Journal of Machine Learning Research 4 (2003) 369-383 Submitted 11/01; Published 7/03

An Empirical Study of the Use of Relevance Information in Inductive Logic Programming

Ashwin Srinivasan ASHWIN@COMLAB.OX.AC.UK
Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD, United Kingdom.

Ross D. King RDK@ABER.AC.UK
Department of Computer Science,
University of Wales, Aberystwyth, Wales, United Kingdom.

Michael E. Bain MIKE@CSE.UNSW.EDU.AU
School of Computer Science and Engineering,
University of New South Wales, Kensington, Australia.

Editors: Richard Dybowski, Kathryn Blackmond Laskey, James Myers and Simon Parsons

Abstract

Inductive Logic Programming (ILP) systems construct models for data using domain-specific background information. When using these systems, it is typically assumed that sufficient human expertise is at hand to rule out irrelevant background information. Such irrelevant information can, and typically does, hinder an ILP system's search for good models. Here, we provide evidence that if expertise is available that can provide a partial-ordering on sets of background predicates in terms of relevance to the analysis task, then this can be used to good effect by an ILP system. In particular, using data from biochemical domains, we investigate an incremental strategy of including sets of predicates in decreasing order of relevance. Results obtained suggest that: (a) the incremental approach identifies, in substantially less time, a model that is comparable in predictive accuracy to that obtained with all background information in place; and (b) the incremental approach using the relevance ordering performs better than one that does not (that is, one that adds sets of predicates randomly). For a practitioner concerned with use of ILP, the implication of these findings are two-fold: (1) when not all background information can be used at once (either due to limitations of the ILP system, or the nature of the domain) expert assessment of the relevance of background predicates can assist substantially in the construction of good models; and (2) good "first-cut" results can be obtained quickly by a simple exclusion of information known to be less relevant.

Keywords: ILP, expert-assistance, relevance of background predicates

1. Introduction

The use of background knowledge to construct explanations is a distinctive feature of Inductive Logic Programming (ILP) systems. Using domain-specific background knowledge, such systems have constructed models for data in software engineering (Bratko and Grobelnik, 1993), stress analysis in engineering (Dolsak and Muggleton, 1992), environmental monitoring (Dzeroski et al., 1994), electronic circuit diagnosis (Feng, 1992), molecular biology and drug design (King et al., 1992, 1996, Muggleton et al., 1992), and natural language processing (Zelle and Mooney, 1993). Crucial to these "real-world" applications has been the inclusion of human expertise, that has, in each case, decided the background information to be used. This sorting of background knowledge—

into relevant and irrelevant—can be extremely important: experiments by Quinlan (1993) suggest that background knowledge that contains large amounts of information that is irrelevant to the problem being considered can, and typically does, hinder an ILP system in its search for a model. Sufficiently skilled human experts may be capable of more than such a coarse sorting of background knowledge. It may be possible, for example, to rank sets of background predicates into some partial-ordering based on relevance to the analysis task. This is particularly the case in scientific domains, where the expert can often call upon a sufficiently large corpus of inter-related experimental and theoretical results ("previous results suggest that the Ames test is very good indicator of carcinogenicity... or "the presence of a zinc-binder is important as we know the target is a metallo-protease"). To the best of our knowledge, there have been no attempts to use or to evaluate the merit of incorporating such expertise into ILP.

The most satisfactory solution to the problem would commence with a refinement to the theoretical specification for an ILP system. Currently, systems for constructing predictive models employ background knowledge encoded as logic programs. Incorporating relevance information of the kind we have outlined would require background information to be encoded using as some form of "labelled" logic program, in the sense described by Gabbay (1991). There, any sentence S can be annotated further to yield a labelled sentence $l : S$. Given additional semantic and proof-theoretic machinery for manipulating labelled sentences, we would then be in a position to contemplate an ILP system that could use such labelled logic programs as background knowledge (the labels would specify the degree of relevance of the corresponding sentences to the modelling task).

While some advances have been made into extending the representation language underlying ILP from logic programs to ones that explicitly incorporate labels (Cussens, 2000, Muggleton, 1996, 2000), we are still some distance from a mature theoretical framework and even further from efficient implementations. In their absence, we investigate a simple extension to current implementations directed at incorporating relevance information. The implementation extension is in the form of an incremental procedure that adds background predicates in decreasing order of relevance. The investigation is in the form of an empirical study using two real-world scientific analysis tasks previously addressed by ILP systems. For each of these, domain expertise provides an ordering of the form described. The experiments are designed to examine the support, if any, for the conjecture that an expert-provided relevance ordering over background predicates can significantly (statistically speaking) improve the performance of an ILP system.

This paper is organised as follows. Section 2 contains a short description of ILP systems for constructing predictive models. Section 3 describes the procedural extension to account for a relevance ordering over background knowledge. The experimental study is in Section 4: Section 4.1 clarifies its precise scope; materials available, including descriptions of the biochemical domains used, are in Section 4.2; the design of experiments in Section 4.3; and results in Sections 4.4. Section 5 concludes this study.

2. Predictive ILP

The term "predictive ILP" has been used to describe the function of a class of programs that construct theories for discriminating accurately amongst two sets of examples ("positive" and "negative"). The partial specifications provided by Muggleton (1994) have been adopted as the basis for deriving programs in this class. With some minor changes to the formulation by Muggleton (1994), an ILP algorithm for constructing predictive theories is taken to be one that conforms to at least the following (we refer the reader to Nienhuys-Cheng and de Wolf, 1997, for definitions in logic programming).

- B is background knowledge consisting of a set of definite clauses $= \{C_1, C_2, \ldots\}$
- I is an optional set of constraints on acceptable hypotheses
- E is a finite set of examples $= E^+ \cup E^-$ where:
  - *Positive Examples.* $E^+ = \{e_1, e_2, \ldots\}$ is a set of definite clauses;
  - *Negative Examples.* $E^- = \{\overline{f_1}, \overline{f_2} \ldots\}$ is a set of Horn clauses; and
  - *Prior Necessity.* $B \not\models E^+$
- $H = \{D_1, D_2, \ldots\}$, the output of the algorithm given B, I and E, is from a predefined language $\mathcal{L}$. A model H is acceptable if the following conditions are met:
  - *Weak Sufficiency.* Each D_i in H is a definite clause that has the property $B \cup \{D_i\} \models e_1 \vee e_2 \vee \ldots$, where $\{e_1, e_2, \ldots\} \subseteq E^+$
  - *Strong Sufficiency.* $B \cup H \models E^+$;
  - *Weak Consistency.* $B \cup H \not\models \Box$; and
  - *Strong Consistency.* (1) $B \cup H \cup E^- \not\models \Box$; and (2) $B \cup H$ is consistent with the I

Any acceptable model H is sometimes said to "explain" the examples. The first requirement under Strong Consistency ensures that H does not contain any "over-general" clauses. Often, implementations do not require clauses to meet this requirement, as some members of E^- are taken to be "noisy". This specification is then refined to include a parameter whose value sets a lower bound on the accuracy required of each clause D_i in the theory.[1] We have deliberately left the second requirement imprecise to allow a flexible interpretation of the I.

The reader would have noted that this specification does not state how the D_i are to be constructed. This is an implementation-dependent detail and some variant of the greedy cover set procedure in Figure 1 is the most popular (reproduced with minor changes from Srinivasan, 1999).

In the implementation described, the function *generalise*$(\ldots)$ does not discriminate amongst background predicates provided to it. It is straightforward to adapt it to include a partial-ordering on equivalence classes of background predicates. This adaptation is described in the following section.

3. Selection of Predicates Using a Relevance Ordering

Srinivasan (2001) describe a procedure for stepwise selection of background predicates. The goal of the procedure is to identify sets of background predicates that can be used to construct optimal models under different decision-theoretic conditions. There, expertise was confined to providing an equivalence relation over the sets of predicates resulting in a partitioning into equivalence classes. Here, we incorporate additional expertise in the form of a partial-ordering on equivalence classes

1. For any clause $D_i \in H$, let $E_p \subseteq E^+$ s.t. $B \cup \{D_i\} \models E_p$. Also let $E_n \subseteq E^-$ s.t. for all $e_n \in E_n$, $B \cup \{D_i\} \cup \{e_n\} \models \Box$. It is usual to call E_p, E_n the positive and negative examples "covered" by D_i. $|E_p| + |E_n|$ is simply the "coverage" of D_i, and $|E_p|, |E_n|$ are its positive and negative coverages respectively. The accuracy of D_i is then $|E_p|/(|E_p| + |E_n|)$.

generalise$(B, I, \mathcal{L}, E)$: *Given background knowledge B, hypothesis constraints I, a finite training set* $E = E^+ \cup E^-$, *returns a model H in* $\mathcal{L}$ *such that H explains the E.*

1. $i = 0$
2. $E_i^+ = E^+$, $H_i = \emptyset$
3. *if* $E_i^+ = \emptyset$ *return* H_i *otherwise continue*
4. *increment i*
5. $Train_i = E_{i-1}^+ \cup E^-$
6. $D_i = search(B, H_{i-1}, I, \mathcal{L}, Train_i)$
7. $H_i = H_{i-1} \cup \{D_i\}$
8. $E_p = \{e_p : e_p \in E_{i-1}^+ \ s.t. \ B \cup H_i \models \{e_p\}\}$.
9. $E_i^+ = E_{i-1}^+ \backslash E_p$
10. *Go to Step 3*

Figure 1: An ILP implementation. The function *search*$(\ldots)$ in Step 6 is some search procedure that returns one clause from all candidates in $\mathcal{L}$. The requirement that D_i be consistent with all the negative examples may be weakened by specifying a lower bound on the accuracy of any D_i. This would then have to be provided as an additional parameter to the search procedure.

of predicates to direct a simple incremental search procedure.[2] Unlike the procedure of Srinivasan (2001), a single model results: the ordering therefore applies to the data analysis task as a whole (the problem and the cost conditions that prevail). The procedure is in Figure 2.

In the experiments that follow, we distinguish between the "informed" incremental strategy embodied in Figure 2 and the "uninformed" incremental strategy in Figure 3.

4. Empirical Evaluation

The incremental procedure of the previous section is evaluated empirically on two real-world biochemical datasets. For each, sufficient human expertise exists to not just identify relevant background information, but also provide a partial ordering, based on relevance to the analysis task, over equivalence classes of background predicates.

2. Providing a partial-ordering on equivalence classes of background predicates is tantamount to defining a quasi-ordering on the predicates themselves. The following results concerning orderings are relevant. (1) A quasi-order Q in a set S is a dyadic relation over S that satisfies the properties: for every $a \in S, aQa$ (reflexivity) and if aQb and bQc then aQc (transitivity); (2) For every Q we can define an equivalence relation E as follows: aEb iff aQb and bQa; (3) Any equivalence relation E over a non-empty set S results in a partition of S into disjoint subsets (equivalence classes); (4) any quasi-order Q results in a partial order P over a set of equivalence classes of elements in S; and (5) for every partial ordering P over a set S it is possible to construct a total ordering T or chain over S that is consistent with P.

generalise'$(B,P,I,\mathcal{L},E)$: *Given background knowledge B, a partial-ordering P over equivalence classes of predicates in B, hypothesis constraints I, a finite training set* $E = E^{+} \cup E^{-}$*, returns a model H in* $\mathcal{L}$ *such that H explains the E.*

1. *Let there be k equivalence classes and* $< b_1, b_2, \ldots, b_k >$ *be a total ordering over them that is consistent with P*
2. $i = 0$
3. $H_i = \emptyset, B_i = \emptyset$
4. *if* $i \geq k$ *return* H_i *otherwise continue*
5. *increment i*
6. $B_i = B_{i-1} \cup b_i$
7. $H_i = generalise(B_i, I, \mathcal{L}, E)$
8. *if* $H_i \sim H_{i-1}$ *return* H_{i-1} *otherwise go to Step 4*

Figure 2: An incremental ILP implementation guided by an ordering over disjoint subsets of background predicates. The function *generalise*$(\ldots)$ in Step 7 is as in Figure 1. The relation $\sim$ in Step 8 holds if the models compared are "equivalent". This is usually in the sense of there being no significant difference in estimates of some statistic (like predictive accuracy) measured for each model. Correctly, the procedure should either be given a quasi-ordering over *B*, from which the equivalence classes can be obtained, or be provided with the equivalence relation. We have ignored this for simplicity.

4.1 Experimental Aim

Our intention is to investigate support for the conjecture presented in Section 1, namely that an expert-provided relevance ordering over background predicates can significantly improve the performance of an ILP system. The "null hypothesis" for the experiment is thus:

H_0 : The expert-provided relevance ordering over background predicates has no significant effect on the performance of an ILP system.

For the purposes of this study, the performance of an ILP system will be given by the pair of numbers (A, T) , denoting respectfully, estimates of the predictive accuracy of the model constructed and the time taken to construct the model.[3]

4.2 Materials

Below, we introduce the domains, background knowledge and algorithms used in the remainder of the paper.

3. "Explanatory value" is an important attribute for any model, and should warrant inclusion in the performance measure. For Muggleton et al. (1998) the explanatory value of a model is a Boolean property defined by the use, or otherwise, of particular types of descriptors by the model. There has however been no consensus on this property and we have confined ourselves to the clearly measurable properties of accuracy and time.

generalise$''(B,P,I,\mathcal{L},E)$: *Given background knowledge B, a partial-ordering P over equivalence classes of predicates in B, hypothesis constraints I, a finite training set* $E = E^+ \cup E^-$, *returns a model H in* $\mathcal{L}$ *such that H explains the E.*

1. *Let* $b_1, b_2, \ldots, b_k$ *be the k equivalence classes over which P is defined*
2. $i = 0$
3. $H_i = \emptyset, B_i = \emptyset$
4. $B'_i = \{b_1, b_2, \ldots, b_k\}$
5. *if* $i \geq k$ *return* H_i *otherwise continue*
6. *increment i*
7. $s_i = random(B'_{i-1})$
8. $B_i = B_{i-1} \cup s_i$
9. $B'_i = B'_{i-1} \backslash \{s_i\}$
10. $H_i = generalise(B_i, I, \mathcal{L}, E)$
11. *if* $H_i \sim H_{i-1}$ *return* H_{i-1} *otherwise go to Step 4*

Figure 3: An incremental ILP implementation that adds subsets of background predicates at random. The function *random*(S) in Step 7 randomly selects an element from S. Other functions and relations are as in Figure 2.

4.2.1 DOMAINS

The following is adapted from Srinivasan (2001).

Mutagenesis. Models are to be constructed for discriminating amongst nitroaromatic chemicals with varying mutagenic activity.[4] The data pertain to chemicals examined by Debnath and colleagues (Debnath et al., 1991), with the view of constructing a linear model to predict levels of biological activity using molecular properties as regressor variables. These properties were extended by King et al. (1996), Srinivasan et al. (1996) to include general structural information (atoms, bond connectivity, *etc.*: described in the following section). The original chemical study listed the mutagenic activity of 230 compounds. These were taken to be composed of two disparate groups of 188 and 42. Each compound has an associated mutagenicity value obtained from a procedure known as the Ames test. We concentrate only on the subgroup of 188 compounds, as these are sufficient for the purposes of the study here. The classification of compounds is binary: those with positive log mutagenicity are labelled “active” and those which have zero or negative log mutagenicity are labelled “inactive”. Of the 188 chemicals, 125 (67%) are active.

Carcinogenesis. The data pertain to chemicals undergoing rodent carcinogenicity tests conducted within the U.S. National Toxicology Program (NTP). These tests are conducted on a very wide range of chemicals and the data describing these chemicals have been used to construct models to discriminate true carcinogens from a pool of potential carcinogens (this is harder than discriminat-

4. Mutagenic chemicals mutate DNA sequences. Nitroaromatics occur in automobile exhaust fumes and are also common intermediates in the synthesis of many thousands of industrial compounds. Highly mutagenic nitroaromatics have been found to be carcinogenic and it is of considerable interest to the chemical and pharmaceutical industry to determine which molecular features result in compounds having mutagenic activity.

ing the former from non-carcinogens, Benigni, 1998). Outcome from rodent tests with approximately 300 chemicals are available. As with the mutagenesis problem, the data on bulk molecular descriptors have been augmented with structural information (see the following section) and have previously formed the basis for a popular "challenge" problem for symbolic machine learning techniques (Srinivasan et al., 1997, 1999). Compounds are again classified in one of two classes: "true carcinogens" and "others". The dataset refers to 337 chemicals of which 182 are classified as true carcinogens (54%).

4.2.2 Background Knowledge and Relevance Ordering

Data constituting the background knowledge for mutagenesis is contained in the definition of 29 predicates, partitioned into the following equivalence classes:

- M0. Molecular description at the atomic level. This includes the atom and bond structure, the partial charges on atoms, and arithmetic constraints (equalities and inequalities) on these charges. There are 5 predicates in this group;

- M1. Structural properties identified by experts as being specifically relevant to the mutagenic activity of the chemicals considered in this dataset. These are: the presence of three or more benzene rings, and membership in a class of compounds called acenthrylenes. There are 2 predicates in this group;

- M2. Chemical properties identified by experts as being generally relevant to mutagenic activity, along with arithmetic constraints (equalities and inequalities) on these properties. The chemical properties are: the energy level of the lowest unoccupied molecular orbital ("LUMO") in the compound, an artificial property related to this energy level (Debnath et al., 1991), and the hydrophobicity of the compound. There are 6 predicates in this group;

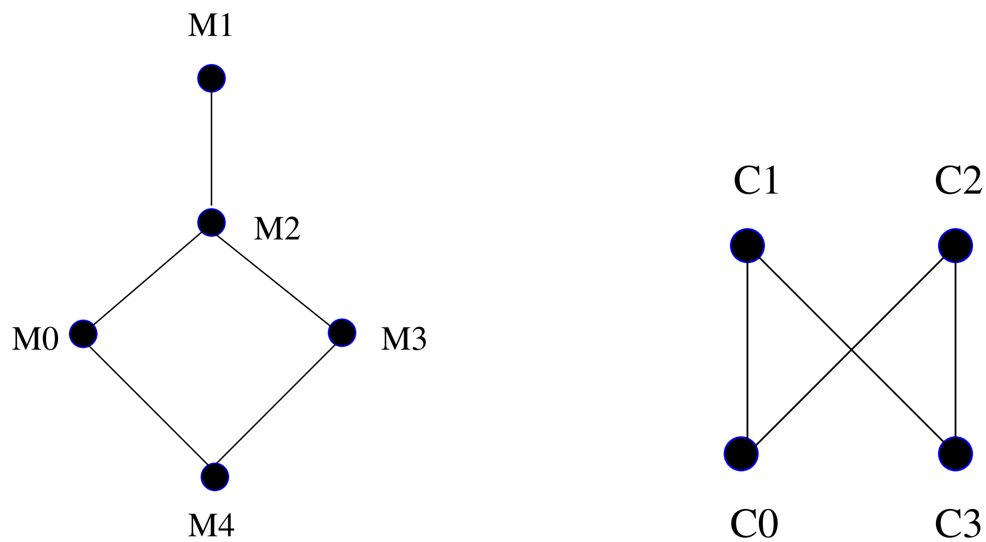
- M3. Generic planar groups. These include generic structures like benzene rings, methyl groups, *etc.*, and predicates to determine connectivity amongst such groups. There are 14 predicates in this group; and

- M4. Three-dimensional structure. These are the distances between centres of the generic planar groups in M3. They are calculated from the positions in space of the individual atoms constituting the groups. There are 2 predicates in this group.

The following statements can be made about M0–M4: M1 is specifically aimed at the compounds considered; M2 has been shown to be useful in constructing linear models for the data (Debnath et al., 1991); M0, M3 and M4 are generic to all chemical compounds. Chemists prefer the higher level concepts in M3 to the low-level atom-bond structure in M0. The extent to which M4 is relevant to mutagenic activity is unclear as mutagencity tends to be a toxic property affecting entire cells. This suggests the partial ordering in Figure 4(a).

The background knowledge for carcinogenesis is contained in the definition of 41 predicates, partitioned into the following equivalence classes:

- C0. Molecular description at the atomic level. This is similar to M0 above and is comprised of 5 predicates;



(a) Mutagenesis (b) Carcinogenesis

Figure 4: Partial ordering over equivalence classes of background predicates based on relevance information available.

C1. Toxicity properties identified by experts as being related to carcinogenic activity, and arithmetic constraints on these. These are an interpretation of the descriptions by Ashby and Tennant (1991), and are contained within the definitions of 5 predicates;

C2. Short-term assays for genetic risks. These include the *Ames* test, *Salmonella* assay, in-vivo tests for the induction of micro-nuclei in rat and mouse bone marrow *etc.* The test results are simply "positive" or "negative" depending on the response and are encoded by a single predicate definition; and

C3. Generic planar groups. These are similar to M3 above, extended to 30 predicate definitions.

The following statements can be made about C0–C3: C1 and C2 contain very good indicators of carcinogenecity (although our translation of indicators described by Ashby and Tennant, 1991, is not perfect, thus making C2 probably more reliable than C1); C0 and C3 are generic to all chemical compounds. Chemists prefer the higher level concepts in C3 to the low-level atom-bond structure in C0. This suggests the partial ordering in Figure 4(b).

4.2.3 ALGORITHMS AND MACHINES

All experiments use the ILP system P-Progol (Version 2.7.5). This is available at: `www.comlab.ox.ac.uk/oucl/research/areas/machlearn/PProgol/pprogol.pl`. The experiments were performed on machine equipped with a 266 Mhz Pentium II processor and 128 megabytes of random access memory.

4.3 Method

We adopt the following method:

For each problem (Mutagenesis and Carcinogenesis):

1. Obtain a model using all background information and the procedure described in Figure 1. Obtain estimates of the predictive accuracy of the model and time for model construction using a N-fold cross-validation design (Weiss and Kulikowski, 1991).
2. Obtain a model using the "informed" incremental procedure described in Figure 2. Obtain estimates of the predictive accuracy of the model and time for model construction using the same cross-validation design as in Step 1 above.
3. Repeat T times: obtain a model using the "uninformed" incremental procedure described in Figure 3; and obtain estimates of the predictive accuracy of the model and time for model construction using the same cross-validation design as in Step 1 above. Overall estimates of predictive accuracy and time are obtained by averaging the results from the T trials.
4. The results from Step 2 are compared against those from Step 1 and Step 3 (see details below).

The following details are relevant: (i) For the purposes of this study, N and T are assigned the value 10; (ii) All reports suggest that the data are "noisy" and clauses need not be consistent with all negative examples. As stated in Section 2, the presence of noise in the data requires the specification of the minimum accuracy required for any clause found. It is difficult to establish an appropriate value of the minimum accuracy of a clause. We have adopted 70%, which has been used with some empirical justification for at least one of the problems (Srinivasan and King, 1997). Other parameters for P-Progol remain at default settings; (iii) The procedure in Figure 2 employs a total ordering over equivalence classes that is consistent with the partial ordering provided. In general, several total orderings are possible. The procedure we have used results in the following: M1, M2, M3, M4, M0; and C2, C1, C3, C0. The choice of C2 ahead of C1 is based on the lower reliability of C1 (due to our approximate encoding). The choice of M4 ahead of M0 is less clear-cut, being based on the fact that chemists rarely call on the low-level atom-bond constructs of M0. (iv) The incremental procedures in Figures 2 and 3 judge a pair of models as equivalent if their predictive accuracies are within 1 standard error of each other. This is equivalent to the rule used by programs like CART (Breiman et al., 1984) to judge equivalence of models; and (v) We use the following working definition for comparing performances of a pair of procedures: the procedure yielding a model with with significantly higher predictive accuracy is better. If the predictive accuracies are statistically equivalent, then the procedure that constructs a model significantly faster is better. If times for model construction are also equivalent, then the two procedures are equivalent. We will use the same "1-SE" rule as in (iv) above to judge significant differences in model construction time.

4.4 Results

Figure 5 tabulates the performance of the three procedures being compared. The principal details in Figure 5 are these: (1) For both problems, the "informed" incremental procedure identifies, in significantly lesser time, a model equivalent in predictive accuracy to the procedure that uses all of the background information. This is not the case for the uninformed incremental procedure; (2) For both problems, it is evident that the performance of the informed incremental procedure is better than the uninformed procedure.

Procedure	Mutagenesis		Carcinogenesis	
	Acc	Time	Acc	Time
All	0.89 (0.02)	62.55 (3.01)	0.62 (0.03)	236.81 (26.01)
Informed	0.89 (0.02)	3.79 (0.36)	0.61 (0.03)	9.54 (1.00)
Uninformed	0.88 (0.02)	115.93 (30.29)	0.53 (0.05)	460.51 (197.25)

Figure 5: Comparative performance of procedures in Figure 1 ("All"), Figure 2 ("Informed") and Figure 3 ("Uninformed"). "Acc" refers to the estimated predictive accuracy of the model returned by the procedure, and "Time" to the time taken in seconds, to construct the model. The numbers in parentheses are standard-error estimates.

From Section 4.3, the reader would recall that for the "informed" procedure we selected particular total orderings that were consistent with the partial orderings shown in Figure 4 (M1, M2, M3, M4, M0 for mutagenesis; and C2, C1, C3, C0 for carcinogenesis). While we believe there are good reasons for selecting these orderings (see point (iii) in Section 4.3), it is nevertheless instructive to examine the extent to which the results in Figure 5 depend on these choices. The other total orderings consistent with Figure 4 are: M1, M2, M3, M0, M4 (mutagenesis); and C1, C2, C3, C0 (carcinogenesis). The only change that results from using these alternative orderings is in the time taken by the informed procedure for carcinogenesis. This changes from 9.54s to 36.53s (with standard error 5.67s), which is still significantly lower than the time recorded for either of the other two procedures in Figure 5.

These results suggest that for both problems, the performance of the ILP system is significantly better when using the relevance ordering to guide model construction. This provides evidence against the null hypothesis stated at the outcome of the experiments, namely:

H_0 : The expert-provided relevance ordering over background predicates has no significant effect on the performance of an ILP system.

and, in turn, some evidence for the conjecture that the performance of an ILP system can benefit significantly from incorporating expertise in the form of a relevance ordering.

It is instructive to compare some further properties of the models returned. In the light of the results in Figure 5, we will concentrate here only on "All" and "Informed". Comparative features of the models returned are in Figure 6. They indicate that the predominant gain in using procedure "All" appears to be a model that can explain a greater proportion of the instances. While this results in a better descriptive fit, the cross-validated accuracies in Figure 5 suggest that the extra effort does not appear to have resulted in a significant better predictive model.

For purely illustrative purposes, the rules obtained for carcinogenesis are shown in Figure 7. It is worth noting that rules using procedure "Informed" involve no relational aspects (that is, they are entirely propositional in nature).

For the practitioner concerned with ILP, the main implications of these tabulations are the following:

Procedure	Mutagenesis			Carcinogenesis		
	Rules	Unexp	Preds	Rules	Unexp	Preds
All	4	7	M0,M1,M3,M4	23	19	C0,C1,C2,C3
Informed	5	12	M1,M2	4	60	C2

Figure 6: Some further features of the models returned. "Rules" is the number of non-trivial rules (that is, they cover two or more positive examples); "Unexp" is the number of positive examples left unexplained by the rules; "Preds" is the equivalence classes of predicates used by the rules.

1. When all background information cannot be used at once (either due to limitations of the ILP system, or the nature of the domain) expert assessment of the relevance of background predicates can assist substantially in the construction of good models.

2. Good "first-cut" results can be obtained quickly by a simple exclusion of information known to be less relevant.

Both these points would come as no surprise to any experienced data analyst. Nevertheless, it is surprising that existing ILP systems appear to have made no provisions to take advantage of such expertise. One additional point, although not directly relevant to the research conjecture of this study, is nevertheless of interest. Much of the power of relational learning appears to be used in modelling "difficult" instances: those for which available expertise is not highly relevant. For these instances, the relatively low-level reasoning afforded by the use relational predicates appears to be particularly useful (in contrast to predicates provided by experts which appear to encode high-level propositions).

5. Conclusion

The ability to include and use logical specifications of prior knowledge is a characteristic feature of all ILP systems. This should allow a form of automated data analysis that naturally benefits from domain-specific expertise. To date, the primary use made of this expertise by ILP systems has been in the form of providing sytactic and semantic constraints on models; and in the form of identifying background knowledge that is irrelevant to the analysis task. It is quite possible that a sufficiently skilled expert may be capable of more than this, and an ILP system interacting with him or her should benefit from this. While expertise may be available on a number of aspects, we have concentrated here on the issue of ordering background predicates based on relevance. This is clearly an issue requiring substantial prior domain knowledge, and one that a general-purpose ILP system is unlikely to be able to resolve simply by examining the data provided. For tasks with substantial background information, there are good reasons to seek such additional expertise: complexity results (see for example Muggleton, 1995) suggest that as the number of predicates in the background increases, the size of the space searched by an ILP system can increase greatly. Our results here suggest that a simple incremental procedure that is guided by an expert-provided relevance ordering will allow accurate models to be constructed quickly. Of course, to be more

Procedure "All":

```
Rule 1: Pos cover = 51 Neg cover = 21
active(A) :- atm(A,B,c,10,C), has_property(A,cytogen_sce,p), has_property(A,cytogen_ca,p).

Rule 2: Pos cover = 22 Neg cover = 7
active(A) :- atm(A,B,h,1,C), gteq(C,0.329), has_property(A,ames,p).

Rule 3: Pos cover = 11 Neg cover = 2
active(A) :- atm(A,B,c,29,C).

Rule 4: Pos cover = 12 Neg cover = 3
active(A) :- atm(A,B,o,40,C), lteq(C,-0.532), has_property(A,salmonella,p).

Rule 5: Pos cover = 23 Neg cover = 1
active(A) :- has_property(A,drosophila_slrl,p).

Rule 6: Pos cover = 9 Neg cover = 3
active(A) :- atm(A,B,o,49,C), has_property(A,mouse_lymph,n).

Rule 7: Pos cover = 11 Neg cover = 4
active(A) :- has_property(A,chromaberr,n), has_property(A,cytogen_ca,n), has_property(A,salmonella,n).

Rule 8: Pos cover = 8 Neg cover = 1
active(A) :- atm(A,B,o,50,C), gteq(C,-0.203), atm(A,D,c,22,E).

Rule 9: Pos cover = 5 Neg cover = 0
active(A) :- atm(A,B,c,10,C), C= -0.003, has_property(A,ames,p).

Rule 10: Pos cover = 7 Neg cover = 1
active(A) :- alert(di23,A,B), has_property(A,ames,p).

Rule 11: Pos cover = 5 Neg cover = 0
active(A) :- alert(di10,A,B), has_property(A,cytogen_sce,n), has_property(A,salmonella,p).

Rule 12: Pos cover = 8 Neg cover = 3
active(A) :- atm(A,B,c,10,C), gteq(C,0.474), has_property(A,ames,n).

Rule 13: Pos cover = 9 Neg cover = 1
active(A) :- atm(A,B,br,94,C), alert(halide10,A,D), ar_halide(A,D).

Rule 14: Pos cover = 6 Neg cover = 0
active(A) :- has_property(A,micronuc_f,n), ether(A,B).

Rule 15: Pos cover = 3 Neg cover = 0
active(A) :- atm(A,B,c,10,C), C= -0.121.

Rule 16: Pos cover = 2 Neg cover = 0
active(A) :- atm(A,B,h,3,C), C=0.073.

Rule 17: Pos cover = 3 Neg cover = 0
active(A) :- atm(A,B,n,35,C), lteq(C,-0.764).

Rule 18: Pos cover = 2 Neg cover = 0
active(A) :- atm(A,B,cl,93,C), gteq(C,-0.077), has_property(A,cytogen_sce,p).

Rule 19: Pos cover = 2 Neg cover = 0
active(A) :- atm(A,B,c,22,C), C= -0.163, alert(di10,A,D), has_property(A,cytogen_ca,n).

Rule 20: Pos cover = 11 Neg cover = 2
active(A) :- atm(A,B,h,3,C), gteq(C,0.184), has_property(A,ames,n).

Rule 21: Pos cover = 10 Neg cover = 1
active(A) :- atm(A,B,o,50,C), atm(A,D,h,1,E), gteq(E,0.315).

Rule 22: Pos cover = 2 Neg cover = 0
active(A) :- atm(A,B,c,10,C), C= -0.085.

Rule 23: Pos cover = 2 Neg cover = 0
active(A) :- alert(di10,A,B), alert(di10,A,C), connected(C,B).
```

Procedure "Informed":

```
Rule 1: Pos cover = 99 Neg cover = 40
active(A) :- has_property(A,ames,p).

Rule 2: Pos cover = 3 Neg cover = 0
active(A) :- has_property(A,chromaberr,p), has_property(A,mouse_lymph,n).

Rule 3: Pos cover = 17 Neg cover = 5
active(A) :- has_property(A,ames,n), has_property(A,chromaberr,n).

Rule 4: Pos cover = 3 Neg cover = 0
active(A) :- has_property(A,drosophila_rt,n), has_property(A,salmonella,n).
```

Figure 7: Rules obtained (in Prolog) for the carcinogenesis problem.

convinced of the reality of the effect observed requires further experimentation (positive outcomes from two experiments clearly leaves considerable room for chance).

Even if the effects reported here are confirmed by further experimentation, simply accounting for relevance using a partial ordering is hardly likely to be sufficient. Experts' preferences may be considerably more refined. For example, in a recent study of predicting protein functional classes from sequence data (King et al., 2001), background knowledge consisted of sequence descriptions in three groups of predicates. It was found that experts preferred rules that called on predicates from any one group (to rules that "mixed up" predicates from several groups). The incremental procedure described here will not directly address such issues. However, when used with other mechanisms for encoding domain expertise—like the specification of constraints on hypotheses—it could help an ILP system identify quickly models that are both accurate and understandable.

Acknowledgments

A. S. acknowledges the support of the Nuffield Trust, Green College Oxford, and the European Union project IST-1999-11495:SolEuNet. Special thanks are due to Donald Michie and the Dept. of CSE, University of NSW, especially Claude Sammut for advice and guidance on issues raised in this paper. Thanks are also due to Steve Moyle for the generous use of his computer.

References

J. Ashby and R.W. Tennant. Definitive relationships among chemical structure, carcinogenicity and mutagenicity for 301 chemicals tested by the U.S. NTP. *Mutation Research*, 257:229–306, 1991.

R. Benigni. (Q)SAR prediction of chemical carcinogenicity and the biological side of the structure activity relationship. In *Proceedings of The Eighth International Workshop on QSARs in the Environmental Sciences*, 1998. Held in Baltimore, May 16–20, 1998.

I. Bratko and M. Grobelnik. Inductive learning applied to program construction and verification. In *Third International Workshop on Inductive Logic Programming*, pages 279–292, 1993. Available as Technical Report IJS-DP-6707, J. Stefan Inst., Ljubljana, Slovenia.

L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, 1984.

J. Cussens. Stochastic logic programs: Sampling, inference and applications. In *Proceedings of the Sixteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-2000)*, pages 115–122, San Francisco, CA, 2000. Morgan Kaufmann.

A.K. Debnath, R.L Lopez de Compadre, G. Debnath, A.J. Schusterman, and C. Hansch. Structure-Activity Relationship of Mutagenic Aromatic and Heteroaromatic Nitro compounds. Correlation with molecular orbital energies and hydrophobicity. *Journal of Medicinal Chemistry*, 34(2):786 – 797, 1991.

B. Dolsak and S. Muggleton. The application of Inductive Logic Programming to finite element mesh design. In S. Muggleton, editor, *Inductive Logic Programming*, pages 453–472. Academic Press, London, 1992.

S. Dzeroski, L. Dehaspe, B. Ruck, and W. Walley. Classification of river water quality data using machine learning. In *Proceedings of the Fifth International Conference on the Development and Application of Computer Techniques Environmental Studies*, 1994.

C. Feng. Inducing temporal fault dignostic rules from a qualitative model. In S. Muggleton, editor, *Inductive Logic Programming*, pages 473–486. Academic Press, London, 1992.

D. Gabbay. Labelled deductive systems. Technical report, Centrum fur Informations und Spachverarbeitung, Universitat Munchen, 1991.

R.D. King, A. Karwath, A. Clare, and L. Dehaspe. The Utility of Different Representations of Protein Sequence for Predicting Functional Class. *Bioinformatics*, 17:445–454, 2001.

R.D. King, S.H. Muggleton, A. Srinivasan, and M.J.E. Sternberg. Structure-activity relationships derived by machine learning: The use of atoms and their bond connectivities to predict mutagenicity by inductive logic programming. *Proc. of the National Academy of Sciences*, 93: 438–442, 1996.

R.D. King, S.H. Muggleton, and M.J.E. Sternberg. Drug design by machine learning: The use of inductive logic programming to model the structure-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proc. of the National Academy of Sciences*, 89(23): 11322–11326, 1992.

S. Muggleton. Inductive Logic Programming: derivations, successes and shortcomings. *SIGART Bulletin*, 5(1):5–11, 1994.

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

S. Muggleton. Stochastic logic programs. In L. DeRaedt, editor, *Advances in Inductive Logic Programming*, pages 254–264. IOS Press, 1996.

S. Muggleton. Semantics and derivation for stochastic logic programs. In *UAI-2000 Workshop on Fusion of Domain Knowledge with Data for Decision Support*, 2000.

S. Muggleton, R. King, and M. Sternberg. Predicting protein secondary structure using inductive logic programming. *Protein Engineering*, 5:647–657, 1992.

S.H. Muggleton, A. Srinivasan, R.D. King, and M.J.E. Sternberg. Biochemical knowledge discovery using Inductive Logic Programming. In H. Motoda, editor, *First Conference on Discovery Science*. Springer-Verlag, 1998.
URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/discovery98.ps.gz`.

S. Nienhuys-Cheng and R. de Wolf. *Foundations of Inductive Logic Programming*. Springer, Berlin, 1997.

J. R. Quinlan. FOIL: a midterm report. In *European Conference on Machine Learning*. Springer-Verlag, Berlin, 1993.

A. Srinivasan. A study of two sampling methods for analysing large datasets with ILP. *Data Mining and Knowledge Discovery*, 3(1):95–123, 1999.
URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/dami98b.ps.gz`.

A. Srinivasan. Extracting context-sensitive models in Inductive Logic Programming. *Machine Learning*, 44:301–324, 2001.
URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/relev.ps.gz`.

A. Srinivasan and R.D. King. Carcinogenesis predictions using ILP. In N. Lavrac and S. Dzeroski, editors, *Proceedings of the Seventh International Workshop on Inductive Logic Programming (ILP97)*, volume 1297 of *LNAI*, pages 273–287, Berlin, 1997. Springer. URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/ilp97.ps.gz`. A version also in Intelligent Data Analysis in Medicine, Kluwer.

A. Srinivasan, R.D. King, and D.W. Bristol. An assessment of submissions made to the Predictive Toxicology Evaluation Challenge. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence (IJCAI-99)*. Morgan Kaufmann, Los Angeles, CA, 1999. URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/ijcai99.ps.gz`.

A. Srinivasan, R.D. King, S.H. Muggleton, and M.J.E. Sternberg. The Predictive Toxicology Evaluation Challenge. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence (IJCAI-97)*. Morgan Kaufmann, Los Angeles, CA, 1997. URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/ijcai97.ps.gz`.

A. Srinivasan, S.H. Muggleton, R.D. King, and M.J.E. Sternberg. Theories for mutagenicity: a study of first-order and feature based induction. *Artificial Intelligence*, 85:277–299, 1996. URL `ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/Papers/AS/aij96.ps.gz`.

S.M. Weiss and C.A. Kulikowski. *Computer systems that learn*. Morgan Kaufmann, San Mateo, CA, 1991.

J. Zelle and R. Mooney. Learning semantic grammars with constructive inductive logic programming. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 817–822. Morgan Kaufmann, 1993.