1

Biochemical knowledge discovery using Inductive Logic Programming

Stephen Muggleton, Ashwin Srinivasan, R.D. King and M.J.E. Sternberg

ABSTRACT

Machine Learning algorithms are being increasingly used for knowledge discovery tasks. Approaches can be broadly divided by distinguishing discovery of procedural from that of declarative knowledge. Client requirements determine which of these is appropriate. This paper discusses an experimental application of machine learning in an area related to drug design. The bottleneck here is in finding appropriate constraints to reduce the large number of candidate molecules to be synthesised and tested. Such constraints can be viewed as declarative specifications of the structural elements necessary for high medicinal activity and low toxicity. The first-order representation used within Inductive Logic Programming (ILP) provides an appropriate description language for such constraints. Within this application area knowledge accreditation requires not only a demonstration of predictive accuracy but also, and crucially, a certification of novel insight into the structural chemistry. This paper describes an experiment in which the ILP system Progol was used to obtain structural constraints associated with mutagenicity of molecules[1]. In doing so Progol found a new indicator of mutagenicity within a subset of previously published data. This subset was already known not to be amenable to statistical regression, though its complement was adequately explained by a linear model. According to the combined accuracy/explanation criterion provided in this paper, on both subsets comparative trials show that Progol's structurally-oriented hypotheses are preferable to those of other machine learning algorithms.

[1] The results in this paper are published separately in [KMSS96, SMKS96]

Book Title, Edited by Editors' Names

1.1 INTRODUCTION

Within the AI literature, the distinction between procedural and declarative knowledge was first introduced by McCarthy [McC59], though it is strongly related to Ryle's difference between "knowing how" and "knowing that" [Ryl49]. While procedural knowledge can often be conveniently described in algorithmic form, logical sentences are usually used to capture declarative knowledge.

Most of Machine Learning has been concerned with the acquisition of procedural knowledge. For instance, the nested if-then-else rules used in decision tree technology largely describe the flow of procedural control. As a consequence of this procedural bias, the emphasis in the testing methodology has been on predictive accuracy, while communicability, the primary hallmark of declarative knowledge, has largely been disregarded in practice. This is despite early and repeated recognition of the importance of comprehensibility in machine learning [Mic83, Mic86, MST94]. In certain domains client requirements dictate the need for inductive discovery of declarative knowledge. This is the case in the area of rational drug design (see James Black's Nobel Lecture reprinted in [Bla89]).

A range of specialists are involved within the the pharmaceutical industry. These include computational chemists, molecular biologists, pharamcologists, synthetic and analytical chemists. The bottleneck in the process of drug design is the discovery of appropriate constraints to reduce the large number of candidate molecules for synthesis and testing. Since such constraints need to be used by synthetic chemists in the molecular design process, they must be stated in appropriately structural, and ideally 3-D terms. The constraints will describe both structural attributes which enhance medicinal activity as well as those which should be absent, owing to toxic side-effects. Such design-oriented constraints are declarative in nature.

For the development of such constraints, Inductive Logic Programming (ILP) [MR94], with its emphasis on the declarative representation of logic programs, is the obvious Machine Learning technique. This is not to say that other Machine Learning approaches cannot be used when, for instance, appropriate known structural properties have been encoded as molecular attributes. However, in industrial drug discovery tasks such 'indicator' attributes are typically either unknown, or only understood to a limited degree. Without such attributes, computational chemists usually apply linear regression to pre-computed bulk properties of the molecules involved. Common bulk properties include the octanol/water partition coefficient $log(P)$, and molecular reactivity as measured by the electronic property ϵ_{LUMO}. Although the derived regression equations can be used for testing candidate molecules within a molecular modelling package, they provide synthetic chemists with no structural clues as to how candidate molecules should be designed.

In this paper we use the problem of mutagenicity prediction to compare the ILP algorithm Progol [Mug95] with various other state-of-the-art machine learning, neural and statistical algorithms. Mutagenicity is a toxic property of molecules related to carcinogenicity (see 1.2). The algorithms are compared both with and without provision of specially defined structural attributes. The derived hypotheses are compared both in terms of predictive accuracy, and in terms of declarative, structural description provided. Progol is capable of using low-level atom and bond relations to define structural properties not found in the original attribute language.

For the purposes of such a comparison it is not immediately obvious how one should define a performance criterion which combines both predictive accuracy and declarative explanation.

Below we provide a working definition of such a combining function, appropriate for this problem domain.

Definition 1 Combination of Accuracy and Explanation. *If the predictive accuracies of two hypotheses are statistically equivalent then the hypothesis with better explanatory power will be preferred. Otherwise the one with higher accuracy will be preferred.*

Clearly no preference is possible in the case in which the hypotheses are equivalent in terms of both accuracy and explanation. In the experiments reported, hypotheses are judged to have "explanatory power" (a boolean property) if they provide insight for molecular design based on their use of structural descriptors, and not otherwise.

1.2 MUTAGENESIS

Mutagenic compounds cause mutations in the DNA sequence. These compounds are often, though not always, carcinogenic (cancer-producing). It is of considerable interest to the pharmaceutical industry to determine molecular indicators of mutagenicity, since this would allow the development of less hazardous new compounds. This paper explores the power of Progol to derive Structure-Activity Relations (SARs). A data set in the public domain is used, which was originally investigated using linear regression techniques in [DdCD+91]. This set contains 230 aromatic and hetero-aromatic nitro compounds that were screened for mutagenecity by the widely used Ames test (with *Salmonella typhimurium* TA 98). The data present a significant challenge to SAR studies as they are chemically diverse and without a single common structural template.

In the course of the original regression analysis, the data were found to have two sub-groups, with 188 compounds being amenable to linear modelling by regression and the remainder being regression "unfriendly". The 188 compounds have also been examined with good results by neural-network techniques [VCC93]. In keeping with attribute-based approaches to SAR, these studies rely on representing compounds in terms of global chemical properties, notably $log(P)$ and ϵ_{LUMO} (see previous section). Usually, some degree of structural detail is introduced manually in the form of logical indicator attributes to be used in the regression equation. For example, in [DdCD+91] the authors introduce the variable I_1 to denote the presence of 3 or more fused rings, and I_a to denote the class "acenthrylenes". Regression and neural-network models have then been constructed using these 4 basic attributes, namely $log(P)$, ϵ_{LUMO}, I_1, and I_a.

1.3 EXPERIMENT

1.3.1 Hypothesis

The experimental hypothesis is as follows.

Hypothesis. The ILP program Progol has better performance – using the criterion described in Definition 1 – than the attribute-based algorithms embodied in linear regression, decision-trees and neural networks.

1.3.2 Materials

We first describe the algorithms used in the comparative trial. The description of Progol is somewhat more detailed, given that it is less likely to be as well known to the reader as the attribute-based learners.

Progol

A formal specification for Progol is as follows.

- B is background knowledge consisting of a set of definite clauses $= C_1 \wedge C_2 \wedge \ldots$
- E is a set of examples $= E^+ \wedge E^-$ where
  - E^+ or 'positive examples' $= e_1 \wedge e_2 \wedge \ldots$ are definite clause s and
  - E^- or 'negative examples' $= f_1 \wedge f_2 \wedge \ldots$ are non-definite Horn clauses.
- $H = D_1 \wedge D_2 \wedge \ldots$ is an hypothesised explanation of the examples in terms of the background knowledge.

Progol can be treated as an algorithm A such that $H = A(B, E)$ is an hypothesis from predefined language $\mathcal{L}$. The language $\mathcal{L}$ consists of legal forms of the predicates in B that can appear in H. Each D_i in H has the property that it can explain at least one positive example and the set H is consistent with the negative examples. That is, $B \wedge D_i \models e_1 \vee e_2 \vee \ldots$, $\{e_1, e_2, \ldots\} \subseteq E^+$ and $B \wedge H \wedge E^- \not\models \square$. If more than one such H exists then Progol returns the first one in an arbitrary lexicographic ordering $\mathcal{O}(\mathcal{L})$. Figure 1.1 is a diagrammatic view of the space searched by a program like Progol when constructing the individual clauses D_i.

Progol is thus provided with a set of positive and negative examples together with problem-specific background knowledge B. The aim is to generate an hypothesis, expressed as a set of rules, which explains all the positive examples in terms of the background knowledge whilst remaining consistent with the negative examples. To achieve this, *Progol* 1) randomly selects a positive example e_i; 2) uses inverse entailment ([Mug95]) to construct the most specific hypothesis $\bot(B, e_i)$ which explains e_i in terms of B; 3) finds a rule D_i which generalises $\bot(B, e_i)$ and which maximally compresses a set of entailed examples E_i; and 4) adds D_i to the hypothesis H and repeats from 1) with examples not covered so far until no more compression is possible. Compression is here defined as the difference, in total size of formula, between E_i and D_i. Compression formalises the reduction of information provided by the rule in describing the data. Figure 1.1 illustrates the reduced, finite, search space of *Progol*. The reduction in the search space is such that, unlike most machine learning algorithms, it is generally feasible to find the optimal rule satisfying conditions (3) and (4) above in reasonable time (more detailed complexity arguments can be found in [Mug95]).

The results in this paper were obtained using a Prolog implementation of Progol, called *P-Progol*. Details of obtaining and using a C implementation of Progol can be found in [Mug95]. *P-Progol* is available by ftp access to *ftp.comlab.ox.ac.uk*. The relevant directory is *pub/Packages/ILP*. The implementation includes on-line documentation that clarifies the points of difference with the C version. The theory underlying both versions is the same and is described fully in [Mug95]. However, differences in search strategies and pruning facilities imply that given the same language and resource restrictions, the two versions can compute

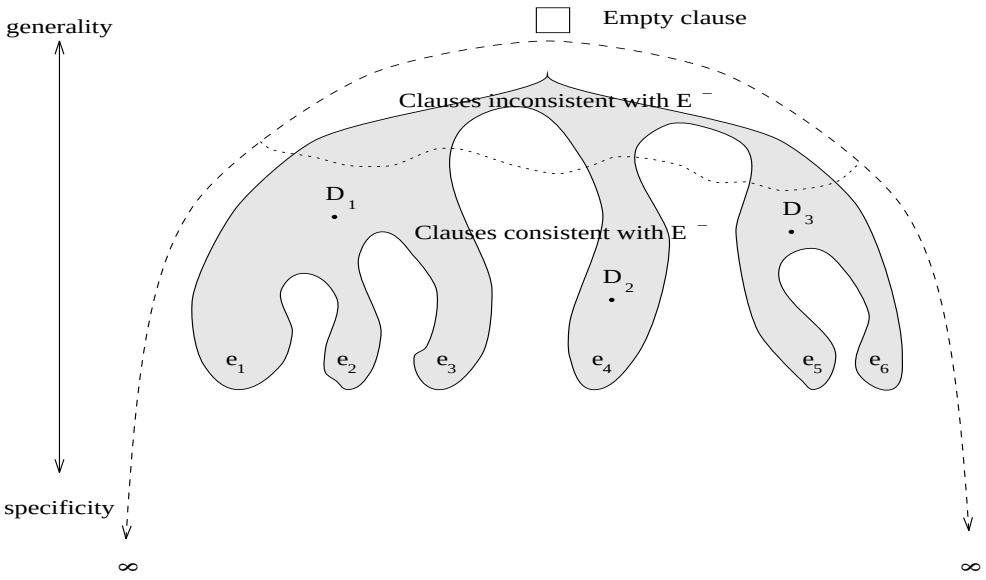

Figure 1.1 The space of clauses searched by logic-based learning algorithms. The space enclosed by the parabolic broken line contains all clauses that can potentially be included in a theory. Clauses near the top are usually too general, that is, are inconsistent with the negative examples. Clauses become progressively more specific as one proceeds downwards from the empty clause. For example, given a set of positive examples, $E^+ = e_1 \wedge e_2 \wedge \ldots e_6$ and negative examples E^-, the shaded area shows the part of the space searched by Progol. In this space, the clauses D_i represent clauses that maximally compress the examples 'under' them in the search space. Progol ensures that it only searches those parts in the grey space.

different answers. For convenience, in the rest of this paper, we shall refer to *P-Progol* as Progol.

Attribute-based algorithms

The regression was achieved using the Minitab package (Minitab Inc, Pennsylvania State University, Pa). The learning rule used by the neural technique relies on the back-propagation of errors. Changes in weight are calculated by solving a set of differential equations as described in [Gea71]. This technique removes the need for learning-rate or momentum parameters [OF89]. The actual implementation of the back-propagation algorithm was supplied by J Hirst of the Imperial Cancer Research Fund. Finally, the procedure embodied by the CART algorithm [BFOS84] as implemented by the Ind package [Bun92] is used to construct classification trees.

Data

For this study, we consider the following characterisation for mutagenic molecules. All molecules whose mutagenic activity, as measured by the Ames test, is greater than 0 are termed "active". All others are taken to be "inactive". Figure 1.2 shows the distribution of molecules within these classes for the two different subsets identified in [DdCD+91].

Subset	"Active"	"Inactive"	Total
"Regression friendly"	125	63	188
"Regression unfriendly"	13	29	42
All	138	92	230

Figure 1.2 Class distribution of molecules

For each of the two subsets, the task is to obtain a characterisation of the "active" molecules. Unlike the attribute-based algorithms, Progol can employ an explicit 2-dimensional atom and bond representation of the molecules. This representation supports the encoding of generic chemical background knowledge. This is described in the next section.

ILP representation

The obvious generic description of molecules is to use atoms and their bond connectivities. For the chemicals described here, these were obtained automatically using the molecular modelling program QUANTA[2]. For the experiments here, each chemical was entered manually via a molecular sketchpad. All atoms in the chemical are then automatically typed by QUANTA to consider their local chemical environment, an estimate of the electronic charge on each atom estimated, and connecting bonds are classified (as single, double, aromatic, etc.). It is worth noting here that the manual use of a sketchpad is clearly not mandatory – with appropriate software, the chemicals could have been extracted directly from a database and their structure computed. Further, the choice of QUANTA was arbitrary, any similar molecular modelling package would have been suitable.

There is a straightforward translation of the QUANTA facts into Prolog facts suitable for Progol. The result is that each compound is represented by a set of facts such as the following.

```
atm(127, 127_1, c, 22, 0.191)
bond(127, 127_1, 127_6, 7)
```

This states that in compound 127, atom number 1 is a carbon atom of QUANTA type 22 with a partial charge of 0.191, and atoms 1 and 6 are connected by a bond of type 7 (aromatic). Again, this representation is not peculiar to the compounds in this study, and can be used to encode arbitrary chemical structures. With the *atm* and *bond* predicates as basis, it is possible to use background knowledge definitions of higher-level 2-dimensional chemical substructures that formalise concepts like methyl groups, nitro groups, 5- and 6- membered rings, aromatic

[2] Distributed by Molecular Simulations Inc, USA

rings, heteroaromatic rings, connected rings, etc. For example, the following Prolog program fragment can be used to detect benzene rings.

```
% benzene - 6 membered carbon aromatic ring
% recall that QUANTA type 7 indicates an aromatic bond
benzene(Drug,Ring_list) :-
        atoms(Drug,6,Atom_list,[c,c,c,c,c,c]),
        ring6(Drug,Atom_list,Ring_list,[7,7,7,7,7,7]).

% get N lexico-graphically ordered atoms and their elements
atoms(Drug,N,[Atom|Atoms],[Elem|Elems]) :-
...

% find 6 connected atoms and their bond types
ring6(Drug,[Atom1|List],[Atom1,Atom2,Atom4,Atom6,Atom5,Atom3],
        [Type1,Type2,Type3,Type4,Type5,Type6]) :-
        ...
```

A list of definitions used for the experiments is given in Appendix A.

1.3.3 Method

The comparative trials reported here were conducted under the following conditions.

1. Analysis of the two subsets of compounds in Figure 1.2 (188 and 42) were conducted separately.
2. For each subset, Progol and attribute-based algorithms were compared a) using the molecular properties $log(P)$ and ϵ_{LUMO} and b) using additional expert-derived structural indicator attributes I_1 and I_a. In both cases, Progol also had access to the definitions of the generic chemical properties described earlier. Owing to the lack of a relational description language, these properties could not be provided effectively to the attribute-valued algorithms, though they would be readily available in a real-world situation.
3. Predictive accuracy estimates were obtained using cross-validation. For the subset of 188 compounds, 10-fold cross-validation was employed, and for the 42 compound a leave-one-out procedure was used. Comparison of estimates use McNemar's test. Details are available in [SMKS96].
4. Explanatory power of Progol's hypotheses was compared to those of the attribute-based algorithm with highest predictive accuracy. Hypotheses were judged to have explanatory power if and only if they made use of structural attributes of molecules.
5. Relative performance was assessed in each case by combining accuracy and explanation using Definition 1.

1.3.4 Results

There are two cases to consider. These are analysis without (case $\overline{I}$) and with (case I) the indicator attributes $I_{1,a}$.

Accuracy

Figure 1.3 tabulates the predictive accuracies of theories obtained on the subsets of 188 and 42 compounds for cases $\overline{I}$ and I. In the tabulation REG denotes linear regression, DT denotes decision-tree and NN a neural-network with 3 hidden units.

Algorithm	188		42	
	$\overline{I}$	I	$\overline{I}$	I
REG	0.85 (0.03)	0.89 (0.02)	0.67 (0.07)	0.67 (0.07)
DT	0.82 (0.02)	0.88 (0.02)	0.83 (0.06)	0.83 (0.06)
NN	0.86 (0.02)	0.89 (0.02)	0.64 (0.07)	0.69 (0.07)
Progol	0.88 (0.02)[1]	0.88 (0.02)[1]	0.83 (0.06)[2]	0.83 (0.06)[2]
Default class	0.66 (0.03)	0.66 (0.03)	0.69 (0.07)	0.69 (0.07)

Figure 1.3 Predictive accuracy estimates of theories of mutagenicity. Accuracy is defined as the proportion of correct predictions obtained from cross-validation trials. Estimates for the 188 compounds are from a 10-fold cross-validation, and those for the 42 are from a leave-one-out procedure. The "Default class" algorithm is one that simply guesses majority class. Estimated standard error is shown in parentheses after each accuracy value. Superscripts on Progol results refer to the pairwise comparison in turn of Progol against the attribute-based programs. With the null hypothesis that Progol and its adversary classify the same proportions of examples correctly, superscript 1 indicates that the probability of observing the classification obtained is ≤ 0.05 for DT, and 2 that this probability is ≤ 0.05 for REG and NN only (but not DT).

It is evident that on the "regression-friendly" subset (188 compounds), Progol yields state-of-the-art results (at $P < 10\%$). It is also unsurprising that regression and neural-network perform well on this subset which had been identified as being amenable to a regression-like analysis. It is apparent that the predictive accuracy of Progol on the "regression-unfriendly" data is the same as that obtained using a decision-tree. Figure 1.4 summarises the comparison of Progol's predictive accuracy against the best attribute-based algorithm for the two subsets. For the 188 compounds, the best attribute-based algorithm is in turn NN (case $\overline{I}$) and REG (case I). For the 42 compounds, the best attribute-based algorithm is DT in both cases.

Explanation

Appendix B lists the theories obtained by Progol and the attribute-based algorithm with the highest predictive accuracy – with the exception of case $\overline{I}$ on the 188 compounds. Here the best attribute-based algorithm is the neural-network and we do not list the weights assigned to each node. Figure 1.5 summarises the comparative assessment of the explanatory power of these theories, using the criterion in Step 4, Section 1.3.3.

Performance

The relative overall performance of Progol against the best attribute-based algorithm is obtained by combining the results in Figures 1.4, 1.5 using Definition 1. This is shown in Figure 1.6.

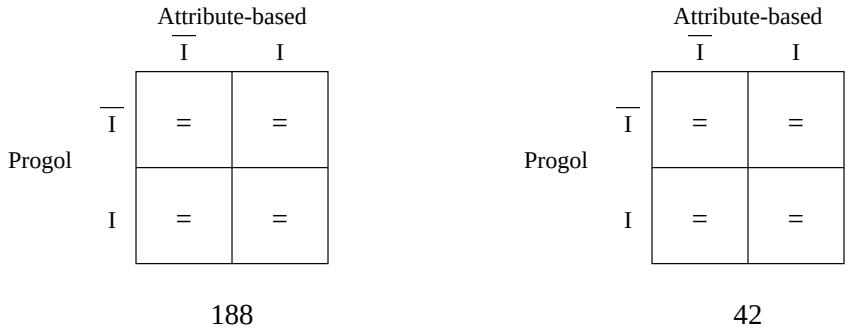

Figure 1.4 Comparing the predictive accuracies of Progol and the best attribute-based algorithm. '=' means statistically equivalent at $P = 0.05$.

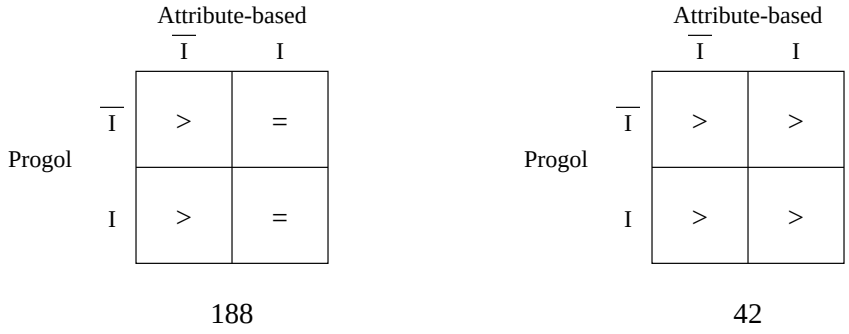

Figure 1.5 Comparing the explanatory power of Progol and the best attribute-based algorithm. '=' means the hypotheses of both algorithms use structural attributes. '>' means Progol's hypothesis uses structural attributes and the other algorithm's hypothesis does not.

The experimental hypothesis holds in all but two of the eight cases. The expected case in real life drug design is the $\overline{I}, \overline{I}$ one, in which Progol outperforms all other algorithms tested.

1.4 FUTURE WORK

A next step in representing chemical structure is to incorporate three-dimensional information for the compounds studied. We are currently considering a straightforward extension to the representation described in Section 1.3.2 to include the 3-dimensional co-ordinates for every atom in the molecule. These co-ordinates are then used to obtain a 3-dimensional co-ordinate specification for each structural group in Appendix A. This specifies the X,Y,Z co-ordinates of the nominal "centre" of the group. For example, the earlier Prolog program fragment to detect benzene rings is now modified as follows:

```
% benzene - 6 membered carbon aromatic ring centred at X,Y,Z
benzene(Drug,Atom_list,X,Y,Z) :-
   atoms(Drug,6,Atom_list,[c,c,c,c,c,c],Coordinates),
   ring6(Drug,Atom_list,Coordinates,Ring_list,[7,7,7,7,7,7],X,Y,Z).
```

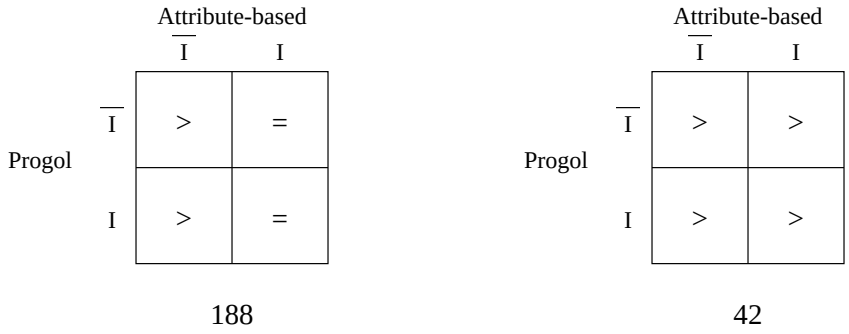

Figure 1.6 Comparing the performance of Progol and the best attribute-based algorithm. The combination of accuracy and explanation from Figures 1.4 and 1.5 is according to Definition 1.

}}

Euclidean distances between such group-centres within a molecule are easily calculable, and form the basis of a rudimentary reasoning about the 3-dimensional structure of the molecule. This approach is similar to recent work reported in [MPS96], where Progol is used to identify a pharmacophore common to a series of active molecules. A pharmacophore is a 3-dimensional description of a molecule that consists of 3–5 functional groups and the distances between them (angles and other geometric properties may also be used). In [MPS96] it is shown that in a blind-test Progol was able to identify the pharmacophore generally thought responsible for ACE inhibition.

1.5 CONCLUSION

This paper investigates Machine Learning algorithms' ability to discover accurate declarative knowledge in a drug design domain. The primary conclusion from the experiments described was that the ILP program Progol achieved this end to a greater degree than the other algorithms tested. The edge that an ILP program maintains over the other methods in this paper is that the rules are structurally oriented, and are thus in a position to make direct contributions to the synthesis of new compounds. The usefulness of the rules was taken into account in the comparative performance measure introduced in Definition 1. To our knowledge this is the first attempt to introduce a combined measure of this kind. It was only possible to define such a measure since the application domain dictates the way in which the rules would be used. One avenue of future research for knowledge discovery systems would be to use such a measure directly as a utility function in the hypotheses space search, thus maximising the chances of constructing accurate and comprehensible hypotheses. Such a function would be quite different in nature to the simple differential cost functions used in algorithms such as CART [MST94].

The ILP-constructed rules can be viewed as "knowledge-compilations" in the Michie sense – namely not just expressing the structural relationships that hold between response and

measured attributes, but doing so in a form " . . . that is meaningful to humans and evaluable in the head" (page 51, [MST94]).

Acknowledgements

This research was supported partly by the Esprit Basic Research Action ILP (project 6020), the SERC project 'Experimental Application and Development of ILP' and an SERC Advanced Research Fellowship held by Stephen Muggleton. Stephen Muggleton is also supported by a Research Fellowship of Wolfson College Oxford. Ross King and Michael Sternberg are supported by the Imperial Cancer Research Fund. The authors are indebted to to Donald Michie for his advice and interest in this work.

REFERENCES

[BFOS84] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, 1984.

[Bla89] J. Black. Drugs from emasculated hormones: the principle of syntopic antagonism. *Bioscience reports*, 9(3), 1989. Published in *Les Prix Nobel* 1988, printed in Sweden by Norstedts Tryckeri, Stockholm, Sweden.

[Bun92] W. Buntine. Ind package of machine learning algorithms. Technical Report 244-17, Research Institute for Advanced Computer Science, NASA Ames Research Center, Moffett Field, CA 94035, 1992.

[DdCD+91] 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.

[Gea71] C.W. Gear. *Numerical Initial Value Problems is Ordinary Differential Equations*. Prentice-Hall, Edgewood Cliffs, NJ, 1971.

[KMSS96] R. King, S. Muggleton, A. Srinivasan, and M. Sternberg. Structure-activity relationships derived by machine learning: the use of atoms and their bond connectives to predict mutagenicity by inductive logic programming. *Proceedings of the National Academy of Sciences*, 93:438–442, 1996.

[McC59] J. McCarthy. Programs with commonsense. In *Mechanisation of thought processes*, volume 1. Her Majesty's Stationery Office, London, 1959. Reprinted (with an added section on 'Situations, Actions and Causal Laws') in *Semantic Information Processing*, ed. M. Minsky (Cambridge, MA: MIT Press (1963)).

[Mic83] R.S. Michalski. A theory and methodology of inductive learning. In R. Michalski, J. Carbonnel, and T. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, pages 83–134. Tioga, Palo Alto, CA, 1983.

[Mic86] D. Michie. The superarticulacy phenomenon in the context of software manufacture. *Proceedings of the Royal Society of London*, A 405:185–212, 1986.

[MPS96] S. Muggleton, C.D. Page, and A. Srinivasan. An initial experiment into stereochemistry-based drug design using ilp. In *Proceedings of the Sixth Inductive Logic Programming Workshop*, LNAI, Berlin, 1996. Springer-Verlag.

[MR94] S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19,20:629–679, 1994.

[MST94] D. Michie, D.J. Spiegelhalter, and C.C. Taylor. *Machine learning, neural and statistical classification*. Ellis Horwood, London, 1994.

[Mug95] S. Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.

[OF89] A. J. Owens and D. L. Filkin. Efficient training of the back-propagation network by solving a system ofstiff ordinary differential equations. In *Proceedings IEEE/INNS International Joint Conference of Neural Networks*, pages 381–386, Washington DC, 1989.

[Ryl49] G. Ryle. *The Concept of Mind*. Hutchinson, 1949.

[SMKS96] 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.

[VCC93] D. Villemin, D. Cherqaoui, and J.M. Cense. Neural network studies: quantitative structure-activity relationship of mutagenic aromatic nitro compounds. *J. Chim. Phys*, 90:1505–1519, 1993.

A. Some elementary chemical concepts defined in terms of the atom and bond structure of molecules

The following are Prolog definitions for some simple chemical concepts that can be defined directly using the atomic and bond structure of a molecule.

```
% In the following QUANTA bond type 7 is aromatic.

% Three benzene rings connected linearly
anthracene(Drug,[Ring1,Ring2,Ring3]) :-
        benzene(Drug,Ring1),
        benzene(Drug,Ring2),
        Ring1 @> Ring2,
        interjoin(Ring1,Ring2,Join1),
        benzene(Drug,Ring3),
        Ring1 @> Ring3,
        Ring2 @> Ring3,
        interjoin(Ring2,Ring3,Join2),
        \+ interjoin(Join1,Join2,_),
        \+ members_bonded(Drug,Join1,Join2).

% Three benzene rings connected in a curve
phenanthrene(Drug,[Ring1,Ring2,Ring3]) :-
        benzene(Drug,Ring1),
        benzene(Drug,Ring2),
        Ring1 @> Ring2,
        interjoin(Ring1,Ring2,Join1),
        benzene(Drug,Ring3),
        Ring1 @> Ring3,
        Ring2 @> Ring3,
        interjoin(Ring2,Ring3,Join2),
        \+ interjoin(Join1,Join2,_),
        members_bonded(Drug,Join1,Join2).


% Three benzene rings connected in a ball
ball3(Drug,[Ring1,Ring2,Ring3]) :-
        benzene(Drug,Ring1),
        benzene(Drug,Ring2),
        Ring1 @> Ring2,
        interjoin(Ring1,Ring2,Join1),
        benzene(Drug,Ring3),
```

```
        Ring1 @> Ring3,
        Ring2 @> Ring3,
        interjoin(Ring2,Ring3,Join2),
        interjoin(Join1,Join2,_).

members_bonded(Drug,Join1,Join2) :-
        member(J1,Join1),
        member(J2,Join2),
        bondd(Drug,J1,J2,7).


ring_size_6(Drug,Ring_list) :-
        atoms(Drug,6,Atom_list,_),
        ring6(Drug,Atom_list,Ring_list,_).

ring_size_5(Drug,Ring_list) :-
        atoms(Drug,5,Atom_list,_),
        ring5(Drug,Atom_list,Ring_list,_).

% benzene - 6 membered carbon aromatic ring
benzene(Drug,Ring_list) :-
        atoms(Drug,6,Atom_list,[c,c,c,c,c,c]),
        ring6(Drug,Atom_list,Ring_list,[7,7,7,7,7,7]).

carbon_5_aromatic_ring(Drug,Ring_list) :-
        atoms(Drug,5,Atom_list,[c,c,c,c,c]),
        ring5(Drug,Atom_list,Ring_list,[7,7,7,7,7]).

carbon_6_ring(Drug,Ring_list) :-
        atoms(Drug,6,Atom_list,[c,c,c,c,c,c]),
        ring6(Drug,Atom_list,Ring_list,Bond_list),
        Bond_list \== [7,7,7,7,7,7].

carbon_5_ring(Drug,Ring_list) :-
        atoms(Drug,5,Atom_list,[c,c,c,c,c]),
        ring5(Drug,Atom_list,Ring_list,Bond_list),
        Bond_list \== [7,7,7,7,7].

hetero_aromatic_6_ring(Drug,Ring_list) :-
        atoms(Drug,6,Atom_list,Type_list),
        Type_list \== [c,c,c,c,c,c],
        ring6(Drug,Atom_list,Ring_list,[7,7,7,7,7,7]).

hetero_aromatic_5_ring(Drug,Ring_list) :-
        atoms(Drug,5,Atom_list,Type_list),
        Type_list \== [c,c,c,c,c],
        ring5(Drug,Atom_list,Ring_list,[7,7,7,7,7]).

atoms(Drug,1,[Atom],[T]) :-
        atm(Drug,Atom,T,_,_),
        T \== h.
atoms(Drug,N1,[Atom1|[Atom2|List_a]],[T1|[T2|List_t]]) :-
        N1 > 1,
        N2 is N1 - 1,
        atoms(Drug,N2,[Atom2|List_a],[T2|List_t]),
        atm(Drug,Atom1,T1,_,_),
```

```
        Atom1 @> Atom2,
        T1 \== h.

ring6(Drug,[Atom1|List],[Atom1,Atom2,Atom4,Atom6,Atom5,Atom3],
        [Type1,Type2,Type3,Type4,Type5,Type6]) :-
        bondd(Drug,Atom1,Atom2,Type1),
        memberchk(Atom2,[Atom1|List]),
        bondd(Drug,Atom1,Atom3,Type2),
        memberchk(Atom3,[Atom1|List]),
        Atom3 @> Atom2,
        bondd(Drug,Atom2,Atom4,Type3),
        Atom4 \== Atom1,
        memberchk(Atom4,[Atom1|List]),
        bondd(Drug,Atom3,Atom5,Type4),
        Atom5 \== Atom1,
        memberchk(Atom5,[Atom1|List]),
        bondd(Drug,Atom4,Atom6,Type5),
        Atom6 \== Atom2,
        memberchk(Atom6,[Atom1|List]),
        bondd(Drug,Atom5,Atom6,Type6),
        Atom6 \== Atom3.


ring5(Drug,[Atom1|List],[Atom1,Atom2,Atom4,Atom5,Atom3],
        [Type1,Type2,Type3,Type4,Type5]) :-
        bondd(Drug,Atom1,Atom2,Type1),
        memberchk(Atom2,[Atom1|List]),
        bondd(Drug,Atom1,Atom3,Type2),
        memberchk(Atom3,[Atom1|List]),
        Atom3 @> Atom2,
        bondd(Drug,Atom2,Atom4,Type3),
        Atom4 \== Atom1,
        memberchk(Atom4,[Atom1|List]),
        bondd(Drug,Atom3,Atom5,Type4),
        Atom5 \== Atom1,
        memberchk(Atom5,[Atom1|List]),
        bondd(Drug,Atom4,Atom5,Type5),
        Atom5 \== Atom2.

nitro(Drug,[Atom0,Atom1,Atom2,Atom3]) :-
        atm(Drug,Atom1,n,38,_),
        bondd(Drug,Atom0,Atom1,1),
        bondd(Drug,Atom1,Atom2,2),
        atm(Drug,Atom2,o,40,_),
        bondd(Drug,Atom1,Atom3,2),
        Atom3 @> Atom2,
        atm(Drug,Atom3,o,40,_).

methyl(Drug,[Atom0,Atom1,Atom2,Atom3,Atom4]) :-
        atm(Drug,Atom1,c,10,_),
        bondd(Drug,Atom0,Atom1,1),
        atm(Drug,Atom0,Type,_,_),
        Type \== h,
        bondd(Drug,Atom1,Atom2,1),
        atm(Drug,Atom2,h,3,_),
        bondd(Drug,Atom1,Atom3,1),
```

```
        Atom3 @> Atom2,
        atm(Drug,Atom3,h,3,_),
        bondd(Drug,Atom1,Atom4,1),
        Atom4 @> Atom3,
        atm(Drug,Atom4,h,3,_).


% intersection(+Set1, +Set2, ?Intersection)

interjoin(A,B,C) :-
        intersection(A,B,C),
        C \== [].

bondd(Drug,Atom1,Atom2,Type) :-  bond(Drug,Atom2,Atom1,Type).

member(X,[X|_]).
member(X,[_|T]):-

connected(Ring1,Ring2):-
        Ring1 \= Ring2,
        member(Atom,Ring1),
        member(Atom,Ring2), !.
```

B. Theories obtained

Progol

On the subset of 188 compounds, the Progol theory for "active" molecules for case $\overline{I}$ is as follows.

```
active(A) :-
   logp(A,B), gteq(B,4.180).

active(A) :-
   lumo(A,B), lteq(B,-1.937).

active(A) :-
   logp(A,B), gteq(B,2.740), ring_size_5(A,C).
```

For case I, the rules obtained are as follows (again, ground facts are not shown).

```
active(A) :-
   ind1(A,1.000).
active(A) :-
   atm(A,B,o,40,-0.384).
active(A) :-
   atm(A,B,c,29,0.017).
active(A) :-
   atm(A,B,c,29,C), gteq(C,0.010).
active(A) :-
```

```
   atm(A,B,o,40,-0.389), bond(A,C,B,2), bond(A,D,C,1).
active(A) :-
   bond(A,B,C,1), bond(A,D,B,2), ring_size_5(A,E).
```

On the subset of 42 compounds, cases $\overline{I}$ and I yielded the description below.

```
active(A) :-
   bond(A,B,C,2), bond(A,D,B,1), atm(A,D,c,21,E).
```

Attribute-based algorithms

On the subset of 188 compounds, the best attribute-based algorithm (that is, with highest predictive accuracy) for case $\overline{I}$ is a neural-network. We refrain from reproducing the weights on the nodes here. For case I on the same subset, the best algorithm is regression. The equation obtained is below.

$$\begin{aligned} Activity = \quad & -2.94(\pm 0.33) + 0.10(\pm 0.08) logP - 1.42(\pm 0.16) LUMO \\ & - 2.36(\pm 0.50) Ia + 2.38(0.23) I1 \end{aligned} \tag{1.1}$$

On the subset of 42 compounds, the following decision-tree has the highest predictive accuracy.