 1.2	93.8 ± 0.9 *	95.7 ± 2.7	95.9 ± 3.4	**98.0 ± 1.4**
Horse-Colic	66.4 ± 3.2	76.5 ± 2.8	83.3 ± 2.1	**86.0 ± 3.1 ***	83.0 ± 6.2	80.3 ± 5.4	82.1 ± 5.4
Sonar	51.7 ± 0.0	51.7 ± 0.0	64.8 ± 5.9	66.6 ± 4.9 *	78.3 ± 5.7	**80.7 ± 8.2**	74.6 ± 6.6
Letter	76.2 ± 0.8	50.6 ± 1.1	88.8 ± 0.4	88.8 ± 0.4 *	92.6 ± 0.6	**95.1 ± 0.4**	82.1 ± 0.8
Segment	94.8 ± 1.2	96.1 ± 0.8	96.2 ± 0.8	97.7 ± 0.5 *	96.8 ± 0.8	**98.0 ± 0.6**	92.5 ± 1.2
Soybean	44.2 ± 2.0	60.1 ± 5.5	52.0 ± 1.9	59.7 ± 1.8 *	93.0 ± 3.0	93.8 ± 2.2	**94.9 ± 1.4**
Hepatitis	81.8 ± 0.0	82.3 ± 2.6	81.8 ± 0.0	**84.1 ± 3.0**	78.6 ± 6.4	82.2 ± 5.7	84.0 ± 6.7
Spectf	77.4 ± 2.4	78.9 ± 0.0	81.3 ± 4.0	**81.8 ± 2.2 ***	80.1 ± 5.4	79.0 ± 5.7	79.8 ± 1.2
Wpbc	78.9 ± 3.7	78.6 ± 0.0	81.1 ± 3.2	**87.9 ± 2.9 ***	79.1 ± 5.3	76.5 ± 5.4	79.8 ± 1.2

sets, especially high dimensional data sets will also be considered. The effectiveness of other fusion techniques such as the mixture of experts needs to be examined. The effectiveness of rule sharing among different populations is also a possible direction for future work.

7. REFERENCES

[Bacardit and Butz 2007] Bacardit, J. and Butz, M. V. 2007. Data mining in learning classifier systems: comparing xcs with gassist. Proceedings of the 2003-2005 international conference on Learning classifier systems.

[Bacardit and Krasnogor 2006] Bacardit, J. and Krasnogor, N. 2006. Smart crossover operator with multiple parents for a pittsburgh learning classifier system. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pp. 1441–1448. ACM.

[Bernado et al. 2003] Bernado, E., Mansilla, and Garrell-Guiu, J. M. 2003. Accuracy-based learning classifier systems: models, analysis and applications to classification tasks. *Evol. Comput. 11,* 3, 209–238. UCS.

[Breiman 2001] Breiman, L. 2001. Random forests. In *Machine Learning*, pp. 5–32.

[Bull et al. 2007] Bull, L., Studley, M., Bagnall, A., and Whittley, I. 2007. Learning classifier system ensembles with rule-sharing. *Evolutionary Computation, IEEE Transactions on 11,* 4, 496–502.

[Butz et al. 2006] Butz, M., Pelikan, M., Llorà, X., and Goldberg, D. 2006. Automated global structure extraction for effective local building block processing in xcs. *Evolutionary Computation 14,* 3, 345–380.

[Debie et al. 2013a] Debie, E., Shafi, k., Lokan, C., and Merrick, K. 2013a. Investigating differential evolution based rule discovery in learning classifier systems. In *Proceedings of IEEE Symposium Series on Computational Intelligence (IEEE SSCI 2013)*, Singapore. IEEE Press.

[Debie et al. 2013b] Debie, E., Shafi, k., Lokan, C., and Merrick, K. 2013b. Reduct based ensemble of learning classifier system for real-valued classification problems. In *Proceedings of IEEE Symposium Series on Computational Intelligence*, Singapore. IEEE Press.

[Holland 1976] Holland, J. H. 1976. Adaptation. In A. Press (Ed.), *Progress in Theoretical Biology IV*, pp. 263–293. Academic, New York.

[Hu et al. 2007] Hu, Q., Yu, D., Xie, Z., and Li, X. 2007. Eros: Ensemble rough subspaces. *Pattern Recogn. 40,* 12, 3728–3739.

[Iorio and Li 2011] Iorio, A. and Li, X. 2011. Improving the performance and scalability of differential evolution on problems exhibiting parameter interactions. *Soft Computing-A Fusion of Foundations, Methodologies and Applications 15,* 9, 1769–1792.

[Komorowski and Ohrn 1997] Komorowski, J. and Ohrn, A. 1997. RosettaŮŮa rough set toolkit for analysis of data. In *Fifth International Workshop on Rough Sets and Soft Computing, Tokyo, Japan*, pp. 403–407.

[Morales-Ortigosa et al. 2008] Morales-Ortigosa, S., Orriols-Puig, A., and Bernadó-Mansilla, E. 2008. New crossover operator for evolutionary rule discovery in xcs. In *Hybrid Intelligent Systems, 2008. HIS'08. Eighth International Conference on*, pp. 867–872. IEEE.

[Morales-Ortigosa et al. 2009] Morales-Ortigosa, S., Orriols-Puig, A., and Bernado-Mansilla, E. 2009. Analysis and improvement of the genetic discovery component of xcs. *International Journal of Hybrid Intelligent Systems 6,* 2, 81–95.

[Price et al. 2005] Price, K., Storn, R., and Lampinen, J. 2005. *Differential evolution: a practical approach to global optimization*. Springer-Verlag New York Inc.

[Shafi et al. 2009] Shafi, K., Kovacs, T., Abbass, H., and Zhu, W. 2009. Intrusion detection with evolutionary learning classifier systems. *Natural Computing 8,* 1, 3–27.

[Tusar and Filipic 2007] Tusar, T. and Filipic, B. 2007. *Differential Evolution versus Genetic Algorithms in Multiobjective Optimization*, Volume 4403 of *Lecture Notes in Computer Science*, Chapter 22, pp. 257–271. Springer Berlin Heidelberg.

[Vinterbo and Øhrn 2000] Vinterbo, S. and Øhrn, A. 2000. Minimal approximate hitting sets and rule templates. *International Journal of Approximate Reasoning 25,* 2, 123–143.

[Wang and Wang 2001] Wang, J. and Wang, J. 2001. Reduction algorithms based on discernibility matrix: The ordered attributes method. *Journal of Computer Science and Technology 16,* 6, 489–504.

[Wilson 1995] Wilson, S. W. 1995. Classifier fitness based on accuracy. *Evol. Comput. 3,* 2, 149–175.