SMART+: A Multi-Strategy Learning Tool*

Marco Botta and Attilio Giordana
Dipartimcnto di Informatica, University di Torino
Corso Svizzera 185,10149 Torino, Italy
Tel. (+39) 11 771-2002, Fax. (+39) 11 751-603
E-mail: {botta,attilio}@di.unito.it

Abstract

Inducing concept descriptions in First Order Logic is inherently a complex task. There are two main reasons: on one hand, the task is usually formulated as a search problem inside a very large space of logical descriptions which needs strong heuristics to be kept to manageable size. On the other hand, most developed algorithms are unable to handle numerical features, typically occurring in real-world data. In this paper, we describe the learning system SMART+, that embeds sophisticated knowledge-based heuristics to control the search process and is able to deal with numerical features. SMART+ can use different learning strategies, such as inductive, deductive and abductive ones, and exploits both backgruond knowledge and statistical evaluation criteria. Furthermore, it can use simple Genetic Algorithms to refine predicate semantics and this aspect will be described in detail. Finally, an evaluation of SMART+ performances is made on a complex task.

1 Introduction

In the recent literature, inducing concept descriptions in First Order Logic is receiving an increasing interest [Bergadano *et* a/., 1988, Michalski, 1980, Quinlan, 1990, Gemello and Mana, 1991, Pazzani and Kibler, 1992].

In general, the task of learning concept descriptions (or relations) is formulated as a search problem inside a space of logical formulas, built up starting from a set of initial predicates, evaluable on the learning events. Two crucial points attract the attention of many researchers: the design of strategies that limit the search space, and the choice of the representation language. As discussed in [Utgoff, 1986] choosing inappropriate features for describing the learning events can prevent the induction algorithm from finding good concept descriptions.

The system SMART+, described in this paper, proposes solutions for both problems and proved to be effective in coping with tasks more complex than the ones approached so far in the literature. SMART+ is an improved version of ML-SMART [Bergadano *et* a/., 1988, Bergadano and Giordana, 1990], and offers sophisticated learning strategies that can either be selected in alternative or combined together, depending on the choice of the teacher. On the one hand, induction can be guided by a domain theory using deductive and/or abductive reasoning. On the other hand, besides the well known information gain rule [Quinlan, 1990], other statistical criteria are available that can be more effective in complex problems.

Moreover, SMART+ offers a mechanism for dealing with numerical features in order to help solving the hard problem of choosing an appropriate concept description language. In particular, it is able to infer a quantitative definition of numerical features according to the context in which those features are used. For instance, suppose we have a symbolic language in which terms such as *short, medium* and *long* are defined to characterize the length of an object: an object is considered *short* if its length is less than a given value k_1 *long* if its length is greater than a given value k2 and *medium* if its length is between k_1 and k2. How to choose values for k1 and k2 depends on the context those features are used in: if we are dealing with bicycles, cars and airplanes, suitable values might be k1 = 7 feet, k2 = 30 feet, but if we are dealing with different kinds of aircrafts (pipers, military jets, passenger cargos), suitable values might be k1 = 50 feet, k2 = 100 feet. In SMART+ the teacher is only requested to provide the system with a range of possible meaningful values for a given feature and the system itself should find the *best* assignments for the task at hand.

The paper is organized as follows: an overview of the knowledge representation formalism in SMART+ is given in Section 2, whereas Sections 3 through 6 describe the available learning strategies. Finally, some experiments on a complex artificial domain, simulating a problem of pattern recognition, will be discussed in Section 7, and conclusive remarks follow in Section 8.

2 Knowledge Representation

SMART+ is a problem solver designed for the task of learning concept descriptions from examples. Given a set Ho of concepts and a set Fo of labelled instances, SMART* generates as output a classification theory described in a Horn clause language L extended with functions, negation and numerical quantifiers. In particular, a well formed formula (wff) in the language L has the form:

* This work has been partially supported by EEC, project BLEARN II,no.7274.

$$H_i \wedge \varphi(t_1, \ldots, t_n) \xrightarrow{w} H_j \quad (1)$$

being $H_i \subset H_j \subset H_0$ predicates denoting sets of concepts and <p a logical formula stating a condition over the terms $t_1, \ldots, t_n$. Expression (1) actually means that if an event f, to be classified, is an instance of a concept h belonging to the set H_i and the condition $\varphi(t_1, \ldots, t_n)$ is true of f, then h is included in Hj. As the set of concepts Hj implied by a rule is, in general, in the body of another rule, the knowledge learned by SMART+ defines a structured classification theory, which can be described as a discrimination graph. In order to deal with noise, partially incorrect classification rules are allowed. The value of the weight w, in (1), is an estimate of the probability that the rule give the correct classification; it is evaluated as the ratio between the number of correct instances matched and the number of total instances matched to the learning events Fo.

Formula $\varphi(t_1, \ldots, t_n)$ is built up by using predicates in a set P, connectives A and -. and quantifiers ATM, ATL and EX. As described in [Yager, 1983], these quantifiers stand for ATMost, ATLeast and EXactly, respectively, and can be considered as an extension of the standard existential quantifier (similar to the numeric quantifiers used in the INDUCE system [Michalski, 1980]).

In order to cope with the fuzziness inherent in real-world data, a continuous-valued semantics is adopted for the predicates in L. For each predicate $p \in \mathbb{P}$, a corresponding semantic functiofp, mapping a numerical base variable to the interval [0,1], must be defined by the teacher in order to specify how to evaluate the truth of p on the learning events. As a matter of fact, *fp* is a function that evaluates the membership μ_p of a numerical feature V_p with respect to a trapezoidal fuzzy set Sp_t The value of S_p can be defined by means of parameters that will occur in the predicate p as variable terms. Terms in a predicate p are subdivided into two categories: objects and parameters. The first ones, denoted by the symbols $x1, \ldots, x_n$, can only be bound to items corresponding to learning instances or components of them, whereas the second ones, denoted by the symbols k1, ..., km, correspond to fuzzy set parameters and can only be bound to numerical constants.

Trapezoidal fuzzy sets are considered and defined as in Figure 1(a), in a way that allows the fuzzy set to be identified by a pair of real parameters, *k1* and k2. Special cases are reported in Figure 1(b) and 1(c), in which open intervals are represented; in these cases, one numerical parameter is sufficient to define the fuzzy set.

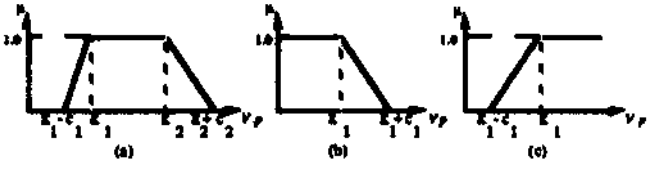

Figure 1 - Example of trapezoidal fuzzy sets used to define the semantics of predicates. The ordinate μ represents the troth value, whereas the abscissa *Vp* represents the values of the base variable.

For each parameter k, the teacher must specify the range in which k has to be searched for and the approximation requested for tuning its value. A default value can also be specified; this will be used by the system in the case that this learning capability is explicitly disabled by the teacher.

The semantics for the logical connectives can be chosen by the teacher, as well, in order to implement a specific evidential calculus. By default, SMART+ adopts a pair of t-norm and t-conorm functions [Bergadano *et* a/., 1988] for the AND and OR connectives, respectively. The semantics of the quantifiers can be expressed by means of these connectives [Yager, 1983, Bergadano *et* a/., 1988].

3 Main Learning Strategy

SMART+ shares with FOIL [Quinlan, 1990] a general-to-specific search strategy for inductive learning, but it has a richer set of specializing operators and uses more sophisticated strategies.

The basic search strategy has been combined with a policy of *reduction to subproblems.* The main goal of the reduction to subproblems is that of producing a structured knowledge base; in particular, it can be considered as a form of *constructive* learning, where intermediate concepts, corresponding to subsets of H_O, are generated automatically. In fact, the classification rules are actually intended to be valid only in the context of already hypothesized concepts, as described in Section 2. Then, the learned rules do not constitute a "flat** set; on the contrary, they are organized into a graph G of rules, called *subproblem graph.* Nodes of G correspond to sets of concepts, and logical formulas label edges. A node (H,F) of G is called a "subproblem", because it represents the problem of discriminating among the classes it contains. The root (H_O,F_O) of the graph corresponds to the initial problem, i.e. the problem of distinguishing among the whole set of concepts. Notice that the sets of examples occurring in sibling nodes need not be disjoint.

A complete and consistent solution of a problem (H_O, F_O) is obtained when all the rule weights are equal to 1, and the leaves of G contain sets F_i whose union equals FQ. This is the ideal case, when no noise and errors are present. However, in real-world problems this assumption cannot be accepted and the final solution turns out to be (in general) inconsistent and/or incomplete.

The use of a subproblem graph, instead of flat classification rules, enhances classification efficiency, since common parts of different rules will only be tested once. Moreover, it makes it easier to learn the rules, because it separates the task into smaller contexts which can be solved separately. Finally, this structure of the knowledge base is well suited for a diagnostic process based on a multi-stage refinement strategy. In Figure 2 the process of generating the subproblem graph is schematically represented.

For each subproblem the learning process follows the same search scheme, as described in the following. The search within a subproblem SP; = (Hi,Fi) develops a *specialization tree* according to a general-to-specific strategy, similar to the one used in FOIL [Quinlan, 1990] or FOCL [Pazzani and Kibler, 1992]. The difference is that many search, strategies, going from best-first to beam-search, are available in addition to the popular hill climbing one. Moreover, SMART+ has a reacher set of operators for

building a more specific formula; in particular it can use *numerical quantification* and *numerical optimization.*

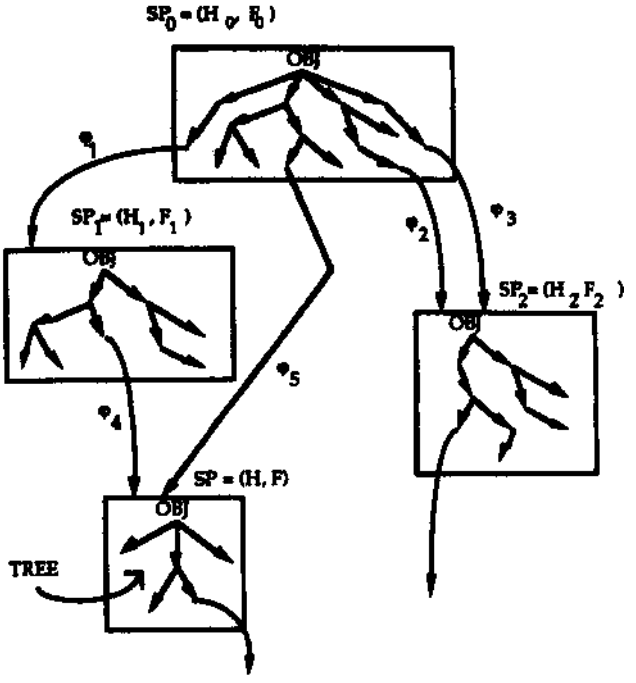

Figure 2 - Subproblem reduction process.

When an inductive hypothesis of the type $\varphi \rightarrow H_j$ is found (being Hj a proper subset of H_i), a rule of type (1) is created and the events $F' \subset F_i$ belonging to the extension of φ are declared "solved". Consequently, the focus of the search is moved on those formulas, on the frontier of the tree, which contain in their extension not yet solved events. The development of the specialization tree is guided by combining reasoning about background knowledge, supplied to the system, and statistical criteria. The primary goal of the control strategy is to find rules that are both statistically significant, because verified by many learning events, and meaningful with respect to the background knowledge. In this phase only consistent rules (i.e. with w=l) are accepted.

The search stops when all the events *Fi* are covered by some rule of type (1) or when there is no more hope of discovering such rules. In this case the system can also decide to accept rules with weight $w<1$. Afterwards, all the events matched by rules having the same conclusion Hj are merged together into a single set Fj and a new subproblem (Hj,Fj) is defined. This process offers an important advantage: in every subproblem, all the events, which were previously subdivided among the extensions of many rules, are OR-ed into a single relation. In this way, the inductive strategy (relaying on statistical criteria) acquires new strength. A deeper description of this search procedure can be found in [Bergadano *et* a/., 1988].

4 Guiding Induction Using Domain Theories

A fundamental characteristic of SMART+ is its ability of guiding induction by exploiting a (possibly incomplete) domain theory. This is achieved by integrating induction with deduction and/or with abduction. The mechanism far integrating induction and deduction has been described in [Bergadano and Giordana, 1990]. The specialization tree, generated by the induction process, has the same structure of the deduction tree generated by a theorem prover based on SLD. Therefore, a domain theory can be used to construct the initial part of the specialization tree, and, then, induction can be used to complete the proof, discovering information missing in the domain theory. If the theory is perfect, the method becomes an EBG [Mitchell *et al.,* 1986, DeJong and Mooney, 1986].

The abductive mechanism has been introduced more recently and, for some aspects, shares some ideas with systems like CIGOL [Muggleton and Buntine, 1988] and CLINT [De Raedt and Bruynooghe, 1991]. Given a Horn clause theory T and a formula 9, associated to a node *n* of the specialization tree, the formula φ^G, obtained by exhaustively applying the absorption rule [Sammut and Bannerji, 1986] between φ and clauses in T, is said to be the *generalization* of <p through the theory T. Formula φ^G is said to be an *explanation* of φ with respect to T, in the sense that T, $\varphi^G \vdash \varphi$ [Poole, 1989, Torasso and Console, 1989]. Therefore, the basic inductive procedure is modified as follows:

(a) Let ψ be the formula selected for specialization at a given step; SMART+ determines the set Λ of literals that can be used to specialize ψ with a non-negative information gain.

(b) For each literal $L_j \in \Lambda$, all the possible generalizations φ^G, with respect to T, are computed for formulas $q_i = \psi \wedge L_j$.

(c) Then, all the formulas φ_i are evaluated, using a criterion which takes into account both the information coming from the extension of φ_i on the learning set F_O and the generalization obtained with respect to the theory T. The best N formulas are then added to the specialization tree. The value of N can be 1, if a greedy strategy is used, or greater than 1 if a beam-search or a best-first strategy is used.

Using this abductive strategy, a domain theory will bias only to a limited extent the choice of a literal, without imposing a determinate choice, as it happens using the purely deductive strategy. Our claim is that this abductive strategy could be a knowledge-based alternative to the one based on determinate literals, introduced to resolve the impasse of greedy algorithms.

5 Search Control strategies

A choice among several search control strategies, ranging from FOIL'S hill climbing to the best-first and beam-search ones, is offered by SMART+ for guiding the inductive process. The evaluation rule for each strategy is also a matter of choice. Given a formula ψ in the specialization tree, the score $s(\psi)$ of ψ is computed as the sum of two independent terms:

$$s(\psi) = a \cdot \mu(\psi) + b \cdot G_T(\psi)$$

where $\mu(\psi)$ is a measure of the quality of a hypothesis ψ, and $G_T(\psi)$ is a measure that tries to capture how we ψ s "explained" by a given domain theory T. The coefficients a

and b can be chosen in such a way to weigh the contribution of the two terms with respect to the global evaluation $s(\psi)$.

In turn, the quality measure $\mu(\psi)$ of a formula ψ is represented as a weighed sum of two components:

$$\mu(\psi) = \alpha * \mu_1(\psi) + \beta * \mu_2(\psi) \quad (3)$$

being $\mu_1(\psi)$ the information gain obtained by φ with respect to its immediate ancestor in the specialization tree and $\mu_2(\psi)$ an evaluation of the completeness and consistency of ψ. By assigning $\alpha=1$ and $\beta=0$ we obtain the information gain rule, whereas by putting $\alpha=0$ and $\beta=1$ we obtain the ML-SMART control rule [Bergadano *et al.*, 1988]. As a matter of fact, the information gain evaluation $\mu_1(\psi)$ has been defined by extending the classical rule in order to deal with many classes at one time.

Two kinds of statistical information are extracted from the extension ψ^* of a formula ψ: the *instance distribution histogram* (denoted by $\mathbf{m}(\psi)$) and of the *object distribution histogram* (denoted by $\mathbf{m}^*(\psi)$) over the concepts in the set H_0. In particular, $\mathbf{m}(\psi)$ is a n-dimensional vector with $n=|H_0|$: each component m_i ($1 \le i \le n$) corresponds to the number of events $f \in F_0$ which are instances of the concept $h_i \in H_0$ and are covered by ψ. Analogously, $\mathbf{m}^*(\psi)$ is an n-dimensional vector, whose generic component m^*_i corresponds to the global number of different bindings in ψ^* between variables in ψ and objects occurring in learning instances of class h_i. Moreover, for each subproblem $sp = \langle H,F \rangle$, an analogous pair of vectors $\mathbf{M}(sp)$ and $\mathbf{M}^*(sp)$, reporting distributions analogous to $\mathbf{m}$ and $\mathbf{m}^*$, evaluated on all the instances in F, are also defined.

Suppose that formula ψ has been obtained by making another formula φ more specific; the problem is how to evaluate the information gain of ψ with respect to φ. First of all, the classes having a non null component in $\mathbf{m}(\psi)$ are partitioned into two groups: the ones which are likely to be classified, and the ones which are likely to be excluded later on. This partition is necessary in order to compute the information gain given by a literal, according to the formula used in FOIL [Quinlan, 1990]. The partition into two groups of classes is described by a Boolean vector $\mathbf{t}(\psi)$, whose generic element $t_i(\psi)$ is set to 1 if $\frac{m_i(\psi)}{M_i(sp)}$ is greater than a user defined threshold v_{min} and the information gain $g_i(\psi) = m^*_i(\psi)*(I_i(\varphi)-I_i(\psi)) > 0$, where

$$I_i(\varphi) = -\log_2\left(\frac{m^*_i(\varphi)}{\sum_{j=1}^{n} m^*_j(\varphi)}\right) \text{ and } I_i(\psi) = -\log_2\left(\frac{m^*_i(\psi)}{\sum_{j=1}^{n} m^*_j(\psi)}\right) \quad (4)$$

and set to 0 otherwise. Vector $\mathbf{t}(\psi)$ is said to be the *target* of ψ.

The first condition asks for the classes in the target to have a still significant amount of instances in order to be meaningful, i.e. not less than a user-defined degree of acceptability. The second condition on the information gain requests that formula ψ has positive support for those classes. Then, the information gain of ψ is computed according to the target $\mathbf{t}(\psi)$ in the following way:

$$\mu_1(\psi) = E^+(\psi) * \left(\log_2\left(\frac{E^+(\psi)}{E^+(\psi)+E^-(\psi)}\right) - \log_2\left(\frac{E^+(\varphi)}{E^+(\varphi)+E^-(\varphi)}\right)\right) \quad (5)$$

$$E^+(\psi) = \sum_{i=1}^{n} \begin{cases} m^*_i(\psi) & t_i=1 \\ 0 & t_i=0 \end{cases}, \quad E^-(\psi) = \sum_{i=1}^{n} \begin{cases} m^*_i(\psi) & t_i=0 \\ 0 & t_i=1 \end{cases}.$$

On the other hand, measure $\mu_2(\psi)$ is computed as follows:

$$\mu_2(\psi) = \frac{e^+(\psi)}{M^+(sp)} * \frac{e^+(\psi)}{e^+(\psi) + e^-(\psi)} \quad (6)$$

where $M^+(sp) = \sum_{i=1}^{n} \begin{cases} M_i(sp) & t_i=1 \\ 0 & t_i=0 \end{cases}$,

$$e^+(\psi) = \sum_{i=1}^{n} \begin{cases} m_i(\psi) & t_i=1 \\ 0 & t_i=0 \end{cases}, \quad e^-(\psi) = \sum_{i=1}^{n} \begin{cases} m_i(\psi) & t_i=0 \\ 0 & t_i=1 \end{cases}.$$

In (6), the first factor accounts for completeness and the second one for consistency.

Finally, the measure $G_T(\psi)$ is obtained by considering the generalization ψ^G of ψ with respect to the theory T and the saturation ψ^S of ψ [Rouveirol and Puget, 1990]. The saturation can be simply obtained by adding to ψ all the literals in the head of the clauses used to abduce ψ^G. Then $G_T(\psi)$ is evaluated as:

$$G_T(\psi) = 1 - (N_{lit}(\psi^G) / N_{lit}(\psi^S)) \quad (7)$$

being $N_{lit}(\psi^G)$ and $N_{lit}(\psi^S)$ the number of literals occurring in ψ^G and ψ^S, respectively. Equation (7) is an empirical estimate of how well ψ is explained by the theory T; in fact $G_T(\psi)$ increases when $N_{lit}(\psi^G)$ decreases, i.e., when simple, non-trivial explanations are found. If there is no theory, i.e., T is empty, $G_T(\psi)$ is always equal to 0, because $\psi^G = \psi^S = \psi$. Then, rule (2) implicitly reduces to a pure statistical evaluation of ψ.

6 Optimising Numerical Predicates

Numerical parameters defining fuzzy set boundaries are learned in two steps. The first one occurs when a formula φ is made more specific using a literal containing some parameter k. In this way, literals are added one at a time and, consequently, parameter values are determined by taking into account only a partial context. Of course, this strategy can lead to a non-optimal choice; hence, a second optimization step, trying to perform a global optimization, can be applied before adding a rule to the subproblem graph. This is done using a simple genetic algorithm [DeJong, 1975].

6.1 Local Optimisation Algorithm

The mechanism for making the local choice for parameters is a simple extension of the one used for adding a literal without parameters. The difference is that a predicate containing parameters does not correspond to a single literal, but rather to the whole set of literals obtainable by sampling the range specified by the teacher, according to the assigned parameter precision. For each literal L_i obtained in this way, the formula $\varphi \wedge L_i$ is evaluated using

l rule (2). Of course, different literals can produce the same (target For each target, the literal with the best evaluation p. ,f (if a hill climbing strategy is used) or the best n ones (if a beam-search strategy is used) are selected for specialisation.

The algorithm for searching the values of the parameters of a predicate p is reported in Figure 3. It receives as input arguments a formula φ, a predicate p and, for each parameter k occurring in p, a triple <min, max, step> describing the range of variability and the granularity of k. A set of suitable values, to be used in order to specialise formula φ with predicate p, is returned as a result

Informally, vectors m and m* are evaluated for all the possible assignments of values to the parameters of p, according to the range and the granularity. Then, the promising m's are identified and the corresponding targets are determined. Finally, for every target, the most suited setting of the parameters is selected. The FLOP algorithm has an exponential complexity in the number of parameters of the predicate p; therefore, a coarse granularity is used in order to limit the size of the number of m and m* to be evaluated.

```
Algorithm FLOP (φ, p, {(min_i, max_i, step_i) : ∀ i ∈ [1..n]})
  ∀ i ∈ [1..n] k_i = min_i;
  i = 1;
  Let BEST be a table whose entries are sets of triples (g, t,
       (k_1,...,k_n)) indexed by a target mask t
  while i ≤ n do
       m=instance distribution histogram for φ∧p(k_1,...,k_n);
       m*=object distribution histogram for φ ∧ p(k_1,...,k_n);
       t = target mask for m and m*
       g = gain(p) with respect to t
       if g > 0 and g > BEST[t]→g then BEST[t] = (g, t,
       (k_1,...,k_n))
       endif
       k_i = k_i + step_i;
       while i ≤ n and k_i > max_i do
            k_i = min_i; i = i + 1;
            if i ≤ n then k_i = k_i + step_i endif
       enddo
       if i ≤ n then i = 1  endif
  enddo
  BESTPARS = {}
  for each entry in BEST do
       add the triple (g, t, (k_1,...,k_n)) to BESTPARS
  endfor
  return (BESTPARS)
```

Figure 3 - Algorithm to find locally optimal parameter assignments.

However, the use of a coarse granularity can lead to inaccuracy in this learning phase. Furthermore, the specialisation process leads to learn parameter values for a given predicate when the ones of other predicates have been already chosen. In many cases this strategy prevents learning optimal values. On the other hand, a contextual optimisation of all the parameters in the multidimensional space of numerical features would be impossible in this learning phase. Therefore, the refinement step described in the following can be useful in order to increase the accuracy of the classification theory.

6.2 Global Optimisation Using a Genetic Algorithm

Genetic Algorithms [Goldberg, 1989] can be an excellent tool for optimising multimodal numerical functions, such as the semantic functions implicitly associated with the classification rules. In the present case a simple Genetic Algorithm of the type proposed by DeJong [DeJong, 1975] has been used and implemented as described in [Goldberg, 1989].

Given a formula φ, all the parameters k_i occurring in φ define a string of reals $rs = \#:[k_1, ..., k_n]$; as a predicate p can occur more than once in φ, for each occurrence of p in φ the parameters of p will have a different occurrence in *rs*. A string *rs* is then converted into a bit string representation in order to use the standard Genetic Algorithm.

In order to prevent the crossover operator from generating values outside the range assigned by the teacher to the parameters, the following conversion algorithm is adopted to map a string of reals *rs* into the corresponding bit string *bs*:

a) Given a parameter k_i and the range $[min_i, max_i]$ assigned by the teacher, k_i is represented as follows:

$$k_i = min_i + \delta_i / \Delta_i \qquad (8)$$

being $0 \le \delta_i \le 2^{N_i}$, where N_i is the number of bits chosen by the teacher for representing the increment δ_i. The value of Δ_i is obtained from N_i according to the following equation:

$$\Delta_i = \frac{2^{N_i}}{(max_i - min_i)} \qquad (9)$$

b) Given a string $rs = \#:[k_1, ..., k_n]$, for each parameter k_i in *rs* a field of N_i bits will be defined in *bs*, where the value of the corresponding δ_i is represented.

Given a formula φ, the Genetic Algorithm is then applied, trying to increase the completeness and the consistency of φ. An initial population $A(\varphi)$ of bit strings is generated at random and the fitness of each individual *bs* is evaluated by computing the extension $\varphi^*(bs)$ and by ranking the resulting formulas according to the evaluation given by (3). Afterwards the usual genetic evolution begins.

Actually, two types of crossovers, c_1 and c_2, are used. Crossover c_1 is the standard *single point crossover*, whereas crossover c_2 is defined as follows: given two parents bs_1 and bs_2, a parameter k_c is selected at random and two offsprings bs_1' and bs_2' are generated in three steps:

a) The parameters to the left of k_c in bs_1' and bs_2' are obtained by copying the parameters in bs_1 and bs_2, respectively.
b) The parameter k_c in both offsprings is obtained by averaging the two corresponding values in the parents.
c) The parameters to the right of k_c are obtained in bs_1' by copying the corresponding ones in bs_2 and viceversa in bs_2'.

The choice between c_1 and c_2 is controlled by two probabilities p_1 and p_2, respectively, such that:

$$p_1 + p_2 \le 1$$

The standard mutation operator is applied with a probability $p_m = 0.001$.

7 Evaluation on a Test Case

The application domain is quite complex, even though artificial, and bears many features of a real-world one. Ten capital letters of the English alphabet have been chosen, a Horn clause theory has been invented and used by a random choice theorem prover in order to generate instances of these letters, some of which are shown in Figure 4. Each letter is represented as a set of segments that are described by the initial and final (x,y) coordinates in a Cartesian plane. From these attributes, other features can be extracted, such as the length of a segment, its orientation, its preceding and following segments, and so on. Some of these features are numerical by nature and then parameterised fuzzy semantics definition have been adopted for the corresponding symbolic literals.

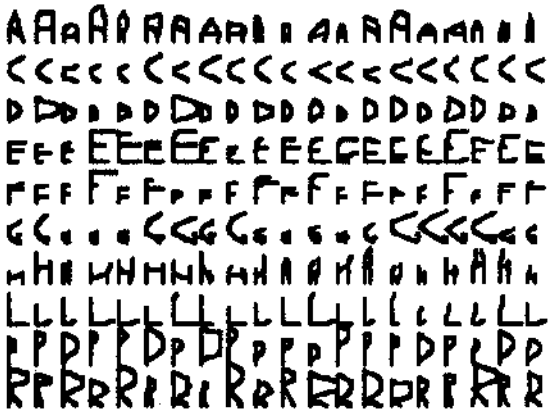

Figure 4 – Some instances taken from the artificial character domain.

SMART+ has been run under many different conditions, by using hill-climbing and beam-search strategies coupled with evaluation rule (2) in which a and b have been both set to 1, μ is given by expression (3) with (α=1,β=0) or (α=0,β=1) depending on the evaluation rule to be applied and $G_T(\psi)$ is 0 when no theory is used. Table I reports the results obtained by using a learning set of 1000 instances (100 per class) in the following cases: (a) using the default values for numerical parameters, as assigned by the teacher and a hill-climbing strategy coupled with information gain selection rule (α=1,β=0+No OPT); (b) performing only a local optimisation according to the algorithm described in Section 6.1 and a hill-climbing strategy coupled with information gain selection rule (α=1,β=0+Local OPT); (c) performing only a local optimisation and hill-climbing search strategy coupled with ML-SMART evaluation rule (α=0,β=1 and Local OPT); (d) performing only a local optimisation, hill-climbing search strategy, rule (2) and using two partial theories: T1 (α=0,β=1, Local OPT and T1), which describes the feature of a closed region in a capitol A, and T2 (α=0,β=1, Local OPT and T2), describing, in addition, the set of segments forming a semicircle. Such theories are very weak and have been proposed by a human expert that was not aware of the theory actually used to generate the characters. As the same results have been obtained by setting α=1,β=0, i.e. by using the information gain rule, they are not explicitly reported. (e) using local optimisation plus the global optimisation performed by Genetic Algorithm, hill-climbing, theory T2 and rule (2) (a=0,B=I, Local+GA OPT and T2).

The second column in Table I refers to the number of formulas in the learned knowledge. The next three columns refer to the recognition rate (the correct label is assigned to a sample), the error rate (cases in which a wrong label is assigned to a sample) and the ambiguity rate (cases in which no label is assigned to a sample), respectively, evaluated on an independent test set of 5000 events (500 per class).

Table I - Results obtained by SMART+ on the test example.

	Number of learned formulas	Recog-nition Rate	Error Rate	Ambiguity Rate
α=1,β=0+No OPT	81	41.48	3.82	54.70
α=1,β=0+Local OPT	43	80.2	1.16	18.64
α=0,β=1 and Local OPT	48	82.62	1.02	16.36
α=0,β=1, Local OPT and T1	43	91.68	1.02	7.3
α=0,β=1, Local OPT and T2	47	98.68	0.12	1.20
α=0,β=1, Local+GA OPT and T2	28	99.70	0.0	0.30

As can be noted from Table I, the use of even incomplete domain theories greatly help the inductive search, by letting the system perform significantly better. Moreover, the capability of learning numerical parameters, i.e. adjusting the concept description language semantics as needed, appears determinant to the success of the application. In fact, using the default values the performances achieved were very poor, whereas by only using local optimisation the recognition rate doubled, and raised up to 98% with the use theory T2. A further significant improvement is then added by the global optimisation algorithm that brings the recognition rate to 99.7%. On the other hand the simplicity and compactness of the knowledge base increases in accordance with the performances, as it is possible to grasp from the values reported in the second column, of course, at the cost of a greater computation time (Smart+ using Local+GA OPT and theory T2 took twice as much as without Genetic Algorithm).

A second experiment concerns the use of theory T2: it should be pointed out that when the domain theory is used to initialize the search tree in an EBL fashion and, then, pure induction is applied, the performances obtained are quite poor, as shown in Table II. This is probably due to the incompleteness of the domain theory which led the system too far in wrong directions. If the theory is also used to bias the inductive search, as done by using rule (2) to score hypotheses, after the initial EBL process, the results are closer to the ones obtained without theory. Thus, abductive bias turned out to be more reliable than the deductive one in the case of weak domain theories.

Table II - Results of Smart+ using an initial EBL process + Inductive Search.

Strategy	Recognition Rate	Error Rate	Ambiguity Rate
EBL+α=1,β=0	64.92	14.54	20.54
EBL+α=0,β=1 and T2	83.44	0.86	15.7

A last comment concerns the comparison between hill-climbing and beam-search strategies, in absence of theory. Even though extensive tests have not been performed at this time, the beam-search strategy seems to be more effective, by letting the system explore a smaller part of the search space but converging more quickly towards robust knowledge.

However, this test is not enough for stating the superiority of one strategy with respect to the other; on the other hand, what can be said (also on the basis of other applications) is that in learning structured concepts a global evaluation rule like (2), associated with a best-first or beam-search strategy, may result in a more effective and simpler knowledge base.

8 Conclusion

As an alternative to the empirical strategy based on determinate literals, we suggested the use of a domain theory to guide the induction algorithm. Such a theory does not need to be complete or consistent and can be limited to a rough description of the structural aspects of the problem. An abductive reasoning mechanism has been embedded in SMART+ in order to guide the induction process. Moreover, a new evaluation rule, combining both information coming from data with information coming from the theory, has been proposed. In this way, bias on induction due to the theory is much more gentle than when a top-down deduction is used as in a previous version of the system.

Furthermore, SMART+ has been equipped with a method for learning fuzzy set values in first order logic environments, which combines an extension of the methodology developed for learning relations with genetic algorithms. In particular, genetic algorithms can be an effective tool for refining numerical parameters such as thresholds, weights and coefficients which control the flexible matching of a symbolic expression against a real world instance. The applicability of the method has been demonstrated on a non-trivial learning problem of pattern recognition.

However, the conclusion we can make from this experiment is that the symbolic approach proposed by Artificial Intelligence can be extended in order to be effective in dealing with patterns of the real world such as complex signals. In particular this approach can compete quite well with other methods, such as Neural Networks, in domains where structural knowledge is relevant and where there exists background knowledge which can be exploited.

References

[Bergadano et al, 1988] F. Bergadano, A. Giordana, and L. Saitta. Automated Concept Acquisition in Noisy Environment. IEEE Transaction on Pattern Analysis and Machine Intelligence, PAMI-10,555-575,1988.

[Bergadano and Giordana, 1990] F. Bergadano and A. Giordana. Guiding Induction with Domain Theories. In Machine Learning: An Artificial Intelligence Approach, III, R.S. Michalski and Y. Kodratoff (Eds), Morgan Kaufmann, 474-492, Los Altos, CA, 1990.

[De Raedt and Bruynooghe, 1991] L. De Raedt and M. Bruynooghe. Towards Friendly Concept-Learners. Proc. of the IJCAI-91,849-854, Sidney, Australia, 1991.

[DeJong, 1975] K.A. Delong. Analysis of the Behavious of a Class of Genetic Adaptive Systems. Ph.D. dissertation. Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI, 1975.

[DeJong and Mooney, 1986] G. DeJong and R. Mooney. Explanation based learning: An alternative view. Machine Learning, 1,47-80,1986.

[Gemello and Mana, 1991] R. Gemello and F. Mana. RIGEL: An Inductive Learning System. Machine Learning, 6,7-36,1991.

[Goldberg, 1989] D.E. Goldberg. Genetic Algorithms. Addison-Wesley Publishing Company, Reading, MA, 1989.

[Michalski, 1980] R.S. Michalski. Pattern recognition as a rule-guided inductive inference. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-2,349-361,1980.

[Mitchell et al., 1986] T. Mitchell, R. Keller, and S. Kedar-Cabelli. Explanation Based Generalization: A Unifying View. Machine Learning, 1,47-80,1986.

[Muggleton and Buntine, 1988] S. Muggleton and W. Buntine. Machine Invention of First-Order Predicates by Inverting Resolution. Proc. of the Fifth International Conference on Machine Learning, 339-352, Ann Arbor, MI, 1988.

[Pazzani and Kibler, 1992] MJ. Pazzani and D. Kibler. The Utility of Knowledge in Inductive Learning. Machine Learning, 9,57-94, 1992.

[Poole, 1989] D. Poole. Explanation and Prediction: An Architecture for Default and Abductive Reasoning. Computational Intelligences, 97-110,1989.

[Quinlan, 1990] R. Quinlan. Learning Logical Definitions from Relations. Machine Learning, 5,239-266,1990.

[Rouveirol and Puget, 1990] C. Rouveirol and J.F. Puget. Beyond Inverse of Resolution. Proc. of the Seventh International Conference on Machine Learning, 122-131, Austin, TX, 1990.

[Sammut and Bannerji, 1986] C. Sammut and R. Banerji. Learning concepts by asking questions. In Machine Learning: An Artificial Intelligence Approach, II, R.S. Michalski, J. Carbonell, and T.M. Mitchell (Eds), Morgan Kaufmann, Los Altos, CA, 1986.

[Torasso and Console, 1989] P. Torasso and L. Console. Diagnostic problem solving. Van Nostrand Reinhold, 1989.

[Utgoff, 1986] P. Utgoff. Shift of Bias for Inductive Concept Learning. In Machine Learning: An Artificial Intelligence Approach, II, R.S. Michalski, J. Carbonell, and T.M. Mitchell (Eds), Morgan Kaufmann, 107-148, Los Altos, CA, 1986.

[Yager, 1983] R. Yager. Quantified Propositions in a Linguistic Logic. International Journal of Man-Machine Studies, 19,195-227,1983.