Technical Report AI 91-153, Artificial Intelligence Lab, University of Texas at Austin, March, 1991

Theory Refinement with Noisy Data*

Raymond J. Mooney and Dirk Ourston
Department of Computer Sciences
University of Texas
Austin, TX 78712
email: mooney@cs.utexas.edu [(512)471-9558],
dirk@cs.utexas.edu [(512)471-9715]

April 1, 1992

Abstract

This paper presents a method for revising an approximate domain theory based on noisy data. The basic idea is to avoid making changes to the theory that account for only a small amount of data. This method is implemented in the EITHER propositional Horn-clause theory revision system. The paper presents empirical results on artificially corrupted data to show that this method successfully prevents over-fitting. In other words, when the data is noisy, performance on novel test data is considerably better than revising the theory to completely fit the data. When the data is not noisy, noise processing causes no significant degradation in performance. Finally, noise processing increases efficiency and decreases the complexity of the resulting theory.

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

1 Introduction

Theory revision, or knowledge-base refinement, is a difficult problem that has important applications in the development of expert systems. An approximate domain theory obtained from an expert or a textbook can be substantially improved by observing the errors it makes on a set of empirical data and automatically modifying the theory to correct for these errors. Previous work on this problem [Ginsberg, 1990; Ourston and Mooney, 1990a] has focussed on basic mechanisms for refining theories to completely fit a set of noise-free data. Unfortunately, real-world data is frequently noisy and contains incorrect categorizations and feature values. In such cases, modifying the theory to account for all of the data may result in "over-fitting" by unnecessarily changing and complicating the theory to accommodate noisy examples. Although the noise problem has been confronted in standard inductive learning systems [Quinlan, 1986a; Mingers, 1989], the problem of refining an existing domain theory using noisy data has not previously been addressed.

This paper presents a method for using noisy data to revise domain theories. We are developing a system called EITHER that is capable of revising arbitrarily imperfect propositional Horn-clause domain theories. In [Ourston and Mooney, 1990a] we described an initial version of the system that modifies a theory to completely fit a set of empirical data. This paper describes important additions to EITHER for handling noisy data. Basically, the system is prevented from making complicated changes to the theory in order to account for only a small amount of data. We present empirical results that show this method successfully prevents over-fitting. In other words, when the data is noisy, accuracy on novel test data is significantly better than revising the theory to completely fit the data. When the data is not noisy, noise processing does not significantly effect performance. By avoiding unnecessary revisions, noise processing also increases efficiency and decreases the complexity of the resulting theory.

2 Problem Definition

Figure 1 gives a definition of the problem EITHER addresses. It is difficult to precisely define the phrases "minimally revised" and "most of the examples." 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. In order to prevent over-fitting, EITHER avoids making changes to the theory that account for only a single piece of data.

As in [Mitchell *et al.*, 1986], concepts are defined as predicates over some universe of instances and a domain theory is defined as a logical theory that defines a set of concepts. The current version of EITHER is restricted to Horn-clause theories expressed in a propositional

Given: An imperfect domain theory for a set of categories and a set of classified examples that are described by a set of observable features and that may suffer from feature and category noise.

Find: A minimally revised version of the domain theory that correctly classifies most of these examples without over-fitting the theory to noise in the data.

Figure 1: Problem Definition for Theory Refinement with Noise

logic whose atomic formulae include feature-value pairs and numerical thresholds as well as binary propositions. Figure 2 shows a sample domain theory for animals. 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.

For the purpose of refinement, a closed-world assumption is made. If the theory does not prove that an example is a member of a concept, then it is assumed to be a negative example of that concept. A domain theory is required to be acyclic and therefore defines a directed acyclic graph (DAG). Propositions that are used to describe the examples (e.g. (color black)) are called *observables*. To avoid problems with negation as failure, only observables can appear as negated antecedents. Propositions that represent the final concepts in which examples are to be classified (e.g. penguin) 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 (e.g bird) are called *intermediate concepts*.

3 Review of EITHER's Theory Revision Algorithm

EITHER's original theory refinement algorithm is presented in [Ourston and Mooney, 1990a] and an extension to this algorithm for multi-category theories is presented in [Ourston and Mooney, 1990b]. The EITHER algorithm is 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. This is due to additional rules or rules with missing or overly-general antecedents. 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. This is due to missing rules or rules with extra or overly-specific antecedents. 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 candidate antecedents that conflict with the facts pertaining to the failing

(mammal) ← (body-covering hair)
(mammal) ← (feed-young milk)
(mammal) ← (birth live)
(bird) ← (body-covering feathers)
(bird) ← (birth egg) (fly)
(ungulate) ← (mammal) (foot-type hoof)
(ungulate) ← (mammal) (ruminate)
(carnivore) ← (eat-meat)
(carnivore) ← (teeth pointed) (foot-type clawed)
(giraffe) ← (ungulate) (neck-length ?n) ($\geq$?n 5) ($\leq$?n 6) (color tawny)
(pattern spots) (pattern-color black)
(zebra) ← (ungulate) (color white) (pattern stripes) (pattern-color black)
(cheetah) ← (mammal) (carnivore) (color tawny) (pattern spots)(pattern-color black)
(tiger) ← (mammal) (carnivore) (color tawny) (pattern stripes)(pattern-color black)
(dolphin) ← (mammal) (fore-appendage fin) (color gray)(body-covering moist-skin)
(body-length ?b) ($\geq$?b 4) ($\leq$?b 6)
(whale) ← (mammal) (fore-appendage fin) (color gray) (body-covering moist-skin)
(body-length ?b) ($\geq$?b 10) ($\leq$?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) ($\geq$?n 3) ($\leq$?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

Figure 2: Sample Domain Theory

positive examples. The condition on these antecedents is that if they are removed from the theory, then the failing positive examples will be provable. EITHER discovers these antecedents through *partial proofs* of the failing positive examples: proofs that would be complete if the candidate antecedents were removed from their associated rules.

At each iteration of the covering algorithm, the system calculates a benefit-to-cost ratio for each set of candidate antecedents that will fix a failing positive, and the set with the most examples fixed per antecedent is added to the cover. This continues until all of the failing positives are provable. However, simply removing antecedents from some rules may over-generalize the theory. If retracting antecedents causes 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 which each rule participates and the rule retraction that removes the most proofs is added to the cover. This continues until all faulty proofs of all failing negatives are removed. As with generalization, this initially-suggested change to the theory may be an over-correction. If a given rule retraction causes additional failing positives, then additional antecedents are inductively learned that discriminate between these failing positives and the erroneously proven negative examples.

EITHER also contains a number of extensions to this basic procedure which are concerned with generalizing and specializing existing antecedents; using existing rules in the theory to construct new features; extracting new intermediate concepts from inductively learned rules; and finding the appropriate level in the theory to modify. These extensions omitted from this discussion since they are beyond the scope of this paper. The interested reader is referred to [Ourston and Mooney, 1990b].

4 Modifications to Handle Noisy Data

For examples represented as attribute-value lists, there are two primary sources of noise. The first is mis-specification: the value for an attribute or category may be incorrect, for a variety of reasons including typographical errors, errors in measurement, and perception errors. The second type of noise is called residual variation in [Mingers, 1989], and refers to additional factors that affect the results, but that are not recorded. This is a fairly common source of noise in real-world applications and can occur because those recording the data were either unaware of the affect of the additional factors, or simply unable to record them. With either source of noise, the effect is examples that are either inconsistent (i.e. two examples that

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

have identical attributes, but that are labeled in different categories), or examples that have features that should correlate with their category, but that do not because of noise masking the correlation.

One of the main problems with learning algorithms that don't account for noise is that they may tend to "overfit" the data. That is, elaborate rules or decision structures are generated that serve to account for a small number of anomalous examples among the training set. Generating the additional rules not only increases the processing time and the complexity of the result, but in general will actually decrease performance since the learning algorithm is learning rules that only apply to the noisy examples and may misclassify correct examples.

Response to noisy data shows up at three points in EITHER: in the generation of the minimum covers, in the corrections EITHER makes to rules without calling the inductive learner, and in the learning of rules and additional antecedents by the inductive learner. When noisy data are submitted to EITHER, the resulting theory corrections can no longer be guaranteed to be consistent with the training data (as mentioned above, examples in the training data may not be consistent with each other). When noise processing is invoked, the corrected theory may not even be consistent with *noise-free* training data. The next three subsections describe EITHER's response to noise in more detail.

4.1 Accounting for Noise in the Minimum Covers

The principle behind handling noise in cover generation is that one is more confident in corrections to rules that apply to several failing examples. If the correction only applies to a single example, then it could well be that it was the *example*, rather than the rule that was in error. However, if the same (erroneous) rule is implicated by several failing examples, it is unlikely that all of the examples just happen to have noisy attributes that cause them to all focus on the same rule.

To be specific, for failing positives, assume several of the partial proofs of the examples happen to use the same rule, with the same set of antecedents in conflict with the features of the example. This is unlikely to happen if the examples are subject to random feature noise. For failing negatives, assume several of the incorrect proofs of the examples use the same rule. This is unlikely to happen if the incorrect proofs of the examples are due to random feature noise in the examples.

EITHER's response to noise in these cases is to prune the minimum covers provided to the rule correction algorithms. For antecedent covers, failing antecedents are grouped into separate sets as necessary to account for the failing positive examples associated with each rule. That is, a new antecedent set is created whenever the conflicting antecedents for a particular example are different from any of the antecedent sets already associated with the rule. When the cover is completed, noise processing removes those antecedent sets from the cover that correspond to only a single example. In minimum rule cover processing, terms in the cover that only correspond to a single example are also removed. In this case, any rule

in the cover that only corresponds to a single failing negative is removed.

Because EITHER is now using a partial cover, the corrected theory will not in general be consistent with the training set. However, the benefit of the partial cover in responding to noise should actually improve as the number of training examples increases. This is because the likelihood of a bad rule only affecting a single example decreases as the number of examples increases, and since EITHER is attempting to select rules which correspond to multiple examples, the likelihood of selecting a spuriously correlated rule will also decrease.

4.2 Accounting for Noise Outside the Inductive Learner

EITHER must also provides a response to noisy data in the cases where it modifies the theory without calling the inductive component since inconsistent examples may cause problems with these changes. This occurs in the following situations:

- When a rule is generalized by generalizing its antecedents.
- When a rule is generalized by retracting an antecedent.
- When the theory is specialized by removing a rule from the theory.

To account for noise, EITHER pre-processes the input data prior to entering the calculations described above. The pre-processing removes inconsistent examples in the following way:

- EITHER checks the input examples for inconsistency. Inconsistency is discovered when two examples with differing category labels are found to have the same attribute values.
- For each set of inconsistent examples, if a majority belong to a particular category, EITHER deletes the examples in other categories from the input.

This approach causes EITHER to ignore examples with obvious noise problems and provide corrections that are consistent with the remaining examples.

4.3 Accounting for Noise in the Inductive Learner

Since noisy data may be passed on to the inductive component when learning new rules and new antecedents, EITHER employs the chi-squared procedure [Quinlan, 1986a] in its inductive component based on ID3. In this method, when an attribute is being considered as a possible test feature in the decision tree, it is first checked to see if the values for the attribute are independent of the example categories, using a chi-squared statistical test. If so, this indicates that the attribute values may include noise and will not be predictive of the category. When all of the remaining attributes fail the chi-squared test, the current node is

made a leaf node (i.e. the decision tree will be "pruned" of all possible subtrees that could have originated from this node). The leaf is labelled with the majority category among the training examples input to the node.

The decision trees formed using this approach are no longer guaranteed to be consistent with the training examples. However, it has been shown that the pruned trees can provide improved accuracy on the test data [Mingers, 1989; Quinlan, 1986a]. The trees are also simpler, in that they do not include the complex subtrees necessary to account for the noisy training examples. Since EITHER transforms these decision trees directly into rules, the resulting rules should be syntactically simpler and show improved performance over the test data.

5 Experimental Results

The changes to EITHER described above were tested on revising domain theories using data artificially corrupted with noise. Feature and category noise were added to the training examples, and feature noise only to the test examples. Feature noise was added by substituting a random value from the feature's domain. Category noise was added by substituting a random category. Category noise was not added to the test examples in order to identify how well EITHER was able to identify the correct categories for the test examples. The noise level identifies the frequency of noisy data: a noise level of 0.1 means that 10% of the features and categories were randomly corrupted.

Data was automatically generated for the animal theory shown in Figure 2 and a similar theory for classifying different types of computers based on their appearance. 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. A core example was formed for each possible proof for each category. For example, below are the core examples of ducks.

1. (body-covering feathers) (foot-type webbed) (fly)
2. (birth egg) (foot-type webbed) (fly)

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. The errors added to the animal theory are shown in Figure 3 (this theory has an accuracy of only 36%). The faults introduced include missing rules, additional antecedents, and missing antecedents. These theories were given to EITHER to revise.

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

Figure 3: Corrupted Animal Theory

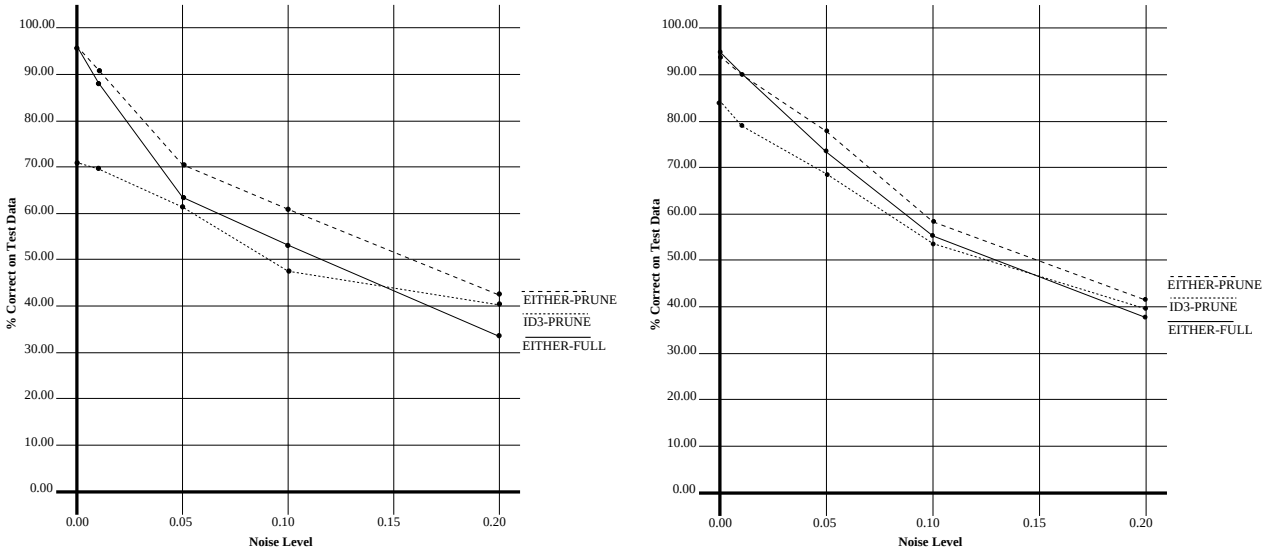

Figure 4: Test Accuracy vs. Noise Level for Animal and Computer Domains

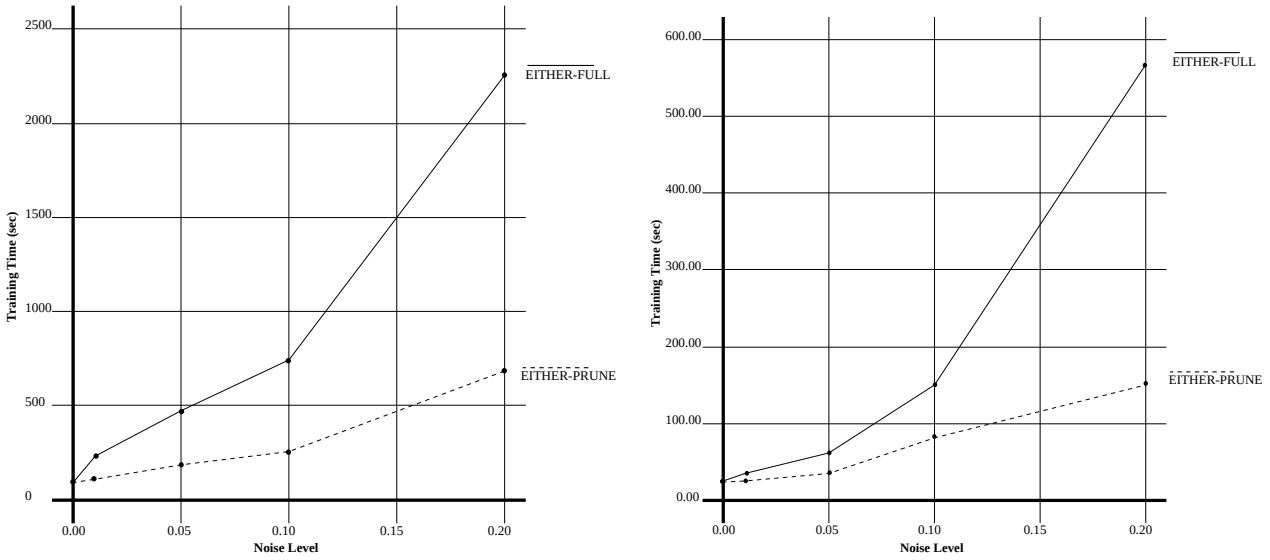

Figure 5: Train Time vs. Noise Level for Animal and Computer Domains

Figure 4 shows the degradation in predictive accuracy with increasing levels of noise for EITHER with cover truncation (EITHER-PRUNE), EITHER without cover truncation (EITHER-FULL), and ID3 (with chi-squared pruning). Since EITHER uses ID3 as its inductive component, ID3 is the same as EITHER when it is not given an initial theory. The animal results are averaged over 8 trials of 60 training and 100 test examples. The computer results are averaged over 23 trials of 40 training and 100 test examples. Due to its initial theory, EITHER-PRUNE always performs better than ID3. A statistical t-test shows that the difference is significant at each noise-level plotted except 0.2. With little or no noise (noise levels of 0.0 and 0.01) both versions of EITHER perform approximately the same. However, as the noise level is increased, the effect of pruning becomes more pronounced. The difference is statistically significant for levels of noise greater than 0.01. At noise levels of 0.1 and 0.2, pruning provides approximately a 9% advantage on animals and 4% advantage on computers. Unlike EITHER-PRUNE, EITHER-FULL eventually performs worse than ID3 at a noise-level of 0.2. Of course, one would expect the accuracy of all systems to converge to random chance as noise-level is increased to 1.0.

Unlike EITHER-FULL, EITHER-PRUNE's correctness on training data also decreases as noise is increased. Therefore, the difference between training correctness and test correctness is much smaller for EITHER-PRUNE (26% for animals at noise-level 0.2) than for EITHER-FULL (66% for animals at noise-level 0.2). This confirms that EITHER-PRUNE avoids much of the over-fitting performed by EITHER-FULL. Ideally, a system's performance on training and test data should be the same in order to completely avoid over-fitting.

Since cover truncation prunes computation as well as theory changes, EITHER-PRUNE is

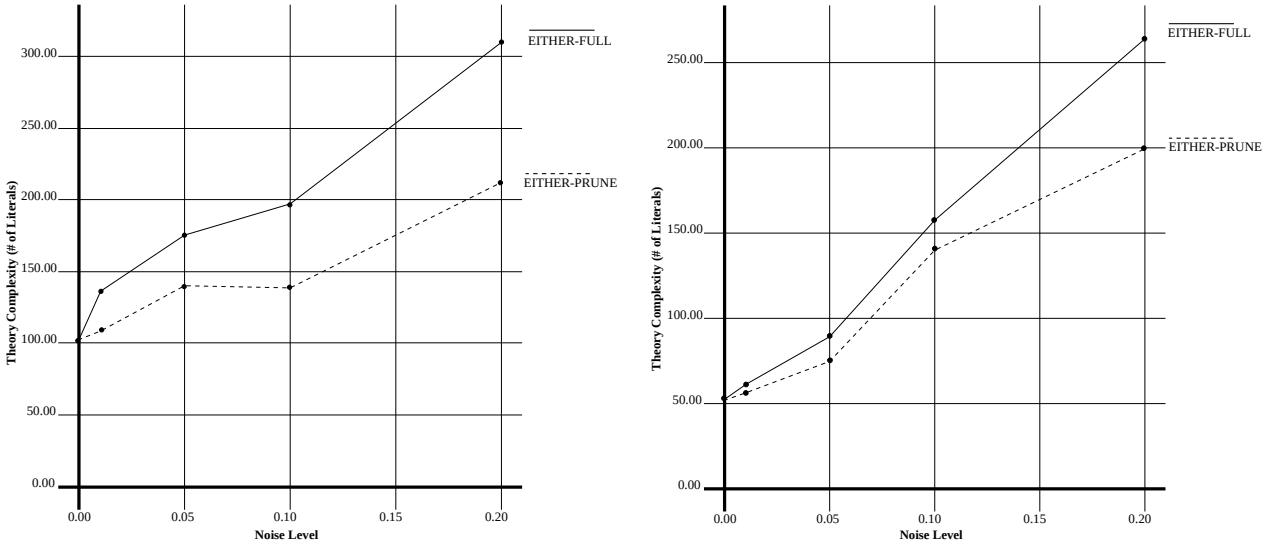

Figure 6: Theory Complexity vs. Noise Level for Animal and Computer Domains

also more efficient than EITHER-FULL. Figure 5 shows the training time for both systems as noise is increased. EITHER-PRUNE is clearly faster than EITHER-FULL and the difference increases with noise-level since more pruning is taking place at the higher noise levels.

Finally, cover truncation decreases the number of changes to the theory and therefore decreases the over-all complexity of the revised theory as well. Figure 6 shows the number of literals (i.e. total number of consequents and antecedents) in the revised theory as a function of noise-level. EITHER-PRUNE clearly produces simpler theories than EITHER-FULL and again the difference increases with noise-level due to increased pruning. However, the fact that EITHER-PRUNE's theory-complexity still increases with noise-level indicates that it is perhaps not doing enough pruning.

EITHER was also tested on a version of the DNA data used by [Towell *et al.*, 1990]. This data involves learning the concept of a *promoter*, a sequence of nucleotides that initiates the expression of a gene. An initial theory and data for this problem was assembled from the biological literature. Figure 7 shows accuracy vs. noise curves for this data. The results are averaged over 19 trials of 100 training examples and 100 test examples. EITHER-PRUNE only becomes significantly better than EITHER-FULL at a a noise level of 0.1. The differences at lower noise levels are not statistically significant. Both versions of EITHER perform significantly better than ID3. EITHER-PRUNE's performance degrades with noise at about the same rate as ID3's. EITHER-PRUNE's training time is consistently about 6% less than EITHER-FULL's.

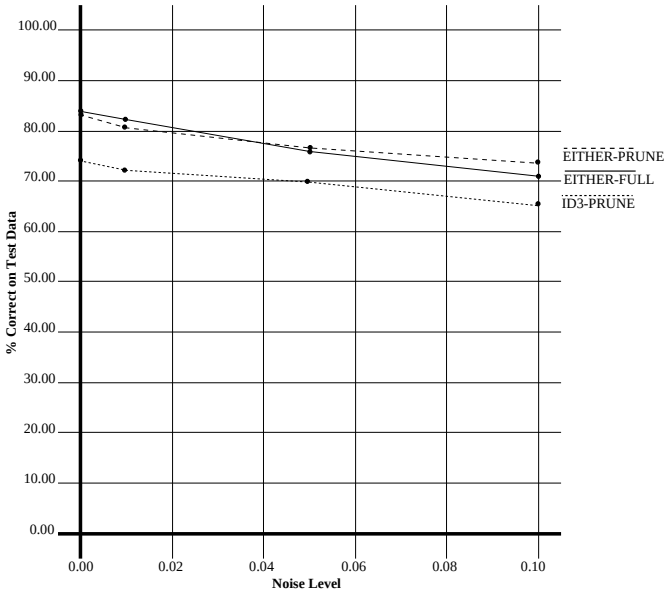

Figure 7: Test Accuracy vs. Noise Level for DNA Domain

6 Related Work

Most previous system that use imperfect domain theories to aid concept learning [Wilkins, 1988; Danyluk, 1989; Pazzani, 1989; Flann and Dietterich, 1989; Cohen, 1990; Ginsberg, 1990] have not dealt with the issue of noisy data. KBANN [Towell *et al.*, 1990], which converts a theory into a neural net and then refines it using backpropagation, has been shown to be able to handle noisy data. However, KBANN produces a neural-net classifier rather than a revised theory and does not perform any special noise processing to specifically avoid over-fitting. IVSM [Hirsh, 1989] uses an imperfect domain theory to aid learning and is noise tolerant; however, it cannot make use of arbitrarily imperfect theories and does not produce a revised domain theory.

The most closely related work within standard inductive learning is the cover truncation performed in AQ15 [Michalksi *et al.*, 1986]. However, AQ15's covers consist of disjuncts that match examples while EITHER's covers consist of theory changes that fix failing positive and negative examples. In addition, AQ15 prunes complete concept definitions after they were formed, while EITHER prunes potential theory changes before they are fully computed. Consequently, pruning in EITHER improves efficiency as well as accuracy and theory complexity.

7 Conclusions

In order to prevent over-fitting, a theory revision system must have methods to prevent complicated changes to a theory that only account for noisy instances. This paper has presented an effective method for dealing with noise during theory refinement. The basic approach is to avoid making changes to a theory that account for only a single data-point. This method has been incorporated into the EITHER theory refinement system and tested on artificially corrupted data. Experimental data supports the conclusion that this method is effective at preventing over-fitting and that it increases accuracy while decreasing both run-time and theory complexity.

8 Acknowledgements

We wish to thank Jude Shavlik and Michiel Noordewier for providing the DNA data and theory.

References

[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.

[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.

[Hirsh, 1989] H. Hirsh. Combining empirical and analytical learning with version spaces. In *Proceedings of the Sixth International Workshop on Machine Learning*, pages 29–33, Ithaca, NY, June 1989.

[Michalksi *et al.*, 1986] R.S. Michalksi, I. Mozetic, J. Hong, and N. Lavrac. The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 1041–1045, Philadelphia, PA, Aug 1986.

[Mingers, 1989] J. Mingers. An empirical comparison of pruning methods for decision tree induction. *Machine Learning*, 4(2):227–243, Nov 1989.

[Mitchell *et al.*, 1986] T. M. Mitchell, R. Keller, and S. Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine Learning*, 1(1):47–80, 1986.

[Ourston and Mooney, 1990a] 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.

[Ourston and Mooney, 1990b] D. Ourston and R. J. Mooney. Improving shared rules in multiple category domain theories. Technical Report AI90-150, Artificial Intelligence Laboratory, University of Texas, Austin, TX, December 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, 1986a] J. R. Quinlan. The effect of noise on concept learning. In R. S. Michalski, Jaime G. Carbonell, and Tom M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach, Volume II*, pages 149–166. Morgan Kaufman, 1986.

[Quinlan, 1986b] 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.

[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.