Inductive Logic Programming
(A Short Introduction and a Thesis Abstract)

Johannes Fürnkranz
E-mail: juffi@ai.univie.ac.at
Austrian Research Institute for Artificial Intelligence*
Schottengasse 3, A-1010 Vienna, Austria

Inductive Logic Programming

Inductive Logic Programming (ILP) can be viewed as research in the intersection of Logic Programming and inductive Machine Learning. Informally speaking the field is concerned with the induction of PROLOG programs. Being able to express the discovered knowledge in a first-order logic representation language can overcome some of the limitations of classical learning algorithms. The representational power of these algorithms is usually restricted to propositional domain theories such as decision trees in the well-known ID3 family (Quinlan 1986) or propositional Horn clauses as in AQ (Michalski, Mozetič, Hong, and Lavrač 1986) or CN2 (Clark and Niblett 1989). ILP algorithms, on the other hand, can not only test attributes for specific values, but also make use of relations (like equality) between the values of different attributes.

Inductive Logic Programming has quickly developed to a very active research field. The expressiveness of first order logic has lead to many different research areas within ILP. *Relational Learning* algorithms learn classification rules for a concept and are thus closely related to research in Machine Learning. The program typically receives a large collection of positive and negative examples from real-world databases as well as background knowledge in the form of relations. The prototypical example for this research is FOIL (Quinlan 1990) and its various successors, but there are several other approaches like LINUS (Lavrač and Džeroski 1993) and GOLEM (Muggleton and Feng 1990). On the other end of the spectrum are algorithms that rely more on concepts from logic programming. For example systems like CIGOL ("logic" backwards) (Muggleton and Buntine 1988) try to invert one step of a resolution proof, i.e. they are given a consequent

*The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry of Science and Research.

Given:

- a set of positive and negative training examples for the *target relation*
- background knowledge (typically in the form of relational tables)

Find:

- a logic program that is able to derive all of the positive and none of the negative training examples using the relations provided in the background knowledge.

Figure 1: A Generic Relational Learning Task

and preconditions and try to guess a rule that can prove the consequent from the preconditions. To invent recursive predicates, typically all resolution steps involving the recursive predicate should be considered at once, a process that has been called inverting implication (Lapointe, Ling, and Matwin 1993). Between these two extremes, research is prospering on problems like predicate invention (Stahl 1993), theory revision (Shapiro 1982; De Raedt 1992; Wrobel 1994) or discovery of full first-order clauses (De Raedt and Bruynooghe 1993).

An excellent overview of the history of the field can be found in (Sammut 1993), a selection of some of the most important papers in (Muggleton 1992). (Lavrač and Džeroski 1993) is an introductory book on Inductive Logic Programming with a strong focus on relational learning systems, in particular on LINUS and mFOIL. An excellent introduction with a strong emphasis on the logic programming foundations of ILP can be found in (Muggleton and De Raedt 1994).

Relational Learning

Relational Learning systems typically are designed to learn classification rules from real-world databases. Main characteristics of real-world databases are that they are large and unreliable. Algorithms designed to work on them must therefore be efficient and noise-tolerant.

Relational Learning systems are designed for this purpose and have already been applied to various real-world problems, like finite-element mesh design, medical diagnosis, chess endgames, natural language understanding, and many more. Some of the systems even produced new knowledge that was of considerable interest for researchers in the application domain and has been published in journals of their subject area (Muggleton et al. 1992; Sternberg et al. 1992; King et al. 1992). For a short overview of applications of relational learning systems consult (Bratko and King 1994).

Different methods have been found to solve this problem:

Top-Down Induction: Many algorithms use the so-called *separate-and-conquer* strategy which is well-known from propositional learning algorithms like AQ or CN2. It tries to find a complete and consistent theory by iterative *specialization* of the most general (empty) theory. The prototypical system for this approach is FOIL (Quinlan 1990).

Bottom-Up Induction: As opposed to top-down induction, the bottom-up approaches search for a complete and consistent theory by iterative *generalization* of clauses. The most common generalization operators are based on *inverse resolution* as in CIGOL (Muggleton and Buntine 1988) or on *relative least general generalization* (Plotkin 1971) as in GOLEM (Muggleton and Feng 1990). A generic framework for bottom-up induction can be found in (Rouveirol, Adé, and de Raedt 1993).

Representation Change: The systems of this class approach the problem by reformulating it in a language that conventional Machine Learning systems can use. Prototypical for this approach is the LINUS system (Lavrač and Džeroski 1993) that transforms the relational learning problem into an attribute-value representation that can be used by propositional learning algorithms.

All three approaches impose different restrictions on the class of programs they can learn. LINUS e.g. is not able to learn recursive predicates, while GOLEM is limited to a restricted form of ground background knowledge.

Efficient Pruning Methods for Relational Learning

As real-world problems very often contain erroneous values and incomplete or misclassified examples, noise handling is an important problem that has to be faced. *Pruning* is the common framework for methods that avoid to *overfit* the noise in the data. The basic idea is to incorporate a bias towards more general and simpler theories that misclassify some of the training examples in the hope that those examples that are not explained by the simple theory are in fact erroneous.

"Efficient Pruning Methods for Relational Learning" was the topic of the author's recent doctoral thesis (Fürnkranz 1994a).[1] The main contribution of this thesis are three new algorithms that improve relational top-down induction with new efficient noise-handling strategies. We describe the robust pre-pruning algorithm FOSSIL, a *Top-Down Pruning* (TDP) approach that combines pre-

[1] The complete thesis appeared as technical report OEFAI-TR-94-28 of the Austrian Research Institute for Artificial Intelligence. It is also available via the Internet (`ftp://ftp.ai.univie.ac.at/papers/oefai-tr-94-28.ps.Z`).

and post-pruning, and finally *Incremental Reduced Error Pruning* (I-REP), a very efficient integration of pre-and post-pruning.

Pre-pruning methods deal with noise during learning. Instead of trying to find a theory that is complete and consistent with the given training data, heuristics — so-called *stopping criteria* — are used to relax this constraint by stopping the learning process although some positive examples might not yet be explained and some of the negative examples may still be covered by the current theory. We review some of the basic algorithms that use this approach, most importantly FOIL, which is the ancestor of most relational learning algorithms. Thereafter we introduce a new system, FOSSIL (Fürnkranz 1994b). While it uses the same basic *separate-and-conquer* learning strategy as FOIL, FOSSIL employs a new search heuristic based on statistical correlation which has several interesting properties that distinguish it from other approaches. Most importantly, we show how the correlation heuristic can effectively deal with noise when used with the simple *cutoff* stopping criterion. This new simple and efficient criterion does not depend on the number of training examples or on the amount of noise in the data and thus seems to be more robust than alternative approaches.

Another family of algorithms deals with noise after learning. These *post-pruning* algorithms typically first induce a theory that is complete and consistent with the training data. Then this theory is examined and clauses and literals that seem to only explain characteristics of the particular training set used and thus not reflect true regularities of the domain are discarded. The quality of the found clauses and literals is commonly evaluated on a separate set of training examples that have not been seen during learning. We review two common post-pruning methods, *Reduced Error Pruning* (REP) (Brunk and Pazzani 1991) and GROW (Cohen 1993), a variant that uses a top-down search. While both methods prove to be very effective in noise-handling, they are also very inefficient.

One of the reasons for the inefficiency of post-pruning methods is that the intermediate theory resulting from the initial overfitting phase can be much more complex than the final theory. Thus a lot of effort has to be wasted in generating and subsequently simplifying this intermediate theory. One way for reducing this problem is to *combine* pre- and post-pruning. For this purpose pre-pruning heuristics are used to reduce (not entirely prevent) the amount of overfitting, so that learning and pruning will be more efficient. A new method, *Top-Down Pruning* (TDP) (Fürnkranz 1994), uses FOSSIL's simple cutoff stopping criterion to systematically vary the overfitting avoidance bias. Theories pruned to different degrees are generated in a top-down, general-to-specific order. The accuracies of the theories are evaluated on a separate set of data and the most specific theory with an accuracy comparable to the accuracy of the best theory so far will be submitted to a subsequent post-pruning phase. Experiments show that this initial top-down search for a better starting theory can be more efficient than the overfitting phase of classical post-pruning algorithms. As this search will typically return a theory that is closer to the final theory, we can also achieve a significant

speed-up in the post-pruning phase along with a slight gain in accuracy.

Motivated by the success of this method, we have developed a more rigorous approach that tightly *integrates* pre- and post-pruning. Instead of learning an entire theory and pruning it thereafter, *Incremental Reduced Error Pruning* (I-REP) (Fürnkranz and Widmer 1994) prunes single clauses right after they have been learned. This new algorithm entirely avoids the learning of an overfitting theory by using post-pruning methods as a pre-pruning stopping criterion and thus significant speedups can be achieved in noisy domains. As it avoids some problems with other approaches that incorporate post-pruning, I-REP also learns more accurate theories.

In a series of experiments in propositional and relational domains we have tried to confirm the quality of our algorithms, and found that in noisy domains, I-REP is significantly faster than all other algorithms an also brings a slight gain in accuracy. However, this advantage was not as clear in propositional test domains as it was in domains that require relations. Pre-pruning as used in FOSSIL appears to be the most stable method and should be used when nothing is known about the domain. A setting of 0.3 for its cutoff parameter has yielded good results in almost all tested domains, which illustrates the robustness of this algorithm against varying noise levels and varying training set sizes.

References

Bratko, I. and R. King (1994). Applications of Inductive Logic Programming. *SIGART Bulletin 5*(1), 43–49.

Brunk, C. A. and M. J. Pazzani (1991). An investigation of noise-tolerant relational concept learning algorithms. In *Proceedings of the 8th International Workshop on Machine Learning*, Evanston, Illinois, pp. 389–393.

Clark, P. and T. Niblett (1989). The CN2 induction algorithm. *Machine Learning 3*(4), 261–283.

Cohen, W. W. (1993). Efficient pruning methods for separate-and-conquer rule learning systems. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, Chambery, France, pp. 988–994.

De Raedt, L. (1992). *Interactive Theory Revision: An Inductive Logic Programming Approach*. Academic Press.

De Raedt, L. and M. Bruynooghe (1993). A theory of clausal discovery. In R. Bajcsy (Ed.), *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, Chambery, France, pp. 1058–1063. Morgan Kaufmann.

Fürnkranz, J. (1994a). *Efficient Pruning Methods for Relational Learning*. Ph. D. thesis, Vienna University of Technology.

Fürnkranz, J. (1994b). FOSSIL: A robust relational learner. In *Proceedings of the European Conference on Machine Learning*, Catania, Italy, pp. 122–137. Springer-Verlag.

Fürnkranz, J. (1994). Top-down pruning in relational learning. In *Proceedings of the 11th European Conference on Artificial Intelligence*, Amsterdam, The Netherlands, pp. 453–457.

Fürnkranz, J. and G. Widmer (1994). Incremental Reduced Error Pruning. In *Proceedings of the 11th International Conference on Machine Learning*, New Brunswick, NJ, pp. 70–77.

King, R., S. Muggleton, R. Lewis, and M. Sternberg (1992). 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).

Lapointe, S., C. Ling, and S. Matwin (1993). Constructive Inductive Logic Programming. In R. Bajcsy (Ed.), *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, Chambery, France, pp. 1030–1036. Morgan Kaufmann.

Lavrač, N. and S. Džeroski (1993). *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood.

Michalski, R. S., I. Mozetič, J. Hong, and N. Lavrač (1986). The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. In *Proceedings of the 5th National Conference on Artificial Intelligence*, Philadelphia, PA, pp. 1041–1045.

Muggleton, S. (Ed.) (1992). *Inductive Logic Programming*. London: Academic Press Ltd.

Muggleton, S. and L. De Raedt (1994). Inductive Logic Programming: Theory and methods. *Journal of Logic Programming 19,20*, 629–679.

Muggleton, S., R. King, and M. Sternberg (1992). Protein secondary structure prediction using logic-based Machine Learning. *Protein Engineering 5*(7), 647–657.

Muggleton, S. H. and W. L. Buntine (1988). Machine invention of first-order predicates by inverting resolution. In *Proceedings of the 5th International Conference on Machine Learning*, pp. 339–352.

Muggleton, S. H. and C. Feng (1990). Efficient induction of logic programs. In *Proceedings of the 1st Conference on Algorithmic Learning Theory*, Tokyo, Japan, pp. 1–14.

Plotkin, G. D. (1971). *Automatic methods of inductive inference*. Ph. D. thesis, University of Edinburgh, Scotland.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning 1*, 81–106.

Quinlan, J. R. (1990). Learning logical definitions from relations. *Machine Learning 5*, 239–266.

Rouveirol, C., H. Adé, and L. de Raedt (1993). Bottom up generalization in I.L.P. In *Proceedings of the IJCAI-93 Workshop on Inductive Logic Programming*, pp. 59–70.

Sammut, C. (1993). The origins of Inductive Logic Programming: A prehistoric tale. In *Proceedings of the 3rd International Workshop on Inductive Logic Programming*, Bled, Slovenia, pp. 177–216.

Shapiro, E. Y. (1982). *Algorithmic Program debugging*. Cambridge: The MIT Press.

Stahl, I. (1993). Predicate Invention in ILP — an overview. In P. B. Brazdil (Ed.), *Machine Learning: ECML-93*, Number 667 in Lecture Notes in Artificial Intelligence, Vienna, Austria, pp. 313–322. Springer-Verlag.

Sternberg, M., R. Lewis, R. King, and S. Muggleton (1992). Modelling the structure and function of enzymes by machine learning. *Proceedings of the Royal Society of Chemistry: Faraday Discussions 93*, 269–280.

Wrobel, S. (1994). Concept formation during interactive theory revision. *Machine Learning 14*(2), 169–191.