Compact Rulesets from XCSI

Stewart W. Wilson

Department of General Engineering
The University of Illinois, Urbana-Champaign IL 61801, USA
Prediction Dynamics, Concord MA 01742, USA
wilson@prediction-dynamics.com

Abstract. An algorithm is presented for reducing the size of evolved classifier populations. On the Wisconsin Breast Cancer dataset, the algorithm produced compact rulesets substantially smaller than the populations, yet performance in cross-validation tests was nearly unchanged. Classifiers of the rulesets expressed readily interpretable knowledge about the dataset that should be useful to practitioners.

1 Introduction

Desirably, machine learning systems for data inference not only perform well on new data but permit the user to see and understand the "knowledge" that the system has acquired. In principle, learning classifier systems (LCS) have both properties. XCS [6], a recently developed LCS, shows good performance and knowledge visibility in many problem domains. Good performance results from evolution of rules (classifiers) that accurately predict reinforcement (payoffs, rewards) to the system, resulting in correct decisions (yes/no, turn left/turn right/go straight, etc.) that maximize the reinforcement the system receives. At the same time, the rules evolve to be as general as possible while still maintaining accuracy; they compactly express input domain regions over which the same decision is appropriate. Accurate, general rules are a natural and understandable way of representing knowledge to users. Thus XCS has both performance and knowledge visibility properties suitable for data inference.

XCS takes binary inputs, but many data inference problems involve integer attributes. XCSI is an adaptation of XCS for the integer domain, but the essential aspects of XCS are carried over. XCSI was recently applied [8][3] to the Wisconsin Breast Cancer (WBC) dataset [1] with performance results comparable to or exceeding those of other machine-learning methods. In addition, the evolved rules suggested interesting patterns in the dataset, especially when the evolutionary process was continued for very large numbers of random dataset instances. The rules were plausible medically, but knowledge visibility was impeded because there were more than 1000 distinct rules in all. It is natural to ask whether there exists a minimal subset of the classifiers that is sufficient to solve the problem, i.e. to correctly process all instances in the dataset. If so, how small—"compact"—would such a subset be and would its rules be relatively simple?

We present a technique that on the WBC problem reduces the evolved classifier set to approximately two dozen classifiers while maintaining performance on both training and test sets. The rules of this compact set appear to "make sense medically"[1] and may be amenable to use by practitioners in several ways, one of which is illustrated. The technique for deriving the compact rulesets may be extensible to other data domains.

The next section briefly discusses XCSI, the WBC dataset, and previous experiments. Section 3 introduces a *macrostate* concept, important for visualizing classifier populations and in forming compact rulesets. The Compact Ruleset Algorithm is presented in Section 4, followed in the next section by its application to the WBC dataset. A cross-validation experiment is reported in Section 6. Section 7 has conclusions.

2 XCSI and the Wisconsin Breast Cancer Problem

We used the "original" Wisconsin Breast Cancer Database which contains 699 instances (individual cases). Each instance has nine attributes such as Clump Thickness, Marginal Adhesion, etc. Each attribute has an integer value between 1 and 10 inclusive. Sixteen instances contain an attribute whose value is unknown. The outcome distribution is 458 Benign (65.5%), 241 Malignant (34.5%). The upper line of Figure 1 uses a graphic notation to illustrate a particular instance. Each interval between vertical bars represents an attribute. The attribute's value is indicated by the position of the star within the interval. For example, the value of the first attribute (which happens to be Clump Thickness) is 5, the second is 1, etc. The outcome of the instance is indicated by the digit at the end of the line, 0 for Benign and 1 for Malignant; this case was benign.

```
|    *     |*         |*         |*         | *        |*         |  *       |*         |*         |  0
|0000000...|0000000000|0000000000|00........|00........|0000000000|0000000000|0000000000|0000000000|  0 1000
```

Fig. 1. Example of an input data instance (upper line) matched by a classifier (lower line). See text for notation explanation.

The second line of Figure 1 shows a classifier that matches the data instance on the first line. The vertical bars separate attribute fields of the classifier's *condition*. Note that each field has exactly ten positions, corresponding to the ten possible attribute values. The string of "`0`" symbols in the first field indicates that that field matches the first attribute of an instance if the attribute's value is between 1 and 7, inclusive. Similarly, the second field matches *any* value for its attribute, the fourth matches values 1 or 2, and so on. A classifier matches an instance if and only if each of its fields matches the corresponding attribute value.

[1] Personal communication, 25 April 2001, William H. Wolberg, MD. Dr. Wolberg compiled the WBC dataset.

The last two numbers in the classifier are its *action* (its decision) and its payoff *prediction*, respectively. In the illustrated classifier, the action is 0 (benign); the prediction is 1000, which is the highest possible payoff, indicating that this classifier is probably always correct (its action is the same as the outcome of every instance it matches).

Note that the illustrated classifier is quite *general*: it potentially matches a large number of possible instances. At the same time, observe that the fourth and fifth fields are restricted to just two possible values, and the first field is somewhat restricted. The assertion made by this classifier can be read, "If Clump Thickness is less than 8 and Marginal Adhesion is less than 3 and Single Epithelial Cell Size is less than 3, then the case is benign".

Internally, XCSI represents the classifier's condition by a concatenation of *interval predicates* $int_k = (l_k, u_k)$ where l_k and u_k are integers and denote the lower and upper limits, inclusive, of the range represented in Figure 1 by the "O" strings. The action is an integer. The prediction is one of several estimates associated with the classifier; the others are *prediction error*, *fitness*, and *action set size*. Two further associated quantities are the classifier's *numerosity* and *experience*.

A full description of XCSI may be found in [8]. Information on basic XCS is in [6], [7], and in the updated formal description of [2] that XCSI follows most closely. The present work concerns a method for *post-processing* classifier populations already evolved by XCSI. We therefore omit a review of XCSI's mechanisms except for some observations relevant to the post-processing.

XCSI trains on a dataset like WBC roughly as follows. An instance is randomly selected from the dataset and presented to XCSI which forms a *match set* [M] consisting of classifiers that match the instance. The system then randomly picks an action from among those advocated by the matching classifiers and presents that action to a separate *reinforcement program* which knows the correct answer (outcome of the case) and gives the system a reward of 1000 if the system was right and 0 if it was wrong. The system then updates the prediction and other quantities mentioned above of all match set classifiers that advocated the selected action—the so-called *action set* [A]. Broadly, the update process is designed to associate high fitness with classifiers that, when they are in the action set, *accurately* predict the reward that will be received.

Each presentation of an instance is called a *time-step*; the training continues for a predetermined number of time-steps or until some other termination criterion is met.

On some time-steps, a genetic algorithm takes place in [A]. Since fitness depends on accuracy, over time the population comes to consist of increasingly accurate classifiers. In addition, however, the classifiers become more general while still remaining accurate. This is because classifiers that are more general tend to occur in more action sets, giving them more reproductive opportunities. Genetic operations are constantly varying classifiers' interval predicates; variations toward greater generality win out over variations toward greater specificity as long as the more general versions retain accuracy–i.e., don't start making mistakes.

Increasing population generality can be observed in Figure 2, taken from [8]. In this experiment, XCSI was trained on the full WBC dataset for 2 million training time-steps. To monitor the system's performance, the training time-steps (called *explore problems* on the graph) alternated with testing time-steps or *test problems*. On a test problem, XCSI made its action choice deterministically (i.e., it picked the action with the highest likely payoff), and a moving average of the fraction correct was plotted.

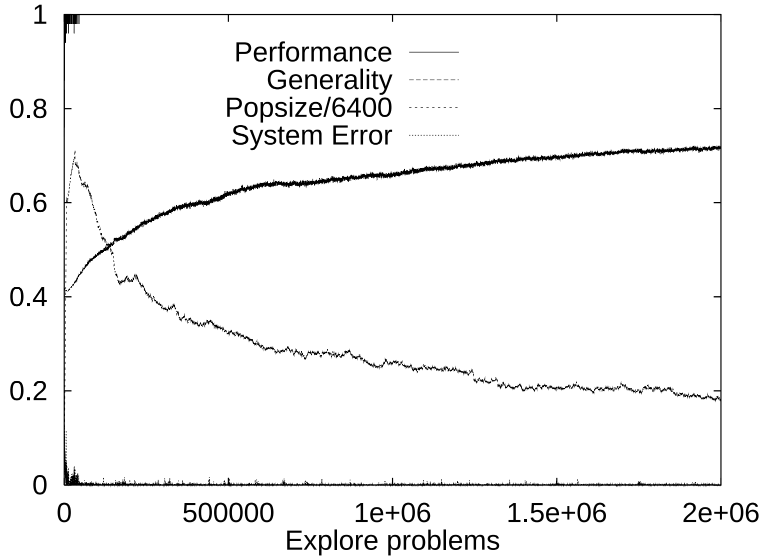

Fig. 2. Performance (fraction correct), generality, population size (/6400), and system error vs. number of explore problems in a training experiment on the Wisconsin Breast Cancer dataset.

Note that the performance curve reaches 100% early, at approximately 50,000 explore problems. The system error curve also falls quickly to a low level (system error measures the error in the system's expectation of payoff). The other two curves, however, develop over the whole experiment and are still changing at its end. The generality curve averages individual classifier generality over the whole population. Classifier generality is defined as the sum of the widths $u_k - l_k - 1$ of the interval predicates, all divided by the maximum possible value of this sum, in this case 90. Generality increases via genetic operations on the interval predicates as explained above. The result is to increase the widths of the interval predicates, some of which go to the full width of 10 and in effect remove the corresponding attributes from consideration in matching (they match unconditionally) and the classifier becomes simpler. Note that while generality steadily increases, the performance stays at 100%: the classifiers though more general are still accurate.

Corresponding to the generality increase, the population size decreases. Specialized classifiers are replaced by more-general ones, so fewer classifiers are needed. At the end, however, the population still has over 1000 classifiers in it, with extension of the experiment unlikely to produce much further improvement. The question arises: are all these classifiers really necessary (for high performance), or can a subset—ideally a *small* subset—be found that performs just as well? The remainder of the paper is devoted to providing an answer. In the next section we describe a simple technique for examining populations.

3 Macrostates

Classifier populations are basically unordered and no aspect of the operation of a classifier system depends on the order in which the classifiers occur. It may be that the classifiers actually form a list structure and those at the beginning were more recently generated than those further on. This makes no difference in principle, however, since the matching and GA processes are conceptually parallel. Nevertheless, as a tool for analysis, it can be useful to order the population according to certain classifier properties. This places classifiers of interest at the head of the list and brings out properties of the population as a whole. Such an ordering of the population is termed a *macrostate*[2].

As an example, consider a *numerosity macrostate* in which the classifiers are ordered in descending order of numerosity. A classifier that is often selected for the GA tends to attain high numerosity. When, as often happens under the GA, an offspring is unchanged from one of its parents (it was not mutated or crossed, or crossing had no effect), it does not enter the population as a separate classifier but the parent's numerosity is instead incremented by one. Thus if a classifier is frequently selected, its numerosity will grow more rapidly than that of less frequently selected classifiers. In general, the frequency of selection depends on fitness as well as the frequency with which a classifier occurs in an action set, which depends on its generality. Consequently, classifiers that are both accurate and maximally general will tend to have high numerosity and so will tend to appear at the top of the numerosity macrostate.

For both performance and knowledge visibility, we would regard accurate and maximally general classifiers as among the "best" in the population, so that calculating the numerosity macrostate is a way of finding these classifiers and bringing them to the fore. Other macrostates, for example based directly on fitness or on experience (the number of times a classifier has occurred in an action set) also tend to place "good" classifiers at the top. Not all orderings do this, however. For example, a generality macrostate will emphasize some highly general but *in*accurate classifiers that have not yet been eliminated from the population.

---

[2] There is a parallel with the macrostate concept of statistical mechanics; see [5].

Certain macrostates, then, tend to place the "best" classifiers at the top. In the next section, an algorithm is introduced that makes use of this fact to derive high-performance compact rulesets from the macrostates.

4 Compact Ruleset Algorithm

To state the algorithm it is useful first to define a macrostate $\mathbf{M}$ as an ordered set of classifiers c_i, $0 < i < M$, where i is an ordering index, the c are the members of a population [P], M is the total number of classifiers in [P], and the c_i are ordered according to a classifier property *prop* such that $i < j$ if $prop_{c_i} > prop_{c_j}$. For example, *prop* could be numerosity. $\mathbf{M}$ would then be in descending order of classifier numerosity, except that classifiers with equal numerosity would be adjacent in the ordering.

Define also a *sequential subset* $\mathbf{M}_n$ of $\mathbf{M}$ to be the subset consisting of all classifiers c_i such that $i \leq n$, $0 \leq n \leq M$. The nth sequential subset of $\mathbf{M}$ then consists of all classifiers starting with the *0*th (at the "top" of $\mathbf{M}$) down through the nth.

The basic idea of the algorithm is heuristic: it is assumed that classifiers closer to the top of a macrostate are "better" (in terms of performance and generality) than those further down. The algorithm has three steps.

1. Find the smallest $\mathbf{M}_n$ that achieves 100% performance when tested on the dataset D. Letting n^* equal the corresponding value of n, call this set $\mathbf{M}_{n^*}$.
2. Then, because there might have been some c_i such that the performance of $\mathbf{M}_i$ was not greater than the performance of $\mathbf{M}_{i-1}$, $0 < i < n^*$, eliminate such c_i from $\mathbf{M}_{n^*}$. Call the resulting set $\mathbf{B}$.
3. Now process $\mathbf{B}$ as follows. First find the classifier in $\mathbf{B}$ that matches the largest number of instances of D. Place that classifier in an initially empty set $\mathbf{C}$ and remove the matched instances from D. Again find the classifier of $\mathbf{B}$ that matches the most instances of D, add it to $\mathbf{C}$, and remove the matched instances from D. Iterate this cycle until D is empty. Call the final $\mathbf{C}$ a *compact ruleset* for $\mathbf{M}$, and denote it by $\mathbf{M}_{comp}$.

Steps 1 and 2 do most of the work; step 3 places the classifiers of $\mathbf{B}$ in descending order of marginal contribution with respect to D, which is one way of measuring their relative importance. In principle, step 3 can also result in $\mathbf{M}_{comp}$ being slightly smaller than $\mathbf{B}$.

On the basis of experience and analysis of the algorithm, it is believed that if $\mathbf{M}$ has 100% performance on D, so will $\mathbf{M}_{comp}$.

Note that there is no guarantee that $\mathbf{M}_{comp}$ is *minimal*, i.e. that there is not some smaller subset of $\mathbf{M}$ that would also produce 100% performance. And it is not clear *a priori* that $\mathbf{M}_{comp}$ will be appreciably smaller than $\mathbf{M}$. Still, the heuristic of "seeing how many of the best you actually need" seemed reasonable, and worth testing experimentally.

5 Compact Rulesets for the WBC data

The compact ruleset algorithm (CRA) was applied to the numerosity macrostate of the final population of the experiment of Figure 2, which consisted of 1155 classifiers. The resulting compact ruleset (CR), shown in Figure 3, contained 25 classifiers, a reduction of 97.8%. As expected, the ruleset scored 100% on the dataset, but it is interesting to examine how much of the ruleset is needed for a given level of performance.

```
 0. |0000000000|0000000000|0000......|0000000000|0000000000|00........|0000000000|00........|0000000000|  0 1000
 1. |......0000|....000000|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|  1 1000
 2. |0000000000|0000000000|0000000000|0000000000|0000000000|.....00000|....000000|0000000...|0000000000|  1 1000
 3. |0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|.........0|0000000000|  1 1000
 4. |0000000...|0000000000|0000000000|00........|00........|0000000000|0000000000|0000000000|0000000000|  0 1000
 5. |00000.....|0000000000|.....00000|.000000000|0000000000|0000000000|0000000000|0000000000|0000000000|  1 1000
 6. |......0000|0000000000|0000000000|0000000000|0000000000|.......000|0000000000|0000000000|0000000000|  1 1000
 7. |000000....|0000000000|000000000.|000000000.|..00000000|000000....|000000000.|.0000000..|00........|  0 1000
 8. |0000000000|0000000000|..00000000|...0000000|0000000000|0000000...|000000....|00000000..|0000000000|  1 1000
 9. |00000000..|0000000000|0000000000|0000000000|0000000000|0000......|00........|0000000000|0000000000|  0 1000
10. |0000000000|.00000....|0000000000|0000......|..00000000|.....00000|0000000000|0000000000|0000000000|  1 1000
11. |........00|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|  1 1000
12. |0000000000|000.......|0000000000|00000.....|0000000000|..0000000.|0000000000|.00000....|0000000000|  1 1000
13. |.....00000|0000......|.000000000|0000000000|....000000|000.......|0000000000|0000000000|0000000000|  1 1000
14. |00000000..|0000000000|000000000.|....0.....|..00000000|...0000000|000000....|0000000...|000000000.|  0 1000
15. |0000000000|....000000|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|.000000000|  1 1000
16. |0000000000|0000000000|..00000000|.....00000|0000000000|0000000000|000000....|00000000..|0000000000|  1 1000
17. |000000....|...00000..|.000000...|000000....|..00000...|...00000..|0000......|..00000000|000000....|  0 1000
18. |0000000000|0000000000|0000000000|.000000000|0.........|...0000...|0000000000|0000000000|0000000000|  1 1000
19. |0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|.......000|0000000000|0000000000|  1 1000
20. |.....000..|.000000000|0000000000|.000000...|0000......|0000......|0000000000|0000000...|0000......|  0 1000
21. |....000000|000.......|0000000000|0000000000|00000.....|.000000...|...0......|0000......|000.......|  1 1000
22. |0000000000|0000000000|0000......|0000000000|0000000000|0000......|000.......|0.........|0000000000|  0 1000
23. |....000000|.000......|...0000...|....0.....|00000000..|...0000000|.....00000|0000000000|0000000...|  0 1000
24. |.00000000.|0000......|000000....|......0000|000000000.|..0000....|000000....|00000000..|000000....|  0 1000
```

Fig. 3. Compact ruleset derived from the numerosity macrostate of the final population of Figure 2 using the algorithm of Section 4.

Step 3 of the compact ruleset algorithm orders the classifiers according to their marginal contribution; that is, the classifiers are ordered such that a given one matches the maximum number of data instances not matched by previous classifiers in the set. Figure 4 shows how performance increases as classifiers are added (in effect, with a numbering adjustment, the graph shows performance vs. sequential subsets of the CR).

The first classifier in the set, number 0 in Figure 3, is alone sufficient to achieve 57.7% performance on the dataset; the first five classifiers raise performance to 90.0%; the first ten to 95.6%, and so on. Here, performance means number correct divided by the size of the dataset; since all classifiers of the CR are accurate, the number not correct consists entirely of instances that are not matched, i.e., not recognized. So the figures mean that classifier 0 has enough knowledge of the dataset to correctly classify 57.7% of it, etc. The CRA clearly shows that the knowledge about the WBC dataset that was contained in the

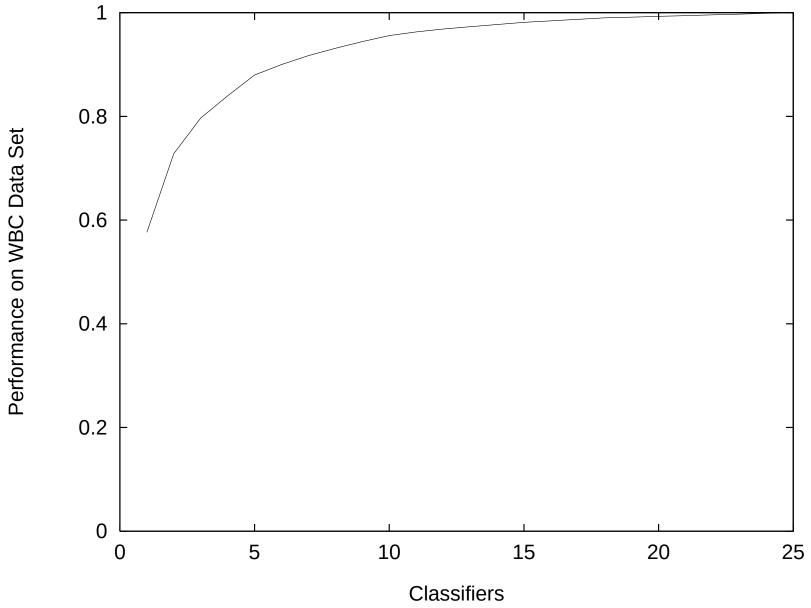

Fig. 4. Performance (fraction correct) on the WBC dataset using progressively increasing numbers of classifiers from the compact ruleset of Figure 3, starting with the first classifier.

original final population can be completely represented by a small number of selected classifiers.

It is interesting to ask why the original final population was so large—even though action of the GA and subsumption deletion [8] had reduced it considerably. In a nine-attribute, integer-valued problem like the WBC, the search processes (crossover and mutation) result in a huge number of candidate "better" classifiers, which are only very slowly either eliminated or emphasized. Furthermore, since the dataset occupies only a minuscule fraction of the input space, many different accurate generalizations are possible. They overlap only partially, so the GA and subsumption mechanisms have difficulty eliminating the less general of them. Fortunately, however, a small number of classifiers sufficient to accurately process the dataset are evolved, among all the others, and these can apparently be identified and selected by the CRA.

The classifiers of the CR in Figure 3 are straightforward to translate into English language descriptions, as was done in Section 2 for the classifier of Figure 1 (which is number 4 of the CR). In English (or other language), the rules could be directly useful to physicians, especially the first 10 rules (covering 95.6% of cases), most of which depend on just one, two or three attributes. These rules could be applied to cases or studied in themselves, as representing broad classes of cases.

```
        1          2          3          4          5          6          7          8          9

   |  *       |*         |*         |*         | *        | *        |  *       |*         |*         |  0
0. |0000000000|0000000000|0000......|0000000000|0000000000|00........|0000000000|00........|0000000000|  0 1000
1. |0000000...|0000000000|0000000000|00........|00........|0000000000|0000000000|0000000000|0000000000|  0 1000
2. |0000000000|0000000000|0000......|0000000000|0000000000|0000......|000.......|0.........|0000000000|  0 1000

   |         *|  *       |     *    | *        |  *       |    *     |   *      |         *| *        |  1
0. |0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|.........0|0000000000|  1 1000
1. |........00|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|0000000000|  1 1000

   |       *  |   *      |    *     |*         | *        |**********|      *   |  *       |*         |  1
0. |0000000000|0000000000|0000000000|0000000000|0000000000|.....00000|....000000|0000000...|0000000000|  1 1000
1. |......0000|0000000000|0000000000|0000000000|0000000000|.......000|0000000000|0000000000|0000000000|  1 1000

   |*         |*         |*         |*         |   *      |  *       |*         |*         |*         |  0
0. |00000000..|0000000000|0000000000|0000000000|0000000000|0000......|00........|0000000000|0000000000|  0 1000
1. |0000000000|0000000000|0000......|0000000000|0000000000|0000......|000.......|0.........|0000000000|  0 1000
```

Fig. 5. WBC cases and matching classifiers from the compact ruleset of Figure 3. Notation is the same as for Figure 1.

A somewhat different way the rules could be employed is illustrated in Figure 5. As in Figure 1, the star rows represent particular cases (the block of stars is an example of a missing attribute value, matched unconditionally). With each case are the two or three classifiers from the CR that matched it. It can be imagined that a physician, with a case in hand, has the system display the classifiers that match the case in order to see what knowledge the ruleset, and through it the underlying dataset, might contribute.

For example, the first two classifiers matching the first case suggest that if either attributes 6 and 8 or attributes 4 and 5 have values less than 3, the case is benign. Each classifier has another non-general attribute, but it is not so sharply defined and may not matter so much. The third classifier is more complicated, but it also appears in the fourth case. Together, the two cases suggest that low values of attributes 6 and 7 also imply benign, and that if they are indeed low, the value of attribute 8 doesn't matter.

The second case suggests strongly that either a 10 on attribute 8 or 9-10 on attribute 1 implies malignant. Recall that a classifier tends to be generalized up to the point where it begins making mistakes. The narrow dependence of these two classifiers on high values of attributes 8 and 1, respectively, suggests that cases with values even slightly less will need to have additional attributes looked at (by other classifiers), but that values within these ranges are decisive by themselves. The third case in Figure 5 contains an unknown attribute value, but since the decision is correct, it probably depends on other attributes. The key variables are probably the relatively high value of either attribute 1 or attribute 7, since the other attributes are generalized out (except for the counter-intuitive predicate in field 8 of the first classifier, which might have generalized out had the evolution gone farther), An interesting observation about Figure 5 is that attributes 2 and 9 are irrelevant to all four decisions. In fact, from looking at further cases, it turns out that attribute 2 is sometimes important, but attribute 9 hardly ever.

These examples are meant to indicate how the compact ruleset can assist a practitioner in gaining understanding of a database, and thus the real-world problem behind it. They bring out which attributes are important and which not, and they show how a given case can often be decided from more than one perspective. It seems likely that access to software that matched compact ruleset classifiers to cases would aid practioners in the important process of developing intuition.

6 Cross Validation Experiment

The CRA produced a quite compact and apparently insightful ruleset for the breast cancer data that retained 100% performance. However, it is important to know whether such rulesets have good performance on *new* cases. Since truly new cases were not available, stratified ten-fold cross-validation (STFCV) experiments were performed using the original dataset.

An STFCV is typically done as follows. The dataset is divided into 10 parts called "folds". The system is then tested on each fold (which functions as a set of "new" cases), after being trained on the balance of the dataset (the *fold complement*). Then the results on the 10 test folds are averaged giving the final score. The folds are made as equal in size as possible, given the size of the actual dataset. "Stratification" means that in each fold the possible outcomes have approximately the same prevalences as in the whole dataset. For more details, see [8] and [9].

Table 1 gives the results of four experiments. The second column reproduces performance results from the STFCV experient in [8]. For each fold, XCSI was trained for 50,000 problems (on the fold complement), and the final population was tested with the performance (fraction correct) given in the table. For the third column, these populations were further trained (on the same fold complements) out to 2,000,000 problems and then tested on the same folds as before. For the fourth column, the numerosity macrostate was computed for each of these populations, and the CRA applied to yield a numerosity compact ruleset, "nCR". Then the ten nCR's were tested on the folds with the results shown. The fifth column is like the third, except that *experience* compact rulesets, "eCR's", were computed and tested.

The table suggests several inferences. First, the means of the second and third columns are close, indicating that for full populations, the generalization achieved by training out to 2,000,000 problems does not come at the expense of performance. (The generalization and population size reductions were similar to those in the experiment of Figure 2). Second, the results on the CR's, though worse, are only slightly so, suggesting that the large ruleset size reduction achieved by the CRA comes without significant performance cost and thus that "the knowledge" made visible by the CRA is essentially as correct as the

Fold	Pop. @ 50,000	Pop. @ 2 mill.	nCR of Pop. @ 2 mill.	eCR of Pop. @ 2 mill.
1	0.9714	.9714	.9429	.9714
2	0.9857	.9857	.9714	.9857
3	0.9286	.9143	.9000	.9143
4	0.9429	.9429	.9000	.9286
5	0.9286	.9429	.9286	.9286
6	0.9143	.9429	.9571	.9429
7	1.0000	.9429	.9143	.9286
8	0.9857	.9857	.9286	.9571
9	0.9420	.9565	.9420	.9275
10	0.9571	.9429	.9429	.9286
Mean	0.9556	.9528	.9328	.9413

Table 1. Results of stratified tenfold cross-validation tests on WBC dataset. Each line has results for one fold. Last line is mean of fold scores. See text for explanation of tests.

knowledge in the full populations. Finally, it is interesting that the eCR performs slightly better on this problem than the nCR, suggesting that classifier experience is at least as good a basis as numerosity for forming compact rulesets.

7 Conclusion

This paper presented a technique, the Compact Ruleset Algorithm, for reducing the size of evolved classifier populations. On the Wisconsin Breast Cancer dataset problem, the CRA produced compact rulesets substantially smaller than the evolved populations, yet performance on new data was nearly unchanged. Classifiers of the CR's expressed readily interpretable knowledge about the dataset, and may be usable by practitioners in a variety of ways.

The CRA technique should be tested on other datasets, and variations in the technique itself need to be explored. The value of compact rulesets that make knowledge visible and can be easily worked with is clear, even if their performance is slightly less than that of full evolved populations. In a developed software system, it would be desirable to make available *both* the evolved population—for maximum performance on new data instances—and the corresponding compact ruleset—as a summary of "the knowledge", a tool for quick evaluations, and an aid to intuition.

References

1. C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998. http://www.ics.uci.edu/~mlearn/MLRepository.html.

2. Martin V. Butz and Stewart W. Wilson. An Algorithmic Description of XCS. In Lanzi et al. [4].
3. Chunsheng Fu, Stewart W. Wilson, and Lawrence Davis. Studies of the XCSI classifier system on a data mining problem. In Lee Spector, Erik Goodman, Annie Wu, William B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, page 985. Morgan Kaufmann: San Francisco, CA, 2001.
4. Pier Luca Lanzi, Wolfgang Stolzmann, and Stewart W. Wilson, editors. *Advances in Learning Classifier Systems: Proceedings of the Third International Workshop (IWLCS-2000)*, LNAI-1996. Springer-Verlag, Berlin, 2001.
5. Stewart W. Wilson. Classifier Systems and the Animat Problem. *Machine Learning*, 2:199–228, 1987.
6. Stewart W. Wilson. Classifier Fitness Based on Accuracy. *Evolutionary Computation*, 3(2):149–175, 1995.
7. Stewart W. Wilson. Generalization in the XCS classifier system. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David E. Goldberg, Hitoshi Iba, and Rick Riolo, editors, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 665–674. Morgan Kaufmann: San Francisco, CA, 1998.
8. Stewart W. Wilson. Mining oblique data with XCS. In Lanzi et al. [4].
9. Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, San Francisco, CA, 2000.