Mining the Space of Generality with Uncertainty-Concerned Cooperative Classifiers

Jorge Muruzábal
ESCET
University Rey Juan Carlos
28933 Móstoles, Spain
j.muruzabal@escet.urjc.es

Abstract

Generality is a recurrent theme in automated inductive systems. Induction of general patterns/rules is of course complicated by several factors. For example, higher levels of uncertainty and error are naturally introduced by generality. Moreover, it is not clear what sort of trade-off should be sought between increasing generality and decreasing predictive power. As a result, specific criteria to guide the search for useful general rules do not abound. In this paper, I reconsider these issues in the context of the generalized, fuzzy-like classifier system first proposed by Frey and Slate (1991) and later equipped with a Bayesian learning component by Muruzábal (1998). A crucial feature of this approach is that uncertainty is probabilistically measured at each classifier in the population. A new reinforcement policy exploiting this probabilistic structure and priming cooperation among general classifiers is introduced and shown to promote the stability of niches of reasonably high predictive power. The underlying genetic algorithm contributes effectively to learning although it somehow counteracts the built-in bias towards generality.

1 INTRODUCTION

Generality is perhaps the most desirable feature of inductive systems: it guarantees that findings based on the training sample apply to the broadest possible range of future test cases. General rules often provide as well easy-to-grasp, potentially far-reaching information nuggets for the human interpreter. However, good performance upon induction does not require generality. Nor is it clear how much generalization is possible or desirable in a given task. Since generality typically introduces some loss in predictive power, we face among others the dilemma of having to "decide" how big such a loss is worth trading by an increase in generality.

Meanwhile, the size of the data sets available for analysis continues to grow, and the need for methods capable of extracting simple general patterns is only exacerbated. For example, in some data mining applications highly successful classification rules may already be available, yet the wish to generalize this very detailed classification and abstract out a small number of important concepts remains (Klösgen, 1996). Or, as Kohavi, Sommerfield and Dougherty (in press) put it, "In practice, of course, the user of a data mining tool is interested in accuracy, efficiency and comprehensibility for a specific domain, just as the car buyer is interested in power, gas mileage and safety for specific driving conditions".

Evolutionary systems provide a highly flexible framework for both optimization and "rule-set assembly" problems. For the latter type, learning classifier systems (LCS) have been proposed for a variety of tasks in both their Michigan and Pittsburgh versions (Booker, 1989; Janikow, 1993; Wilson, 1995; Holmes, 1997). In recent years, theoretical research within LCS has concentrated on niching, cooperation and generalization issues (Mahfoud, 1995; Horn and Goldberg, 1996; Lanzi, 1998). Many of these issues are shared by other modern techniques such as neural nets (Kosko, 1996; Whitehead and Choate, 1996).

In this paper, the BYPASS prototype (Muruzábal, 1998) is tested and shown to promote the stability of niches of general classifiers providing reasonably high predictive power. BYPASS stems from the generalized fuzzy-like stimulus-response classifier system proposed by Frey and Slate (1991) and builds on ideas laid out by Wilson (1987), Booker (1989) and Packard

(1990) among others. A previous version was discussed and evaluated in (Muruzábal and Muñoz, 1994). The current version can be seen as a Michigan-style LCS which incorporates probability distributions (PD) in the action part of classifiers and disregards the classical notion of strength in favor of Frey and Slate's (1991) utility-based survival mechanism. PDs seem well-suited to describe the uncertainty underlying general rules. Endowed with a simple Bayesian updating component, BYPASS classifiers modify their PD and learn from the stream of data. This probabilistic structure is further exploited in that the system's response, the reinforcement policy and the genetic algorithm are all based on the subset of matched PDs. Overall, BYPASS emphasizes predictive ability and thus shares the point of view that a model with good predictive ability "must have captured some regularities that also reflect properties of the data generating process" (Kontkanen, Myllymäki and Tirri, 1996).

The paper is organized as follows. Section 2 reviews the details of the architecture. Section 3 presents experimental results. Section 4 establishes some links with related ideas in the literature and section 5 summarizes and points out some directions for further research.

2 BYPASS ARCHITECTURE

The present BYPASS architecture is concerned with *classification* tasks. Assume a sample of pairs (or rows) (x_i, y_i), where the *response* y_i is one of k output labels and x_i is a vector of n *predictors*. Following Frey and Slate's fuzzy-like design, all real-valued predictors (columns) in a given data set are linearly transformed and rounded to a 0-15 integer scale prior to training. Assume also the usual processing cycle comprising matching, system response, reinforcement and (triggered) rule generation steps. BYPASS performs also an utility check at the end of each cycle (Frey and Slate, 1991). Let us consider these steps in detail.

2.1 CLASSIFIERS

Each classifier has the familiar form $Q \xrightarrow{\sigma} R$, where Q and R are called the classifier's *receptive field* and *prediction* respectively. Q's coordinates can be either an integer between 0 and 15 or the standard "don't care" #. R is a (time-varying) probability distribution over the set of output labels. $\sigma = (\kappa, \rho, \lambda, \alpha)$ encodes statistics summarizing previous experience with the classifier. Specifically, $\kappa > 0$ and $\rho > 0$ are respectively the classifier's *raw utility* and *accuracy*, $\lambda > 0$ reflects the average size of the match sets to which the classifier belonged, and α simply counts the number of data items presented to the system since the classifier was incorporated.

All members of σ are updated after each match. As described below, the raw utility κ is increased whenever the classifier seems responsible for a correct prediction. The utility measure μ is defined as $\mu = \frac{\kappa}{\alpha}$. Utility is key to survival: if μ becomes low, the classifier is prone to deletion. The accuracy summary ρ reflects in turn the uncertainty in R: the sharper (or more concentrated) this distribution, the lower ρ. In general, the more *specific* Q, the sharper R. Overall, ρ and μ dynamically encode the usefulness of a rule on a stand-alone basis and given the remaining units in the population respectively.

The traditional approach to R considers single output labels only. In BYPASS, classifiers build their predictions from two k-dimensional vectors c and $a > 0$ called respectively the *likelihood* and *prior* vectors. The latter is fixed once and for all at the time of creation; the former evolves over time according to the subset of responses observed when the classifier is matched. Computation of R is straightforward once a standard Multinomial-Dirichlet model is built into each classifier. Specifically, a standard prior density of Dirichlet type with parameter a is assumed over the space of conditional distributions of the response y given that the classifier is matched. This readily leads to a posterior predictive distribution (given all previous data *filtered* by Q) with coordinates $R_j = \frac{c_j + a_j}{c_+ + a_+}$ where c_+ and a_+ equal the sum of their entries. Clearly, $a_j > 0$ implies $R_j > 0$ at all times.

2.2 INITIALIZATION

The initial population is always generated according to EXM, a simple extension of the well-known *exemplar-based generalization* method (Frey and Slate, 1991). EXM randomly selects a single data item (x, y) and builds a single classifier (Q, R, σ). The receptive field Q is constructed by first setting $Q = x$ and then parsing through its coordinates $l = 1, \ldots, n$: with some generalization probability π, the current value x_l is switched to #, otherwise $Q_l = x_l$ is maintained. The likelihood vector c is naturally set to 0, whereas a places .7 mass at y and $\frac{.3}{k-1}$ everywhere else. Naturally, $\alpha = \kappa = 0$; initial values for ρ and λ are set after the first match. Relatively small initial populations (100 for the letter data, 50 for the DNA data) were used in all experiments below.

Frey and Slate (1991) and others consider only $\pi = .5$.

In this paper EXM (with $\pi = .8$ or larger) is the major supplier of generality. For large π, EXM may easily generate the trivial receptive field containing all #'s. This classifier is considered useless and is not allowed into the system. Unlike Wilson (1995), no replicated receptive fields are allowed either.

2.3 MATCHING

As usual, the x part of each training item determines the set M of matched classifiers; a classifier is matched if all the non-# coordinates in Q are. For a boolean predictor l, exact matching is required: $|x_l - Q_l| < 1$. For integer-valued predictors, matching requires only $|x_l - Q_l| < 2$. The system can process any combination of integer and boolean predictors. The size of M, say m, is used to update the λ estimate of all classifiers in M. In case M is empty, a new unit is automatically created by EXM and incorporated immediately (so $m \geq 1$ at all times).

2.4 SYSTEM PREDICTIONS

Two system predictions are computed and their performance tracked over time, namely, the *single-winner* (SW) and *mixture-based* (MIX) predictions. SW selects the classifier in M with the lowest uncertainty evaluation ρ as its best chance to predict on the basis of a single unit. The *maximum a posteriori* (MAP) class label y_{SW} is then determined from this single R as $y_{SW} = \arg\max_{1 \leq j \leq k} R_j$. On the other hand, MIX combines first the m matched R's into a (uniformly weighted) mixture distribution $R_{MIX} = \frac{1}{m} \sum R_s$ and then obtains y_{MIX} by MAP selection as before. The idea in R_{MIX} is much like in (feed-forward) pattern recognition neural networks: each rule in the population is devoted to a general feature (satisfied by many instances) and it is only the concurrent activation of several such general-feature detectors what dissolves the uncertainty and yields a correct recognition in each case.

Several system decisions are based on whether y_{MIX} is correct or not. For example, in the present suite of experiments, the rule-generation mechanism is always *triggered* by MIX failure (regardless of the reward policy in effect). Another central quantity is the *score* of each individual classifier in M: $S_y = -\log(R_y) > 0$. Obviously, the lower S_y, the better R at this prediction time. Any upper bound on S_y can be expressed equivalently as $R_y \geq prob$; for ease of reference, $prob$ values are also provided below. As discussed next, classifiers with the lowest S_y in each M should be the prime contributors to a correct MIX prediction and therefore ought to be rewarded accordingly.

2.5 REINFORCEMENT AND CLASSIFIER UPDATE

Reinforcement in BYPASS takes place every cycle and essentially involves the updating of both the accuracy ρ and the raw utility κ in the light of the predicted label y_{SW} (or y_{MIX}) and the current observation y. Other components need also be updated. For example, all classifiers have their α counters naturally increased by 1, be they matched or not. Classifiers not in M remain otherwise untouched, only matched units are reinforced as follows. First, each c_y is increased by 1. ρ is updated via the MAM procedure, see (Wilson, 1995): while $t \leq \tau$, ρ simply equals the average of all previous t matches. Once $t > \tau$, $\rho_{t+1} = (1 - \frac{1}{\tau})\rho_t + \frac{1}{\tau} S_y$. Estimate λ is updated similarly; τ is set to 50 in both cases. In the long run ρ_t converges to the standard entropy measure of the multinomial distribution determined by Q and the training data. Thus, for finite t, ρ can be seen as an estimate (subject to random fluctuations) of that uncertainty measure.

The way in which κ is updated depends in the first place on whether the system is providing y_{SW} or y_{MIX} as its "accountable" prediction. If the system relies on y_{SW}, and this turns out to be correct, then only the relevant classifier is rewarded. Specifically, this unit's κ is added an amount $w(y) > 0$. Since m may be rather large, this reward policy may appear rather "greedy". The actual $w(y)$ depends in turn on whether the user-specified option EQFT (for "all output labels at *eq*ual *foot*ing") is set to *True* or not. If so, then $w(j) \equiv 1$; otherwise, $w(j) = \frac{1}{kf(j)}$ where $f(j)$ denotes the relative frequency of the j-th output label in the training sample. The rationale is of course that different categories may turn up with different frequencies, in which case it seems unfair to reward all classifiers evenly.

Two reward policies are examined for y_{MIX}. A simple idea is to reward the p lowest S_y when y_{MIX} is correct. The κ counter of these units is updated as before, that is, the whole $w(y)$ is given to each κ. However, the fact that y_{MIX} failed does not mean that all matched classifiers are useless. Thus, the second policy behaves exactly as the first when y_{MIX} is correct but also reinforces selected classifiers when y_{MIX} makes a mistake. Specifically, *all* units with scores below certain system threshold γ are rewarded as before. The two reward schemes can thus be parameterized by p and $\gamma \geq 0$. The idea is to help units with promising low scores to survive until a number of *cooperative* classifiers are found and similarly maintained by the system. At that time, these rules will hopefully begin to work

together forming a "critical mass" and getting their reward from *correct* y_{MIX}! Empirical evidence suggesting that this "patient" strategy can work nicely in some cases is provided below; for another example of patient strategy see (Friedman and Fisher, 1997).

2.6 A HYBRID GENETIC ALGORITHM

The present *hybrid* genetic algorithm (HGA) is always triggered by y_{MIX} failure (Booker, 1989). However, in each individual activation either the GA itself or the EXM routine may create a new rule depending on the system's score threshold θ. This parameter can be used to strike a balance between the purely random (and typically slow) search carried out by EXM and the more *focused* alternative provided by traditional genetic operators. Specifically, the HGA first checks whether there are at least two scores S_y in M lower than θ. If so, standard crossover and mutation are applied as usual over the set of matched *receptive fields*; otherwise, a single classifier is generated by EXM. Again, the trivial receptive field and exact copies of existing receptive fields are precluded in either case. The resulting mating strategy is thus doubly restricted: parent classifiers must belong to the same match set and must also have seen a substantial number of instances of the target y.

Standard uniform and single-point crossover are implemented; in either case, a single receptive field is produced by crossover and mutation acts on this offspring with some coordinatewise mutation probability ϵ. The final Q is endowed with $\kappa = 0, c_j \equiv 0$ and a following y as in EXM above. Receptive fields may be selected for mating according to the standard roulette-wheel procedure (with weights given by the normalized inverses of their S_y), or directly as the two lowest scores in M.

2.7 CHECKING FOR LOW UTILITY

At the end of each cycle, all classifiers have their utility checked for "services rendered" to date. Units become candidates for deletion as soon as their utility μ drops below a given system threshold μ_0. This powerful parameter also helps to promote generality: if μ_0 is relatively high, specific classifiers will surely become extinct no matter how low is their accuracy ρ. It is convenient to view μ_0 as $\mu_0 = \frac{1}{v}$ with the interpretation (under EQFT=*True*) that classifiers must be rewarded on average once every v cycles to survive. Early versions of the system simply deleted all classifiers with $\mu < \mu_0$ at once. It was later thought a good idea to avoid sudden shocks to the population as much as possible. Therefore, only one classifier is currently deleted per cycle, namely the largest λ (Wilson, 1995). Also, because sometimes a relatively high μ_0 is used, a *mercy* period of guaranteed survival α_0 is granted to all classifiers, that is, no unit with $\alpha \leq \alpha_0$ is deleted. This gives classifiers some time to refine their likelihood vectors before they can be safely discarded.

2.8 EFFECTIVE TRAINING AND COOLING

Two kinds of training phases are distinguished: during *effective* training, the system can produce as many new units as it needs to. During *cooling*, the HGA is turned off and no new rules are generated (except those due to empty match sets); all other system operations continue as usual. Utility constraints typically reduce the size of the population considerably during this latter phase.

3 EMPIRICAL WORK

I present experiments for both the letter (Frey and Slate, 1991) and the DNA data (Neri and Saitta, 1996), see also (Michie, Spiegelhalter and Taylor, 1994). Both data sets are available from the UCI machine learning data repository; the following table summarizes their characteristics. Note that some ambiguous data were removed from the DNA file; non-ambiguous data were (redundantly) encoded by assigning four bits to each nucleotide. Training and test samples were randomly selected once and for all in each case. Raw utility weights $w(j)$ for the DNA data are 1.475, 1.462 and .610 respectively; for the letter data, $w(j) \equiv 1$. Mercy periods α_0 were 200 and 1,000 respectively.

	letter	DNA
n	16	240
Predictors	all integer	all boolean
k	26	3
Training sample size	16,000	2,000
Test sample size	4,000	1,175

A few selected runs for the letter data are discussed first. The first experiments aim to show that the cooperative reward policies $(p, 0)$ and (p, γ) can manage to organize some predictive abilities under either relatively high generalization rate π or survival pressure μ_0 or both. Some interesting questions regarding the role of the GA in this quest for generality will be addressed later in the more systematic analysis of the DNA data.

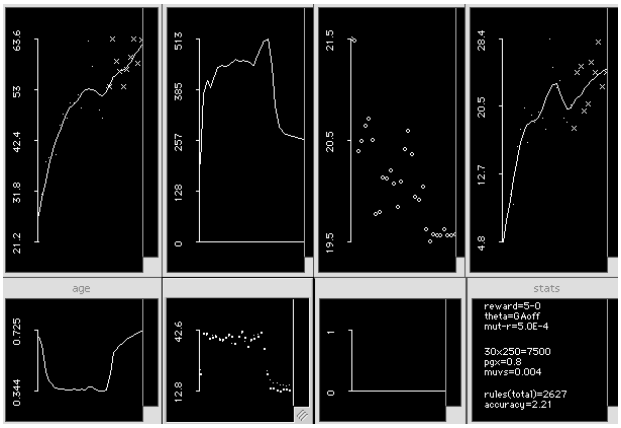

Figure 1: Quick learning under $\gamma = 0$.

3.1 EXPLORING THE LIMITS OF GENERALITY

A number of experiments were carried out comparing SW and MIX performance under the SW greedy reward policy and $\theta = 0$ (no GA, only EXM generates new rules). Results (not shown here) generally indicate that rules tend to be more specific and have great difficulty in organizing cooperation. A reference test success rate of about 58% is achieved in $30 + 10$ thousand cycles by either y_{SW} or y_{MIX} under $\mu_0 = 1/1000$ and π equal to .5 or .8. The SW reward policy only obtains a 42% test recognition rate under $\mu_0 = 1/250$ and $\pi = .5$, see also (Muruzábal, 1998).

Figures 1, 2 and 3 refer to three runs under $\theta = 0$ but using patient reward schemes for y_{MIX}. The following table describes the specific configurations used in each case (the last column refers to the largest μ_0 considered by Frey and Slate, 1991).

	Fig. 1	Fig. 2	Fig. 3	F&S91
π	.8	.9	.8	.5
μ_0	1/250	1/50	1/250	1/1000
(p, γ)	$(5, 0)$	$(10, 2.5)$	$(5, 2)$	SW
Effective	5,000	20,000	15,000	80,000
Cooling	2,500	–	5,000	16,000

Figures 1, 2, and 3 are snapshots of the computer screen updated and redrawn every 250 cycles. Given the complex dynamics of the system, it is crucial to be able to grasp as much detail as possible of its internal work. In particular, these images reveal phenomena that go unnoticed by only looking at, say, the overall success rate. Each figure consists of the following eight panels (top row first, from left to right):

1. The MIX success rate. This is the performance indicator of foremost interest; the related SW success rate is plotted indirectly in panel 4. Smoothed curves (provided by the LOWESS procedure) are superimposed to improve trend perception. Cooling periods are signaled by X's.

2. Total number of classifiers and total number of genetic classifiers. Under $\theta = 0$ the lower line surely remains at 0.

3. Average specificity of the population (expressed as a percentage of the total number of coordinates n).

4. The *edge* or difference between MIX and SW success rates. If this drops below 0, the system is confused in exploiting a "distributed" knowledge base as described above.

5. The *aging factor* of the population. This is computed as the ratio between the average α in the population and the total number of data items presented so far. An increasing trend here indicates that a body of rules is consolidating.

6. The average size of the match set. Two symbols are actually plotted at each abscise corresponding to separate averages for correct and wrong y_{MIX}.

7. The average number of rewarded units when y_{MIX} is wrong. This is of course 0 unless $\gamma > 0$. Reference lines at 1 and 2 are plotted for clarity.

8. Some run parameters and further statistics of interest. The latter include the median accuracy ρ in the final population and the total number of rules created by the system.

In Fig. 1 we see rather good signs of cooperation as given by an uprising edge of at least 20%. We also see the early rapid increase in MIX performance followed by a (typical) further increment due to cooling and population condensation. The average specificity is slightly below the postulated .8, whereas the number of rules is moderate. This final population achieved a MIX test success rate of 57% (comparable to the above given rate for the greedy reward policy) with a notable edge of 23.5%.

In Fig. 2 a high generalization rate is combined with a rather generous reward policy ($p = 10$ and $\gamma = 2.5 <=> prob = .082$). We see a similar trend except now specificity "escapes" the hypothesized .9 upward, the edge is larger and the average accuracy ρ is higher. While the present edge well above 30% shows great cooperation among units, the overall success rate seems to settle around a modest 52%.

In Fig. 3 the amount of reward is reduced a bit ($p = 5$ and $\gamma = 2$ <=> $prob = .135$). Note the increasing number of rewarded units during effective training. In five runs this version of the system achieved a median test recognition rate of 67.4% with a maximum of 68.9%. These seem comparable to Frey and Slate's 68.5% with 252 rules under the choices listed above. Yet the two populations are very different. BYPASS requires about three times as many rules, but Frey and Slate's smaller population presents an average specificity of 42.8% (or about double of Fig. 3). Overall, the suggestion is strong that the new reward policies promote cooperation among general rules and facilitate rapid learning even in the absence of any genetic input.

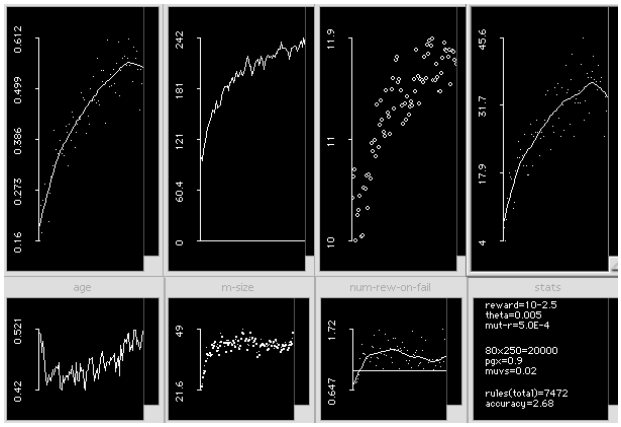

Figure 2: Learning under $\pi = .9$ and high survival pressure.

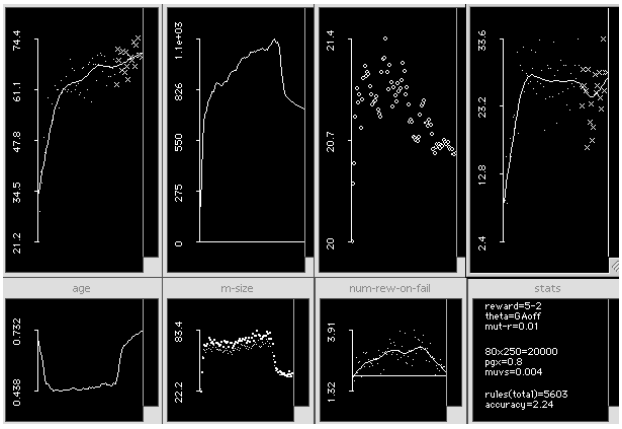

Figure 3: Achieving substantial predictive power under $\gamma > 0$ and the (data-supported) $\pi = .8$.

The user can modify the values of system parameters in the light of the displayed information. For example, in order to study the possible contribution of the GA, the population diversity finally achieved in Fig. 2 should provide an excellent starting point. Therefore, this run went on for an additional 40,000 trials of effective training (no cooling) under modified $\pi = .85$ and $\theta = 1.4$ ($prob = .247$); all remaining parameters maintained their values. Uniform crossover, $\varepsilon = .0005$ and roulette-wheel selection were used for the GA. The result (not shown here) is a disappointing 5 point decrease in both MIX and edge after consideration of nearly 14,000 genetic rules (about 70 of which made it into the final population). Specificity went up faster and hence match sets became smaller. Overall, the impression is that the GA may have disrupted a bit the progress made earlier. What are the reasons for this behaviour?

3.2 THE ROLE OF THE GA

We now switch to the DNA data and try four types of HGA as described in the following table; single-point crossover, $\pi = .985$, $\mu_0 = 1/10$, $p = 3$ and $\gamma = .8$ ($prob = .449$) were fixed in ten runs by each algorithm (each run consisted of 15,000 trials of effective training and 5,000 trials of cooling). A summary of these runs is provided in Fig. 4.

Algorithm type	1	2	3	4
θ	0	1	1	.75
ϵ	-	.0025	.0005	.00001
roulette-wheel	-	N	Y	Y

Type 1, the reference based on EXM alone as in the previous section, achieves the largest edge (about 10% on average). However, since its SW rate is also 10 points behind the other algorithms, type 1 ranks worst in MIX performance. The large edge is due in part to the generality maintained by the system in this case: the average specificity is only about 1.15% (or just 2.75 defined bits per receptive field), whereas the average accuracy ρ is about .93, a remarkably high figure – recall that the default predictive distribution $(\frac{1}{4}, \frac{1}{4}, \frac{1}{2})$ has an entropy of about 1.04.

Looking now at the remaining algorithms in Fig. 4, we see that the GA does change quite a bit the behaviour of the system. First, better MIX rates are obtained using fewer rules in all cases. The final rules are relatively more specific, less uncertain, and are discovered sooner by the system. However, the average edge achieved by type 2 is 0. On the other hand, type 4 gets the best MIX rate while maintaining many of the nice features of type 1: more general rules, higher ρ and at least 5 points of edge. This suggests that the GA may need to be appropriately constrained to yield the

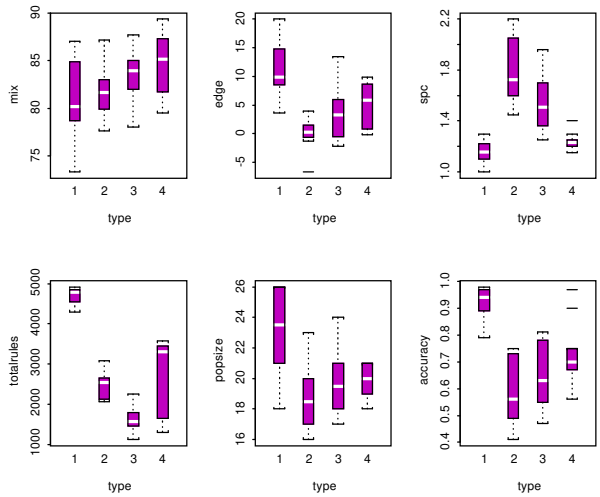

Figure 4: Box-and-whiskers plots of key test statistics for the DNA data. The white notches signal the median value over runs; the dark boxes summarize the spreads of the underlying distributions. The lower panels portray respectively the total number of rules explored and the (final) population sizes and median accuracies.

```
                                MAG GTRAGT
09876543210987654321:0987654321 1234567890:12345678901234567890  accuracy
                    :            cT   c    :        a              .859
    c               :                G     :                 a     .890
          t         :             T        :                       .969
          t         :        t    T t      :                       .927
                    :            G         :                      1.008
                    :             T        :    a                 .965
                    :        g    T        :                       .999
                    :                G     :                       .933
                    :            GT        :                       .565
09876543210987654321:0987654321 1234567890:12345678901234567890
                    :             T        :                      1.055
                    :                G     :                 a     .890
   g                : t          G tg      :                       .920
           a        :             Tt       :     a                 .991
            c       :                G     :                       .904
         g          :             g  G     :                       .940
                    :            G       g :                      1.084
                    :       g     T  a     :                      1.029
09876543210987654321:0987654321 1234567890:12345678901234567890
                    :            GT        :              a   d    .563
  g                 :            GT        :a                      .541
                    :            GT        :a                      .521
                    :      g     GT        :a                      .551
                    :            GT        :                       .616
09876543210987654321:0987654321 1234567890:12345678901234567890
```

Figure 5: DNA data: Classifiers for class 1 (EI) in three populations. Each capital (resp. lower case) letter requires the presence (resp. absence) of the corresponding nucleotide at the given site. The *donor* site is signaled by the central empty column. Other sites are numbered +1, +2, ... on the right of the donor site, and -1, -2, ... on the left. The top line recalls the core of the domain theory: the conventions `M` = `A` or `C`, `R` = `A` or `G`, `Y` = `C` or `T` are standard.

best results. It also seems better to use lower mutation rates to prevent the advent of more specific rules with better scores that end up "canibalizing" the more general rules.

Let us look at individual classifiers and contrast them with the current domain theory, see e.g. (Neri and Saitta, 1996). Fig. 5 lists receptive fields and accuracies of classifiers whose MAP individual prediction is 1 (or EI in the domain's jargon) in three final populations; Fig. 6 lists units pointing similarly to class 2 (IE) taken from the same three populations. The top and middle populations classified correctly all but 2 test instances of the EI class and all but 7 test instances of class IE respectively. These were both obtained by type 1 and became the best of 40 as far as this partial misclassification rates are concerned (test recognition rates by y_{MIX} were just 79.3% and 73.3% respectively). The third population was obtained by type 4 and corresponds to the overall best test rate (89.4%). Classifiers for class 3 (neither EI or IE) are largely (relatively uninteresting) negatives of these units.

The current theory establishes that *central* sites (enclosed by colons) are most important for classification. Figs. 5 and 6 are in clear agreement: all symbols arising in this central part (shown in bold face) do not contradict those in the first line. Further, outside the central portion we only find a few lower case letters randomly scattered here and there. However, not all the symbols in the theory are equally supported by the listed classifiers. For example, in Fig. 5 the `T` at location +2 is very much supported, whereas the `G` at -1 is rather unsupported. Likewise, compare in Fig. 6 the `A` at -2 and the `G` at +1. Note that classifiers obtained under $\theta = 0$ are indeed more general, but those obtained under $\theta > 0$ are able to locate lower-uncertainty patterns that manage to survive under the demanding $\mu_0 = .1$. In particular, the dominating role of the strings `GT` in Fig. 5 and `AG` in Fig. 6 is now accentuated.

4 RELATED WORK

In this section I briefly review some related contributions from the recent literature. Horn and Goldberg (1996) provide theoretical results concerning the way in which LCS can successfully achieve "cooperative diversity" via reward sharing and niching. These results assume all rules share the same specificity and hence concentrate on the underlying "hard" *covering* prob-

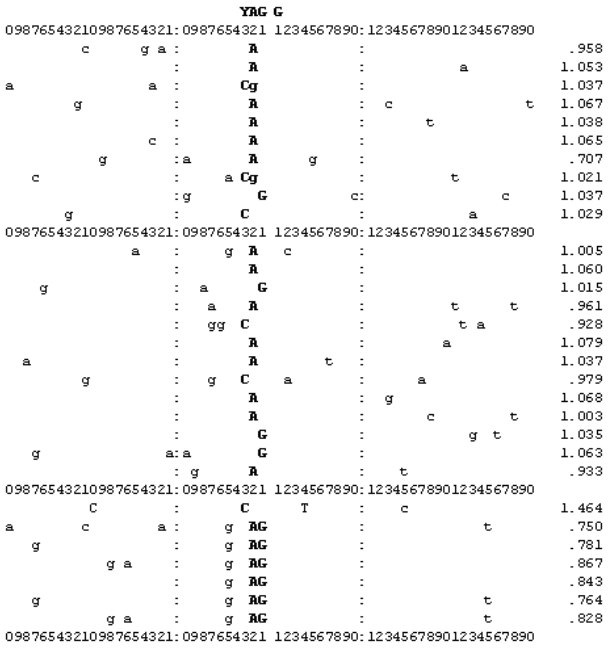

Figure 6: DNA data: Associated lists of classifiers for class 2 (IE), see Fig. 5. Three of these units showed some redundancy at nucleotide -2 (meaning that we simultaneously find `A` and `d` in the associated substring); no such redundancy was observed in Fig. 5.

lem. Horn and Goldberg point out that the extension to the more general unrestricted case would require a user-input combination of two "conflicting objectives, with no general way to combine them into a single scalar fitness measure". Cooperation vs competition of units is also a central theme in the work (cast in terms of RBF neural nets) of Whitehead and Choate (1996). In this context, covering is more important than generalization, and a more flexible representation of receptive fields allowing different centers and *widths* for different RBF units is used, see also (Kosko, 1996).

Holmes (1997) encountered two problems related to generality in his EpiCS. The system initially showed a tendency to create very general rules leading to poor predictions. It was then decided to automatically specialize every rule with specificity of 15% or less. As a result, a preponderance by highly specific, *overfitting* classifiers led again to poor performance. Finally, classifiers pointing in the wrong direction were penalized. This penalty factor was found to promote survival of useful general units and certain equilibrium was reached. Lanzi (1998) summarizes some of the issues related to generality in the context of Wilson's (1995) XCS. Lanzi points out that *overgeneralization* may "corrupt" the population in the sense of leading to worse performance. In particular, Lanzi discusses the "specify" operator, a special mechanism intended to counterbalance generalization pressure.

A previous approach to data mining based on a (hierarchical) evolutionary architecture is described in (Radcliffe and Surry, 1994). A number of commercial sites providing data mining services based on proprietary evolutionary systems are easily found in the WWW these days.

5 CONCLUDING REMARKS

The BYPASS reward system and genetic algorithm have been empirically tested and shown to yield various learning behaviours of interest. Effective cooperation can be achieved with rather general units (although one must be willing to sacrifice some predictive ability). Probability distributions successfully encode the information needed to achieve that cooperation. No ad-hoc procedures are needed to regulate generality since a few simple probabilistic constructs suffice to control the system. In particular, patient reward strategies seem helpful to collect all needed bits of knowledge before cooperation can yield the highest number of correct predictions.

It appears that the underlying GA should be adjusted so it does not disrupt the progress made by "independently" generated rules. The latter probably lead to maximum cooperation but need to be enhanced somehow to render better predictive power. Lower mutation rates are preferred, although it is not clear how low (Franconi and Jennison, 1997). Once adjusted, the GA has been found able to locate more specific (but still general) rules that simplify somewhat the interpretation but could reduce population diversity.

A number of interesting possibilities can be listed for future work. The extension to the regression case should be rather straightforward (Muruzábal, 1995). The intuition is strong that making the a priors for new genetic units appropriately dependent on their parents should accelerate learning. More sophisticated weights for the mixture R_{MIX} should be investigated. Finally, one could also assign "skewed" utility weights $w(j)$ with the idea of concentrating the effort on a few selected labels of interest (Friedman and Fisher, 1997).

Acknowledgments

Financial support from CICYT grants HID98-0379-C02-01 and TIC98-0272-C02-01 (Spain) is acknowledged.

References

Booker, L.B. (1989). Triggered Rule Discovery in Classifier Systems. *Proceedings of the Third ICGA* (J.D. Schaffer, Ed.), 265–274. Morgan Kauffman.

Franconi, L. & Jennison, C. (1997). Comparison of a Genetic Algorithm and Simulated Annealing in an Application to Statistical Image Reconstruction. *Statistics and Computing* 7, 193–207.

Frey, P.W. & Slate, D.J. (1991). Letter Recognition Using Holland-style Adaptive Classifiers. *Machine Learning* 6, 161–182.

Friedman, J.H. & Fisher, N.I. (1997). Bump Hunting in High-Dimensional Data. Technical Report, Department of Statistics, Stanford University.

Holmes, J.M. (1997). Discovering Risk of Disease with a Learning Classifier System. *Proceedings of the Seventh ICGA* (T. Bäck, Ed.). Morgan Kauffman.

Horn, J. & Goldberg, D.E. (1996). Natural Niching for Evolving Cooperative Classifiers. *Proceedings of the First Genetic Programming Conference* (J.R. Koza, D.E. Goldberg, D.B. Fogel & R.L. Riolo, Eds.), 553–564. MIT Press.

Janikow, C.Z. (1993). A Knowledge-Intensive Genetic Algorithm for Supervised Learning. *Machine Learning* 13, 189–228.

Klösgen, W. (1996). Explora: a Multipattern and Multistrategy Discovery Assistant. In *Advances in Knowledge Discovery and Data Mining* (U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth & R. Uthurusamy, Eds.), 249–271. AAAI/MIT Press.

Kohavi, R., Sommerfield, D. & Dougherty, J. (in press). Data Mining using MLC++. A Machine Learning Library in C++. To appear in the *International Journal on Artificial Intelligence Tools.*

Kontkanen, P., Myllymäki, P. & Tirri, H. (1996). Predictive Data Mining with Finite Mixtures. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (E. Simoudis, J. Han & U. Fayyad, Eds.), 176–182. AAAI Press.

Kosko, B. (1996). Additive Fuzzy Systems: from Function Approximation to Learning. In *Fuzzy Logic and Neural Network Handbook* (C.H. Chen, Ed.). McGraw-Hill.

Lanzi, P.L. (1998). Generalization in Wilson's Classifier System. *Proceedings of PPSN V* (A.E. Eiben, T. Bäck, M. Schoenauer & H.-P. Schwefel, Eds.). Springer-Verlag Computer Science Lecture Notes Series, No. 1498.

Mahfoud, S.W. (1995). Niching Methods for Genetic Algorithms. Ph.D. Thesis, Department of General Engineering, University of Illinois at Urbana-Champaign.

Michie, D., Spiegelhalter, D.J. & Taylor, C.C. (1994). *Machine Learning, Neural and Statistical Classification.* Ellis Horwood.

Muruzábal, J. & Muñoz, A. (1994). Diffuse Pattern Learning with Fuzzy ARTMAP and PASS. *Proceedings of PPSN III* (Y. Davidor, H.-P. Schwefel & M. Reiner, Eds.), Springer-Verlag Computer Science Lecture Notes Series, No. 866, 376–385.

Muruzábal, J. (1995). Fuzzy and Probabilistic Reasoning in Simple Learning Classifier Systems. *Proceedings of the 2nd IEEE International Conference on Evolutionary Computation* (D.B. Fogel, Ed.), 262–266. IEEE Press.

Muruzábal, J. (1998). Bayesian Adaptation of Classifiers in a Simple Learning Classifier System. Technical Report 98-18, ESCET, University Rey Juan Carlos, Madrid, Spain.

Neri, F. & Saitta, L. (1996). Exploring the Power of Genetic Search in Learning Symbolic Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18(11), 1135–1141.

Packard, N.H. (1990). A Genetic Learning Algorithm for the Analysis of Complex Data. *Complex Systems*, Vol. 4, 543–572.

Radcliffe, N.J. & Surry, P.D. (1994). Co-operation through Hierarchical Competition in Genetic Data Mining. Technical Report 94-09. Edinburgh Parallel Computing Centre, University of Edinburgh, Scotland, UK.

Whitehead, B.A. & Choate, T.D. (1996). Cooperative-Competitive Genetic Evolution of Radial Basis Function Centers and Widths for Time Series Prediction. *IEEE Transactions on Neural Networks* 7(4), 869–880.

Wilson, S.W. (1987). Classifier Systems and the Animat Problem. *Machine Learning* 2, 199–228.

Wilson, S.W. (1995). Classifier Fitness Based on Accuracy. *Evolutionary Computation* 3(2), 149–175.