Appears in *Machine Learning: A Multistrategy Appraoch, Vol. IV*
R.S. Michalski & G. Teccuci (Eds.), pp.141-164, Morgan Kaufman, San Mateo, CA, 1994

A MULTISTRATEGY APPROACH TO THEORY REFINEMENT

Raymond J. Mooney
(University of Texas at Austin)

Dirk Ourston
(British Petroleum Research)

Abstract

This chapter describes a multistrategy system that employs independent modules for deductive, abductive, and inductive reasoning to revise an arbitrarily incorrect propositional Horn-clause domain theory to fit a set of preclassified training instances. By combining such diverse methods, EITHER is able to handle a wider range of imperfect theories than other theory revision systems while guaranteeing that the revised theory will be consistent with the training data. EITHER has successfully revised two actual expert theories, one in molecular biology and one in plant pathology. The results confirm the hypothesis that using a multistrategy system to learn from both theory and data gives better results than using either theory or data alone.

1 INTRODUCTION

The problem of revising an imperfect 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.

We have developed a multistrategy approach to revising an arbitrarily incorrect propositional Horn-clause domain theory to fit a set of preclassified training instances. The system we have developed, EITHER (Explanation-Based and Inductive THeory Extension and Revision), is modular and contains independent subsystems for deduction, abduction, and induction. Each of these reasoning components makes an important contribution to the overall goal of the system. EITHER can also be viewed as integrating knowledge-intensive (deductive and abductive) and data-intensive (inductive) learning methods.

The remainder of this paper is organized as follows. Section 2 presents an overview of the EITHER system and the problem it is designed to solve. Sections 3-6 discuss each of EITHER's reasoning strategies (deduction, abduction, and induction) as well as the minimal covering algorithms that coordinate the interaction of these components. Section 7 presents empirical results on two real knowledge bases that EITHER has refined. Section 8 discusses how EITHER compares to related work and section 9 presents some conclusions and directions for future work.

2 EITHER OVERVIEW

The EITHER system combines deduction, abduction, and induction to provide a focused correction to an incorrect theory. The deductive and abductive parts of the system identify the failing parts of the theory, and constrain the examples used for induction. The inductive part of the system determines the specific corrections to failing rules that render them consistent with the supplied examples.

2.1 Problem definition

Stated succinctly, the purpose of EITHER is:

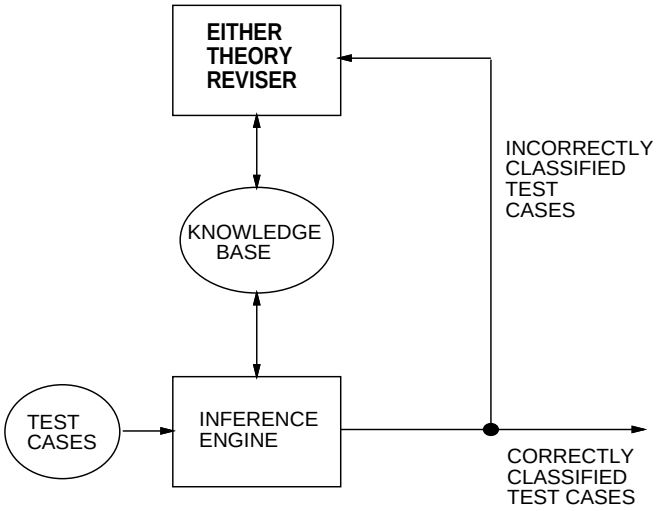

Figure 1: EITHER Components

Given: An imperfect domain theory for a set of categories and a set of classified examples each described by a set of observable features.

Find: A minimally revised version of the domain theory that correctly classifies all of the examples.

Figure 1 shows the architecture for the EITHER system. So long as the inference engine correctly classifies test cases, no additional processing is required. In the event that a misclassified example is detected, EITHER is used to correct the error.

Horn-clause logic (if-then rules) was chosen as the formalism for the EITHER system. This provides a relatively simple and useful language for exploring the problems associated with theory revision. Theories currently are restricted to an extended propositional logic that allows numerical and multi-valued features as well as binary attributes. In addition, domain theories are required to be acyclic and therefore a theory defines a directed acyclic graph (DAG). For the purpose of theory refinement, EITHER makes a closed-world assumption. If the theory can not prove that an example is a member of a category, then it is assumed to be a negative example of that category. Due to the restrictions of propositional logic, EITHER is primarily useful for classification – assigning examples to one of a finite set of predefined categories.

1. cup ← stable ∧ liftable ∧ open-vessel
2. stable ← has-bottom ∧ flat-bottom
3. liftable ← graspable ∧ lightweight
4. graspable ← has-handle
5. graspable ← width=small ∧ styrofoam
6. graspable ← width=small ∧ ceramic
7. open-vessel ← has-concavity ∧ upward-pointing-concavity

Figure 2: The Cup Theory

Propositions that are used to describe examples (e.g. color=black) are called *observables*. To avoid problems with negation as failure, only observables can appear as negated antecedents in rules. Propositions that represent the final concepts in which examples are to be classified are called *categories*. It is currently assumed that the categories are disjoint. In a typical domain theory, all of the sources (leaves) of the DAG are observables and all of the sinks (roots) are categories. Propositions in the theory that are neither observables nor categories are called *intermediate concepts*.

It is difficult to precisely define the adjective "minimal" used to characterize the revision to be produced. Since it is assumed that the original theory is "approximately correct" the goal is to change it as little as possible. Syntactic measures such as the total number of symbols added or deleted are reasonable criteria. EITHER uses various methods to help insure that its revisions are minimal in this sense. However, finding a revision that is guaranteed to be syntactically minimal is clearly computationally intractable. When the initial theory is empty, the problem reduces to that of finding a minimal theory for a set of examples.

A sample theory suitable for EITHER is a version of the cup theory [Winston *et al.*, 1983] shown in Figure 2. Figure 3 shows six examples that are consistent with this theory, three positive examples of cup and three negative examples. Each example is described in terms of twelve observable features. There are eight binary features: has-concavity, upward-pointing-concavity, has-bottom, flat-bottom, lightweight, has-handle, styrofoam and ceramic; three multi-valued features: color, width, and shape; and a single real-valued feature: volume. Given various imperfect versions of the cup theory and these six examples, EITHER can regenerate the correct theory. For ex-

	has-concavity	upward-pointing	has-bottom	flat-bottom	lightweight	has-handle	styrofoam	ceramic	color	width	volume	shape	
1. +	X	X	X	X	X		X		red	sm	8	hem	
2. +	X	X	X	X	X	X		X	blue	med	16	hem	
3. +	X	X	X	X	X	X		X	tan	med	8	cyl	
4. -	X	X	X	X	X				gray	sm	8	cyl	
5. -	X	X	X	X	X		X		red	med	8	hem	
6. -	X	X	X	X	X			X	blue	med	16	hem	

Figure 3: Cup Examples

ample, if rule 4 is missing from the theory, examples 2 and 3 are no longer provable as cups. If the antecedent width=small is missing from rule 5, then negative example 5 becomes provable as a cup. EITHER can correct either or both of these errors using the examples in Figure 3.

The need for EITHER processing is signaled by incorrectly classified examples, as shown in the Figure 1. In making corrections, EITHER operates in batch mode, using as input a set of training examples. The incorrectly classified examples, or *failing* examples, are used to identify that there is an error and to control the correction. The correctly classified examples are used to focus the correction and to limit the extent of the correction. An important property of the EITHER algorithm is that it is guaranteed to produce a revised theory that is consistent with the training examples when there is no noise present in the data. That is, the following statements will be true for every example:

$$T \cup E \models C_E, \tag{1}$$

$$\forall C_i \, (C_i \neq C_E \Rightarrow \quad T \cup E \not\models C_i) \tag{2}$$

where T represents the corrected theory, E represents the conjunction of facts describing any example in the training set, C_E is the correct category

of the example, and C_i is any arbitrary category. A proof of this consistency property is given in [Ourston, 1991].

2.2 Types of theory errors

Figure 4 shows a taxonomy for incorrect propositional Horn-clause theories. At the top level, theories can be incorrect because they are either overly-general or overly-specific. An overly-general theory entails category

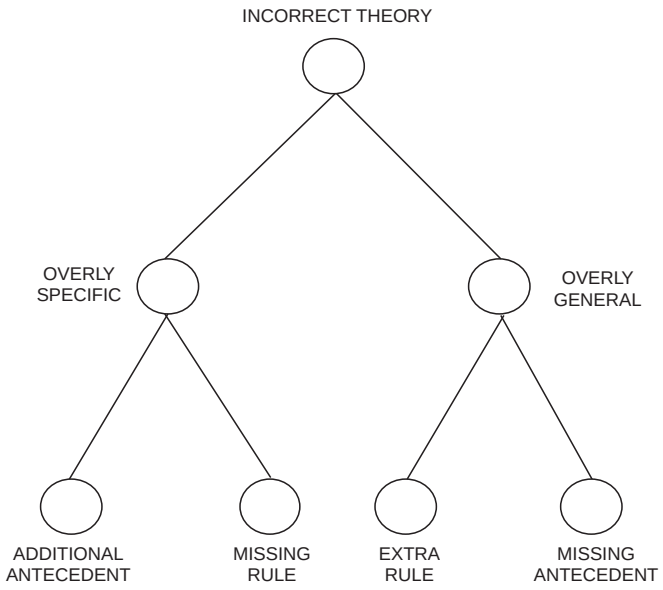

Figure 4: Theory Error Taxonomy

membership for examples that are not members of the category. One way a theory can be overly-general is by having rules that lack required antecedents, providing proofs for examples that should have been excluded. Another way a theory can be overly-general is by having completely incorrect rules. By contrast, an overly-specific theory fails to entail category membership for members of a category. This can occur because the theory is missing a rule, or because an existing rule has additional antecedents that exclude category members. Note that these definitions allow a theory to be both overly-general *and* overly-specific.

The following terminology is used in the remainder of this chapter. "The example is provable," is used to mean "the example is provable as a member of its own category." A *failing positive* refers to an example that is not provable as a member of its own category. A *failing negative* refers to an example that is provable as a member of a category other than its own. Notice that a single example can be both a failing negative and a failing positive.

2.3 EITHER components

As shown in Figure 5, EITHER uses a combination of methods to revise a theory to make it consistent with the examples. It first attempts to fix failing positives by removing antecedents and to fix failing negatives by removing rules since these are simpler and less powerful operations. Only if these operations fail does the system resort to the more powerful technique of using induction to learn rules to fix failing positives and to add antecedents to existing rules to fix failing negatives.

Horn-clause deduction is the basic inference engine used to classify examples. EITHER initially uses deduction to identify failing positives and negatives among the training examples. It uses the proofs generated by deduction to find a near-minimal set of rule retractions that would correct all of the failing negatives. During the course of the correction, deduction is also used to assess proposed changes to the theory as part of the generalization and specialization processes.

EITHER uses abduction to initially find the incorrect part of an overly-specific theory. Abduction identifies sets of assumptions that allow a failing positive to become provable. These assumptions identify rule antecedents that, if deleted, would properly generalize the theory and correct the failing positive. EITHER uses the output of abduction to find a near-minimal set of antecedent retractions that would correct all of the failing positives.

Induction is used to learn new rules or to determine which additional antecedents to add to an existing rule. In both cases, EITHER uses the output of abduction and deduction to determine an appropriately labelled subset of the training examples to pass to induction in order to form a consistent correction. EITHER currently uses a version of ID3 [Quinlan, 1986] as its inductive component. The decision trees returned by ID3 are translated into equivalent Horn-clause rules [Quinlan, 1987]. The remaining components

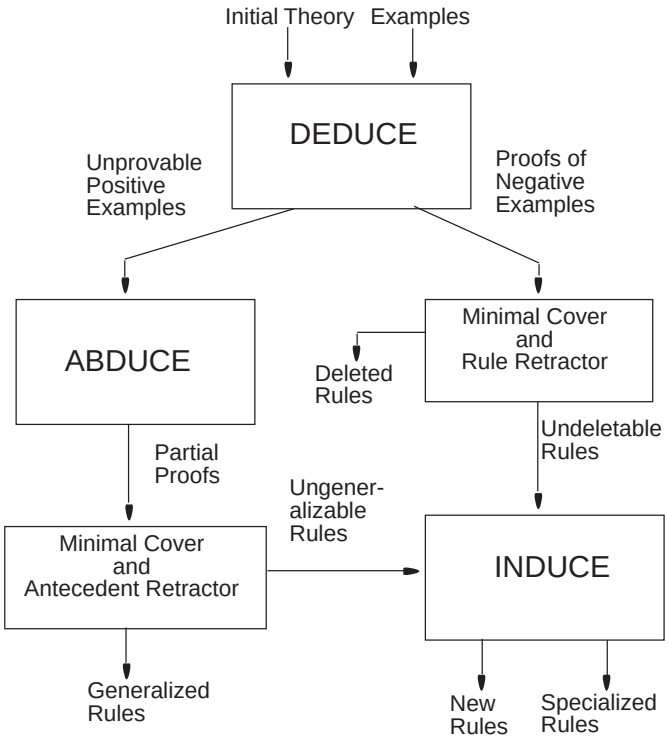

Figure 5: EITHER Architecture

of the EITHER system constitute generalization and specialization control algorithms that identify the types of corrections to be made to the theory.

One of the main advantages of EITHER's architecture is its modularity. Because the control and processing components are separated from the deductive, inductive, and abductive components, these latter components can be modified or replaced as better algorithms for these reasoning methods are developed.

The following sections describe each of EITHER's components and their interactions in details. The discussion focuses on the basic multistrategy approach employed in EITHER. Recent enhancements to the system are discussed in [Ourston and Mooney, 1991; Mooney and Ourston, 1991a; Mooney and Ourston, 1991b] and a complete description is given in [Ourston, 1991].

3 THE DEDUCTIVE COMPONENT

The deductive component of EITHER is a standard backward-chaining, Horn-clause theorem prover similar to Prolog. Our particular implementation is based on the deductive retrieval system from [Charniak *et al.*, 1987]. Deduction is the first step in theory revision. The system attempts to prove that each example is a member of each of the known categories. Failing positives (examples that cannot be proven as members of the correct category) indicate overly-specific aspects of the theory and are passed on to the abductive component. Failing negatives (examples that are proven as members of incorrect categories) indicate overly-general aspects of the theory and are passed on to the specialization procedure. In order to satisfy the requirements of specialization, the deductive component finds all possible proofs of each incorrect category and returns all of the resulting proof trees.

The deductive component also forms the basic performance system and is used during testing to classify novel examples. If during testing an example is provable as a member of multiple categories, the system picks the provable category that is most common in the training set. If an example fails to be provable as a member of any category, it is assigned to the most common category overall. This assures that the system always assigns a test example to a unique category.

The deductive component is also used to check for over generalization and over specialization when certain changes to the theory are proposed. For example, when an antecedent retraction is proposed, all of the examples are reproven with the resulting theory to determine whether any additional failing negatives are created. Analogously, when a rule retraction is proposed, all of the examples are reproven to determine whether any additional failing positives are created. If so, then the proposed revision is not made and the system resorts to learning new rules or adding antecedents. Better bookkeeping methods, such as truth-maintenance techniques, could potentially be used to avoid unnecessary reproving of examples. For example, existing proofs of all examples could be used to more directly determine the effect of a rule deletion and partial proofs of negative examples could be used to more directly determine the effect of antecedent deletions. However such methods would be fairly complicated and incur a potentially large overhead.

4 THE ABDUCTIVE COMPONENT

If an example cannot be proven as a member of its category, then abduction is used to find minimal sets of assumptions that would allow the example to become provable. The normal logical definition of abduction is:

Given: A domain theory, T, and an observed fact O.

Find: All minimal sets of atoms, A, called *assumptions*, such that $A \cup T$ is logically consistent and $A \cup T \models O$.

The assumptions A are said to *explain* the observation. Legal assumptions are frequently restricted, such as allowing only instances of certain predicates (*predicate specific abduction*) or requiring that assumptions not be provable from more basic assumptions (*most-specific abduction*) [Stickel, 1988].

In EITHER, an observation states that an example is a member of a category (in the notation introduced earlier, $E \rightarrow C_E$). In addition, EITHER's abductive component backchains as far as possible before making an assumption (most-specific abduction), and the consistency constraint is removed. As a result, for each failing positive, abduction finds all minimal sets of most-specific atoms, A, such that:

$$A \cup E \cup T \models C_E \tag{3}$$

where minimal means that no assumption set is a subset of another. The proof supported by each such set is called a *partial proof*. EITHER currently uses an abductive component that employs exhaustive search to find all partial proofs of each failing positive example [Ng and Mooney, 1989]. Partial proofs are used to indicate antecedents that, if retracted, would allow the example to become provable. The above definition guarantees that if all of the assumptions in a set are removed from the antecedents of the rules in their corresponding partial proof, the example will become provable. This is because not requiring a fact for a proof has the same generalizing effect as assuming it.

As a concrete example, assume that rule 4 about handles is missing from the cup theory. This will cause example 2 from Figure 3 to become a failing positive. Abduction finds two minimal sets of assumptions: {width=small$_6$} and {width=small$_5$, styrofoam$_5$}. The subscripts indicate the number of the rule to which the antecedent belongs, since each antecedent of each rule must

be treated distinctly. Notice that removing the consistency constraint is critical to the interpretation of assumptions as antecedent retractions. Assuming width=small is inconsistent when width=medium is known; however, retracting width=small as an antecedent from one of the graspable rules is still a legitimate way to help make this example provable.

5 THE MINIMUM COVER COMPONENTS

In EITHER, a cover is a complete set of rules requiring correction. There are two types of covers: the *antecedent cover* and the *rule cover*. The antecedent cover is used by the generalization procedure to fix all failing positives. The rule cover is used by the specialization procedure to fix all failing negatives. There is an essential property that holds for both types of cover: If all of the elements of the cover are removed from the theory, the examples associated with the cover will be correctly classified. Specifically, if all of the antecedents in the antecedent cover are removed, the theory is generalized so that all of the failing positives are fixed and if all of the rules in the rule cover are removed, the theory is specialized so that all of the failing negatives are fixed. In each case, EITHER attempts to find a *minimum* cover in order to minimize change to the initial theory. The details of the minimum cover algorithms are given in the next two subsections. EITHER initially constructs covers containing only rules at the "bottom" or "leaves" of the theory; however, if higher-level changes are necessary or syntactically simpler, non-leaf rules can also be included [Ourston and Mooney, 1991; Ourston, 1991].

5.1 The minimum antecedent cover

The partial proofs of failing positives generated by the abductive component are used to determine the minimum antecedent cover. In a complex problem, there will be many partial proofs for each failing positive. In order to minimize change to the initial theory, EITHER attempts to find the minimum number of antecedent retractions required to fix *all* of the failing positives. In other words, we want to make the following expression true:

$$E_1 \wedge E_2 \wedge ... \wedge E_n \tag{4}$$

where E_i represents the statement that the ith failing positive has at least one completed partial proof, that is,

$$E_i \equiv P_{i1} \vee P_{i2} \vee ... \vee P_{im} \tag{5}$$

where P_{ij} represent the statement that the jth partial proof of the ith failing positive is completed, that is,

$$P_{ij} \equiv A_{ij1} \wedge A_{ij2} ... \wedge A_{ijp} \tag{6}$$

where the A_{ijk} means that the antecedent represented by the kth assumption used in the jth partial proof of the ith example is removed from the theory. In order to determine a minimum change to the theory, we need to find the minimum set of antecedent retractions (A's) that satisfy this expression. Pursuing the example of the cup theory that is missing the rule for handles, both failing positives (examples 2 and 3) have the same partial proofs, resulting in the expressions:

$$E_2 \equiv \mathsf{width} = \mathsf{small}_6 \vee (\mathsf{width} = \mathsf{small}_5 \wedge \mathsf{styrofoam}_5)$$
$$E_3 \equiv \mathsf{width} = \mathsf{small}_6 \vee (\mathsf{width} = \mathsf{small}_5 \wedge \mathsf{styrofoam}_5)$$

In this case, the minimum antecedent cover is trivial and consists of retracting the single antecedent $\mathsf{width}{=}\mathsf{small}_6$.

Since the general minimum set covering problem is NP-Hard [Garey and Johnson, 1979], EITHER uses a version of the *greedy covering algorithm* to find the antecedent cover. The greedy algorithm does not guarantee to find the minimum cover, but will come within a logarithmic factor of it and runs in polynomial time [Johnson, 1974]. The algorithm iteratively updates a partial cover, as follows. At each iteration, the algorithm chooses a partial proof and adds the antecedent retractions associated with the proof to the evolving cover. The chosen partial proof is the one that maximizes *benefit-to-cost*, defined as the ratio of the additional examples covered when its antecedents are included, divided by the number of antecedents added. The set of examples that have the selected partial proof as one of their partial proofs are removed from the examples remaining to be covered. The process terminates when all examples are covered. The result is a near-minimum set of antecedent retractions that fix all of the failing positives.

Once the antecedent cover is formed, EITHER attempts to retract the indicated antecedents of each rule in the cover. If a given retraction is not an

over generalization (i.e. it does not result in any additional failing negatives as determined by the deductive component), then it is chosen as part of the desired revision. If a particular retraction does result in additional failing negatives, then the inductive component is instead used to learn a new rule (see section 6.1).

5.2 The minimum rule cover

The proofs of failing negatives generated by the deductive component are used to determine the minimum rule cover. In order to minimize change to the initial theory, EITHER attempts to find the minimum number of leaf-rule retractions required to fix *all* of the failing negatives. In analogy with the previous section, we would like to make the following expression true:

$$\neg E_1 \wedge \neg E_2 \wedge ... \wedge \neg E_n \tag{7}$$

where E_i represents the statement that the ith failing negative has a complete proof, that is,

$$\neg E_i \equiv \neg P_{i1} \wedge \neg P_{i2} \wedge ... \wedge \neg P_{im} \tag{8}$$

where P_{ij} represent the statement that the jth proof of the ith failing negative is complete, that is,

$$\neg P_{ij} \equiv \neg R_{ij1} \vee \neg R_{ij2} ... \vee \neg R_{ijp} \tag{9}$$

where $\neg R_{ijk}$ represents the statement that the kth leaf rule used in the jth proof of the ith failing negative is removed, i.e. a proof is no longer complete if at least one of the rules used in the proof is removed.

As with assumptions, EITHER attempts to find a minimum cover of rule retractions using greedy covering. If this case, the goal is to remove all proofs of all of the failing negatives. Note that in computing retractions, EITHER removes from consideration those rules that do not have any disjuncts in their proof path to the goal since these rules are needed to prove *any* example. At each step in the covering algorithm, the eligible rule that participates in the most faulty proofs is added to the evolving cover until all the faulty proofs are covered.

As an example, consider the cup theory in which the width=small antecedent is missing from rule 5. In this case, example 5 becomes a failing

negative. The minimum rule cover is the overly-general version of rule 5:

graspable ← styrofoam

since it is the only rule used in the faulty proof with alternative disjuncts (rules 4 and 6).

Once the rule cover is formed, EITHER attempts to retract each rule in the cover. If a given retraction is not an over specialization (i.e. it does not result in any additional failing positives as determined by the deductive component), then it is chosen as part of the desired revision. If a particular retraction does result in additional failing negatives, then the inductive component is instead used to specialize the rule by adding additional antecedents (see section 6.2).

6 THE INDUCTIVE COMPONENT

If retracting an element of the antecedent cover causes new failing negatives or if retracting an element of the rule cover causes new failing positives, then the inductive component is used to learn new rules or new antecedents, respectively. The only assumption EITHER makes about the inductive component is that it solves the following problem:

Given: A set of positive and negative examples of a concept C described by a set of observable features.

Find: A Horn-clause theory, T that is consistent with the examples, i.e. for each positive example description, P, $P \cup T \models C$ and for each negative example description, N, $N \cup T \not\models C$.

As mentioned previously, EITHER currently uses ID3 as an inductive component by translating its decision trees into a set of rules; however, any inductive rule-learning system could be used. An inductive system that directly produces a multi-layer Horn-clause theory would be preferable but current robust inductive algorithms produce decision trees or DNF formulas. In EITHER, *inverse resolution* operators [Muggleton, 1987; Muggleton and Buntine, 1988] are used to introduce new intermediate concepts and produce a multi-layer theory from a translated decision tree [Mooney and Ourston, 1991a].

6.1 Rule addition

If antecedent retraction ever over-generalizes, then induction is used to learn a new set of rules for the consequent (C) of the corresponding rule. The rules are learned so they cover the failing positives associated with the antecedent retraction without introducing any new failing negatives. The positive examples of C are those that have a partial proof completed by the given antecedent retraction (i.e. the failing positives covered by the given assumptions). The negative examples of C are examples that become failing negatives when C is assumed to be true (i.e. this is a proof by contradiction that they are $\neg C$ since a contradiction is derived when C is assumed).

Again consider the example of missing rule 4 from the cup theory. Based on the antecedent cover, EITHER first attempts to remove width=small from rule 6; however, this results in example 6 becoming a failing negative. Therefore, induction is used to form a new rule for graspable. The positive examples are the original failing positives, examples 2 and 3. The negative examples are examples 4, 5 and 6, which become provable when graspable is assumed to be true. Since has-handle is the only single feature that distinguishes examples 2 and 3 from examples 4, 5 and 6, the inductive system (ID3) generates the required rule:

graspable $\leftarrow$ has-handle

If an element of the antecedent cover is not an observable, then a rule is learned directly for it instead for the consequent of the rule in which it appears. For example, if both rules for graspable are missing from the cup theory, then most-specific abduction returns the single assumption set: {graspable} for all of the failing positives (examples 1, 2 and 3). Since removing the graspable antecedent results in all of the negative examples becoming failing negatives, EITHER decides to learn rules for graspable. All of the positive examples are used as positive examples of graspable and all of the negatives are used as negative examples of graspable and the system learns the approximately correct rules:

graspable $\leftarrow$ has-handle
graspable $\leftarrow$ width=small $\wedge$ styrofoam

6.2 Antecedent addition

If rule retraction ever over-specializes, then induction is used to learn additional antecedents to add to the rule instead of retracting it. Antecedents are learned so they fix the failing negatives associated with the rule retraction without introducing any new failing positives. The positive examples of C are those examples that become unprovable (failing positives) when the rule is retracted. The negative examples of C are the failing negatives covered by the rule.

For example, again consider the case of missing the antecedent width=small from rule 5. Based on the rule cover, EITHER first removes the overly-general rule 5:

graspable ← styrofoam

and tests for additional failing positives. Since example 1 becomes unprovable in this case, EITHER decides to add additional antecedents. Example 1 (the failing positive created by retraction) is used as a positive example and example 5 (the original failing negative) is used as a negative example. Since width is the only feature that distinguishes these two examples, ID3 learns the rule:

positive ← width=small.

This is combined with the original rule to obtain the correct replacement rule:

graspable ← width=small ∧ styrofoam.

7 EMPIRICAL RESULTS

EITHER has revised two real expert-provided rule bases, one in molecular biology and one in plant pathology. This section presents details on our results in these domains. Further information on these tests, including the actual initial and revised theories, is given in [Ourston, 1991].

7.1 Single category: DNA results

EITHER was first tested on a theory for recognizing biological concepts in DNA sequences. The original theory is described in [Towell *et al.*, 1990], it contains 11 rules with a total of 76 propositional symbols. The purpose of

the theory is to recognize *promoters* in strings of nucleotides (one of A, G, T, or C). 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. The examples used in the tests consisted of 53 positive and 53 negative examples assembled from the biological literature. The initial theory classified none of the positive examples and all of the negative examples correctly, thus indicating that the initial theory was entirely overly specific.

Figure 6 presents learning curves for this domain. In each test, classifi-

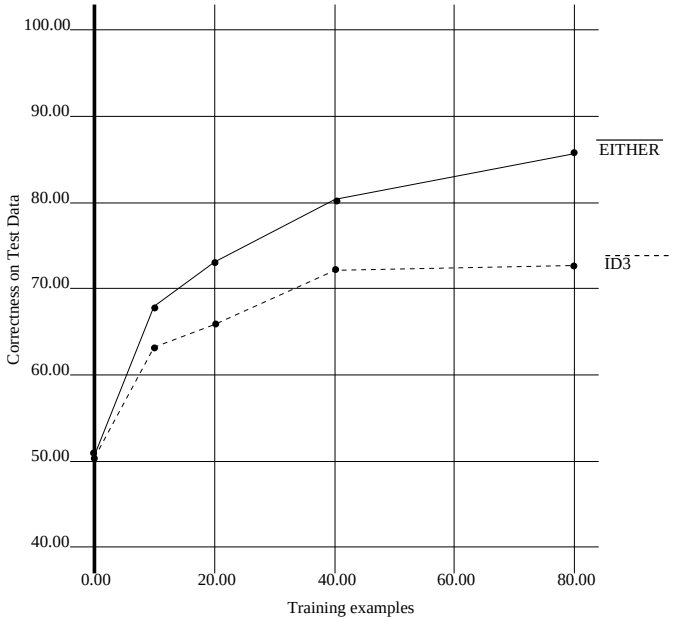

Figure 6: Results for the DNA Theory

cation accuracy was measured on 26 disjoint test examples. The number of training examples was varied from one to eighty, with the training and test examples selected at random. The results were averaged over 21 training/test divisions. ID3's performance is also shown in order to contrast theory refinement with pure induction.

The accuracy of the initial promoter theory is shown in the graph as EITHER's performance with 0 training examples – it is no better than random chance (50%). With no examples, ID3 picks a category at random and

exhibits the same accuracy. However, as the number of training examples increases, EITHER use of the existing theory results in a significant performance advantage compared to pure induction. A one-tailed Student t-test on paired differences showed that the EITHER's superior accuracy compared to ID3 is statistically significant ($p<.05$) for every non-zero point plotted on the learning curves. Overall, EITHER improves the accuracy of the theory by 35 percentage points. An additional reason for including ID3 in the performance graphs is that it represents EITHER's performance without an initial theory, since in this case every example is a failing positive and induction is used to learn a set of rules from scratch. Therefore including ID3's learning curve, provides a clear illustration of the advantage provided by theory-based learning. In fact, if a different inductive system were substituted for ID3, the absolute performance of both learning systems might change, but the relative advantage of EITHER compared to the purely inductive system should remain approximately the same.

Another way of looking at the performance advantage provided by an initial theory is to consider the additional examples required by ID3 in order to achieve equal performance with EITHER. For example, at 75% accuracy, ID3 requires over sixty additional training examples to achieve equal performance with EITHER. Therefore, in some sense, the information contained in the theory is equivalent to 60 examples.

The initial theory is primarily revised by deleting antecedents in various ways. In general, EITHER's changes made sense to the expert. In particular, the subconcept *conformation* was removed from the *promoter* concept in its entirety. This correction was validated by the biologist who encoded the theory (Noordewier), who indicated that conformation was a weakly-justified constraint when it was originally introduced.

This domain theory was also used to test the KBANN system [Towell *et al.*, 1990], which translates the initial theory into an equivalent neural net, and then applies the backpropagation algorithm [Rumelhart *et al.*, 1986] to revise the network. KBANN's accuracy is somewhat better than EITHER's in this domain (a test set accuracy of 92% with 105 training examples). The likely explanation for the performance advantage is that the DNA task involves learning a concept of the form *N out of these M features must be present.* Experiments comparing backpropagation and ID3 report that backpropagation is better at learning *N out of M* functions [Fisher and McKusick, 1989]. Some aspects of the promoter concept fit the *N out of M* format where, for

example, there are several potential sites where hydrogen bonds can form between the DNA and the protein; if enough of these bonds form, promoter activity can occur. EITHER attempts to learn this concept by learning a separate rule for each potential configuration by deleting different combinations of antecedents from the initial rules, which makes this a comparatively difficult learning task for a rule-based system. Finally, it should be noted that when KBANN translated its results into Horn clauses, the resulting theory was significantly more complicated than EITHER's [Towell and Shavlik, 1991]. This is because EITHER's goal is to produce a minimally revised Horn-clause theory and KBANN has no such bias.

7.2 Multiple categories: Soybean results

In order to demonstrate EITHER's ability to revise multiple category theories, EITHER was used to refine the expert rules given in [Michalski and Chilausky, 1980]. This is a theory for diagnosing soybean diseases that distinguishes between nineteen possible soybean diseases using examples that are described with thirty five features. The original experiments compared expert rules to induction from examples. By revising the expert rules to fit the examples, we hoped to show that one could produce better results than using just the examples or just the rules.

The original expert rules associated probabilistic weights with certain disease symptoms. In addition, some groups of disease symptoms were regarded as *significant* while other groups were regarded as *confirmatory*. The rules were translated to propositional Horn-clause format by only including the significant symptoms and by deleting any symptom from the theory that had a weight less than 0.8. After translation, the theory contained 73 rules with 325 propositional symbols.

Unfortunately, the classification performance of the Horn-clause version was seriously deficient compared to the original probabilistic rules. For example, the Horn-clause theory obtained a 12.3% classification performance compared to the accuracy of 73% reported in the original paper. To circumvent the problem, a "flexible" matcher was used to classify the test examples, based on the updated theory provided by EITHER. The flexible matcher accounts for two possible classification problems with a multi-category theory. The first problem occurs when a test example is provable as a member of more than one category. The second problem occurs when a test example is

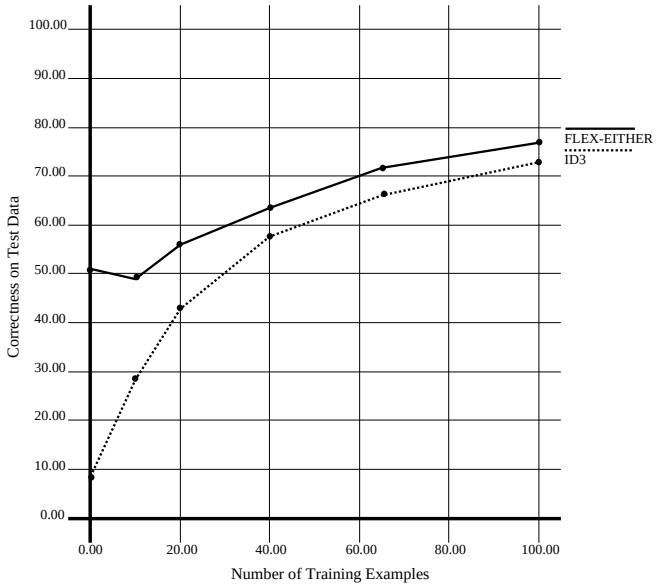

Figure 7: Results for the Soybean Theory

not provable as a member of any category. With the standard EITHER tester, such examples are assigned to the most common category. In contrast, the original soybean tests assigned a match score to each possible category and chose the category with the highest score. The flexible matcher used by EITHER is a simple approximation to the original technique. If an example is assigned to multiple categories, the tester selects the category that makes the most use of the example's features. This is done by choosing the category whose proof of category membership contains the greatest number of features. If an example is assigned to no category, the flexible matcher chooses the category that comes closest to being provable. This is done by choosing the category with a partial proof of category membership that has the least number of assumptions.

Learning curves for the soybean experiments are shown in Figure 7. In each test, accuracy was measured against 75 disjoint test examples. The number of training examples was varied from one to one hundred, with the training and test examples drawn at random from the entire example population. The results were averaged over 22 train/test divisions. Note that even with the flexible matcher, the accuracy of the original rules was only

51%, as compared to 73% for the original results presented in [Michalski and Chilausky, 1980]. Overall, the accuracy of the initial rules is increased by 26 percentage points and EITHER maintains its initial performance advantage compared to pure induction over the entire training interval. A one-tailed Student t-test on paired differences showed that the superior performance of EITHER is statistically significant ($p<.05$) for every point plotted on the learning curves. Therefore, employing both rules and examples is better than using either one alone.

Since comparing a system with flexible matching (FLEX-EITHER) to one without it (ID3) is somewhat inequitable, we also tested a version of ID3 that uses the same flexible matcher as FLEX-EITHER. However, there was no significant difference between the performance of ID3 and FLEX-ID3 in this domain. Since ID3 has a bias for very simple, general descriptions, the case in which an example is not initially categorized into any category (and therefore requires partial matching) rarely occurred.

8 RELATED WORK

As discussed in [Langley, 1989], any incremental inductive learning system can be viewed as continually refining a knowledge base formed from previous examples. However, existing incremental inductive systems [Reinke and Michalski, 1988; Utgoff, 1989] employ "flat" representations such as decision trees and DNF expressions, as opposed to multi-layer theories with intermediate concepts. In addition, attempts to initialize an incremental inductive system with a user-supplied domain theory have met with mixed results [Mahoney and Mooney, 1991; Thompson *et al.*, 1991].

Most systems for integrating explanation-based and similarity-based methods cannot refine arbitrarily imperfect theories. Some systems are only capable of generalizing an overly-specific (incomplete) theory [Wilkins, 1988; Danyluk, 1989; Whitehall, 1990] while others are only capable of specializing an overly-general theory [Flann and Dietterich, 1989; Mooney and Ourston, 1989; Cohen, 1990]. Many systems do not revise the theory itself but instead revise the *operational definition* of a concept [Bergadano and Giordana, 1988; Hirsh, 1990; Pazzani *et al.*, 1991]. Still other systems rely on active experimentation rather than a provided training set to detect and correct errors [Rajamoney, 1990].

Recent experiments with ANAPRON [Golding and Rosenbloom, 1991], a

system that integrates case-based and rule-based reasoning, also demonstrate that combining initial rules and data performs better than either individually. However, this system stores specific cases as exceptions to rules rather than modifying the rules themselves.

KBANN [Towell *et al.*, 1990] and RTLS [Ginsberg, 1990] are theory revision systems that come the closest to handling as many types of imperfections as EITHER. However, neither integrates independent components for deduction, abduction, and induction. Modularity allows EITHER to easily take advantage of improvements in each of these areas of automated reasoning. Also, neither KBANN nor RTLS guarantee consistency with all of the training data when given an arbitrary initial theory.

The DUCTOR [Cain, 1991] is a recent EITHER-inspired system that integrates deduction, abduction, and induction. However, it does not generate all proofs and partial proofs nor attempt to find a minimum cover of theory changes. Consequently, it is less focused on finding a *minimal* revision to the initial theory.

9 FUTURE RESEARCH

EITHER suffers from a number of shortcomings that we hope to address in future research. First, although it has been efficient enough to apply to several real theories, the current implementation can be quite slow at revising large rule bases (its takes 90 minutes on a Texas Instruments Explorer II to fit the soybean theory to 100 examples). Computation of all the partial proofs of failing positives is the primary bottleneck in the current system. By incorporating improved deductive and abductive components, we hope to dramatically improve system efficiency. Specifically, we plan to employ an abduction system that uses heuristic search to compute only a subset of the partial proofs with the fewest assumptions [Ng and Mooney, 1991]. Another approach to improving efficiency is to only partially fit the theory to the training data. Existing experiments with a version of EITHER that computes only partial covers of the failing positive and failing negative examples have demonstrated that this technique can significantly increase efficiency without significantly affecting accuracy [Mooney and Ourston, 1991b]. This method was originally developed to deal with noisy data.

Second, the current system cannot handle theories that employ *negation as failure*. Antecedents of the form not(P) complicate the revision process

since generalizing or learning a rule that concludes P actually specializes the overall theory by preventing this antecedent from being satisfied. Conversely, specializing or eliminating a rule for P may actually generalize the overall theory. Therefore, the system will have to consider standard generalization operators as specializers in certain contexts and vice versa.

Third, the current system assumes all examples are instances of exactly one of the top-level categories. It cannot directly accept examples of intermediate concepts nor deal with overlapping categories. A truly robust theory revision system should be able to accept examples of any of its concepts and use them to revise the rules for that concept directly or to revise other concepts indirectly.

Fourth, EITHER is basically restricted to revising propositional theories. We have already developed a prototype of a successor system called FORTE which is capable of revising first-order Horn-clause theories [Richards and Mooney, 1991]. This system is undergoing continued development to improve its efficiency and capabilities.

Finally, EITHER is restricted to revising purely logical theories and many rule bases employ some form of probabilistic reasoning. Some previous work has addressed the problem of refining the probabilities or certainty factors attached to rules [Ling and Valtorta, 1991; Ginsberg *et al.*, 1988]; however, such numerical adjustments have not been integrated with more symbolic revisions such as learning new rules. We are currently developing a system that first "tweaks" certainty factors until no more improvement is possible and then resorts to learning new rules. The system cycles between "tweaking" and rule learning until it converges to 100% accuracy on the training data.

10 CONCLUSIONS

The development and testing of EITHER has demonstrated how deduction, abduction, and induction can be successfully integrated to revise imperfect domain theories. By combining such diverse methods, EITHER is able to handle a wider range of imperfect theories than other systems while guaranteeing that the revised theory will be consistent with the training data. Results on revising two actual expert theories has demonstrated EITHER's abilities and generality and confirmed the conjecture that learning from both theory and data gives better results than using either one alone.

Acknowledgements

We would like to thank Mick Noordewier and Jude Shavlik for providing the DNA theory and data and helping us interpret the results; Jeff Mahoney for translating the soybean theory and data and implementing the flexible matcher; and Hwee Tou Ng for providing the abduction component. This research was supported by the NASA Ames Research Center under grant NCC 2-629 and by the National Science Foundation under grant IRI-9102926. Equipment was donated by the Texas Instruments Corporation.

References

[Bergadano and Giordana, 1988] F. Bergadano and A. Giordana. A knowledge intensive approach to concept induction. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 305–317, Ann Arbor, MI, June 1988.

[Cain, 1991] T. Cain. The DUCTOR: A theory revision system for propositional domains. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 485–489, Evanston, IL, June 1991.

[Charniak *et al.*, 1987] E. Charniak, C. Riesbeck, D. McDermott, and J. Meehan. *Artificial Intelligence Programming (2nd Ed)*. Lawrence Erlbaum and Associates, Hillsdale, NJ, 1987.

[Cohen, 1990] William W. Cohen. Learning from textbook knowledge: A case study. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 743–748, Boston, MA, July 1990.

[Danyluk, 1989] A. P. Danyluk. Finding new rules for incomplete theories: Explicit biases for induction with contextual information. In *Proceedings of the Sixth International Workshop on Machine Learning*, pages 34–36, Ithaca, NY, June 1989.

[Fisher and McKusick, 1989] D. H. Fisher and K. B. McKusick. An empirical comparison of ID3 and backpropagation. In *Proceedings of the Eleventh International Joint conference on Artificial intelligence*, pages 788–793, Detroit, MI, Aug 1989.

[Flann and Dietterich, 1989] N. S. Flann and T. G. Dietterich. A study of explanation-based methods for inductive learning. *Machine Learning*, 4(2):187–226, 1989.

[Garey and Johnson, 1979] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, NY, 1979.

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

[Ginsberg, 1990] A. Ginsberg. Theory reduction, theory revision, and retranslation. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 777–782, Detroit, MI, July 1990.

[Golding and Rosenbloom, 1991] A. R. Golding and P. S. Rosenbloom. Improving rule-based systems through case-based reasoning. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 22–27, Anaheim, CA, July 1991.

[Hirsh, 1990] H. Hirsh. *Incremental Version-Space Merging: A General Framework for Concept Learning*. Kluwer Academic Publishers, Hingham, MA, 1990.

[Johnson, 1974] D. S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

[Langley, 1989] P. Langley. Unifying themes in empirical and explanation-based learning. In *Proceedings of the Sixth International Workshop on Machine Learning*, pages 2–4, Ithaca, NY, June 1989.

[Ling and Valtorta, 1991] X. Ling and M. Valtorta. Revision of reduced theories. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 519–523, Evanston, IL, June 1991.

[Mahoney and Mooney, 1991] J. J. Mahoney and R. J. Mooney. Initializing ID5R with a domain theory: Some negative results. Technical Report AI91-154, Artificial Intelligence Laboratory, University of Texas, Austin, TX, March 1991.

[Michalski and Chilausky, 1980] R. S. Michalski and S. Chilausky. 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, 1980.

[Mooney and Ourston, 1989] R. J. Mooney and D. Ourston. Induction over the unexplained: Integrated learning of concepts with both explainable and conventional aspects. In *Proceedings of the Sixth International Workshop on Machine Learning*, pages 5–7, Ithaca, NY, June 1989.

[Mooney and Ourston, 1991a] R. Mooney and D. Ourston. Constructive induction in theory refinement. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 178–182, Evanston, IL, June 1991.

[Mooney and Ourston, 1991b] R. J. Mooney and D. Ourston. Theory refinement with noisy data. Technical Report AI91-153, Artificial Intelligence Laboratory, University of Texas, Austin, TX, March 1991.

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

[Muggleton, 1987] S. Muggleton. Duce, an oracle based approach to constructive induction. In *Proceedings of the Tenth International Joint conference on Artificial intelligence*, pages 287–292, Milan, Italy, Aug 1987.

[Ng and Mooney, 1989] H. T. Ng and R. J. Mooney. Abductive explanations for text understanding: Some problems and solutions. Technical Report AI89-116, Artificial Intelligence Laboratory, University of Texas, Austin, TX, August 1989.

[Ng and Mooney, 1991] H. T. Ng and R. J. Mooney. An efficient first-order Horn-clause abduction system based on the ATMS. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 494–499, Anaheim, CA, July 1991.

[Ourston and Mooney, 1991] D. Ourston and R. Mooney. Improving shared rules in multiple category domain theories. In *Proceedings of the Eighth*

International Workshop on Machine Learning, pages 534–538, Evanston, IL, June 1991.

[Ourston, 1991] D. Ourston. *Using Explanation-Based and Empirical Methods in Theory Revision.* PhD thesis, University of Texas, Austin, TX, August 1991.

[Pazzani *et al.*, 1991] M. Pazzani, C. Brunk, and G. Silverstein. A knowledge-intensive approach to learning relational concepts. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 432–436, Evanston, IL, June 1991.

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

[Quinlan, 1987] J. R. Quinlan. Generating production rules from decision trees. In *Proceedings of the Tenth International Joint conference on Artificial intelligence*, pages 304–307, Milan, Italy, Aug 1987.

[Rajamoney, 1990] S. A. Rajamoney. A computational approach to theory revision. In J. Shrager and P. Langley, editors, *Computational Models of Scientific Discovery and Theory Formation*, pages 225–254. Morgan Kaufman Publishers, San Mateo, CA, 1990.

[Reinke and Michalski, 1988] R. E. Reinke and R. S. Michalski. Incremental learning of concept descriptions. In J. E. Hayes, D. Michie, and J. Richards, editors, *Machine Intelligence (Vol. 11)*. Oxford University Press, Oxford, England, 1988.

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

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

[Stickel, 1988] M. E. Stickel. A Prolog-like inference system for computing minimum-cost abductive explanations in natural-language interpretation.

Technical Report Technical Note 451, SRI International, Menlo Park, CA, September 1988.

[Thompson *et al.*, 1991] K. Thompson, P. Langley, and W. Iba. Using background knowledge in concept formation. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 554–558, Evanston, IL, June 1991.

[Towell and Shavlik, 1991] G. Towell and J. Shavlik. Refining symbolic knowledge using neural networks. In *Proceedings of the International Workshop on Multistrategy Learning*, pages 257–272, Harper's Ferry, W.Va., Nov. 1991.

[Towell *et al.*, 1990] G. G. Towell, J. W. Shavlik, and Michiel O. Noordewier. Refinement of approximate domain theories by knowledge-based artificial neural networks. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 861–866, Boston, MA, July 1990.

[Utgoff, 1989] P. E. Utgoff. Incremental induction of decision trees. *Machine Learning*, 4(2):161–186, 1989.

[Whitehall, 1990] B. L. Whitehall. *Knowledge-Based Learning: An Integration of Deductive and Inductive Learning for Knowledge Base Completion.* PhD thesis, University of Illinois, Urbana, IL, Oct 1990. Also appears as Technical Report UILU-ENG-90-1776.

[Wilkins, 1988] D. C. Wilkins. Knowlege base refinement using apprenticeship learning techniques. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 646–651, St. Paul, MN, August 1988.

[Winston *et al.*, 1983] P. H. Winston, T. O. Binford, B. Katz, and M. Lowry. Learning physical descriptions from functional definitions, examples, and precedents. In *Proceedings of the Third National Conference on Artificial Intelligence*, pages 433–439, Washington, D.C., Aug 1983.