N94-34056

Automated Knowledge-Base Refinement

Raymond J. Mooney
Department of Computer Sciences
University of Texas
Austin, TX 78712
mooney@cs.utexas.edu

Abstract

Over the last several years, we have developed several systems for automatically refining incomplete and incorrect knowledge bases. These systems are given an imperfect rule base and a set of training examples and minimally modify the knowledge base to make it consistent with the examples. One of our most recent systems, FORTE, revises first-order Horn-clause knowledge bases. This system can be viewed as automatically debugging Prolog programs based on examples of correct and incorrect I/O pairs. In fact, we have already used the system to debug simple Prolog programs written by students in a programming languages course. FORTE has also been used to automatically induce and revise qualitative models of several continuous dynamic devices from qualitative behavior traces. For example, it has been used to induce and revise a qualitative model of a portion of the Reaction Control System (RCS) of the NASA Space Shuttle. By fitting a correct model of this portion of the RCS to simulated qualitative data from a faulty system, FORTE was also able to correctly diagnose simple faults in this system.

1 Introduction

The problem of revising an imperfect knowledge base (domain theory) to make it consistent with empirical data is a difficult problem that has important applications in the development of expert systems (Ginsberg et al., 1988). Knowledge-base construction can be greatly facilitated by using a set of training cases to automatically refine an imperfect, initial knowledge base obtained from a text book or by interviewing an expert. The advantage of a refinement approach to knowledge-acquisition as opposed to a purely empirical learning approach is two-fold. First, by starting with an approximately-correct theory, a refinement system should be able to achieve high-performance with significantly fewer training examples. Therefore, in domains in which training examples are scarce or in which a rough theory is easily available, the refinement approach has a distinct advantage. Second, theory refinement results in a structured knowledge-base that maintains the intermediate terms and explanatory structure of the original theory. Empirical learning, on the other hand, results in a decision tree or disjunctive-normal-form (DNF) expression with no intermediate terms or explanatory structure. Therefore, a knowledge-base formed by theory refinement is much more suitable for supplying meaningful explanations for its conclusions, an important aspect of the usability of an expert system.

Over the past five years, we have developed a series of machine learning systems that automatically revise incomplete and incorrect domain theories. Section 2 briefly reviews five systems that we have developed and summarizes results from each of them. Section 3 discusses one of these systems, FORTE, in a little more detail. FORTE revises first-order Horn-clause theories and can be viewed as automatically debugging Prolog programs based on examples of correct and incorrect I/O pairs. We briefly summarize our results with using FORTE to debug simple Prolog programs written by undergraduate students and to induce, revise, and diagnose a qualitative model of a portion of the Space Shuttle Reaction Control System.

2 A Series of Knowledge-Base Refinement Systems

By integrating ideas from both explanation-based learning and inductive learning, my students and I have developed a series of systems for automatically revising imperfect knowledge bases of increasing representational complexity.

First, we developed a method called IOU (Induction Over the Unexplained) (Mooney, 1993) for refining overly-general, propositional, Horn-clause domain theories (i.e. if-then rule bases without variables). IOU uses explanation-based methods to learn part of a concept and uses inductive methods over unexplained aspects of examples to impose additional constraints on the final definition. Experiments on real-world data sets for diagnosis of soybean diseases (Michalski and Chilausky, 1980) and human hearing disorders (Porter et al., 1990) demonstrated IOU's ability to use incomplete theories to learn more accurate concepts from fewer examples than a purely inductive learning method like ID3 (Quinlan, 1986). Results in learnability theory (Ehrenfeucht et al., 1989) where used to prove that, under certain conditions, IOU is guaranteed to learn a PAC (probably approximately correct) concept from fewer examples. Finally, IOU was used to model some recent psychological data demonstrating the effect of background theories on human concept acquisition (Wisniewski, 1989). Unfortunately, IOU was restricted to repairing only a certain type of overly-general theory.

Our next system, EITHER (Explanation-based and Inductive Theory Extension and Revision) (Ourston and Mooney, 1990; Ourston and Mooney, in press; Ourston, 1991) was able to refine arbitrarily incorrect propositional Horn-clause theories. EITHER used generic components for deduction, abduction, and induction (ID3) to learn new rules, delete incorrect rules, add antecedents to existing rules, and remove existing antecedents. EITHER was able to successfully refine real expert rule-bases for recognizing promoters in DNA sequences (Towell et al., 1990) and diagnosing soybean diseases, improving the classification accuracy of both theories 30 percentage points using 100 training examples.

Our third system, FORTE (First-Order Revision of Theories from Examples) (Richards and Mooney, 1991; Richards, 1992) was able to refine *first-order* Horn-clause theories by incorporating recently developed methods in inductive logic programming (Muggleton, 1992). FORTE can be viewed as automatically debugging Prolog programs based on examples of correct and incorrect I/O pairs. In fact, it was successfully used to debug simple Prolog programs written by students in an undergraduate course on programming languages. FORTE was also used to automatically induce and revise qualitative models of several continuous dynamic devices from qualitative behavior traces. In

particular, the system induced and revised a qualitative model of a portion of the Reaction Control System (RCS) of the NASA Space Shuttle. FORTE is discussed in more detail in the next section.

Our fourth theory refinement system, RAPTURE (Revising Approximate Probabilistic Theories Using a Repository of Examples) (Mahoney and Mooney, 1993; Mahoney and Mooney, in press) combines symbolic and neural-network learning to refine a certainty-factor rule base. Therefore, this project extended our methods to knowledge bases involving uncertain reasoning. RAPTURE converts a certainty-factor rule base into a network and uses a modified version of connectionist backpropagation (Rumelhart et al., 1986) to adjust certainty factors. If adjusting certainty-factors is insufficient, a symbolic method based on ID3's information gain metric is used to add new rules. Backpropagation and rule addition continue in a cycle until all of the training examples are classified correctly. RAPTURE has successfully revised knowledge bases for three real-world problems: DNA promoter recognition, soybean diagnosis, and the diagnosis of bacterial infections (a version of the MYCIN rule base from Ma and Wilkins (1991)). On the promoter problem, RAPTURE performs significantly better than our previous system, EITHER, and produces a simpler and slightly more accurate rule base than KBANN (Towell et al., 1990), a more standard neural-network theory revisor.

Our most recent system, NEITHER (New EITHER) (Baffes and Mooney, 1993) is a much faster, redesigned version of EITHER that can revise theories with "M of N" rules (rules that fire if any subset of size at least M of their N antecedents are satisfied). It has been tested on refining the promoter domain theory, producing an even simpler rule base with accuracy similar to that produced by RAPTURE and KBANN.

Figure 1 shows learning curves demonstrating the performance of various systems on the DNA promoter recognition problem. A promoter is a genetic region that initiates the first step in the expression of an adjacent gene (*transcription*) by RNA polymerase. The input features are 57 sequential DNA nucleotides (with values A, G, T or C). The data contains 106 examples evenly split between positive and negative. The expert theory provided with the data set, which has 11 rules with a total of 76 literals, is completely overly-specific (proves none of the examples are promoters) and therefore has an initial classification accuracy of only 50%.

The learning curves were generated as follows. Each data set was divided into training and test sets. Training sets were further divided into subsets, so that the algorithms could be evaluated with varying amounts of training data. After training, each system's accuracy was recorded on the test set. To reduce statistical fluctuations, the results of this process of dividing the examples, training, and testing were averaged over 25 runs. Results are shown for the theory revision systems EITHER, RAPTURE, NEITHER (without "M of N" revisions), NEITHER-M-OF-N (with "M of N" revisions), and KBANN, and for the purely inductive systems ID3 (Quinlan, 1986) and neural-network backpropagation (Rumelhart et al., 1986).

All of the revisions systems greatly improve the accuracy of the initial knowledge base and generally perform better than pure induction. The strict rule-based systems (EITHER, NEITHER, and ID3) perform relatively poorly since some aspects of the promoter concept are knowm to fit an M-of-N format. There are several potential sites where hydrogen bonds can form between the DNA and a protein and if enough of these bonds form, promoter activity can occur. Connectionist, probabilistic, and explicit M-of-N systems can represent such concepts more easily than strict Horn-clause theories, which require "M choose N" separate rules to represent an M-of-N concept.

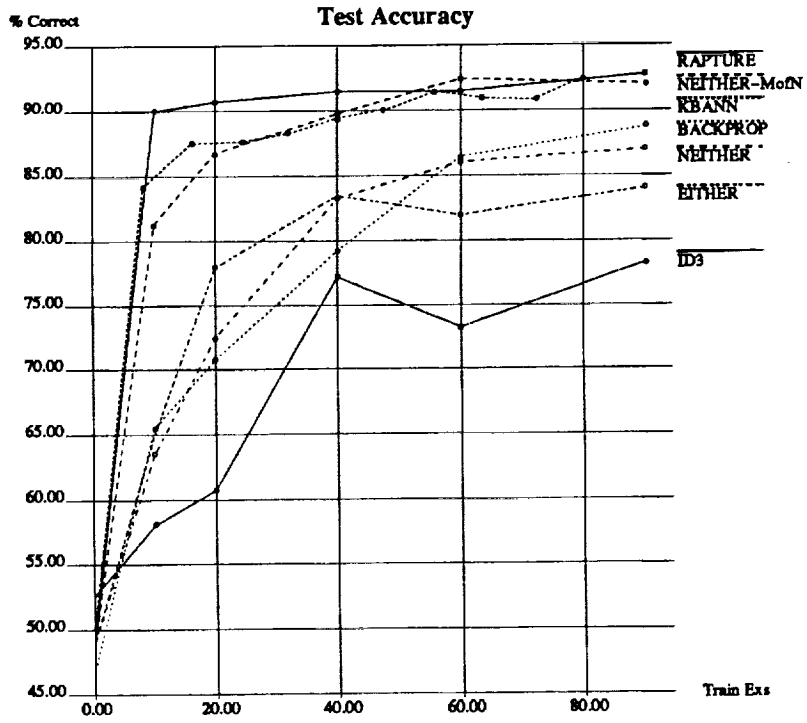

Figure 1: Learning Curves for DNA Promoter Data

3 Overview of FORTE

This section provides a little more detail on our first-order theory revision system. FORTE works by performing a hill-climbing search through a space of specializing and generalizing operators in an attempt to find a minimal revision to a theory that makes it consistent with a batch of training examples.

First, FORTE attempts to prove all positive and negative instances in the training set using the current theory. Positive (negative) instances are tuples of constants that should (should not) satisfy the goal predicate. When a positive instance is unprovable, some program clause needs to be generalized. All clauses that failed during the attempted proof are candidates for generalization. When a negative instance is provable, some program clause needs to be specialized. All clauses that participated in the successful proof are candidates for specialization.

When an error is detected, FORTE identifies all clauses that are candidates for revision. The core of the system consists of a set of operators that generalize or specialize a clause to correctly classify a set of examples. Based on the error, all relevant operators are applied to each candidate clause. The best revision, as determined by classification accuracy on the complete training set, is implemented. This process iterates until the theory is consistent with the training set or until FORTE is caught in a local maximum, i.e. none of the proposed revisions improve overall accuracy.

FORTE's specialization operators include deleting rules and adding antecedents. Several meth-

Program	# of Programs	Training Set Size	Mean Revision Time	% Correct
directed path	4	121 instances	87 seconds	100%
insert after	9	35 instances	82 seconds	100%
merge sort	10	60 instances	199 seconds	100%

Table 1: Summary of program debugging results.

ods are used to determine appropriate antecedents to add to an overly-general clause. One is a hill-climbing method based on the FOIL system Quinlan (1990) for inducing first-order, function-free, Horn-clause rules using an search heuristic based on information gain. Another is called *relational pathfinding* (Richards and Mooney, 1992) and adds a sequence of literals that form a relational path linking all of the arguments of the goal predicate. Since it adds multiple literals at once, relational pathfinding helps overcome local minima problems in FOIL.

FORTE's generalization operators include deleting antecedents and adding rules. Antecedents are chosen for deletion using a greedy algorithm that attempts to maximize the number of additional provable positive examples without causing additional provable negatives. New rules are learned using FOIL and relational pathfinding. FORTE also includes two additional generalization operators (identification and absorption) based on *inverse resolution* as introduced in Muggleton and Buntine (1988). These operators introduce new rules based on repeated patterns of literals found in existing rules.

3.1 Debugging Student Programs

In order to test FORTE's logic program debugging capabilities, we asked students in an undergraduate class on programming languages to hand in their first attempts at writing simple Prolog programs. They gave us their programs after they had satisfied themselves on paper that the programs were correct, but before they tried to run them. The student programs were distributed among three problems: find a path through a directed graph, insert an element into a list, and merge-sort a list. We collected 23 distinctly different incorrect programs, representing a wide variety of errors ranging from simple typographical mistakes to complete misunderstandings of recursion. FORTE was able to debug all of these programs (see Table 1).

3.2 Qualitative Modelling of the Space Shuttle RCS

FORTE has also been used to induce, revise, and diagnosis qualitative models of continuous dynamic systems. Qualitative models suitable for the QSIM qualitative simulation system (Kuipers, 1986) can be represented as Prolog rules by including an antecedent for each of the constraints in the model (such as the flow out of a tank is a monotonically increasing function of the amount in the tank). FORTE can then use qualitative behaviors of the system as examples to revise such a model.

We have applied this approach to qualitative modelling of the Reaction Control System (RCS) of the NASA Space Shuttle. The RCS consists of a number of identical, parallel components; our test domain consisted of one of these components with its valves in fixed positions. Although space prevents us from giving a complete description of the RCS, a simplified view would contain of three

interconnected tanks, plus the thruster outlet. The first tank contains Helium, which it provides at a constant pressure to the fuel tank. The Helium forces fuel out of the fuel tank and into the manifold. From the manifold, the fuel enters the thruster and ignites to provide thrust.

For the purposes of this section, we assume that the valve leading to the thruster is closed (i.e., the thruster is off), the Helium regulator valve is open and providing a constant-pressure supply of Helium, and the valve between the fuel tank and the manifold has just been opened. If the initial pressure in the manifold is lower than the initial pressure in the fuel tank (so that the system is not immediately at equilibrium), then the fuel flows from the fuel tank into the manifold. Providing this single behavior to FORTE allowed FORTE to induce a model for the RCS equivalent to that produced by a QSIM expert (Kay, 1992).

However, since FORTE is a theory refinement system, we can use it in a more sophisticated way. Suppose that the user has a correct system model, but that the system is behaving incorrectly. In this case, we can use theory refinement to revise the correct system model to reflect the actual system behavior. The resulting changes in the model can be viewed as a diagnosis. One of the failures that can occur in the RCS is a leak in one of the manifolds leading from the fuel tank. In order to isolate the leak, the astronauts shut the valve leading from the fuel tank into the manifolds. They then isolate the suspected manifold and reopen the valve connecting the fuel tank and the manifolds. If the leak has been eliminated, the system will quickly reach equilibrium. If the leak has not been isolated, the system will not reach a pressure equilibrium (at least, not before all of the fuel has drained out through the leak).

If FORTE begins with a correct system model along with the system behavior caused by a leak in the manifold, FORTE revises the model by deleting the constraint `minus(D_Amt_Fuel, D_Amt_Man)`. The variable `D_Amt_Fuel` is the amount of fuel leaving the fuel tank and flowing into the manifold. Variable `D_Amt_Man` is the net change in the amount of fuel in the manifold. Normally, the amount of fuel flowing out of the fuel tank should be the same, except for sign, as the net amount of fuel being added to the manifold. Since FORTE deletes this constraint, there must be another influence on the amount of fuel in the manifold, namely, a leak.

4 Conclusion

We have developed a number of systems for automatically refining imperfect knowledge bases by integrating various machine-learning methods. These systems have been successfully tested on a variety of real-world problems, including qualitative modelling of a complex subsystem of the Space Shuttle. We believe our results and those of other researchers in the area demonstrate the promise of automated knowledge base refinement. Hopefully, these methods will continue to be refined and successfully employed to speed the development of knowledge-based systems in additional application areas.

Acknowledgements

Thanks to Brad Richards, Dirk Ourston, Jeff Mahoney, and Paul Baffes as major contributors to the work reported in this paper. This research was supported by the NASA Ames Research Center

under grant NCC 2-629, the National Science Foundation under grant IRI-9102926 and the Texas Advanced Research Program under grant 003658114.

References

Baffes, P., and Mooney, R. (1993). Symbolic revision of theories with M-of-N rules. In *Proceedings of the Thirteenth International Joint Conference on Artificial intelligence*. Chambery, France.

Ehrenfeucht, A., Haussler, D., Kearns, D., and Valiant, L. (1989). A general lower bound on the number of examples needed for learning. *Information and Computation*, 82:267–284.

Ginsberg, A., Weiss, S. M., and Politakis, P. (1988). Automatic knowledge based refinement for classification systems. *Artificial Intelligence*, 35:197–226.

Kay, H. (1992). A qualitative model of the space shuttle reaction control system. Technical Report AI92-188, Artificial Intelligence Laboratory, University of Texas, Austin, TX.

Kuipers, B. J. (1986). Qualitative simulation. *Artificial Intelligence*, 29:289–338.

Ma, Y., and Wilkins, D. C. (1991). Improving the performance of inconsistent knowledge bases via combined optimization method. In *Proceedings of the Eighth International Workshop on Machine Learning*, 23–27. Evanston, IL.

Mahoney, J. J., and Mooney, R. J. (1993). Combining neural and symbolic learning to revise probabilistic rule bases. In Hanson, S., Cowan, J., and Giles, C., editors, *Advances in Neural Information Processing Systems, Vol. 5*, 107–114. San Mateo, CA: Morgan Kaufman.

Mahoney, J. J., and Mooney, R. J. (in press). Combining connectionist and symbolic learning to refine certainty-factor rule-bases. *Connection Science*.

Michalski, R. S., and Chilausky, S. (1980). Learning by being told and learning from examples: An experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis. *Journal of Policy Analysis and Information Systems*, 4(2):126–161.

Mooney, R. J. (1993). Induction over the unexplained: Using overly general domain theories to aid concept learning. *Machine Learning*, 10:79–110.

Muggleton, S., and Buntine, W. (1988). Machine invention of first-order predicates by inverting resolution. In *Proceedings of the Fifth International Conference on Machine Learning*, 339–352. Ann Arbor, MI.

Muggleton, S. H., editor (1992). *Inductive Logic Programming*. New York, NY: Academic Press.

Ourston, D. (1991). *Using Explanation-Based and Empirical Methods in Theory Revision*. PhD thesis, University of Texas, Austin, TX. Also appears as Artificial Intelligence Laboratory Technical Report AI 91-164.

Ourston, D., and Mooney, R. (1990). Changing the rules: a comprehensive approach to theory refinement. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, 815–820. Detroit, MI.

Ourston, D., and Mooney, R. J. (in press). Theory refinement combining analytical and empirical methods. *Artificial Intelligence.*

Porter, B., Bareiss, R., and Holte, R. (1990). Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence*, 45:229–263.

Quinlan, J. (1990). Learning logical definitions from relations. *Machine Learning*, 5(3):239–266.

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

Richards, B., and Mooney, R. (1991). First-order theory revision. In *Proceedings of the Eighth International Workshop on Machine Learning*, 447–451. Evanston, IL.

Richards, B., and Mooney, R. J. (1992). Learning relations by pathfinding. In *Proceedings of the Tenth National Conference on Artificial Intelligence.* San Jose, CA.

Richards, B. L. (1992). *An Operator-Based Approach To First-Order Theory Revision.* PhD thesis, University of Texas, Austin, TX. Also appears as Artificial Intelligence Laboratory Technical Report AI 92-181.

Rumelhart, D. E., Hinton, G. E., and Williams, J. R. (1986). Learning internal representations by error propagation. In Rumelhart, D. E., and McClelland, J. L., editors, *Parallel Distributed Processing, Vol. I*, 318–362. Cambridge, MA: MIT Press.

Towell, G. G., Shavlik, J. W., and Noordewier, M. O. (1990). Refinement of approximate domain theories by knowledge-based artificial neural networks. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, 861–866. Boston, MA.

Wisniewski, E. (1989). Learning from examples: The effect of different conceptual roles. In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, 980–986. Ann Arbor, MI.