Appears in *Proceedings of the Eighth International Machine Learning Workshop*, pp. 534-538, Evanston, IL, June 1991.

Improving Shared Rules in Multiple Category Domain Theories

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

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

Abstract

This paper presents an approach to improving the classification performance of a multiple category theory by correcting intermediate rules which are shared among the categories. Using this technique, the performance of a theory in one category can be improved through training in an entirely different category. Examples of the technique are presented and experimental results are given.

1 Introduction

Frequently, domain theories involve multiple concepts that share intermediate rules. For example, in a theory for a variety of animal species, the concepts for giraffes and tigers might share knowledge about mammals, and the concepts for penguin and duck might share knowledge about birds. Improving rules for the shared concept "mammal" can increase classification accuracy for both giraffes and tigers. However, refining rules for such shared intermediate rules must be done carefully in order to account for the impact on all of the concepts that use them.

Revising rules for shared concepts frequently has the interesting effect of improving performance on test data that is drawn from a completely different population than the training data. For example, if the system is trained on ostriches, it may improve performance when tested on penguins. We refer to this effect as *cross-category transfer*. Standard empirical learning systems are incapable of exhibiting cross-category transfer. Most theoretical work in learning also assumes that the distribution of examples is the same during training and test [Valiant, 1984].

This paper presents an implemented method for revising domain theories with shared intermediate rules. We are developing a system called EITHER that is capable of revising arbitrary propositional Horn-clause domain theories. In [Ourston and Mooney, 1990] we describe an initial version of the system that refines theories for a single concept and has several restrictions on modifying intermediate rules. The techniques described in this paper extend the previous methods to multi-category theories and to corrections to shared rules. The remainder of this paper describes our approach and provides empirical results that illustrate effective transfer to test data that is radically different from the training data.

2 Sample Theory

The theory below is a sample domain theory for animals that will be used as an example throughout the paper. This theory is an extended version of a set of rules given in [Winston and Horn, 1989, pages 388-390]. Leading question marks denote variables, which are only used to define thresholds on numerically-valued features.

(mammal)	$\leftarrow$	(body-covering hair)
(mammal)	$\leftarrow$	(feed-young milk)
(mammal)	$\leftarrow$	(birth live)
(bird)	$\leftarrow$	(body-covering feathers)
(bird)	$\leftarrow$	(birth egg) (fly)
(ungulate)	$\leftarrow$	(mammal) (foot-type hoof)
(ungulate)	$\leftarrow$	(mammal) (ruminate)
(carnivore)	$\leftarrow$	(eat-meat)
(carnivore)	$\leftarrow$	(teeth pointed) (foot-type clawed)
(giraffe)	$\leftarrow$	(ungulate) (neck-length ?n) ($\geq$?n 5)
		($\leq$?n 6) (color tawny)
		(pattern spots) (pattern-color black)
(zebra)	$\leftarrow$	(ungulate) (color white)
		(pattern stripes)(pattern-color black)
(cheetah)	$\leftarrow$	(mammal)(carnivore)(color tawny)
		(pattern spots)(pattern-color black)
(tiger)	$\leftarrow$	(mammal) (carnivore) (color tawny)
		(pattern stripes)(pattern-color black)

(dolphin)	←	(mammal) (fore-appendage fin) (color gray)(body-covering moist-skin) (body-length ?b) (≥ ?b 4) (≤ ?b 6)
(whale)	←	(mammal) (fore-appendage fin) (color gray) (body-covering moist-skin) (body-length ?b) (≥ ?b 10) (≤ ?b 60)
(bat)	←	(mammal) (color black) (pattern none) (pattern-color none) (fly)
(platypus)	←	(mammal) (birth egg) (foot-type webbed)
(ostrich)	←	(bird) (not (fly)) (neck-length ?n) (≥ ?n 3)(≤ ?n 4) (color white) (pattern patch) (pattern-color black)
(penguin)	←	(bird) (color white) (pattern patch) (pattern-color black) (foot-type webbed) (not (fly))
(duck)	←	(bird) (foot-type webbed) (fly)
(grackle)	←	(bird) (color black) (pattern none) (pattern-color none) (fly)

Observable Features: feed-young body-covering birth eat-meat fly teeth fore-appendage foot-type neck-length body-length color pattern pattern-color ruminate
Categories: giraffe zebra cheetah tiger dolphin whale bat platypus penguin ostrich duck grackle

Propositions that are used to describe the examples (for example, (foot-type webbed)) are called *observables.* Propositions that represent the final concepts in which examples are to be classified (for example, penguin) are called *categories.* It is currently assumed that the categories are disjoint. Propositions in the theory that are neither observables nor categories (for example, bird, mammal) are called *intermediate concepts.* A *failing positive* is an example that should be provable as an example of its category, but fails to be provable. A *failing negative* is an example provable in a category other than its own.

3 Revision Approach

EITHER responds to failing negative examples (indicating an overly general theory) by performing rule specialization. Similarly, the response to failing positive examples is rule generalization. EITHER determines the proper location for the rule correction based on a syntactic simplicity measure, subject to a correctability constraint (as will be shown below, some rules responsible for the error cannot be changed to fix the problem). EITHER also uses constructive induction methods in order to effective utilize intermediate concepts in the existing theory and to create new intermediate concepts [Mooney and Ourston, 1991]. The resulting approach tends to focus corrections on rules which are used by multiple categories.

3.1 Rule Generalization

Rule generalization is performed in response to specialization errors in the existing theory, which are indicated by failing positive examples. EITHER initially associates the errors with leaf-level rules[1]. This is done in a batch fashion, and the minimum set of such rules which account for all of the failing positive examples is identified. For each rule EITHER identifies antecedents which, if retracted, would allow proofs of the failing positive examples associated with the rule.

However, simply removing antecedents from the rule may prove to be too much of a generalization (that is, doing so may cause additional failing negatives). If this is the case, EITHER uses the failing positive examples for the rule, and the negative examples which become provable when the consequent of the rule is assumed true, to inductively[2] form a new rule which correctly classifies the positive and negative examples. If the proposed correction results in all of the antecedents for a particular rule being removed, EITHER simply removes the consequent of the rule as an antecedent from all of its parent rules.

The correction process may ultimately result in the correction being placed at a higher level in the theory, as discussed below, but in each case, the generalization algorithm will first attempt to generalize the selected rule by modifying antecedents, and will only inductively learn new rules if this is unsuccessful.

3.2 Rule Specialization

For theory specialization, EITHER first identifies the minimum set of leaf-level rules which, if removed from the theory, would cause all of the failing negative examples to no longer be proven in incorrect categories. As with generalization, this initially-suggested change to the theory may be an over-correction, resulting in examples not being provable in their own categories. If this is the case, then specialized rules are inductively learned which discriminate between the positive examples for the category and the erroneously proven negative examples. If the correction is to be made to a higher level rule, as discussed below, a new rule will be inductively learned.

[1]A "leaf-level" rule is a rule which includes observables among its antecedents.

[2]EITHER uses the ID3 [Quinlan, 1986] as its inductive component.

3.3 The Correctability Problem

In multi-category theories, it may not be possible to make a theory consistent with a set of examples by modifying leaf-level rules. For example, consider the following theory in which the category rules rely on the same leaf-level rule.

$C_1 \leftarrow R$
$C_2 \leftarrow R$
$R \leftarrow A \wedge B$

This is an extremely pathological theory in which any example will either be provable in both categories or neither, and the same remarks apply when any *change* is made to the leaf-level rule. What this means is that the "R" rule cannot be changed to correct the problem. In the animal domain, this type of problem could correspond to trying to fix confusions between grackles and ducks through modifications to the "bird" rules. In general, the problem is detecting that such a condition exists (either directly or indirectly), and then making a correction that causes examples to be classified only in their own categories. For a particular rule, this problem can be detected during specialization when removing the rule from the theory causes a particular example to fail to be proven in its own category, and yet leaving the rule in causes the example to be provable in at least one other category. In this case, the same rule is being used in an erroneous proof of the example and also in its correct proof in its own category. Hence any change to the rule which causes the example not to be erroneously provable will also cause it not to be provable in its own category. For generalization, the problem can be detected when assuming the consequent of a rule to be true causes a particular example to be provable in at least one other category, yet in the original theory the same example is not provable in its own category.

In order to associate the problem with a single rule, the generalization algorithm requires that all failing negatives be removed from the input, and the specialization algorithm requires a strictly overly general theory (that is, it uses the output of the generalization algorithm).

The solution to the correctability problem (the same rule being responsible for proofs of examples in multiple categories), is to make the corrections to rules at higher levels in the theory. This is done by considering the parents[3] of the rule in question and determining if a correctability problem exists for the parents of the rule, and so on recursively, until a set of rules are obtained for which no correctability problem exists. Clearly, there exists such a set of rules, since in the limit the corrections can be made to the *category* rules. Since each such rule implies membership in a particular category, retracting such a rule can only cause examples to be unprovable in the associated category, and no other, and no correctability problem can occur.

[3]Here, parent means a rule which used the given rule in the proof of an antecedent. For specialization, this occurs during the proof of a negative example, whereas for generalization, this occurs during the partial proof of a positive example.

In the implemented EITHER program, for efficiency reasons the correctability algorithm has only been incorporated into the specialization algorithm. However, doing this still guarantees that the output will be consistent with the training set.

3.4 Finding the Simplest Correction

Once a consistent set of rules has been obtained, EITHER begins the process of selecting the syntactically simplest set of rule corrections. For each rule in the current correction set, EITHER compares the change to the rule with the change which would be required by the rule's parents. If the change to the parent rules is simpler than the change to the current rule, the process is continued with the current rule replaced by its parent rules. This process terminates when the change to the parent rule is more complex than the change to the current rule.

As an example of a situation which would result in a higher level rule being chosen, consider the following theory:
$w \leftarrow x$
$x \leftarrow a$
$x \leftarrow b$
$x \leftarrow c.$
and assume that the first rule should correctly be:
$w \leftarrow x \wedge d \wedge e.$
If EITHER was given examples corresponding to the correct theory, and was forced to make the corrections to the theory at the leaf level, than EITHER would obtain the corrected theory (which would be consistent with the input examples):
$w \leftarrow x$
$x \leftarrow a \wedge d \wedge e$
$x \leftarrow b \wedge d \wedge e$
$x \leftarrow c \wedge d \wedge e.$
Clearly, changing the theory at the higher level would result in a much simpler syntactic change to the theory.

On the other hand, consider the theory:

$w \leftarrow x$
$y \leftarrow x$
$z \leftarrow x$
$x \leftarrow a$,
and assume that the correct version of the last rule is:
$x \leftarrow a \wedge b \wedge c$.
If the correction is made at the higher level, then each of the rules "w" "y" and "z" would have to have the antecedents b and c added. In this case, because of syntactic simplicity, EITHER will make the correction to the "x" rule.

Because of these considerations, EITHER tends to make corrections at or below the level of the first shared rule encountered. This implies that the selected rule will be likely to participate in multiple concept definitions, which will allow the correction to enhance the performance in other, related, categories. For efficiency reasons, in the implemented version of EITHER the syntactic simplicity check is always terminated at the first shared rule (that is, a rule having multiple parents).

4 Experimental Results

EITHER has been tested on two multi-category domain theories with shared intermediate rules. This section presents results showing that EITHER can successfully revise shared rules and that such revisions result in positive transfer to test data that is very different from the training data.

Artificial data was automatically generated for the animal theory and a similar theory for classifying different types of computers based on their appearance (categories: pc, mac, macII, sun, hp, explorer, symbolics; intermediate concepts: workstation, micro, lisp-machine, unix-workstation, macintosh). Thirty examples of each category 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. However, adding random values can sometimes make an example provable in another category as well. Consequently, each example was checked to make sure it was provable in only one category before adding it to the final data set. A total of 360 examples of animals and 210 examples of computers were created in this manner.

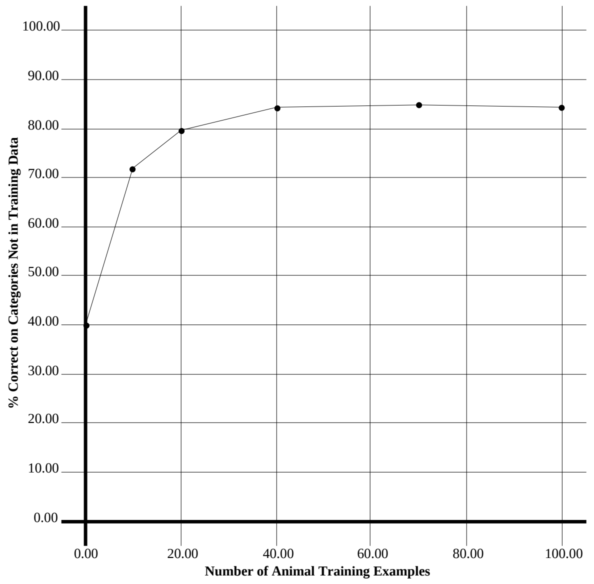
Imperfect versions of the animal and computer theories were also constructed. In each case, the rules for some of the intermediate concepts were corrupted in order to allow for cross-category transfer. The rules corrupted in the animal theory are shown below. Items shown in boldface were added to the theory whereas items shown in italics were deleted from the theory. The faults introduced include missing rules, additional antecedents, and missing antecedents.

(mammal)	←	(body-covering hair) **(fore-appendage leg)**
(mammal)	←	(feed-young milk) **(fore-appendage leg)**
(mammal)	←	(birth live) **(fore-appendage leg)**
(bird)	←	*(body-covering feathers)*
(bird)	←	*(birth egg)* (fly)
(duck)	←	(bird) (foot-type webbed) *(fly)*
(ostrich)	←	... *(not (fly))*
(penguin)	←	... *(not (fly))*

In order to demonstrate cross-category transfer, the system was trained on examples from some of the categories and tested an examples of the remaining categories. In the animal domain, the system was trained on giraffes, cheetahs, dolphins, bats, platypuses, ostriches, and ducks and tested on zebras, tigers, whales, penguins, and grackles. In the computer domain, EITHER was trained on Macs, Suns, and Explorers, and tested on MacII's, HP's, and Symbolics. 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. The final results were averaged over 20 random selections of training and test sets.

The results are shown in Figure 1. EITHER greatly improves its performance despite the fact that the training and test data are drawn from completely different populations. This is because it revises rules for shared intermediate concepts. For example, when trained on ostriches, bats, and dolphins it discovers that flying is neither necessary nor sufficient for birdhood and that mammals don't have to possess legs. It uses the data to make the correct revisions to these rules which allow it to accurately classify penguins and whales during testing. Performance in the animal domain levels out at 85% since some of the faults are in non-shared rules and are only detected when the training set contains examples from all of the categories. In the computer domain, the correct theory is fully reconstructed by 40 training examples in most trials. Notice that a normal inductive learning system would remain at 0% accuracy for this sort of test. If it never encounters any examples of penguins or whales in the training set, it will never classify a test example as a penguin or a whale.

If EITHER is trained on random examples of all categories, it eventually reconstructs the original theories for both domains. About 100 examples are generally sufficient to fix the animal theory, while 60 examples are sufficient for the computer theory. When initialized with the imperfect theories, EITHER's learning rate is significantly better than without an initial theory (in which case it is the same as ID3).



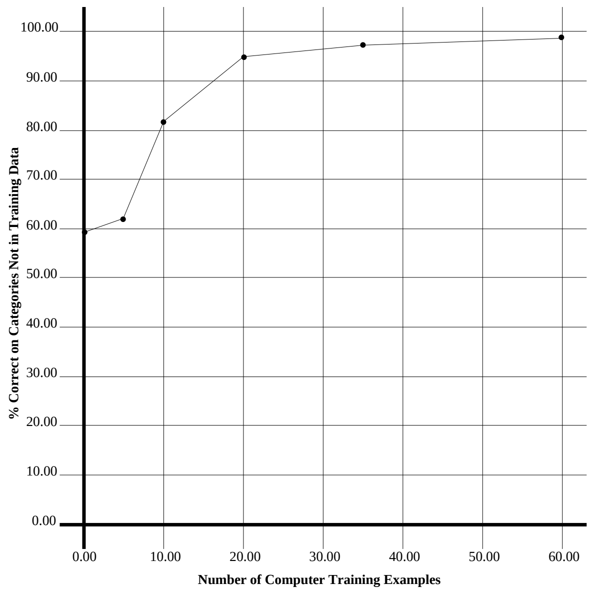


Figure 1: Cross-Category Transfer Results

5 Related Work

As discussed in [Ourston and Mooney, 1990], most systems that use imperfect domain theories in learning can only handle certain types of overly general or overly specific theories [Wilkins, 1988; Flann and Dietterich, 1989; Pazzani, 1989]). Finally, none of these systems explicitly deal with multi-category theories. Ginsberg has recently developed a version of RTLS that produces a revised multi-category theory [Ginsberg, 1990]; however, he explicitly mentions that it cannot revise rules for shared concepts (which he calls *non-eigen-terms*). However, as illustrated in this paper, revising shared rules is extremely important since it allows for cross-category transfer. KBANN [Towell *et al.*, 1990], which converts a theory into a neural net and then refines it using backpropagation, could potentially demonstrate cross-category transfer by refining the weights on shared hidden units. However, KBANN produces a neural-net classifier rather than a revised theory.

6 Conclusions

The ability to improve shared rules allows the corrections to a theory to be focussed on certain key rules, which may play a part in the definition of several concepts within the theory. Learning of this kind permits cross-category transfer, where training in a one category produces performance improvements in related categories.

Multiple category theories also introduce correctability problems which are not present in single category theories, caused by a single rule being used in the proofs of several categories for the same example. EITHER addresses these problems with multiple category theories, while still obtaining rule corrections which are syntactically simple and which apply to shared rules whenever possible.

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

[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.

[Mooney and Ourston, 1991] 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.

[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.

[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.

[Valiant, 1984] L. G. Valiant. A theory of the learnable. *Communications of the Association for Computing Machinery*, 27(11):1134–1142, 1984.

[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 and Horn, 1989] P. H. Winston and B. K. P. Horn. *Lisp*. Addison-Wesley, Reading, MA, 1989.