In: P. Brazdil (ed.), Machine Learning: ECML-93, Proc of the 6th European Conference on Machine Learning. Workshop on Inductive Logic Program ming, Vienna Austria, LNAI 667, Springer, 1993.

Some lower bounds for the Computational Complexity of Inductive Logic Programming

Jörg-Uwe Kietz
German National Research Center for Computer Science, I3-KI,
P.O.Box 1316, D-5205 St.Augustin, Germany

October 3, 1997

Abstract

The field of Inductive Logic Programming (ILP), which is concerned with the induction of Horn clauses from examples and background knowledge, has received increased attention over the last time. Recently, some positive results concerning the learnability of restricted logic programs have been published. In this paper we review these restrictions and prove some lower-bounds of the computational complexity of learning. In particular, we show that a learning algorithm for i2-determinate Horn clauses (with variable i) could be used to decide the PSPACE-complete problem of Finite State Automata Intersection, and that a learning algorithm for 12-nondeterminate Horn clauses could be used to decide the NP-complete problem of Boolean Clause Satisfiability (SAT). This also shows, that these Horn clauses are not PAC-learnable, unless RP = NP = PSPACE.
Keywords: Inductive Logic Programming, PAC-Learning.

1 Introduction

Most success within the field of Machine Learning has been achieved with systems learning in a propositional logic. Also the theory of learnability, e.g. PAC-learnability, was mostly concerned with propositional logic. But, despite their successes, propositional learning approaches suffer from the limited expressiveness of their hypothesis languages and the lack of background knowledge. Therefore, the field of Inductive Logic Programming (ILP), which is concerned with the induction of first-order Horn clauses from examples and background knowledge, has received increased attention recently [19, 4, 11, 12, 13, 14, 17, 20]. The problem tackled by these approaches can be described formally as follows.

Definition 1 (Learning Problem) *Given:*

- *background knowledge B in a Language LB*
- *positive and negative examples $E = E^+ \cup E^-$ in a language LE consistent with B ($B,E \not\models \Box$) and not a consequence of B ($B \not\models E$).*
- *a hypothesis language LH.*

Find a hypothesis $h \in LH$ such that:

(I) ($B,h,E \not\models \Box$), h is consistent with B and E.

(II) ($B,h \models E^+$), h and B explain E^+.

(III) (B, $h \not\models E^-$), h and B dont explain E^-.

The triple (LB,LE,LH) is called the learning problem*. Deciding whether there exists such an $h \in LH$, is called the* consistency problem*. An algorithm which accepts any $B \in LB$ and $E \in LE$ as input and computes such an $h \in LH$ if it exists, or "no" if it does not exist is called a* learning-algorithm *for (LB,LE,LH). If LB, LE and LH are subsets of first-order clauses, (LB,LE,LH) is called the ILP-problem and a learning algorithm for (LB,LE,LH) is called an ILP-algorithm.*

As the ILP problem has been proven to be undecidable in the general case, where LE, LB and LH are first-order clauses [16], current research focussed on the identification of effective subclasses of first-order logic that are efficiently computable. Recently, some positive results on the PAC-learnability of so called constrained Horn clauses [14] and a superset of them called ij-determinate Horn-clauses (with fixed i and j) [5] have been obtained.

In this paper, we show that these hypothesis languages cannot be further extended without the loss of the PAC-learnability property. To prove non-PAC-learnability, we will use an adaptation of a theorem of Pitt and Valiant, also proven as theorem 6.2.1 in [1].

Corollary 0.1 *If a learning problem (LB,LE,LH) is PAC-learnable, then the consistency problem for (LB,LE,LH) is in RP, or turning it the other way around, if the consistency problem is not in RP, then (LB,LE,LH) is not PAC-learnable.*

In the following we will prove, that the ILP problem for determinate Horn clauses is PSPACE-hard, and therefore i2-determinate Horn clauses are not PAC-learnable in general (where i is variable), unless RP = PSPACE. We will also show, that the consistency problem for 12-nondeterminate Horn clauses is NP-hard, and therefore 12-nondeterminate Horn clauses are not PAC-learnable, unless RP = NP. But, first let us review the proposed restrictions of the ILP problem more formally.

2 Common Restrictions of the ILP Problem

The background knowledge used in ILP programs must always be restricted, otherwise the ILP problem inherits the undecidability of the deduction process. A common restriction in ILP programs (e.g.[12, 17]) is the restriction to ground background knowledge and ground unit clauses as examples.

Definition 2 (Ground Background knowledge) *The Backgound knowledge B is ground, if it consists of ground unit clauses only. A clause is ground if it does not contain any variables.*

One way to avoid such a restriction, is the use of generative Horn clauses and a depth-bounded inference process to generate such finite ground background knowledge prior to learning (e.g. Kietz/Wrobel/92,Muggleton/Feng/92).

Definition 3 (Generative Horn Clause) *A Horn clause is generative, if every variable in the head also occurs in the body of the clause.*

Another common restriction applied not only to B, but also to LH is the restriction to function-free clauses (e.g. [11, 17]).

Definition 4 (Function-Free clause) *A clause is called function-free, if all its arguments are either variables, or constants (function symbols of arity 0).*

The restriction to function-free ground unit clauses as examples and function-free ground background knowledge enables us to use $\theta - subsumption$ as a complete inference procedure in the following way: The learning problem (ground B, ground unit E,LH) as defined in definition 1, is equivalent to the learning problem ($\{\}$,E_{new},LH), where E is defined as $E^{+}_{new} := \{e \leftarrow B \mid e \in E^{+}\}$, and $E^{-}_{new} := \{e \leftarrow B \mid e \in E^{-}\}$. Clearly, the examples are now Horn clauses, and between function-free Horn clauses θ-subsumption is a correct and complete inference procedure ($h \models h'$, iff $h\theta \subseteq h'$)

Function-free knowledge has another positive effect on deduction. It has been proven that inferring ground background knowledge can be done completely in time polynomial to the size of B if B consists only of function-free generative Horn clauses ([21]) and there is a fixed maximum arity of predicates in the background knowledge.

Definition 5 (Bounded-arity Background knowledge) *The language for background knowledge LB is of bounded-arity if there exist an integer j, which is greater than the maximum arity of any of the predicates in B.*

This bounded-arity restriction is also used by Page and Frisch [14] to prove the PAC-learnability of a special kind of hypothesis language LH called *constrained clauses.*

Definition 6 (Constrained clause) *A clause is constrained, if all variables in the body also occur in the head.*

Muggleton and Feng [12], have proposed a restriction of LH to ij-determinate Horn clauses.

Definition 7 (Determinate Clauses and determinate depth of terms) *A Horn clause h is determinate with respect to the examples E and the background knowledge B, if every term t in h is linked by a determinate linking-chain. A term occurring in the head is linked by a determinate linking-chain of length 0. Let $h = \{A, \neg B_1, \ldots, \neg B_m, \neg B_{m+1}, \ldots, \neg B_n\}$ be ordered, the term t found in $\neg B_{m+1}$ is linked by a determinate linking-chain of length d+1, if and only if all terms in $\{A, \neg B_1, \ldots, \neg B_m\}$ are linked by a determinate linking-chain of length at most d and for every substitution θ such that $A\theta \in E^+$ and $\{\{B_1\}, \ldots, \{B_m\}\}\theta \subseteq B$ there is a unique substitution δ whose domain is the variables in t such that $\{B_{m+1}\}\theta\delta \in B_G$. The determinate depth of a term is the minimum length of its determinate linking chains.*

A clause with maximum determinate depth of terms i, and maximum arity of literals j, is called ij-determinate.

Muggleton and Feng have used this ij-determinate restriction to prove that the ILP-problem (Ground Background Knowledge, Ground unit clause examples, one ij-determinate Horn clause). is solvable in polynomial time [12].

Recently, it has been proven [5], that a k-disjunction of function-free nonrecursive ij-determinate Horn clauses is PAC-learnable under simple distributions. This was proven by showing that this ILP-learning Problem can be reduced to an equivalent only polynomial larger propositional learning problem, i.e. learning function-free ij-determinate Horn clauses from ground background knowledge and ground examples is not more powerful than learning in propositional logic. What remains is, that this kind of representation is more compact and therefore maybe more user-friendly in the preparation of the input.

So far, there is no answer to the question, if the restriction to ij-determinate Horn clauses can be relaxed without the loss of polynomial computability of the ILP-problem. This paper gives a negative answer to this question, by proving that the consistency problem for non depth-bounded determinate Horn clauses is PSPACE-hard and that the consistency problem for depth bounded non-determinate Horn clauses is NP-hard.

Definition 8 (Linked Horn clause and depth of terms) *A Horn clause is linked, if all its literals are linked. A literal is linked, if at least one of its terms is linked. A term is linked with linking-chain of length 0, if it occurs in the head of the clause. A term in a literal is linked with a linking-chain of length d+1, if another term in the same literal is linked with by linking-chain of length d. The depth of a term is the minimum length of its linking-chains.*

A not determinate clause with maximum depth of terms i, and maximum arity of literals j, is called ij-nondeterminate.

Clearly, terms in nondeterminate clauses need not have a determinate linking-chain. If a term has a *determinate linking-chain*, this is also a (nondeterminate) *linking-chain*, but not vice versa. Therefore, the *determinate depth* of a term is always greater or equal the (nondeterminate) *depth* of a term.

3 Non-Learnability of Determinate Horn clauses

Theorem 1 *The consistency-problem for non depth bounded 2-determinate Horn clauses is PSPACE-Hard.*

We will prove this theorem, by reducing the following PSPACE-complete problem [6] to the consistency-problem as stated above.

Definition 9 (Finite State Automata Intersection) *Given, an alphabet* Σ *and a sequence* $A_1 \ldots, A_n$*, of deterministic finite state automata with input alphabet* Σ*, does there exist a word* $w \in \Sigma^*$ *accepted by each of the* $A_i, 1 \leq i \leq n$.

First let us recall some basics of deterministic finite state automata (DFA). A DFA A is formally described as a 5-tuple $(Q, \Sigma, \delta, s, F)$, where Q is the set of states, Σ is the input alphabet, $s \in Q$ is the initial state, $F \subseteq Q$ is the set of final states, and δ is the transition function mapping $Q \times \Sigma$ to Q The transition function δ is extended to words on Σ^* as follows: $\hat{\delta}(q, \epsilon) = q$,
$\hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a)$. A DFA is said to accept a word$x \in \Sigma^\star$, iff $\hat{\delta}(s, x) \in F$.

Now, we are able to give a useful encoding of DFAs and words as 2-determinate finite state automata.

Definition 10 (Encoding of DFA's and words as Horn clauses)
Let Γ *be a function from* $\Sigma^* \cup (Q, \Sigma, \delta, s, F)$ *to Horn clauses defined as follows:*
$\Gamma_{word}(x_1 x_2 \ldots x_n) = h(Q_0) \leftarrow x_1(Q_0, Q_1), x_2(Q_1, Q_2), \ldots, x_n(Q_{n-1}, Q_n), f(Q_n)$
$\Gamma_{DFA}(A) = \{h(s)\} \cup \{\neg x(q,p) | x \in \Sigma, p, q \in Q, \delta(q, x) = p\} \cup \{\neg f(q) | q \in F\}$

As is easily seen, DFAs can be encoded as ground 2-determinate Horn clauses, i.e. they are suitable as examples and words are encoded as non-ground 2-determinate Horn clauses, i.e. they are suitable as hypotheses. The usefulness of this encoding is shown by the following lemma.

Lemma 1 *A word is recognized by a DFA if and only if the encoding of the word* θ*-subsumes the encoding of the DFA.*
Proof: $\hat{\delta}(s, x) \in F \Rightarrow \Gamma(x) \leq_\theta \Gamma A$
Let $x = x_1 \ldots x_m$*, then there exist a sequence of states (the computation of* A *on* x*)* $q_0, \ldots, q_m$*, such that* $q_0 = s, q_m \in F$*, and* $\delta(q_{i-1}, x_i) = q_i$*. Let* θ *be* $\{Q_i/q_i\}$*, for* $0 \leq i \leq m$*, from the definition of* Γ *it follows that* $\Gamma(x)\theta \subseteq \Gamma(A)$

$\Gamma(x) \leq_\theta \Gamma(A) \Rightarrow \hat{\delta}(s, x) \in F$
Let θ *be* $\{Q_i/q_i\}$*, for* $0 \leq i \leq m$*, from the definition of* Γ *it follows that* $q_0 = s, q_m \in F$*, and* $\delta(q_{i-1}, x_i) = q_i$*, i.e. the sequence* $q_0 \ldots, q_m$ *is an accepting computation of* x *by* A*. q.e.d.*

Now, we are able to prove, that Γ is indeed an encoding of the DFA intersection problem in terms of the learning problem of determinate Horn clauses.

Lemma 2 *Let $A_1 \ldots, A_n$ a sequence of deterministic finite state automata, then there exists a nonempty word $x \in \Sigma^*$ accepted by each of the $A_1 \ldots, A_n$, if and only if there exists a linked determinate Horn clause consistent with the following set of positive and negative examples: $E^+ = \{H | H = \Gamma(A_i), 1 \leq i \leq n\}$ $E^- = \{h(s) \leftarrow f(s), x_1(s,q), \ldots, x_m(s,q), x_1(q,q), \ldots, x_m(q,q),$ $h(s) \leftarrow x_1(s,s), \ldots, x_m(s,s)\}, where \Sigma = \{x_1, \ldots, x_m\}$*

Proof: *"the if direction"*
Let x be the word accepted by by all the $A_1, \ldots, A_n$. By lemma 2.1 $\Gamma(x)$ is a generalization of E^+. $\Gamma(x)$ is not a generalization of the first example, as x must not be empty. $\Gamma(x)$ is not a generalization of the second negative example as it contains a literal $\neg f(Q)$. Therefore $\Gamma(x)$ is a consistent generalization of the examples. Clearly, $\Gamma(x)$ is linked and determinate.

"the only if direction"
Let h be a linked determinate Horn clause consistent with the examples. As h is a generalization of all positive examples each of its literals are either of the form $h(X)$, $\neg f(X)$, or $\neg x(S1, S2)$, where $x \in \Sigma$. As h is a Horn clause $h(Q_0)$ is in h. As h is consistent with the second negative example it contains a literal $f(Q_n)$, and as h is consistent with the first negative example, the variable $f(Q_n)$ is only unifiable with the head variable Q_0 is this unification also unifies a second argument of an $\neg x(S1, S2)$ literal with the head variable. As h is linked the literal $f(Q_n)$ must be linked by a chain of $\neg x(S1, S2)$ literals. The Horn clause C_w consisting only of the head literal $h(Q_0)$, the final literal $f(Q_n)$, and the linking chain of $\neg x(S1, S2)$ literals is a generalization of h and therefore a generalization of all the positive examples. Using the argument above it is not a generalization of the negative examples. C_w is by construction the Γ encoding of a word. By lemma 2.1 this word is accepted by each of the automata $A_1 \ldots, A_n$. q.e.d.

Corollary 1.1 *Non PAC-learnability of determinate Horn clauses Using corollary 1 and theorem 2 we can conclude, that determinate linked Horn clauses are not PAC-learnable, as long as the widely assumed $RP \neq PSPACE$ conjecture is true.*

4 Non Learnability of nondeterminate Horn clauses

Theorem 2 *Consistency of 12-nondeterminate Horn clauses is NP-hard.*

We will prove this theorem, by reducing the well known NP-complete SAT problem [6] to the consistency-problem as stated above.

Definition 11 (SAT) *Given a set $V = \{v_1, \ldots, v_n\}$ of Boolean variables and a set of clauses $C = \{C_1, \ldots, C_m\}$ over V, the question is whether there exists a truth assignment of V that satisfies the clauses in C.*

Now, let us prove, that a SAT instance is satisfiable, iff a special learning problem is solvable. The idea is to construct positive examples, such that every generalisation consists of truth assignements to Boolean variables of the SAT problem and to construct the negative examples so that only hypotheses whose truth assignments satisfy the SAT problem are consistent with them.

Lemma 3 *The SAT instance (V,C) is satisfiable, if and only if there exists a Horn clause consistent with $E = E^+ \cup E^-$ defined as follows:*
Define a set of 2n 1-place predicates $a_1 \ldots, a_{2n}$, each of which are mapped via Ψ to one of the 2n literals $v_1, \ldots, v_n, \bar{v}_1, \ldots, \bar{v}_n$. The positive examples are then coded as following: $E^+ = \{p(X) \leftarrow$
$p(X, X_1), a_1(X_1), \ldots, a_{i-1}(X_1), a_{i+1}(X_1), \ldots, a_{2n}(X_1)$
$p(X, X_2), a_1(X_2), \ldots, a_{n+i-1}(X_2), a_{n+i+1}(X_2), \ldots, a_{2n}(X_2)$
$\}$, for all i: $1 \leq i \leq n$
The negative example is coded as following. $E^- = \{p(X) \leftarrow p(X, X_j), a_i(X_j) \mid \Psi(a_i) \notin C_j, 1 \leq i \leq 2n, 1 \leq j \leq m\}$

Proof: "the if direction"
Assume, that (V,C) is satisfiable. Let L be the set of literals true in an assignment that satisfies (V,C) such that either v_i or $\neg v_i$ is in L, but not both. Let h be a Horn clause defined by

$$h = \{p(X) \leftarrow p(X, Y), a_i(Y) \mid \Psi(a_i) \in L, 1 \leq i \leq 2n\}$$

h θ-subsumes all positive examples, as for every i $1 \leq i \leq n$ it contains either $a_i(X_1)$ or $a_{n+i}(X_1)$, but not both. We now prove by contradiction that h does not θ-subsume the negative example. If h θ-subsumes the negative example, there is an $\theta = \{X/X, Y/X_j\}$, such that

$$h\theta \subseteq \{p(X) \leftarrow p(X, X_j), a_i(X_j) \mid \Psi(a_i) \notin C_j, 1 \leq i \leq 2n\}$$

But, this means that h contains only a_i, such that $\Psi(a_i)$ is not in C_j, which contradicts the assumption that L satisfies all clauses of C.

"the only if direction"
Let h be a Horn clause that is consistent with the positive and negative examples. As h is a generalization of all positive examples, it is of the form

$$p(X) \leftarrow p(X, Y_k), a_1(Y_k), \ldots, a_{2n}(Y_k)$$

where $1 \leq k \leq 2^n$, and for all $i : 1 \leq i \leq n$, at most one of $a_i(Y_k)$ or $a_{n+i}(Y_k)$ is in h. As h does not θ-subsume the negative example, there exists at least one k such that

$$h' = p(X) \leftarrow p(X, Y_k), a_1(Y_k), \ldots, a_{2n}(Y_k) h' \subseteq h$$

and there is no substitution $\theta = X/X, Y_k/X_j$ such that

$$h'\theta \subseteq \{p(X) \leftarrow p(X, X_j), a_i(X_j) \mid \Psi(a_i) \notin C_j, 1 \leq i \leq 2n$$

This means, for every $C_j \in C$ h' contains at least one literal $a_i(Y_k)$, such that $\Psi(a_i) \in C_j$. This means $\{\Psi(a_i) \mid a_i(Y_k) \in h'\}$ is a partial truth assignment for V, that satisfies all clauses in C. q.e.d.

Corollary 2.1 *Using corollary 1 and theorem 2 we can conclude, that 12-nondeterminate linked Horn clauses are not PAC-learnable, as long as the widely assumed $RP \neq NP$ conjecture is true.*

5 Conclusion

In this paper, we have proven some lower bounds of the comptutational complexity of learning within the ILP framework. This results show, that wide classes of Horn clauses cannot be learned by an complete and efficient learning algorithm. This includes e.g. CLINTs [4] languages greater equal L2, the hypohesis space searched by the inverse resolution approaches e.g. [13, 20]. It also shows that fast heuristic search algorithms like FOIL [18] are necessarily incomplete, i.e they do not find a correct definition, even if one exists. Theorem 1 also shows, that the extension of FOIL2 [17], which adds determinate literals, if no good information gain can be achieved by a literal could lead to an explosion of the size of the hypothesis exponential in the number of times this happens.

Acknowledgements: I would like to thank Katharina Morik and Bernhard Nebel for the discussions and suggestions they have provided to this work. I would also like to thank the DFKI for the research visit, they have supported. This work is partially supported by the European Community ESPRIT program under contract number P6020 "Inductive Logic Programming".

References

[1] M. Anthony and N.Biggs. *Computational Learning Theory.* Cambridge University Press, 1992.

[2] Gilles Bisson. Learning of rule systems in a first order representation. Rapport de Recherche 628, LRI, University de Paris-Sud, 1991.

[3] Wray Buntine. Generalized subsumption and its applications to induction and redundancy. *Artificial Intelligence*, 36:149 – 176, 1988.

[4] L. de Raedt and M. Bruynooghe. An overview of the interactive concept-learner and theory revisor clint. In S. Muggleton, editor, *Inductive Logic Programming.* Academic Press, 1992.

[5] S. Džeroski, S. Muggleton, and S. Russell. Pac-learnability of determinate logic programs. In *Proc. of the 5th ACM Workshop on Computaional Learning Theory (COLT)*, 1992.

[6] Michael R. Garey and David S. Johnson. *Computers and Intractability - A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, Cal., 1979.

[7] David Haussler. Learning conjunctive concepts inb structural domains. *Machine Learning*, 4(1):7–40, 1989.

[8] Nicolas Helft. Inductive generalisation: A logical framework. In *Proceedings of the 2nd European Working Session on Learning*, 1987.

[9] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison Wesley, 1979.

[10] Jörg-Uwe Kietz. A comparative study of structural most specific generalizations used in machine learning. In *Proc. of Workshop W18 at the 10th ECAI-92: Logical Approaches to Machine Learning.*, 1992.

[11] Jörg-Uwe Kietz and Stefan Wrobel. Controlling the complexity of learning through syntactic and task-oriented models. In S. Muggleton, editor, *Inductive Logic Programming*, pages 107–126. Academic Press, 1992.

[12] Stephan Muggleton and Cao Feng. Efficient induction of logic programs. In S. Muggleton, editor, *Inductive Logic Programming.* Academic Press, 1992.

[13] Stephen Muggleton and Wray Buntine. Machine invention of first-order predicates by inverting resolution. In *Proc. Fifth Intern. Conf. on Machine Learning*, Los Altos, CA, 1988. Morgan Kaufman.

[14] C. D. Page and A. M. Frisch. Generalisation and learnability: a study in constained atoms. In S. Muggleton, editor, *Inductive Logic Programming.* Academic Press, 1992.

[15] Gordon D. Plotkin. A note on inductive generalization. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 5, chapter 8, pages 153–163. American Elsevier, 1970.

[16] Gordon D. Plotkin. A further note on inductive generalization. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 6, chapter 8, pages 101–124. American Elsevier, 1971.

[17] J. R. Quinlan. Learning logical definitions from relations. In *Proc. of the 12th IJCAI*, 1991.

[18] J.R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3):239 – 266, 1990.

[19] Celine Rouveirol. Completness for inductive procedures. In *Proc. Eigth Intern. Workshop on Machine Learning*, pages 452–456, 1991.

[20] Celine Rouveirol. Semantic model for induction of first order theories. In *Proc. 12th International Joint Conference on Artificial Intelligence*, 1991.

[21] Stefan Wrobel. Higher-order concepts in a tractable knowledge representation. In K. Morik, editor, *GWAI-87 11th German Workshop on Artificial Intelligence*, Informatik-Fachberichte Nr. 152, pages 129 – 138, Berlin, New York, Tokyo, October 1987. Springer.