Five Lessons in Representation based on the Application of ILP (Extended Abstract)

Ashwin Srinivasan

1 Introduction

In "Understanding the nature of learning: issues and research directions" [14] R.S. Michalski is concerned with programs that seek to construct representations of some reality. He notes:

> The representations can be in the form of symbolic descriptions, algorithms, simulation models, control procedures, plans, images, or general formal theories. If one stretches the concept of representation to include physical or physiological imprints occurring in the nervous system when one is acquiring a skill, the above view of learning seems also to cover skill acquisition.
>
> From this viewpoint, a fundamental problem in any research on machine learning concerns the form and method used to represent and modify the knowledge or the skill being acquired.

Since 1990, an on-going collaboration between researchers now variously located at the Universities of Edinburgh, Oxford, and Wales; and the Imperial Cancer Research Fund in London has resulted in prominent applications of symbolic machine-learning to problems in molecular biology and biochemistry [3, 4, 5, 7, 8, 9, 10, 11, 12, 20, 22, 26, 27]. Much of this has been accomplished within the setting of Inductive Logic Programming (ILP: see [17]). This abstract is an anecdotal account of some important lessons learnt in the course of this applied work. They pertain to (a) the nature and content of the domain-specific background knowledge; and (b) representing and searching for results that are understandable and interesting to a domain expert. The lessons are drawn largely from the biological applications cited, and consequently, reference to other reports in the literature is scarce. There is no slight intended in this, and it would be interesting to see the extent to which these experiences are reflected in other laboratories.

Lesson 1: First-order Representation

Much of the molecular biology application work has used a first-order representation. It is evident from the results reported in [8] that predictivity of theories could not be the only motive. There, it was found that the predictivity of ILP theories was often no greater than some propositional learners. However, as in earlier results from chess databases [21], the first-order representation had much to commend itself in terms of expressive power and compactness. Thus, a hypothesis such as

> Every molecule that contains a bonded pair of aromatic rings is mutagenic.

can be expressed straightforwardly as a logic program within an ILP system. A system employing a propositional logic would need precoding of the feature like “contains a bonded pair of aromatic rings”. However, this expressive power comes at a computational cost, and the worst-case time-complexity for an ILP learner is usually of a much higher order than a propositional learner. Thus, while in principle, the effect of all other propositional methods (including statistical ones) could be obtained within the logical setting of ILP, practical constraints placed on ILP programs ensure that it is rarely the case. It has therefore been worthwhile investing some effort in identifying precisely the domain-specific concepts that *really* require a first-order representation. For example, using ILP in conjunction with well-established statistical methods has shown considerable initial promise [1, 6, 26].

Moral: A first-order representation is natural when reasoning with complex structured objects, and can enable the construction of compact theories. However, it is worth investigating thoroughly the extent to which statistical and other machine-learning methods can aid the task.

Lesson 2: Declarative Background Knowledge

The nature of ILP programs lends itself naturally to the use of background knowledge that is “largely” declarative. The qualification is needed here because Prolog programs can be written in a procedural fashion. For example, in [3], one part of the background knowledge required the computation of potential sites for a zinc atom given the descriptions of chemical groups that can bind to zinc. This portion of the background knowledge was highly procedural, although written in Prolog, and included code for intersecting planes and spheres, for Gaussian elimination etc. Fortunately, this did not require interaction with the expert chemist.

By contrast, the part of the background knowledge that defined chemical groups that can bind to zinc was highly declarative, being almost a direct translation of pictorial descriptions provided by the chemist. As a result, where ambiguities or questions arose, or where debugging was necessary, these issues were addressed by actually going through that portion of the background knowledge with the chemist. The experiments in [3] involved at least two major debugging phases. The first resulted when the ILP algorithm Progol [18] was not able to re-discover a description (called a “pharmacophore”) common to a given set of molecules. The problem was tracked down to a possible assumption in the chemical background knowledge. This was corrected, and the experiment repeated. A second phase of debugging ensued when Progol's result was found to be overly-specific. The original "fix" was now found to be incorrect, and the true error in the background knowledge was uncovered. This form of iterative experimentation would have been extremely difficult had the background knowledge been non-declarative.

Moral: In the early stages of an application, a declarative encoding of domain-specific knowledge enables expert-interaction in detecting errors.

Lesson 3: Background Knowledge Encoding

The use of background knowledge to construct explanations of is a distinctive feature of ILP. However, the results from experiments concerned with learning simple logic-programs for list processing [23] suggest that the performance of an ILP system is sensitive to the type and amount of background knowledge provided. 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 an adequate explanation of the data. The two questions that arise naturally are: (1) What should be encoded? and (2) How should it be encoded?

In answering the first question, the applications cited have relied largely on the competence of the domain-expert and on extensive literature searches. Thus, for cutting-edge biological problems, it has been possible to find a consensus opinion on what is likely to be relevant – what is unknown is how these pieces fit together.

It would appear from the earlier section that the answer to the second question would be: "as declaratively as possible". However the utility of a declarative encoding is in ensuring correctness. During the course of experiments, mundane concerns of efficiency were usually found to dominate. For the debugged background knowledge, these concerns have often translated into pre-computation and representation as ground facts or bit-sets, definitions that make a clever use of the argument-indexing mechanisms of the Prolog engine, dynamic storage ("caching") of proofs, or representations that are amenable to more efficient proof procedures[1].

Moral: Use all information thought to be relevant by the expert, and by accounts in the published literature. Once satisfied that it is correct, encode the background knowledge in as efficient a form as possible.

[1] For example, in any large-scale natural language applications it is usually more efficient to use a bottom-up "chart-parsing" technique, than the standard Prolog top-down theorem prover. All background definitions are then re-written to reflect this other style of theorem proving (J. Cussens, personal communication).

Lesson 4: Program Heuristics vs. User Needs

Heuristics built-in to a general purpose learning program sometimes do not match particular problem domains. For example, for the problem in [3], chemical intuition seeks to find the most complex pharmacophore common to all active molecules. This is clearly in conflict with a program which seeks to minimise a function that implements a naive encoding of the Minimum Description Length [24] principle. Similar problems have surfaced in other areas, where the task required minimising a numerical cost function [6, 25], or learning rules in a labelled logic [2].

This obstacle has variously been overcome by manual intervention [3], the use of algorithm that accepts domain-specific search specifications [25], and using a function that maximises the Bayesian posterior probability [19].

Moral: Obtain a clear specification of the cost function to be minimised for a problem. Ensure that the theories are obtained by minimising this function.

Lesson 5: Communicability of Results

Despite early and repeated recognition of the importance of comprehensibility in machine learning [13, 15, 16], the emphasis in the testing methodology has been on predictive accuracy. While accurate theories are expected, the overriding factor in the successful application of ILP to biological problems has been the comprehensibility of results to domain experts. This has made it possible to abstract general biochemical principles worthy of publication in journals in the field.

This "user-friendliness" stemmed not from the Prolog rules – which were usually viewed with some suspicion and considerable distaste – but from English translations of the rules or molecular visualisations of instances entailed by them.

Moral: It is important to communicate the knowledge learnt in a language acceptable to the user. This is rarely Prolog.

Acknowledgements

These lessons were not learnt in isolation. It is a pleasure to acknowledge my teachers and classmates: Professor Donald Michie, Professor Claude Sammut, Stephen Muggleton, Michael Sternberg, Paul Finn, Ross King, David Page, James Cussens, and Rui Camacho.

References

[1] J. Cussens. Part-of-speech tagging using Progol. In *Proceedings of the Seventh Inductive Logic Programming Workshop.* Springer-Verlag, Berlin, 1997. To appear.

[2] James Cussens, Anthony Hunter, and Ashwin Srinivasan. Generating explicit orderings for non-monotonic logics. In *Proc. of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*, pages 420–425. MIT Press, 1993.

[3] P. Finn, S. Muggleton, D. Page, and A. Srinivasan. Discovery of Pharmacophores. *Machine Learning Journal (conditionally accepted)*, 1997.

[4] J.D. Hirst, R.D. King, and M.J.E. Sternberg. Quantitative structure-activity relationships by neural networks and inductive logic programming. ii. the inhibition of dihydrofolate reductase by pyrimidines. *Journal of Computer-Aided Molecular Design*, 8:421–432, 1994.

[5] J.D. Hirst, R.D. King, and M.J.E. Sternberg. Quantitative structure-activity relationships by neural networks and inductive logic programming. ii. the inhibition of dihydrofolate reductase by triazines. *Journal of Computer-Aided Molecular Design*, 8:421–432, 1994.

[6] A. Karalic and I. Bratko. First-Order Regression. *Machine Learning Journal*, 1996. Special Issue on ILP, to appear.

[7] R. King and M.J.E. Sternberg. A machine learning approach for the prediction of protein secondary structure. *Journal of Molecular Biology*, 216:441–457, 1990.

[8] R.D. King, A.Srinivasan, and M.J.E. Sternberg. Relating chemical activity to structure: an examination of ILP successes. *New Gen. Comput.*, 13(3,4), 1995.

[9] R.D. King, S. Muggleton, A.Srinivasan, C. Feng, R.A. Lewis, and M.J.E. Sternberg. Drug design using inductive logic programming. In *Proceedings of the 26th Hawaii International Conference on System Sciences*, Los Alamitos, 1993. IEEE Computer Society Press.

[10] 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.

[11] 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.

[12] R.D. King and A. Srinivasan. Prediction of rodent carcinogenicity bioassays from molecular structure using inductive logic programming. *Environmental Health Perspectives*, 104(5):1031–1040, 1996.

[13] 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.

[14] R.S. Michalski. Understanding the nature of learning: issues and research directions. In R. Michalski, J. Carbonnel, and T. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, volume 2, pages 3–25. Kaufmann, Los Altos, CA, 1986.

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

[16] D. Michie, D.J. Spiegelhalter, and C.C. Taylor, editors. *Machine Learning, Neural and Statistical classification.* Ellis-Horwood, New York, 1994.

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

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

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

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

[21] S.H. Muggleton, M.E. Bain, J. Hayes-Michie, and D. Michie. An experimental comparison of human and machine learning formalisms. In *Proceedings of the Sixth International Workshop on Machine Learning.* Kaufmann, 1989.

[22] S.H. Muggleton, A. Srinivasan, R.D. King, and M.J.E. Sternberg. Biochemical knowledge discovery using Inductive Logic Programming. In R. Michalski, M. Kubat, and I. Bratko, editors, *Methods and Applications of Machine Learning, Data Mining and Knowledge Discovery.* John Wiley, 1996.

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

[24] J. Rissanen. Modeling by Shortest Data Description. *Automatica*, 14:465–471, 1978.

[25] A. Srinivasan and R.C. Camacho. Experiments in numerical reasoning with ILP. *Journal of Logic Programming (accepted)*, 1997.

[26] A. Srinivasan and R.D. King. Feature construction with inductive logic programming: a study of quantitative predictions of biological activity aided by structural attributes. In *Proceedings of the Sixth Inductive Logic Programming Workshop*, LNAI, Berlin, 1996. Springer-Verlag.

[27] 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.