PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON FIFTH GENERATION COMPUTER SYSTEMS 1992,
edited by ICOT. © ICOT, 1992



# Developments in Inductive Logic Programming

Stephen Muggleton
The Turing Institute,
36 North Hanover Street,
Glasgow G1 2AD,
UK.

## Abstract

Inductive Logic Programming (ILP) is a research area formed at the intersection of Machine Learning and Logic Programming. ILP systems develop predicate descriptions from examples and background knowledge. The examples, background knowledge and final descriptions are all described as logic programs. A unifying theory of Inductive Logic Programming is being built up around lattice-based concepts such as refinement, least general generalisation, inverse resolution and most specific corrections. In addition to a well established tradition of learning-in-the-limit results, recently some results within Valiant's PAC-learning framework have been demonstrated for ILP systems. Presently successful applications areas for ILP systems include the learning of structure-activity rules for drug design, finite-element mesh analysis design rules, primary-secondary prediction of protein structure and fault diagnosis rules for satellites.

## 1 Introduction

Deduction and induction have had a long strategic alliance within science and philosophy. Whereas the former enables scientists to predict events from theories, the latter builds up the theories from observations. The field of Inductive Logic Programming [6, 8] unifies induction and deduction within a logical setting, and has already provided notable examples of the discovery of new scientific knowledge in the area of molecular biology [5, 7].

## 2 Theory

In the general setting an ILP system $S$ will be given a logic program $B$ representing background knowledge and a set of positive and negative examples $\langle E^+, E^- \rangle$, typically represented as ground literals. In the case in which $B \not\models E^+$, $S$ must construct a clausal hypothesis $H$ such that

$$B \wedge H \models E^+$$

where $B$, $H$ and $E^-$ are satisfiable. In some approaches [16, 13] $H$ is found via a general-to-specific search through the lattice of clauses. This lattice is rooted at the top by the empty clause and is partially ordered by $\theta$-subsumption ($H$ $\theta$-subsumes $H'$ with substitution $\theta$ whenever $H\theta \subseteq H'$). Two clauses are treated as equivalent when they both $\theta$-subsume each other. Following on from work by Plotkin [12], Buntine [1] demonstrated that the equivalence relation over clauses induced by $\theta$-subsumption is generally very fine relative to the the equivalence relation induced by entailment between two alternative theories with common background knowledge. Thus when searching for the recursive clause for *member/2*, infinitely many clauses containing the appropriate predicate and function symbols are $\theta$-subsumed by the empty clause. Very few of these entail the appropriate examples relative to the base case for *member/2*.

Specific-to-general approaches based on Inverse Resolution [9, 14, 15] and relative least general generalisation [1, 10] maintain admissibility of the search while traversing the coarser partition induced by entailment. For instance Inverse Resolution is based on inverting the equations of resolution to find candidate clauses which resolve with the background knowledge to give the examples. Inverse resolution can also be used to add new theoretical terms (predicates) to the learner's vocabulary. This process is known as *predicate invention*.

Several early ILP authors including Plotkin [12] and Shapiro [16] proved learning in the limit results. Recently, ILP learnability results have been proved within Valiant's PAC framework for learning a single definite clause [11] and in [3] for learning a multiple clause predicate definition assuming the examples are picked from a simple-distribution.

# 3 Applications

ILP is rapidly developing towards being a widely applied technology. In the scientific area, the ILP system Golem [10] was used to find rules relating the structure of drug compounds to their medicinal activity [5]. The clausal solution was demonstrated to give meaningful descriptions of the structural factors involved in drug activity with higher acuracy on an independent test set than standard statistical regression techniques.

In the related area of predicting secondary structure of proteins from primary amino acid sequence [7] Golem rules had an accuracy of 80% on an independent test set. This was considerably higher than results of other comparable approaches.

Golem has also been used for building rules for finite-element-mesh analysis [2] and for building temporal fault diagnosis rules for satellites [4].

# 4 Conclusion

Inductive Logic Programming is developing into a new logic-based technology. The field unifies induction and deduction within a well-founded theoretical framework. ILP is likely to continue extending the boundaries of applicability of machine learning techniques in areas which require machine-construction of structurally complex rules.

# References

[1] W. Buntine. Generalised subsumption and its applications to induction and redundancy. *Artificial Intelligence*, 36(2):149–176, 1988.

[2] B. Dolsak and S. Muggleton. The application of Inductive Logic Programming to finite element mesh design. In S.H. Muggleton, editor, *Inductive Logic Programming*, London, 1992. Academic Press.

[3] S. Dzeroski, S. Muggleton, and S. Russell. Pac-learnability of determinate logic programs. TIRM, The Turing Institute, Glasgow, 1992.

[4] C. Feng. Inducing temporal fault dignostic rules from a qualitative model. In S.H. Muggleton, editor, *Inductive Logic Programming*. Academic Press, London, 1992.

[5] 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 (to appear)*, 1992.

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

[7] S. Muggleton, R. King, and M. Sternberg. Predicting protein secondary-structure using inductive logic programming, 1992. submitted to Protein Engineering.

[8] S.H. Muggleton. *Inductive Logic Programming*. Academic Press, 1992.

[9] S.H. Muggleton and W. Buntine. Machine invention of first-order predicates by inverting resolution. In S.H. Muggleton, editor, *Inductive Logic Programming*. Academic Press, London, 1992.

[10] S.H. Muggleton and C. Feng. Efficient induction of logic programs. In S.H. Muggleton, editor, *Inductive Logic Programming*, London, 1992. Academic Press.

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

[12] G.D. Plotkin. *Automatic Methods of Inductive Inference*. PhD thesis, Edinburgh University, August 1971.

[13] R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[14] C. Rouveirol. Extensions of inversion of resolution applied to theory completion. In S.H. Muggleton, editor, *Inductive Logic Programming*. Academic Press, London, 1992.

[15] C. Sammut and R.B Banerji. Learning concepts by asking questions: In R. Michalski, J. Carbonnel, and T. Mitchell, editors, *Machine*

*Learning: An Artificial Intelligence Approach. Vol. 2*, pages 167–192. Kaufmann, Los Altos, CA, 1986.

[16] E.Y. Shapiro. *Algorithmic program debugging.* MIT Press, 1983.