Tractable Induction and Classification in First Order Logic Via Stochastic Matching

Michèle Sebag
LMS, Ecole Polytechnique
91128 Palaiseau, France
Michele.Sebag@polytechnique.fr

Céline Rouveirol
LRI, Université d'Orsay
91405 Orsay, France
Celine.Rouveirol@lri.fr

Abstract

Learning in first-order logic (FOL) languages suffers from a specific difficulty: both induction and classification are potentially exponential in the size of hypotheses. This difficulty is usually dealt with by limiting the size of hypotheses, via either syntactic restrictions or search strategies.

This paper is concerned with *polynomial induction and use* of FOL hypotheses with no size restrictions. This is done via stochastic matching: instead of exhaustively exploring the set of matchings between any example and any short candidate hypothesis, one stochastically explores the set of matchings between any example and any candidate hypothesis. The user sets the number of matching samples to consider and thereby controls the cost of induction and classification.

One advantage of this heuristic is to allow for resource-bounded learning, without any a priori knowledge about the problem domain.

Experiments on a real-world problem pertaining to organic chemistry fully demonstrate the potentialities of the approach regarding both predictive accuracy and computational cost.

1 Introduction

This paper is concerned with learning from examples expressed in (a restriction of) First-Order Logic (FOL). This language is required to describe examples that include several objects of the same kind (e.g. a car involves several wheels, a molecule involves several atoms), either when these objects cannot be ranked in a canonical way (e.g. atoms in a molecule) or when it makes sense to compare objects of different ranks (e.g. front left and front right wheels).

Learning in FOL, usually referred to as *Inductive Logic Programming* (ILP) (Muggleton & De Raedt, 1994), is receiving growing attention as it appears the only way for machine learning to tackle complex domains such as organic chemistry (King, Srinivasan, & Sternberg, 1995) or natural language (Mooney, 1996). And a number of FOL learners have been developed: *FOIL* (Quinlan, 1990), *ML-Smart* (Bergadano & Giordana, 1990), *FOCL* (Pazzani & Kibler, 1992), *KBG* (Bisson, 1992), *REGAL* (Giordana & Neri, 1994), *PROGOL* (Muggleton, 1995), *RIBL* (Emde & Wettscherek, 1996) and *REMO* (Zucker & Ganascia, 1996) to name a few.

The specific difficulty of ILP is related to the matching step: If a clause involves three literals *part* and the description of a given device involves 40 *parts*, there are 40^3 possible ways (termed matchings) to instantiate the literals in the clause by the literals describing the device. Discriminant induction requires all such matchings to be considered (to ensure the clause discriminates the device); and the same goes for deduction in the worst case (to check whether the clause covers the device). In other words, induction and deduction are exponential in the size of FOL hypotheses. This limitation is dealt with in the literature by inducing short hypotheses, by means of search strategies (Quinlan, 1990; Bergadano & Giordana, 1990; Pazzani & Kibler, 1992; Muggleton, 1995) and/or syntactic restrictions (Giordana & Neri, 1994; Kietz & Lübbe, 1994; Muggleton, 1995).

This paper is interested in polynomial induction and use of FOL hypotheses, with no size restrictions. This is made possible by *stochastic matching*: one only considers some samples of matchings between examples and hypotheses — instead of exhaustively considering all matchings between examples and (short) hypotheses. The considered matchings are constructed by a stochastic sampling mechanism, and the number of samples allowed is supplied by the user. By so-doing, s/he controls both the induction cost (linear in the number of samples) and the quality of the hypotheses (the more samples are considered, the more likely the hypotheses are consistent).

This heuristic is embedded in the Disjunctive Version Space framework (DiVS) (Sebag, 1996), which constructs all consistent hypotheses covering at least one training example. This framework extends that of Version Space (Mitchell, 1982) and likewise implies a single bias, that of the hypothesis language. In particular, as it does not restrict the size of hypotheses, *DiVS* shows intractable for hypothesis languages such as logic programs or constraint logic programs[1] (Sebag & Rouveirol, 1996).

The algorithm combining *DiVS* and stochastic matching is termed *STILL* for *Stochastic Inductive Learning*. Stochastic matching allows *STILL* to learn hypotheses of any size (at most as long as the examples) in polynomial time, and to use them in polynomial time too. Stochastic matching corresponds to a new kind of learning bias (Mitchell, 1991). The stochastic bias contrasts with language and search biases, inasmuch it does not involve any expert knowledge about the problem or about how to find relevant solutions. Rather, it reflects the available computational resources; and the more resources, the better.

This paper is organized as follows. Section 2 briefly presents *DiVS*; the reader is assumed to be familiar with the Version Space (VS) framework (Mitchell, 1982). Section 3 details the stochastic sampling mechanism and outlines the induction and classification algorithms of *STILL*. Experimental validation on the mutagenesis problem (King, Srinivasan, & Sternberg, 1995) is presented in section 4; the influence of the stochastic matching mechanism is discussed, and *STILL* is compared to prominent learners. We conclude with some research perspectives.

2 Disjunctive Version Space

This section recalls how Disjunctive Version Space addresses the limitations of Version Space (failures, exponential complexity). Details on the algorithms are found in (Sebag, 1996; Sebag & Rouveirol, 1996).

2.1 Overcoming VS failures

Basically, VS fails when the maximally specific complete hypotheses (set S) are not more specific than the maximally general consistent hypotheses (set G). But noisy examples lead to over-generalize S and over-specialize G; and examples from a disjunctive concept result in an over-generalization of S if the hypothesis language is conjunctive. This explains why VS fails to handle real-world problems, which include noisy examples and tackle disjunctive concepts most of the time.

VS failures are soundly prevented in three cases: a) when there is no negative example; b) when there is no positive example; and c) when there is exactly one positive example E, provided that E does not belong also to the negative examples (which can easily be checked). The third case is preferred as it is more robust with regard to noise (Sebag, 1996).

The general case, that is, learning from several positive and negative examples, is amenable to the favorable case of a unique positive example by hybridizing Version Space and the AQ algorithm (Michalski, 1983): For each (positive or negative) seed example E, one constructs the *star* $H(E)$ as the set of all consistent hypotheses covering E. If $F_1, ..F_m$ denote the examples not belonging to the same class as E, termed *counter-examples*[2] to E, $H(E)$ exactly is the version space learned from E as unique positive example, and F_i as negative examples.

The disjunction of the stars $H(E)$, for E ranging over the training set, constitutes the Disjunctive Version Space of all consistent partially complete hypotheses.

2.2 Overcoming VS intractability

Building $H(E)$ is intractable even within a propositional language: $H(E)$ is characterized by its lower bound ($S = E$) and its upper bound G, which is the disjunction of an exponential number of conjunctive hypotheses (Haussler, 1988).

DiVS overcomes this limitation by characterizing $H(E)$ as a *conjunction of disjunctions*. This is done as follows. Let $D(E, F_i)$ denote the set of hypotheses covering E and discriminating counter-example F_i. $H(E)$ includes all hypotheses discriminating all F_i, hence it is equal to the conjunction of $D(E, F_i)$, for F_i ranging over the counter-examples to E.

$$H(E) = D(E, F_1) \wedge \ldots \wedge D(E, F_m)$$

In an attribute-value language, $D(E, F_i)$ gets characterized as the disjunction of all maximally general selectors (Michalski, 1983) discriminating E and F_i. Table 1 illustrates how $D(E, F)$ can be built from E and F with linear complexity in the number of attributes; selectors are here restricted to $[att = V]$, where V denotes a value in the domain of a nominal attribute *att*, or an interval in the domain of a linear attribute *att* (selector $[size = (12, +\infty)]$ is written $[size > 12]$ for the sake of simplicity).

Table 1: A pair of examples E and F and D(E, F)

	color	size	shape	weight	class
E	grey	100	wings	3,5 ton	plane
F	white	12	saucer	?	UFO

$D(E,F) = [color = grey] \vee [size > 12] \vee [shape = wings]$

As in most bottom-up approaches, missing values are handled without problems and no preliminary discretization of linear attribute domains is required.

Finally, the complexity of building $H(E)$ is in $\mathcal{O}(N \times P)$, where N denotes the number of examples and P the number of attributes. The whole disjunctive version space is characterized with complexity $\mathcal{O}(N^2 \times P)$.

2.3 Flexible classification with *DiVS*

Classification in *DiVS* much resembles a k-nearest-neighbor (k-NN) classification process: An instance E' is said to be *neighbor* of a training example E iff there exists a hypothesis in $H(E)$ that covers E' (one says for short that E' *belongs to* $H(E)$); and E' is classified according to the majority vote of its neighbors.

[1] Constraint logic programming notably contains the extensions of logic programming concerned with number handling. But this point will not be discussed herein.

[2] The counter-examples to a positive example are the negative examples; and vice versa.

Table 2: FOL and Tabular Representations of E and F

$$\begin{array}{lll} E: & active(a) & :- \ atom(a, a_1, carbon, 22), atom(a, a_2, hydrogen, 3), cc(a, a_1, a_3) \\ F: & inactive(b) & :- \ atom(b, b_1, hydrogen, 2), atom(b, b_2, oxygen, 19) \end{array}$$

$\mathcal{C}$	*active*(X) :−	*atom*(Y	Z	T	U),	*atom*(Y'	Z'	T'	U'),	*cc*(V	W	R).
θ	a	a	a_1	*carbon*	22	a	a_2	*hydrogen*	3	a	a_1	a_3
σ_1	b	b	b_1	*hydrogen*	2	b	b_1	*hydrogen*	2			
σ_2	b	b	b_2	*oxygen*	19	b	b_2	*oxygen*	19			
σ_3	b	b	b_1	*hydrogen*	2	b	b_2	*oxygen*	19			
σ_4	b	b	b_2	*oxygen*	19	b	b_1	*hydrogen*	2			

These neighborhoods, and hence the classification process, can be tuned to cope with noise. Formally, E' belongs to $H(E)$ iff it belongs to $D(E, F_i)$ for all F_i counter-examples to E. This condition is relaxed by allowing ε exceptions (counter-examples F_i such that E' does not belong to $D(E, F_i)$).

Independently, $H(E)$ can also be made more specific to cope with sparse data. Formally, $D(E, F_i)$ is a disjunction of selectors and E' belongs to $D(E, F_i)$ iff it satisfies at least one of these selectors. This is modified by handling from now on $D(E, F_i)$ as an M-of-P concept: E' thereafter belongs to $D(E, F_i)$ iff it satisfies at least M selectors in $D(E, F_i)$.

This way, the set of consistent and partially complete hypotheses is constructed once and for all. Still, classification can employ hypotheses of any degree of consistency and generality — at no extra cost: the complexity of classification is $\mathcal{O}(N^2 \times P)$.

2.4 *DiVS* In First-Order Logic

Let us see how the construction of the set $D(E, F)$ of hypotheses covering E and discriminating F extends to FOL. Let E and F be now described as definite clauses[3], the head of which are built on opposite target predicates (Table 2). Let us express seed E as $E = \mathcal{C}\theta$, where $\mathcal{C}$ is the clause obtained from E by turning every term t_i in E into a distinct variable X_j, and θ is the substitution on $\mathcal{C}$ defined by $\{X_j \ / \ t_i\}$.

Let the hypothesis language be that of constrained clauses $G\rho$, where G generalizes $\mathcal{C}$ and ρ is a conjunction of constraints generalizing θ (a formal presentation is found in (Sebag & Rouveirol, 1996)). Such clauses generalize seed E by construction; they discriminate F iff they do not generalize the clause $\sim F$, built from F by replacing the predicate in its head (e.g. *inactive*) by the opposite target predicate (*active*).

Clause $\mathcal{C}$ allows for a tabular representation of E and $\sim F$ (Table 2). By construction, E is represented by substitution θ. Let the clause built from $\mathcal{C}$ by dropping all predicates absent from F (here *cc*), be still denoted $\mathcal{C}$ by abuse of notation. Then by construction $\sim F$ is subsumed by $\mathcal{C}$, and F is described[4] by the set Σ_F of substitutions σ on $\mathcal{C}$, such that $\mathcal{C}\sigma \subseteq \sim F$. In our example, Σ_F includes four substitutions $\sigma_1 \ldots \sigma_4$, which correspond to the four ways of mapping the two literals *atom* in $\mathcal{C}$ onto the two literals *atom* in F.

This attribute-value reformulation of FOL examples much resembles the *LINUS* and *REMO* approaches (Lavrač & Džeroski, 1994; Zucker & Ganascia, 1996). The difference is twofold. First, the tabular reformulation in *LINUS* and *REMO* operates on the whole dataset; the format of the table is derived from a single clause, specialized if no satisfactory hypothesis is found during the current induction step. Second, the reformulation is one-to-one in *LINUS*, thanks to syntactic restrictions, and it is one-to-many in *REMO* (but the exponential factor is limited by the size of the clause). In contrast, the reformulation in *DiVS* is rather bottom-up than top-down: the format of the table is derived from the current seed example, and one considers at once all information conveyed by the seed. Hence, the reformulation is one-to-one for the seed (E is completely described by θ given $\mathcal{C}$) whereas it is one-to-many for the counter-examples. As the exponential factor is not limited here, *DiVS* turns out to be intractable on real relational problems (see below).

Given this reformulation of E and F, building $D(E, F)$ is amenable to attribute-value discrimination: substitutions on $\mathcal{C}$ can be handled as attribute-value examples and conjunctive constraints on $\mathcal{C}$ can be handled as conjunctions of selectors in the same attribute-value language. Finally, let $\mathcal{P}_F$ denote the set of predicates in E absent from F; then $G\rho$ in the hypothesis language belongs to $D(E, F)$ iff either G includes a predicate in $\mathcal{P}_F$, or ρ belongs to $D(\theta, \sigma)$ for all σ in Σ_F.

Further, an instance E' belongs to $D(E, F)$ iff E' is subsumed by $G\tau$, where G generalizes the body of $\mathcal{C}$ and either G includes a predicate in $\mathcal{P}_F$, or τ belongs to $D(\theta, \sigma)$, for all σ in Σ_F.

3 Stochastic Induction and Classification

This section describes how to construct FOL hypotheses at a polynomial cost, with no size restrictions. Further, these hypotheses can be used at polynomial cost too.

3.1 Stochastic Induction

As seen in section 2.4, *DiVS* can construct hypotheses including as many literals as the seed. The size of the matching set Σ_F can thus be exponential in the size of

[3] The reader interested in learning from constrained clauses is referred to (Sebag & Rouveirol, 1996).

[4] Given the hypothesis language, predicates in F that are absent from E can be omitted with no loss of information.

the examples; e.g. in the mutagenesis dataset, examples are described by up to 40 literals *atoms*; the size of Σ_F hence goes up to 40^{40}...

To overcome this limitation, only a fraction of the substitutions in Σ_F is considered. The number of substitutions to consider, noted η, is supplied by the user, and these substitutions are provided by a stochastic sampling mechanism, as follows. For each literal $p(X_1, X_2, ..)$ in $\mathcal{C}$, a literal $p(t_1, t_2...)$ built on the same predicate is selected with uniform probability in F; substitution σ is iteratively defined by $\sigma = \sigma \cup \{X_i/t_i\}$. More sophisticated sampling mechanisms can also be designed (see section 4.2).

By combining *DiVS* with such a sampling mechanism, *STILL* constructs an approximation of $D(E, F)$, noted $D_\eta(E, F)$:

Definition 1. *Let Σ_η be a set of η substitutions sampled in Σ_F, and let $G\rho$ be in the hypothesis language. $G\rho$ belongs to $D_\eta(E, F)$ iff either G includes a predicate in $\mathcal{P}_F$, or ρ is in $D(\theta, \sigma)$ for all σ in Σ_η.*

By construction, $D_\eta(E, F)$ includes $D(E, F)$ and $D_\eta(E, F)$ goes to $D(E, F)$ as η goes to infinity. The important point is that $D_\eta(E, F)$ is constructed with polynomial complexity: the construction of a sample σ is linear in the number of literals in $\mathcal{C}$, and the construction of $D(\theta, \sigma)$ is linear in the number V of variables in $\mathcal{C}$. As the number of literals is less than V, the complexity of $D_\eta(E, F)$ is $\mathcal{O}(\eta \times V)$.

STILL finally approximates the star $H(E)$ of a seed example E as $H_\eta(E)$, given as the conjunction of $D_\eta(E, F_i)$ for F_i ranging over the counter-examples to E. The Disjunctive Version Space is approximated as the collection of stars $H_\eta(E)$ for E ranging over the training examples. If N still denotes the number of training examples, the complexity of *STILL* induction finally is $\mathcal{O}(\eta \times V \times N^2)$.

3.2 Stochastic Classification

The fact that the hypotheses in $D_\eta(E, F)$ can be *constructed* at a polynomial cost particularly does not imply that they can be *used* at a polynomial cost: as mentioned earlier on, deduction is potentially exponential in the size of FOL hypotheses.

Let E' be the current instance to classify, let the clause obtained from $\mathcal{C}$ by dropping all predicates absent from E' be still denoted $\mathcal{C}$ by abuse of notation, and let $\Sigma_{E'}$ denote the set of substitutions σ on $\mathcal{C}$ such that $\mathcal{C}\sigma \subseteq E'$. Then, checking whether E' is covered by a hypothesis in (for short, *belongs to*) $D_\eta(E, F)$ normally requires to examine all substitutions in $\Sigma_{E'}$; and the size of $\Sigma_{E'}$ is no less than that of Σ_F...

Again, this limitation is overcome by means of stochastic matching: *STILL* only considers K substitutions randomly sampled in $\Sigma_{E'}$.

Definition 2. *Let Σ_K be a set of K substitutions selected in $\Sigma_{E'}$. The instance E' is said to K-belong to $D_\eta(E, F)$ if either E' includes a predicate absent from F or there exists at least one substitution τ in Σ_K such that τ belongs to $D(\theta, \sigma)$ for all σ in Σ_η.*

Note that, if E' K-belongs to $D_\eta(E, F)$, it does belong to $D_\eta(E, F)$; but the converse is not necessarily true. The above definition thus corresponds to a more specific acception of $D_\eta(E, F)$ than the standard logical one. But again, this definition goes to the standard logical definition as K goes to infinity.

The complexity of checking whether E' K-belongs to $D_\eta(E, F)$ is in $\mathcal{O}(K \times \eta \times V)$; and finally the complexity of classification in *STILL* is $\mathcal{O}(K \times \eta \times V \times N^2)$.

4 Experimental Validation

STILL has been experimented and compared to prominent learners on the mutagenesis dataset (King, Srinivasan, & Sternberg, 1995). We also compare the basic stochastic sampling mechanism with a specifically designed sampling mechanism, which incorporates some expert knowledge.

4.1 The Data and Reference Results

The learning goal is to determine among the nitroaromatic molecules occurring in car exhaust fumes, those which might have a carcinogenic effect. The carcinogenicity of a molecule is known to be correlated to its mutagenic activity, but the literature does not yet provide any explicit model for mutagenic activity.

Two descriptions of the mutagenicity problem are available. In a FOL framework, a molecule is represented by a definite clause, the head of which corresponds to its activity (boolean: *active* or *inactive*). The body of the clause describes: a) the atoms of the molecule and the bonds between these atoms; b) the global properties (five attributes) of the molecule, e.g. its hydrophobicity; and c) the chemical structures eventually present in the molecule, e.g. benzenic rings. As witnessed by the size of the matching set (section 3.1), this description is truly relational. It has been processed by *FOIL*, *PROGOL*, and another ILP learner with number handling facilities, *FORS* (Karalic, 1995).
An attribute-value description of the molecules is also available; this second dataset has been processed by linear regression (LR), neural nets (NN) and CART.

Table 3 displays the reference results, obtained by 10-fold crossvalidation and reported from (Srinivasan & Muggleton, 1995) and (Karalic, 1995):

Table 3: Reference Results

	LR	NN	CART	PROGOL	FOIL	FORS
Acc.	89	89	88	88	86	89
±	2	2	2	3	3	6

The computational costs given for *PROGOL*, *FOIL* and *FORS* vary with the description used: *PROGOL* takes from 117,000 to 40,000 seconds (on HP-735), *FOIL* from 9,000 to .5 seconds (HP-735) and *FORS* about 900 seconds (on Sparc-10).

4.2 *STILL* results

In our experimental setting, the dataset was randomly divided into a training set including 90% of the data and a test set, in such a way that the ratio of active/inactive molecules is the same as in the whole dataset (2 to 1). The result was averaged on 25 independent selections of the training/test sets. This procedure is in the same vein as 10-cross fold validation; but a higher number of independent runs is advisable to evaluate stochastic algorithms.

STILL involves 4 parameters: η and K, which respectively control stochastic induction and stochastic deduction; and ε and M, which respectively control the consistency and the generality of the hypotheses (section 2.3), with $\varepsilon = 0$ and $M = 1$ respectively meant as perfect consistency and maximal generality. Parameters η and K are set to 300 and 3; complementary experiments show that doubling η or K increases the predictive accuracy by less than one point. Parameters ε and M respectively vary between 0 and 4 (the value used for *PROGOL*), and 1 and 10.

STILL was experimented with two sampling mechanisms (SM). The first, basic one, was described in 3.1. The "advanced" one uses some naive (authors') knowledge, by rather mapping an atom of a given kind in E, onto an atom of the same kind in F. More precisely, an *atom* is repeatedly selected with uniform probability in $\mathcal{C}$; this atom, say the i-th, is mapped by σ onto the j-atom in F, such that atom j in F is as similar as possible[5] to atom i in E. The substitution σ so defined can be viewed as a "near-miss" with respect to substitution θ.

Table 4 displays the results obtained by *STILL* when combined with both sampling mechanisms. For each value of ε and M, the average predictive accuracy on the test set is given with its standard deviation, as well as the run-time in seconds on a Pentium 166. The predictive accuracy degrades gracefully as ε increases.

Table 4: STILL *results.*

	(a) Advanced SM			(b) Basic SM		
ε	M	Accur.	time	M	Accur.	time
0	6	93.1 ± 2.9	27	1	91.4 ± 5.8	7
0	7	90.6 ± 6.1	27	2	**93.3** ± 4.7	8
0	8	**93.6** ± 4	28	3	93.1 ± 3.3	8
0	9	93.6 ± 4.5	29	4	90.3 ± 6.4	9
2	6	83.9 ± 8.4	27	1	91.1 ± 5.8	7
2	7	85.8 ± 6.4	27	2	92.8 ± 4.4	8
2	8	92.2 ± 4.8	28	3	91.7 ± 5.3	8
2	9	89.4 ± 5.1	29	4	91.7 ± 4.4	9
4	6	83.9 ± 8.4	27	1	77.2 ± 5.1	7
4	7	88.1 ± 6.6	28	2	91.7 ± 5.3	7
4	8	91.1 ± 4.1	28	3	90.3 ± 6	8
4	9	88.1 ± 6.9	29	4	91.9 ± 5.3	9

[5]With same electric charge if possible; otherwise, with same atomtype; otherwise, with same type. The complexity of the advanced SM gets quadratic in the number of literals, instead of linear for the basic SM.

Unexpectedly, *STILL* does not perform better when combined with the "advanced" SM. This fact, rather encouraging with regards to the generality of stochastic induction, can be explained as follows. A substitution σ built by the advanced SM tends to associate atoms in the seed E to the most similar atoms in the counter-example F. As a result, $D(\theta, \sigma)$ lists a few acute discriminant differences, e.g. the electric-charge of atom 17 must be less than .33. In contrast, the basic SM compares any atom in E to any atom in F, and $D(\theta, \sigma)$ therefore lists a number of rough discriminant differences: atom 1 must be a carbon, atom 2 must be an hydrogen, atom 3 must be a carbon,...

During classification, one checks whether an instance E' belongs to $D_\eta(E, F)$, i.e. whether there exists a substitution which satisfies at least M conditions listed in $D(\theta, \sigma)$ for all sampled σ (section 2.3). A substitution built by the advanced SM easily satisfies a few acute conditions: it is sufficient that the concerned atoms in E are considered first, and hence mapped on the similar atoms in E'. Incidentally, this is why high values of M are required in this case (and the computational cost increases with M). In contrast, the ways atoms are "distributed" in E and E' must be close in order for a substitution built by the basic SM to satisfy some out of the discriminant conditions in $D(\theta, \sigma)$ for all sampled σ.

With respect to other ILP learners and on the mutagenesis problem, *STILL* shows quite competitive in terms of predictive accuracy. Further, it is faster by two or three orders of magnitude: e.g. *PROGOL* and *FOIL* process the purely structural description of molecules (atoms and bonds only) in respectively 60,000 and 9,000 seconds (Srinivasan & Muggleton, 1995), whereas *STILL* takes less than two minutes for the same dataset (these times on HP-735 workstation).

5 Discussion and Perspectives

STILL inherits most characteristics of Version Spaces and *DiVS*, notably the absence of restrictions on candidate hypotheses, except consistency and (partial) completeness. This contrasts with most other learners searching for "optimal" candidate hypotheses, no matter whether this optimality refers to the quantity of information, the Gini criterion or the MDL principle (Quinlan, 1990; Bergadano & Giordana, 1990; Muggleton, 1995). It has been suggested that redundancy (and *DiVS* and *STILL* are redundant to an extreme extent) could improve the reliability of the learning output (Webb, 1996). In any case, it avoids the disadvantages of myopic search, such as incurred by decision tree learners.

The price to pay is the readability of the learning output[6]: when expressed as a list (disjunction) of con-

[6]However, *STILL* is definitely not a black box: the classification process constructively exhibits hypotheses relevant to the classification of the current instance. One can thus justify the classification of any instance from an intelligible sub-theory, extracted from the whole theory.

junctive hypotheses, Version Spaces are indeed of exponential size (Haussler, 1988).

Another aspect of *STILL* is that it combines logical aspects and example neighborhoods, in the line of *KBG* (Bisson, 1992), *RISE* (Domingos, 1995) and *RIBL* (Emde & Wettscherek, 1996). The specificity of *STILL* is that it involves neighborhoods which are *constructed by induction*, whereas the above learners rely on a built-in similarity or distance.

But the main originality of *STILL*, due to the stochastic matching heuristic, is to allow a number of expensive hypotheses to be approximately characterized and used. This contrasts with the main trend in ILP, oriented toward the exact characterization of a few affordable hypotheses. Further, stochastic matching allows a fine control of the computational cost, with no expert knowledge on the problem domain.

Note that the use of stochasticity in *STILL* radically differs from what is done in GA-based learners such as *REGAL* (Giordana & Neri, 1994): in *STILL* the stochastic mechanism samples the matching space and it operates as a pre-processor of induction; in contrast, the stochastic mechanism in *REGAL* samples the hypothesis space and so to say replaces[7] induction. Note also that *STILL* does not directly pertain to the Bayesian Inductive Logic Programming framework (Muggleton, 1994), nor to probabilistic induction, in the sense that it does neither assume nor take as input any a priori probability distribution on the hypothesis space.

A main perspective of research is to give *STILL* a PAC model in the sense of Valiant (1984): stochastic induction and deduction approximate standard induction and deduction, and one would like to know how the number of samples η and K relate to the probability for this approximation to be correct.

References

Bergadano, F., and Giordana, A. 1990. Guiding induction with domain theories. In Kodratoff, Y., and Michalski, R., eds., *Machine Learning : an artificial intelligence approach*, volume 3. Morgan Kaufmann. 474–492.

Bisson, G. 1992. Learning in FOL with a similarity measure. In *Proceedings of* 10^{th} *AAAI*.

Domingos, P. 1995. Rule induction and instance-based learning: A unified approach. In *Proceedings of IJCAI-95*, 1226–1232. Morgan Kaufmann.

Emde, W., and Wettscherek, D. 1996. Relational instance based learning. In Saitta, L., ed., *Proceedings of the* 13^{th} *International Conference on Machine Learning*, 122–130.

Giordana, A., and Neri, F. 1994. Search internsive concept induction. *Evolutionary Computation* 3(4):375–416.

Haussler, D. 1988. Quantifying inductive bias : AI learning algorithms and Valiant's learning framework. *Artificial Intelligence* 36:177–221.

[7]But it is true to say that the selection and recombination of hypotheses are based on inductive considerations.

Karalic, A. 1995. *First Order Regression*. Ph.D. Dissertation, Institut Josef Stefan, Ljubljana, Slovenia.

Kietz, J.-U., and Lübbe, M. 1994. An efficient subsumption algorithm for ILP. In Cohen, W., and Hirsh, H., eds., *Proceedings of ICML-94, International Conference on Machine Learning*, 130–137. Morgan Kaufmann.

King, R.; Srinivasan, A.; and Sternberg, M. 1995. Relating chemical activity to structure: an examination of ILP successes. *New Gen. Comput.* 13.

Lavrač, N., and Džeroski, S. 1994. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood.

Michalski, R. 1983. A theory and methodology of inductive learning. In Michalski, R.; Carbonell, J.; and Mitchell, T., eds., *Machine Learning : an artificial intelligence approach*, volume 1. Morgan Kaufmann. 83–134.

Mitchell, T. 1982. Generalization as search. *Artificial Intelligence* 18:203–226.

Mitchell, T. 1991. The need for bias in learning generalizations. In *Readings in Machine Learning*. Morgan Kaufmann. 184–191.

Mooney, R. 1996. ILP for natural language processing. In Muggleton, S., ed., *Proceedings of ILP'96*. Springer-Verlag. forthcoming.

Muggleton, S., and De Raedt, L. 1994. Inductive logic programming: Theory and methods. *Journal of Logic Programming* 19:629–679.

Muggleton., S. 1994. Bayesian inductive logic programming. In Warmuth, M., ed., *Proceedings of COLT-94, ACM Conference on Computational Learning*, 3–11. ACM Press.

Muggleton, S. 1995. Inverse entailment and PROGOL. *New Gen. Comput.* 13:245–286.

Pazzani, M., and Kibler, D. 1992. The role of prior knowledge in inductive learning. *Machine Learning* 9:54–97.

Quinlan, J. 1990. Learning logical definition from relations. *Machine Learning* 5:239–266.

Sebag, M., and Rouveirol, C. 1996. Constraint inductive logic programming. In de Raedt, L., ed., *Advances in ILP*, 277–294. IOS Press.

Sebag, M. 1996. Delaying the choice of bias: A disjunctive version space approach. In Saitta, L., ed., *Proceedings of the* 13^{th} *International Conference on Machine Learning*, 444–452. Morgan Kaufmann.

Srinivasan, A., and Muggleton, S. 1995. Comparing the use of background knowledge by two ILP systems. In de Raedt, L., ed., *Proceedings of ILP-95*. Katholieke Universiteit Leuven.

Valiant, L. 1984. A theory of the learnable. *Communication of the ACM* 27:1134–1142.

Webb, G. 1996. Further experimental evidence against the utility of Occam's razor. *Journal of Artificial Intelligence Research* 4:397–417.

Zucker, J.-D., and Ganascia, J.-G. 1996. Representation changes for efficient learning in structural domains. In Saitta, L., ed., *Proceedings of the* 13^{th} *International Conference on Machine Learning*, 543–551.