Appears in *Proceedings of the Eighth International Machine Learning Workshop*, pp. 178-182, Evanston, IL, June 1991.

Constructive Induction in Theory Refinement

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

Dirk Ourston
Department of Computer Sciences
University of Texas
Austin, TX 78712
dirk@cs.utexas.edu

Abstract

This paper presents constructive induction techniques recently added to the EITHER theory refinement system. These additions allow EITHER to handle arbitrary gaps at the "top," "middle," and/or "bottom" of an incomplete domain theory. *Intermediate concept utilization* employs existing rules in the theory to derive higher-level features for use in induction. *Intermediate concept creation* employs inverse resolution to introduce new intermediate concepts in order to fill gaps in a theory that span multiple levels. These revisions allow EITHER to make use of imperfect domain theories in the ways typical of previous work in both constructive induction and theory refinement. As a result, EITHER is able to handle a wider range of theory imperfections than does any other existing theory refinement system.

1 Introduction

Constructive induction and theory refinement are both attempts to improve the use of domain knowledge in inductive learning. Typical research in constructive induction uses domain rules to form higher-level features from the observable features in the data [Drastal *et al.*, 1989]. Typical research in theory refinement uses induction to fill a gap between existing intermediate concepts in an incomplete domain theory and the observable features in the data [Danyluk, 1989; Ourston and Mooney, 1990]. Both of these processes are important aspects of employing domain theories in inductive learning.

This paper discusses constructive induction techniques recently added to the EITHER theory refinement system [Ourston and Mooney, 1990]. Constructive induction in EITHER makes existing intermediate concepts in an imperfect domain theory usable as additional features during induction. EITHER also adds new intermediate concepts to an incomplete domain theory using *inverse resolution*. These revisions allow EITHER to use domain theories in ways typical of work both in constructive induction and theory refinement. As a result, EITHER can respond to a wide range of imperfect domain theories. This paper presents an overview of constructive induction in EITHER and a number of examples that illustrate its advantages.

(cup)	←	(stable) (liftable) (open-vessel)
(stable)	←	(has-bottom) (flat-bottom)
(liftable)	←	(graspable) (weight ?w) (< ?w 1)
(graspable)	←	(has-handle)
(graspable)	←	(width small) (insulating)
(open-vessel)	←	(has-concavity) (upward-concavity)

Figure 1: Sample Domain Theory

2 Types of Theory Gaps

In this paper we will restrict ourselves to Horn-clause theories expressed in a propositional logic whose atomic formulae include feature value pairs and numerical thresholds as well as binary propositions. Figure 1 shows a variation of the standard domain theory for the concept *cup* that will be used as an example throughout the paper. Leading question marks denote variables, which are used to define thresholds or intervals for numerically-valued features. Propositions that correspond to facts used to describe the examples (e.g. *has-handle*) are called *observables*. Propositions representing classification goals for which explicit ex-

Figure 2: Types of Theory Gaps

amples (and possibly counter-examples) are available (e.g. *cup*) are called *categories*. Propositions in the theory that are neither observables nor categories (e.g. *liftable*) are called *intermediate concepts*.

Research in constructive induction frequently assumes that the existing domain theory consists of a number of rules that define a set of intermediate concepts (derived features) in terms of observables. The rules connecting these intermediate concepts to the categories are assumed to be missing and must be learned using induction. The theory is used to derive truth values for all of the intermediate concepts and these are used as additional input features for induction of the category rules. This is the first situation illustrated in Figure 2 where there is a gap at the "top" of the theory. An example would be if the rule for concluding *cup* was missing from the theory in Figure 1.

Some research in refinement of incomplete theories with missing rules [Danyluk, 1989] assumes that the domain theory has correct rules for inferring categories from intermediate concepts but is instead missing rules connecting observables to intermediate concepts. Partial explanations (incomplete proofs) are used to isolate intermediate concepts that should be provable for some examples but are not. Induction is then used to learn rules for inferring these intermediate concepts from observables. This is the second situation illustrated in Figure 2 where there is a gap at the "bottom" of the theory. An example would be if one of the rules for *graspable* was missing from the *cup* theory.

A third case is illustrated in the final situation in Figure 2, where there is a gap is in the "middle" of the theory. An example would be if the rule for inferring *liftable* was missing from the *cup* theory. None of the previous research seems to directly address this issue.

An ideal system should be able to deal with multiple gaps occurring at arbitrary levels in the domain theory. It should also be able to introduce new intermediate concepts in order to handle the situation in which the gap in the theory spans multiple levels. For example, imagine that all rules for inferring both *liftable* and *graspable* were missing from the theory in Figure 1. In this case, the intermediate concept *graspable* is not even present in the theory and must be created.

The latest version of EITHER combines a number of previous techniques from theory refinement and constructive induction in order to deal with this general problem. In particular, *intermediate concept utilization* allows existing intermediate concepts in the theory to be used as antecedents of learned rules. *Intermediate concept creation* employs inverse resolution to introduce new intermediate concepts in order to fill a gap in the theory than spans multiple levels. We will first review the basic EITHER algorithm and then discuss these recent additions.

3 Overview of EITHER

EITHER's original theory refinement algorithm is presented in [Ourston and Mooney, 1990]. It was designed to correct theories that are either overly general or overly specific or both. An overly general theory is one that causes an example (called a *failing negative*) to be classified in categories other than its own. EITHER specializes existing antecedents, adds new antecedents, and retracts rules to fix these problems. An overly specific theory causes an example (called a *failing positive*) not to be classified in its own category. EITHER retracts and generalizes existing antecedents and learns new rules to fix these problems.

During theory generalization, EITHER uses a greedy covering algorithm to find a near-minimum set of antecedent retractions that correct all of the failing positive examples. At each iteration of the covering algorithm, the system calculates a benefit-to-cost ratio for each set of antecedent retractions that would complete a proof for a failing positive, and the set with the most examples covered per antecedent retracted is added to the cover. This continues until all of the failing positives have been covered. If retracting antecedents from a given rule over-generalizes by creating additional failing negatives, EITHER uses the failing positive examples for the rule, and the negative examples that become provable when the consequent of the rule is assumed true, to inductively[1] form a new rule that correctly classifies these examples.

During theory specialization, EITHER uses a greedy covering algorithm to identify a near-minimum set of leaf-level rule retractions that fixes all of the failing negatives. At each iteration of the covering algorithm, the system determines the number of faulty proofs in

[1]EITHER currently uses a version of ID3 [Quinlan, 1986] as its inductive component.

which each rule participates and the rule retraction that removes the most proofs is added to the cover. This continues until all faulty proofs for all failing negatives are removed. If a given rule retraction overspecializes by causing additional failing positives, additional antecedents are inductively learned that discriminate between the positive examples for the category and the erroneously proven negative examples.

The initial EITHER procedure focuses on rules at the "bottom" of the theory where changes generally have fewer ramifications. Therefore, the basic procedure easily handles the middle case in Figure 2 where there are rules missing at the bottom of the theory. When the second rule for *graspable* was deleted from the theory in Figure 1, EITHER was able to learn it given 50 random examples composed of 50% positive examples of *cup* and 50% near-miss negative examples.

4 Intermediate Concept Utilization

Intermediate concept utilization identifies existing intermediate concepts in the theory that can be used as antecedents in learned rules. First, forward chaining is used to identify truth values of all intermediate concepts for each of the failing examples. These intermediate concepts are then fed to the inductive learner as additional features. In this way, if an intermediate concept is highly correlated with the class of the failing examples, then this concept is returned as an antecedent in the rules formed by the inductive learner. This approach allows the system to learn rules that fill gaps in either the "middle" or the "top" of the theory.

For example, assume that the cup theory is missing the rule for *liftable*. *Liftable* is identified as an unprovable antecedent of the *cup* rule, and EITHER attempts to learn a new rule for *liftable*. Forward chaining on the unprovable positive examples (in this case, all of the positive examples) always adds the feature *graspable*. On the other hand, none of the negative examples will be both graspable and lightweight, since none of them is liftable (remember the negative examples used are those that become provable when *liftable* is assumed true). Consequently, the inductive learner, given enough examples, will select the intermediate concept *graspable* as an antecedent for the new rule for *liftable*. When given 20 random examples, EITHER learns the rule:

(liftable) $\leftarrow$ (graspable) (weight ?G0009)
(< ?G0009 2.2346835)

Intermediate concept utilization also allows EITHER to handle gaps at the very top of the theory as in normal constructive induction. For example, when the rule for *cup* is deleted from the theory in Figure 1, it easily learns it given 30 random examples.

5 Intermediate Concept Creation

Intermediate concept creation serves the twin purposes of compressing the rulebase and identifying new intermediate concepts. The concept creation algorithm used by EITHER is based on inverse resolution [Muggleton, 1987; Muggleton and Buntine, 1988]. In particular, EITHER uses the intra-construction, inter-construction, and absorption operators to identify new intermediate concepts in the rules produced or modified by the basic refinement procedure presented.

In DUCE [Muggleton, 1987], sets of rules are compared in order to identify common patterns, and then combined and compressed using one of the inverse resolution operators. The inter-construction and intra-construction operators introduce new concepts in the process. The basic procedure is an iterative one in which operators are applied repeatedly until no further reduction of the theory is possible.

In inter-construction, a single rule is formed to extract the common pattern associated with the input rules. For example, rules such as $x \leftarrow w \land y \land z$ and $x \leftarrow u \land y \land z$ are combined to form the rules: $x \leftarrow w \land v$ and $x \leftarrow u \land v$ and $v \leftarrow y \land z$, where v is a new intermediate concept.

In intra-construction, new rules are formed representing the differences between the input rules. For example, the same rules as above would be combined as $x \leftarrow v \land y \land z$ where $v \leftarrow w$ and $v \leftarrow u$ where v is again a new intermediate concept. Note that unlike inter-construction, intra-construction requires that both input rules have the same consequent.

Absorption occurs when all of the antecedents for one rule (e.g. $x \leftarrow a \land b$) are contained in the antecedents of another (e.g. $y \leftarrow a \land b \land c$). The consequent for the smaller rule is inserted into the antecedents for the larger rule, in place of the antecedents which the two rules have in common (e.g. $y \leftarrow x \land c$). In the general case, absorption could happen even if there were many rules implying the consequent for the smaller rule, and the combination would represent a generalization to the larger rule. Since the original EITHER algorithm guarantees consistency with the example set, EITHER only allows absorption when there is a single version of the absorbed rule (i.e. a single rule with the given consequent) so that the semantics of the rules are unchanged.

Appears in *Proceedings of the Eighth International Machine Learning Workshop*,
pp. 178-182, Evanston, IL, June 1991.

After EITHER produces a revised theory that is consistent with the training examples, the above operators are used to compress any rules that were modified or created during the revision. In the process, new intermediate concepts are created. The EITHER procedure is slightly different from the original one in DUCE in that it does not employ an oracle, does not actually generalize the input rules, and employs hill-climbing rather than best-first search in order to find a good operator to apply.

Let the original set of rules under consideration for rule reduction be given by:
$X_i \leftarrow A \wedge N_i \qquad (1 \leq i \leq n)$
Where A represents the set of antecedents which are in common among all of the rules, and N_i represents the remaining antecedents for each rule. The objective in choosing A is to produce the greatest syntactic reduction. The computation of A uses a greedy algorithm and is done separately for inter and intra construction, since a different set of rules may be involved in each case. At each iteration, a new literal is chosen to add to A which causes the largest reduction in the input rules. If the reduction with the literal added is less than the previous reduction, the process halts. The overall process is halted when no further reduction is possible. Once A has been computed for each case, the reduction operator which produces the greatest syntactic reduction is chosen. In case of ties, intra-construction is chosen since it focuses the reduction on rules having a single consequent.

As an example of intermediate concept creation, consider the case in which all of the rules for both *liftable* and *graspable* are deleted from *cup* theory. Given 50 examples, EITHER initially learns the rules

(liftable) ← (has-handle) (weight ?G0009)
(< ?G0009 1.1257166)
(liftable) ← (insulating) (width small)
(not (has-handle)) (weight ?G0009)
(< ?G0009 1.1257166)

These rules are then reduced to:

(liftable) ← (intra-0010) (weight ?G0009)
(< ?G0009 1.1257166)
(intra-0010) ← (has-handle)
(intra-0010) ← (insulating) (width small)
(not (has-handle))

The intermediate concept *intra-0010* formed using intra-construction is EITHER's new concept for *graspable*. The extra (not (has-handle)) antecedent on the second rule is a side-effect of translating ID3 decision trees into rules. It does not effect the semantics of the new concept and could be deleted using the sort of rule simplification methods discussed in [Quinlan, 1987].

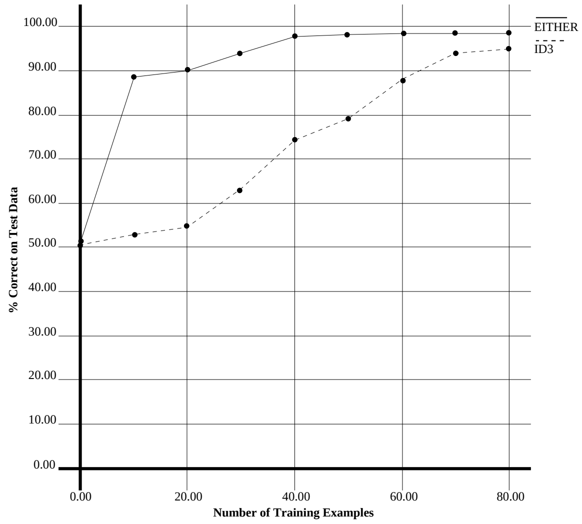

Figure 3: Learning Curves for Liftable Rule

6 Empirical Results on Learning Rate

In order to demonstrate that EITHER's ability to refine arbitrarily imperfect domain theories actually increases it's learning rate, this section presents learning curves for the case of a gap in the middle of the theory. Recall that none of the previous approaches addressed this particular problem. Results for the other types of incomplete theories are similar.

Artificial data was automatically generated for the cup theory shown in Figure 1. Positive examples of *cups* were generated by first forming "core" examples, which contain just the observables needed to complete a proof. For linear features, a value is chosen randomly from the range required for a proof. Next, random values for the remaining observable features were added to the core examples to create full examples. "Cores" for near-miss negative examples were formed by randomly changing one feature in a core-positive example in order to make it unprovable. Again, random values are added for the remaining observables to fill out the examples. Adding random values can sometimes make a core-negative example provable by allowing an alternate proof to succeed. Consequently, each negative example was checked to make sure it was not provable before adding it to the final data set. A total of 50 positive and 50 near-miss negatives were generated in this manner.

EITHER was given the theory in Figure 1 with the rule for *liftable* removed. Learning curves were generated by performing batch training on increasingly larger fractions of a set of training examples and repeatedly testing predictive accuracy on the same disjoint test set of 20 test examples. The final results were averaged over 30 random selections of training and test sets. EITHER was compared to ID3, which is the same as EITHER without an initial theory since ID3 is EITHER's inductive learning component. The learning curves in Figure 3 shows that the initial incomplete theory provides a significant performance advantage. A statistical t-test showed that the difference between the two curves is significant for each non-zero point plotted ($p < .05$). Similar results hold for all of the other incomplete theories discussed in this paper.

7 Related Work

By integrating previous methods for constructive induction and theory refinement, EITHER can employ a wider range of imperfect theories than previous systems. Other systems using imperfect domain theories to aid concept learning [Wilkins, 1988; Danyluk, 1989; Pazzani, 1989; Flann and Dietterich, 1989; Drastal *et al.*, 1989] cannot deal with arbitrary gaps in a domain theory and cannot introduce new intermediate concepts. KBANN [Towell *et al.*, 1990] and RTLS [Ginsberg, 1990] perhaps come the closest to handling as many types of imperfections. However, unlike EITHER, neither of them can deal with actual gaps in the theory (where there are no rules for proving a category or intermediate concept) nor introduce new intermediate concepts. These issues could possibly be addressed in KBANN (which translates a theory into a neural-net, refines it using backpropagation, and then retranslates the result back into rules) by adding extra hidden units and connections to the initial network; however, this would require predetermining the number and type of intermediate concepts to be created. In addition, unlike EITHER, KBANN is not guaranteed to produce a revised theory that is consistent with the training data.

8 Conclusions

In this paper, we have presented constructive induction techniques recently added to the EITHER theory refinement system. *Intermediate concept utilization* employs existing rules in the theory to derive higher-level features for use in induction. *Intermediate concept creation* employs inverse resolution to introduce new intermediate concepts in order to fill gaps in a theory than span multiple levels. These revisions allow EITHER to make use of imperfect domain theories in the ways typical of previous work in both constructive induction and theory refinement. As a result, EITHER is able to handle a wider range of theory imperfections than any other existing theory refinement system.

Acknowledgements

This research was supported by the NASA Ames Research Center under grant NCC 2-629. Equipment was donated by the Texas Instruments Corporation.

References

[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.

[Drastal *et al.*, 1989] G. Drastal, G. Czako, and S. Raatz. Induction in an abstraction space: A form of constructive induction. In *Proceedings of the Eleventh International Joint conference on Artificial intelligence*, pages 708–712, 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.

[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.

[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.

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

[Pazzani, 1989] M. J. Pazzani. Detecting and correcting errors of omission after explanation-based learning. In *Proceedings of the Eleventh International Joint conference on Artificial intelligence*, pages 713–718, Detroit, MI, Aug 1989.

[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.

[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.

[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.