Declarative knowledge discovery in industrial databases

Stephen Muggleton[1]
Oxford University Computing Laboratory,
Wolfson Building, Parks Road,
Oxford, OX1 3QD, U.K.
Email: steve@comlab.ox.ac.uk

Abstract

Industry is increasingly overwhelmed by large-volume-data. For example, the pharmaceutical industry generates vast quantities of data both internally as a side-effect of screening tests and combinatorial chemistry, as well as externally from sources such as the human genome project. Industry is also becoming predominantly knowledge-driven. Increased understanding not only improves products, but is also central in market assessment and strategic decision making. From a computer science point of view, the knowledge requirements within industry often give higher emphasis to "knowing that" (declarative or descriptive knowledge) rather than "knowing how" (procedural or prescriptive knowledge). Mathematical logic has always been the preferred representation for declarative knowledge and thus knowledge discovery techniques are required which generate logical formulae from data. Inductive Logic Programming (ILP) is such a technique. Logic programs provide a powerful and flexible representation for constraints, grammars, plans, equations and temporal relationships. New techniques developed within the 1990s allow general-purpose ILP systems to construct logic programs from a mixture of raw data and encoded domain knowledge. This paper will review the results of the last few years' academic pilot studies involving the application of ILP to problems in the pharmaceutical, telecommunications and

[1] Stephen Muggleton has recently accepted an invitation to take up the new Chair of Machine Learning at the University of York.

automobile industries. While predictive accuracy is the central performance measure of data analytical techniques which generate procedural knowledge (neural nets, decision trees, etc.), the performance of an ILP system is determined both by accuracy and degree of insight provided. ILP hypotheses can be easily stated in English and often automatically exemplified pictorially. This allows cross-checking with other relevant domain knowledge. In several of the comparative trials presented ILP systems provided significant insights where other data analysis techniques do not. The scene appears now to be set for commercially-oriented application of ILP in industry.

1 Introduction

At the moment the average size of a datawarehouse for a large company is about 400 Gb. For example, the pharmaceutical industry generates immense quantities of data both internally as a side-effect of screening tests and combinatorial chemistry, as well as externally from sources such as the human genome project. Similarly the car industry is accumulating vast amounts of data ranging from errors in the manufacturing process to data related to serious injuries involved in crashes. The tendency for increased data collection is being compounded by the exponential growth in data available over the World-Wide-Web.

At the same time industry is also increasingly knowledge-driven. Commercial advantage, whether in the stock exchange or the telecommunications industry, is based on strategic and tactical knowledge related to core business. As an example, each new drug brought to the market by a pharmaceutical company costs over £100M. The costs divide between assessment of medicinal activity and safety testing to avoid toxic side-effects. Much of this cost could be reduced with the availability of improved biological and chemical knowledge. Similarly, knowledge leading to detection of fraud in the banking, credit and telecommunications industries could save billions of pounds every year. The requirement is for technologies which efficiently extract insightful knowledge from databases. One might ask what form such knowledge should take.

Within the AI literature, the distinction between procedural and declarative knowledge was first introduced by McCarthy [14], though it is strongly related to the distinction made by the English philosopher Ryle between "knowing how" and "knowing that" [24]. While procedural knowledge can often be conveniently described in algorithmic form, logical sentences are usually used to capture declarative knowledge. Logic Programming languages, such as Prolog, use a subset of mathematical logic to provide a rich declarative representation language which is also executable. As a knowledge representation language Prolog can be translated automatically into readable English text using simple substitution templates (see examples of this in Section 2). This property makes Prolog ideally suited as a language for informing the user of automatic discoveries derived from a database.

However, most of Machine Learning (ML) 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. Similarly, neural net techniques provide black-box decision procedures without any declarative insight. Procedurally-oriented ML has marked up many successful applications [12]. However, client requirements in many domains, such as rational drug design [1], have dictated the need for inductive discovery of declarative knowledge. The key new ML technology which provides support for learning of declarative knowledge is Inductive Logic Programming (ILP) [15, 21]. ILP provides "white box" descriptions by its use of the declarative representation language of Prolog for examples, background knowledge and hypotheses (see Section 3 for definitions of these technical terms).

Many of the techniques in Machine Learning are now in standard use for the purpose of what one might call "one-way learning", i.e. the machine learns, but its human user does not. The emphasis in ILP is on what one might call "two-way discovery", in which not only the machine, but also its human user learn from the analysis of a database. Allowing experts to encode and debug their own knowledge of a domain in a readable way has proved to be of primary importance in this process. Standard, robust ILP systems are now widely available (see Section 4), and two-way discovery has been demonstrated using ILP in a wide range of applications [2] (see Sections 5 and 6).

This paper is arranged as follows. Section 2 introduces the representation of logic programming via various real-world examples. In Section 3 the technical framework of ILP is introduced and some of the ongoing theoretical issues being explored are referenced. The reader is directed towards various state-of-the-art ILP systems in Section 4. Published results of discovery in databases relevant to the pharmaceutical industry are given in Section 5. Ongoing industrially-oriented ILP discovery projects being carried out at the Oxford ML group are discussed in Section 6. Section 7 concludes the paper and gives an indication of future directions for research.

2 Examples of inductive logic programming

This section uses examples from real-world knowledge discovery problems to introduce the use of logic programs as a knowledge representation language. Each example shows the strength of logic programs as a representation for complex constraints given a sufficient set of terms encoded in the background knowledge. The background knowledge is encoded with the help of a domain expert in the relevant field. Below each example is associated with the partner involved in the corresponding knowledge discovery project.

2.1 Imperial Cancer Research Fund example

Since the late 1980s the author's Machine Learning group has carried out collaborative research [19, 7, 26, 8, 25] with Mike Sternberg's Molecular Modelling group at the Imperial Cancer Research Fund (ICRF). The following is a typical logic program *clause* (usually called a definite clause) expressing a constraint relating the three-dimensional structure of a protein to its amino acid sequence.

```
beta(Prot,Pos) :-
    pref(Prot,Pos,Pref), Pref>0.8,
    coil(Prot,Pos+4), setof(X,beta40(Prot,Pos,X),S),
    length(S,Sz), Sz>=4.
```

Such clauses can be learned using ILP systems such as Progol (see Section 4). Logic program clauses, such as the one above, have the following general form.

$$H : -B_1, B_2, .., B_n.$$

The meaning of the clause is that the *head* of the clause H is true if each one of the constraints B_i in the *body* of the clause is true. Thus the ICRF example clause above can be read to say the following.

> There is a beta-strand at position Pos in protein Prot if there is a preference for betas at Pos of at least 0.8 and there is a coil at Pos+4 and there are at least 4 pieces of beta-strand within 40 positions of Pos.

Predicates such as "pref" are defined by the expert using logic program clauses in the background knowledge. Since a logic program is simply a set of logic program clauses, the syntax and semantics of logic programs is very simple, clean and well-defined [13, 6]. Despite the simplicity, logic programs are powerful enough for a general-purpose programming language. All these properties, together with their ease of comprehension make them ideally suited as a general representation for machine learning within ILP.

2.2 Daimler-Benz example

Reza Nakhaeizadeh's data mining group at Daimler Benz have been investigating the application of ILP to the problem of discovering causes for faults in the process of painting cars. The following is an example of the kind of rule produced by the ILP system Progol (see Section 4).

```
fault(Car,top,dirt) :-
    stop(StationX,TimeX),
    cleaned(StationY,TimeY),
    loc_before(StationX,StationY),
    diff_lt(TimeX,TimeY,30),
    car_entered_station(Car,StationX,TimeX1),
    diff_lt(TimeX,TimeX1,6),
    car_left_station(Car,StationY,TimeY1),
    diff_lt(TimeY1,TimeY,5).
```

This rule can be read as follows.

> There is a paint fault caused by dirt on the top of a car body if there is a station X which is located before a station Y. Station X was stopped less than 30 minutes before station Y was cleaned. The car body entered station X less than 6 minutes after it was stopped. Station Y was cleaned less than 5 minutes after the car body left it.

2.3 British Telecom example

Ken Totton's data mining group at British Telecom are interested in being able to extract user interests from documents accessed from a library database. The work has its academic counterpart in an Oxford MSc thesis in which Rupert Parson used Progol to learn user interests from their accesses of World-Wide-Web pages [23]. The following is an example of the kind of logic program clause which Totton's group are interested in discovering.

```
interested(harry,Doc) :-
        topic(Doc,T), novel(T),
        it(T), size(Doc,S), S<10.
```

This rule can be read as follows.

> Harry is interested in a document Doc if it is new, its subject is IT and it has less than ten pages.

3 Framework and technical results

There are three primary logical constituents of an ILP discovery problem, each of which are logic programs. The constituents are as follows.

Background knowledge. The background knowledge defines domain specific relations such as the amino acid sequence of the given proteins (ICRF example, Section 2.1) or car assembly dependencies (Daimler-Benz example, 2.2). Background knowledge is also used to define problem specific constraints such as "each amino acid in a protein is part of either an alpha helix, a beta-sheet or a coil", as well as to define problem-independent knowledge such as the notion of a temporal interval.

Examples. Examples can be either *positive*, such as "Harry is interested in document1" (BT example, Section 2.3) or *negative* such as "Harry is not interested in document 5".

Hypothesis. This is a set of clauses which explain the examples in terms of the background knowledge, such as any one of the clauses in Sections 2.1, 2.2 and 2.3. It is the automatic construction and acceptance of

> the hypothesis which is the process usually referred to as "machine learning" or in the case of insightful and novel hypotheses "machine discovery".

The aim in ILP is to find an hypothesis H which when added to the background knowledge B allows logical derivation of the examples E. It is essential also that H should be logically consistent with the constraints in both B and E.

Topics of interest within the theory of ILP include the *completeness* of inductive inference mechanisms [10, 11], the rate at which correct prediction increases with increasing numbers of examples [22, 4, 3, 20] as well as statistical criteria for acceptance of hypotheses [16, 18].

4 Implementations

During the 1990s a large number of ILP systems were developed and compared on academic datasets. Most of these implementations and benchtest datasets have recently been collected together and made publicly available by the European Community's network of excellence project ILPnet. The ILPnet systems and datasets can be accessed at the following World-Wide-Web address.

http://www-ai.ijs.si/ilpnet.html

The state-of-the-art ILP system which was used throughout the knowledge discovery applications described in Sections 5 and 6 is Progol [17]. Progol is the ILP system which has been used most widely for applications. It is written in C and includes a built-in Prolog[2] interpreter. The current version, CProgol4.2 has source code, example files and a 40 page manual freely available (for academic research) by anonymous ftp from ftp.comlab.ox.ac.uk in directory pub/Packages/ILP/progol4.2. Progol is also available under license for commercial purposes.

[2] Progol is simply Prolog reversed in the middle.

```
Molecule A is an ACE inhibitor if:
   molecule A can bind to zinc at a site B, and
   molecule A contains a hydrogen acceptor C, and
   the distance between B and C is 7.899 +/- 1.000 Angstroms, and
   molecule A contains a hydrogen acceptor D, and
   the distance between B and D is 8.475 +/- 1.000 Angstroms, and
   the distance between C and D is 2.133 +/- 1.000 Angstroms, and
   molecule A contains a hydrogen acceptor E, and
   the distance between B and E is 4.891 +/- 1.000 Angstroms, and
   the distance between C and E is 3.114 +/- 1.000 Angstroms, and
   the distance between D and E is 3.753 +/- 1.000 Angstroms.
```

Figure 1: The geometric constraint discovered by Progol for ACE inhibition

5 Discovery of biological function

The majority of pharmaceutical R&D is based on finding slightly improved variants of patented active drugs. This involves laboratories of chemists synthesising and testing hundreds of variants of compounds. The average cost of developing a single new drug is more than £100M.

5.1 Drug activity

Drugs are typically small molecules which interact with metabolic proteins, which are large molecules. The shape and charge distribution of drugs must be complementary to that of the "binding site" on the target protein. However, in over 70% of all drug projects carried out by pharmaceutical companies the shape of the binding site is unknown, and has to be inferred from the activities of successful drugs.

A range of specialists are involved within the the pharmaceutical industry. These include computational chemists, molecular biologists, pharmacologists, 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

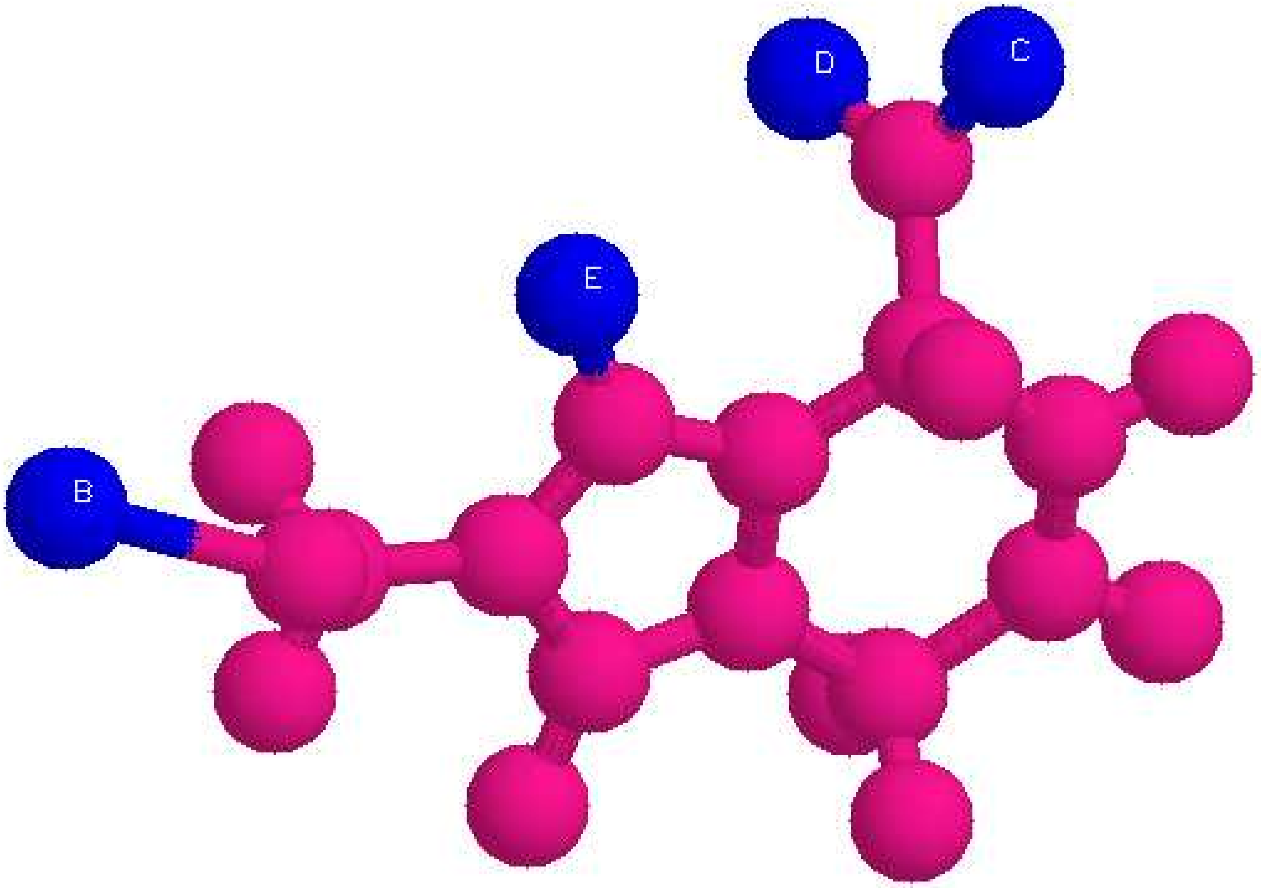

Figure 2: ACE inhibitor number 1 with highlighted 4-point pharmacophore.

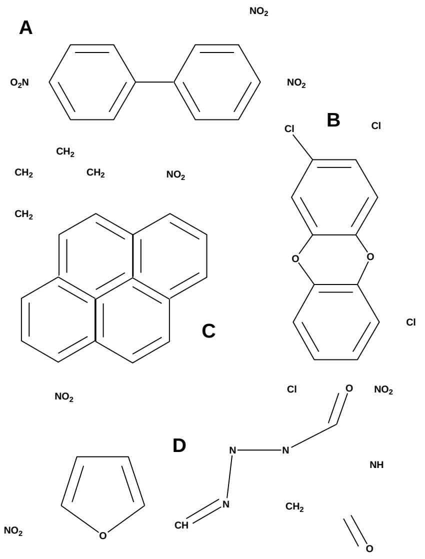

Figure 3: A sample of mutagenic molecules, demonstrating the degree of heterogenicity

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.

Figure 1 shows the English description of such a constraint for Angiotensin-Converting Enzyme (ACE) inhibition. The constraint was discovered in a blind trial during a collaboration between Oxford Machine Learning Group and the Computational Chemistry Group at Pfizer UK, using the ILP system Progol. The work is described in [5], and is an extension of earlier academic work described in [7]. The constraint is illustrated visually in Figure 2, in which one of the example drug molecules is shown with the logical variables from the rule (A,B,C,D,E) superimposed onto the corresponding atoms.

5.2 Drug toxicity

Marketable drugs must not only have medicinal activity, such as the ACE inhibitor in the last section, but also have low toxicity. An important and poorly understood property related to toxicity is mutagenicity. Molecules are mutagenic if they destroy human DNA. The mutagenicity of molecules

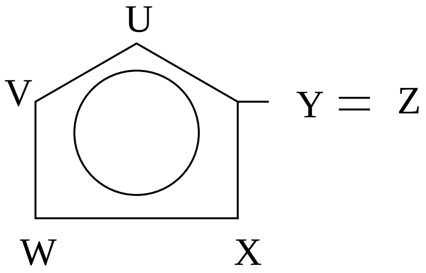

Figure 4: The structural alert for mutagenicity discovered by Progol

is correlated to their carcinogenicity, or tendency to be cancer-producing.

In a study described in [8] the system Progol was used to discover a new structural alert for mutagenicity. Figure 3 shows some of the mutagenic molecules in the examples set. In Figure 4 the substructure which was discovered by Progol to be responsible for 88% of mutagenic molecules, not predicted to be so by linear regression.

Recently a set of rules developed by Progol for predicting carcinogenicity were entered in a global competition run by by the National Toxicology Program (NTP) of the National Institute of Environmental Health Studies in the USA. In the initial results reported in [9] the Progol rule predictions came top out of all systems which were provided with only public data for training. Recent experiments have shown that when Progol's mutagenic rules are added to its other rules derived from the NTP data, the predictive accuracy increases from 64% to 72%, making it the top equal predictor out of all those in the competition.

5.3 Protein shape prediction

Drug design can also be made easier by predicting the shape of the protein binding site. Large amounts of data relating protein amino acid sequence to 3-D shape are now available from the human genome project. Studies reported in [19] and [26] showed that ILP gives accurate predictions of secondary structure and edge relationships within proteins. Many of the patterns discovered using ILP had not been noticed during several years of visual inspection of the proteins by molecular biologists.

6 Applications at Oxford University ML Group

The following is a list of ongoing knowledge discovery projects at the Oxford University Machine Learning group. All of these projects involve applications of the ILP system Progol.

Pharmacophore discovery. This is joint research with Pfizer UK. Some initial results are described in Section 5.1.

Protein topology. This is a joint project with Imperial Cancer Research Fund aimed at discovering the existence of high-level similarities in the fold arrangements of proteins. The project is support under the bio-informatics initiative of the BBSRC and EPSRC.

Peptide motif discovery. This is a collaborative project with Smith-Kline-Beecham.

Tag disambiguation. This is a project aimed at finding rules which help disambiguating parts-of-speech tags for natural language corpora. The research is supported by the ESPRIT Long Term Rsearch project ILP2.

M25 traffic flow analysis. This project is aimed at discovering flow-speed relations in data describing traffic flow on the M25 motorway. The research is being carried out in collaboration with Smith Engineering and the Department of Transport.

Learning WWW user interests. This project aims at finding properties which are related to WWW pages which users find particularly interesting. Such rules can be used easily checked and debugged by users, and can be used in searches for new WWW pages of interest to the user. The project is supported under a CASE studentship by British Telecom.

Fraud detection. This project is aimed at finding unusual patterns of telephone usage, which might indicate fraudulent use of private phones. The project was supported under a British Telecom fellowship.

7 Conclusion

Data is becoming available throughout industry in increasingly larger quantities. Industry is also becoming increasingly knowledge intensive. The need is for a technology which can be used to find insightful declarative knowledge from data. This paper claims that because of its declarative representation language, ILP is the prime candidate for such a technology. Successes of ILP are being progressively demonstrated in the pharmaceutical, automotive and telecommunications industries. The time is ripe for the transfer of ILP from academic laboratories into wide-scale industrial application.

Acknowledgements

This research was supported partly by the Esprit Basic Research Action ILP2 (project 20237), the EPSRC project 'Experiments with Distribution-Based Machine Learning', the SERC project 'Experimental Application and Development of Inductive Logic Programming' and a SERC Advanced Research Fellowship held by Stephen Muggleton. Stephen Muggleton is also supported by a Research Fellowship at Wolfson College, Oxford.

References

[1] 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.

[2] I. Bratko and S. Muggleton. Applications of inductive logic programming. *Communications of the ACM*, 38(11):65–70, 1995.

[3] W. Cohen. Learnability of restricted logic programs. In S. Muggleton, editor, *Proceedings of the 3rd International Workshop on Inductive Logic Programming (Technical report IJS-DP-6707 of the Josef Stefan Institute, Ljubljana, Slovenia)*, pages 41–72, 1993.

[4] S. Dzeroski, S. Muggleton, and S. Russell. PAC-learnability of determinate logic programs. In *Proceedings of the 5th ACM Workshop on Computational Learning Theory*, Pittsburg, PA, 1992.

[5] Paul Finn, Stephen Muggleton, David Page, and Ashwin Srinivasan. Discovery of pharmacophores. *Machine Learning Journal*, 1997. Submitted to special issue on KDD.

[6] C.J. Hogger. *Essentials of logic programming.* Oxford University Press, Oxford, 1990.

[7] R. King, S. Muggleton, R. Lewis, and M. 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. *Proceedings of the National Academy of Sciences*, 89(23), 1992.

[8] 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.

[9] R. King and A. Srinivasan. Prediction of rodent carcinogenicity bioassays from molecular structure using inductive logic programming. *Environmental Health Perspectives*, 104(5), 1996.

[10] P.R. van der Laag and S.H. Nienhuys-Cheng. Subsumption and refinement in model inference. In P. Brazdil, editor, *Proceedings of the 6th European Conference on Machine Learning*, volume 667 of *Lecture Notes in Artificial Intelligence*, pages 95–114. Springer-Verlag, 1993.

[11] P.R. van der Laag and S.H. Nienhuys-Cheng. Existence and nonexistence of complete refinement operators. In Bergadano F. and De Raedt L., editors, *Proceedings of the 7th European Conference on Machine Learning*, volume 784 of *Lecture Notes in Artificial Intelligence*, pages 307–322. Springer-Verlag, 1994.

[12] P. Langley and H. Simon. Applications of machine learning and rule induction. *Communications of the ACM*, 38(11):54–64, 1995.

[13] J.W. Lloyd. *Foundations of Logic Programming.* Springer-Verlag, Berlin, 1984.

[14] 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)).

[15] S. Muggleton. Inductive logic programming. *New Generation Computing*, 8(4):295–318, 1991.

[16] S. Muggleton. Bayesian inductive logic programming. In M. Warmuth, editor, *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory*, pages 3–11, New York, 1994. ACM Press.

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

[18] S. Muggleton. Learning from positive data. In *Proceedings of the Sixth Inductive Logic Programming Workshop*, Stockholm University, 1996.

[19] S. Muggleton, R. King, and M. Sternberg. Protein secondary structure prediction using logic-based machine learning. *Protein Engineering*, 5(7):647–657, 1992.

[20] S. Muggleton and C.D. Page. A learnability model for universal representations. Technical Report PRG-TR-3-94, Oxford University Computing Laboratory, Oxford, 1994.

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

[22] D. Page and A. Frisch. Generalization and learnability: A study of constrained atoms. In S. Muggleton, editor, *Inductive Logic Programming.* Academic Press, London, 1992.

[23] R. Parson. Intelligent agents for the WWW. Master's thesis, Oxford University Computing Laboratory, Oxford, 1995.

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

[25] 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.

[26] M. Sternberg, R. King, R. Lewis, and S. Muggleton. Application of machine learning to structural molecular biology. *Philosophical Transactions of the Royal Society B*, 344:365–371, 1994.