Appears in *Computational Learning Theory and Natural Learning Systems, Vol. 3,*
T. Petsche, S. Judd, and S. Hanson (Eds.), pp.43-53, MIT Press, 1995

A Preliminary PAC Analysis of Theory Revision *

Raymond J. Mooney

Department of Computer Sciences

University of Texas

Austin, TX 78712

mooney@cs.utexas.edu

October 13, 1993

Abstract

This paper presents a preliminary analysis of the sample complexity of theory revision within the framework of PAC (Probably Approximately Correct) learnability theory. By formalizing the notion that the initial theory is "close" to the correct theory we show that the sample complexity of an optimal propositional Horn-clause theory revision algorithm is $O((\ln 1/\delta + d \ln(s_0 + d + n))/\epsilon)$, where d is the *syntactic distance* between the initial and correct theories, s_0 is the size of initial theory, n is the number of observable features, and ϵ and δ are the standard PAC error and probability bounds. The paper also discusses the problems raised by the computational complexity of theory revision.

*This research was supported by the National Science Foundation under grant IRI-9102926, the NASA Ames Research Center under grant NCC 2-629, the Texas Advanced Research Program under grant 003658114.

1 Introduction

Although there has recently been a great deal of empirical work on constraining learning with prior knowledge (Segre, 1989; Birnbaum and Collins, 1991), there has been relatively little theoretical analysis of the problem. One way to use prior knowledge to bias learning is to revise an existing imperfect domain theory to fit empirical data. This approach has important applications to automatically refining knowledge bases for expert systems (Ginsberg et al., 1988). This paper presents a preliminary analysis of theory revision within the framework of PAC (probably approximately correct) learnability theory (Valiant, 1984). Specifically, this paper analyzes the sample complexity (the number of examples required to learn a PAC concept) of propositional Horn-clause theory revision. A number of recent systems modify an existing incorrect/incomplete propositional Horn-clause theory to fit a set of preclassified training examples (Ourston and Mooney, 1990; Ginsberg, 1990; Cain, 1991; Matwin and Plante, 1991).

Empirical results in several artificial and real-world domains have shown that revising an approximate theory results in more accurate definitions from fewer examples than pure induction (Ourston, 1991; Towell, 1991). For example, Figure 1 shows learning curves for revising a theory for recognizing promoter sequences in DNA, a problem introduced by (Towell et al., 1990). It compares classification accuracy on novel test data for the EITHER theory refinement system (Ourston and Mooney, 1990; Ourston, 1991; Mooney and Ourston, in press) and the ID3 inductive system (Quinlan, 1986). Since EITHER uses ID3 as an inductive component, EITHER's performance without an initial theory is the same as ID3's. It clearly show the advantage of theory refinement over pure induction. Although the theoretical results presented in this paper are not directly applicable to existing implemented systems, they provide some theoretical insight into why and when theory revision systems learn from fewer examples.

The basic approach to analyzing theory revision involves formalizing the notion that the initial theory is "close" to the correct theory. We introduce a notion of *syntactic distance* between two theories based on the the number of primitive modifications needed to transform

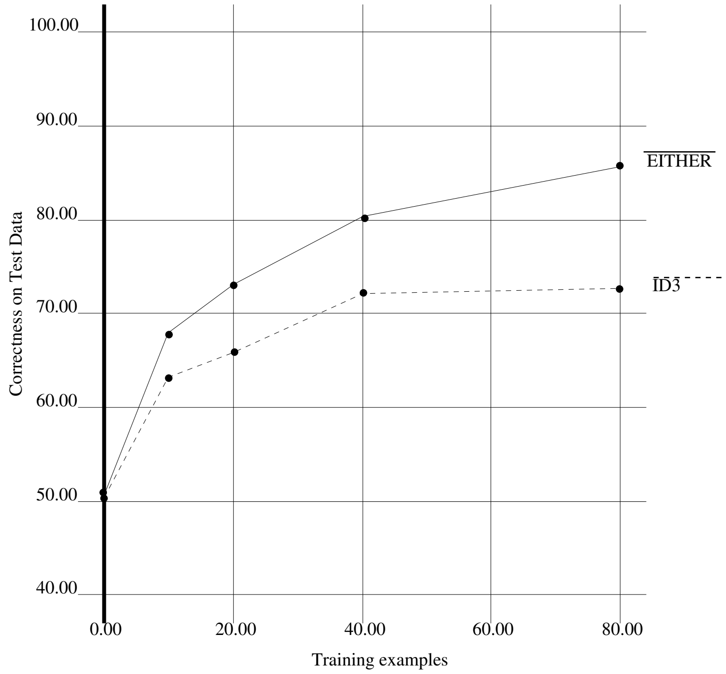


Figure 1: Learning Curves for the DNA Promoter Problem

one theory into another. We show that an upper bound on the number of examples needed to transform an initial propositional theory into a PAC theory is $O(d\log(d))$ in the syntactic distance d between the initial and correct theories. We also discuss the computational complexity of theory revision and what is needed to extend these results to apply to realistic theory revision algorithms.

2 A Relevant Result from PAC Learnability

PAC learnability theory as originated by Valiant (1984) is primarily concerned with determining the number of examples required by a learning algorithm to guarantee that with probability $1-\delta$ the concept description produced by the algorithm has an error rate of at most ϵ.

In this paper, we will use Haussler's notion of the *sample complexity* of a learning algorithm L for a hypothesis space [1] H defined on an instance space X, denoted $S^L_H(\epsilon, \delta)$,

[1] A hypothesis space is also frequently referred to as a *concept class*. Typical examples are pure conjunctive,

which is the minimum number of examples m such that for any target concept $h \in H$ and any distribution on X, given m random examples of h, L produces a hypothesis that, with probability at least $1 - \delta$, has error at most ϵ (Haussler, 1988).

A learning algorithm L is said to *use a hypothesis space H consistently* if it always returns a hypothesis in H that is consistent with the training set or else correctly indicates that no hypothesis in H is consistent with the given examples. A general result is that for any hypothesis space H and any learning algorithm L that uses H consistently: [2]

$$S_H^L(\epsilon, \delta) \leq \frac{1}{\epsilon}(\ln \frac{1}{\delta} + \ln |H|). \tag{1}$$

This result clearly indicates that the number of examples needed to learn a PAC concept description is directly related to the size of the restricted space of concepts in which the target concept is known to belong, and hence formalizes the notion of inductive bias (Haussler, 1988).

3 An Approach to PAC Analysis of Theory Revision

In order to analyze the sample complexity of a theory revision algorithm, we need to formally specify the restricted space of hypotheses that are explored by such an algorithm. The fundamental assumption of theory revision is that although the initial domain theory is flawed, it is still relatively "close" to the correct theory. Therefore, the restricted class of hypotheses explored by theory revision is probably best formalized as those that are within a limited "distance" of the initial theory.

Theory revision systems generally syntactically modify the initial theory to make it fit the training data. Sample modifications include adding and deleting rules and their antecedents (Ourston and Mooney, 1990). Therefore, one way of measuring the distance between two theories is to determine the minimum number of primitive syntactic modifications needed to transform one theory into the other. The notion that the initial theory is "close" to the

DNF, k-DNF, etc..

[2]This is a direct consequence of a result in Blumer et al. (1987).

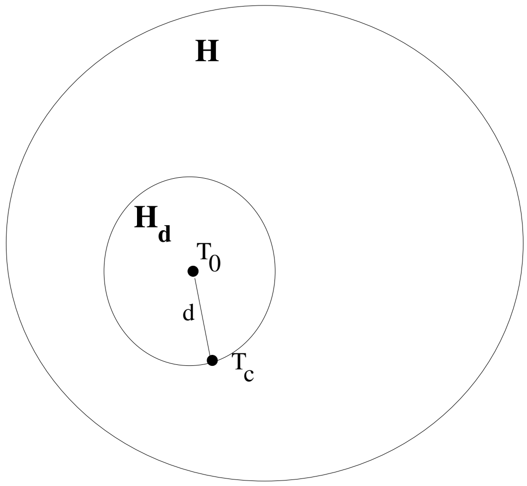


Figure 2: Restricted Hypothesis Space for Theory Revision

correct one can then be captured by assuming that the *syntactic distance* between the two theories is less than some value, d. The hypothesis space can then be limited to all theories within a distance d of the initial theory.

This basic idea is illustrated in Figure 2, where H is the overall hypothesis space (e.g. propositional Horn-clause theories), T_0 is the initial theory, T_c is the correct theory (which is a syntactic distance d from the initial theory), and H_d is the space of all theories within a syntactic distance d of the initial theory. If a learning algorithm is guaranteed to find the closest theory to T_0 that is consistent with the training data, then its hypothesis space can be considered to be H_d.

Therefore, in order to analyze the sample complexity of theory revision, we need to formalize the notion of syntactic distance and determine the size of the corresponding hypothesis space H_d. In the following section, we do this analysis for the case of propositional Horn-clause theories.

4 Sample Complexity of Propositional Revision

As previously mentioned, propositional Horn-clauses are used to represent theories in several recent systems. We will assume that a theory is a set of propositional Horn-clauses for an

C	$\leftarrow$	stable $\wedge$ liftable $\wedge$ open-vessel
stable	$\leftarrow$	has-bottom $\wedge$ flat-bottom
liftable	$\leftarrow$	graspable $\wedge$ lightweight
graspable	$\leftarrow$	has-handle
graspable	$\leftarrow$	width-small $\wedge$ styrofoam
open-vessel	$\leftarrow$	has-concavity $\wedge$ upward-pointing-concavity

Features: has-bottom, flat-bottom, lightweight, has-handle, width-small, styrofoam, has-concavity, upward-pointing-concavity

Table 1: The Cup Theory

instance space defined over n propositional features. A Horn clause is a disjunction of literals, of which at most one is positive. They are normally written in the form of backward-chaining rules (as in Prolog). An example of a propositional Horn-clause theory for the standard "cup" concept (Winston et al., 1983) is shown in Figure 1. We further assume that the theory is acyclic, i.e. the directed graph constructed by adding a directed edge from every antecedent of a rule to its consequent (a_i to c for every rule $c \leftarrow a_1 \wedge \cdots \wedge a_k$) is acyclic. A theory also contains a special literal C representing the concept to be learned. An instance is classified as a positive example of the target concept if and only if, given its features as facts, C can be derived using normal Prolog-style deduction (SLD resolution). A closed-world assumption is also made (Clark, 1978), so that if C cannot be derived, an example is classified as negative ($\neg C$).

For propositional Horn-clause theories, a simple set of primitive syntactic modifications is the addition and deletion of individual literals. The syntactic distance between two propositional Horn-clause theories is then defined as the minimum number of single-literal revisions needed to transform one theory into the other. Literal deletion and addition can have a variety of semantic effects. The deletion of individual antecedents is a straightforward generalizing operator. With respect to determining whether an example is a member of the goal concept using Horn-clause deduction, deleting a consequent has the same effect as deleting the rule. However, when viewed in clause form, there is a difference. Deleting the consequent of a rule $c \leftarrow a_1 \wedge \cdots \wedge a_k$ leaves the clause $\neg a_1 \vee \neg a_2 \vee \cdots \vee \neg a_k$. To truly delete a rule, one must individually delete all of its literals. Adding a literal to the antecedent of a clause is a meaningful specializing operator with a clear semantic effect. We assume that a new clause must be generated by first adding a literal for its consequent. Subsequent modifications can then add antecedents to this initially "empty" rule.

In order to determine the number of theories within a syntactic distance d of the initial theory, we first need to determine the number of possible primitive modifications that can be made to a theory at any given point, i.e. the effective branching factor. Assume the current theory contains a total of s literals. Therefore, the number of possible literal deletions is also s. The maximum number of new clauses that can be begun by adding a consequent literal is $s+1$ since one can start a new rule for one of the existing literals in the theory or begin a rule for a new literal. The maximum number of ways of adding a new antecedent to an existing rule is the number of possible antecedents that can be added times the number of rules to which they can be added. Clearly, there are at most s rules in a theory with s literals. The added literal can either be an existing literal in the theory (at most s possibilities) or one of the literals used to describe examples. Consequently, there are at most $s + n$ literals that can be added, where n is the number of observable (operational) features. We assume that in order to add a brand new literal as an antecedent, we must have previously introduced it as a consequent of a rule. This information is summarized in Table 2. Consequently, the maximum number of single-literal revisions to a theory of size s is:

$$s^2 + (n + 2)s + 1 \tag{2}$$

Since each revision adds at most one literal to the theory, the size of the theory after i revisions (s_i) is at most $s_0 + i$, where s_0 is the size of the original theory. Since the size of the theory changes after each revision, the branching factor also changes. Taking this into account, a bound on the number of possible revised theories after a total of d single-literal

Table 2: Number of Possible Revisions

Revision Type	Maximum Number of Revisions
Delete Literal	s
Add Rule	$s+1$
Add Antecedent	$s(s+n)$
Total	$s^2+(n+2)s+1$

revisions (R_d) is:

$$R_d \leq \prod_{i=0}^{d-1}(s_0+i)^2+(n+2)(s_0+i)+1 \leq [(s_0+d-1)^2+(n+2)(s_0+d-1)+1]^d. \quad (3)$$

Let H_d represent the hypothesis space of all theories that can be constructed with d or fewer single-literal revisions. Summing the number of revised theories generated by d *or fewer* revisions we get:

$$|H_d| \leq \sum_{i=1}^{d} R_i \leq dR_d \leq d[(s_0+d-1)^2+(n+2)(s_0+d-1)+1]^d. \quad (4)$$

Substituting this result into equation 1 we obtain the following upper bound on the sample complexity of any learning algorithm L that uses H_d consistently:

$$S^L_{H_d}(\epsilon,\delta) \leq \frac{1}{\epsilon}[\ln\frac{1}{\delta}+\ln d+d\ln((s_0+d-1)^2+(n+2)(s_0+d-1)+1)]. \quad (5)$$

Simplifying this result using order notation we get:

$$S^L_{H_d}(\epsilon,\delta) = O(\frac{1}{\epsilon}[\ln\frac{1}{\delta}+d\ln(s_0+d+n)]). \quad (6)$$

Assuming the correct theory is within a distance d of the initial theory, a propositional Horn-clause theory revision algorithm that always finds the closest theory to the initial theory that is consistent with the training data uses H_d consistently. This is because there is guaranteed to be at least one theory within a distance d of the initial theory that is consistent with the training data, namely, the correct theory. Consequently, the sample complexity of such an algorithm is given by Equation 6 which is $O(d\log(d))$ in the distance between the initial theory and the correct theory, and logarithmic in the size of the initial theory and the number of features.

This result is also useful for comparing theory revision to inducing a theory from scratch. Since revising an empty theory ($s_0 = 0$) is the same as complete induction, equation 6 also applies in this case. If s_c is the size of the correct theory, then an algorithm L that uses H_d consistently (e.g. an algorithm that always finds the simplest theory consistent with the training data) has a sample complexity of:

$$S^L_{H_d}(\epsilon, \delta) = O(\frac{1}{\epsilon}[\ln\frac{1}{\delta} + s_c \ln(s_c + n)]) \quad (7)$$

since the theory can obviously be constructed with s_c literal additions, and is therefore a distance s_c from the empty theory. Therefore, if the distance from the initial theory to the correct theory is less than the size of the correct theory, the upper-bound on the number of examples needed to learn a PAC concept is lower for theory refinement than that for pure induction. Of course, a guarantee that the sample complexity of theory refinement is lower would require a lower bound on the sample complexity of pure induction of theories with at most s_c literals. An analysis of the *VC-dimension* of this hypothesis space, would provide such a lower bound (Ehrenfeucht et al., 1989).

The above results are closely related to previous results on the sample complexity of learning concepts representable with a limited number of bits (Blumer et al., 1987). With respect to pure induction, the term $s_c \ln(s_c + n)$ in Equation 7 is proportional to the number of bits needed to represent a theory with s_c literals. Since there are at most $s_c + n$ possible literals to pick from (n observables plus at most s_c non-observables), $O(\ln(s_c + n))$ bits are needed to represent each literal. With respect to theory revision, the term $d \ln(s_0 + d + n)$ is proportional to the number of bits needed to encode a theory as a list of changes to the initial theory. Therefore another way of stating the basic result is: If it is simpler to encode a theory as a list of changes to the initial theory than to directly encode its content, then the upper-bound on the number of examples needed to learn a PAC concept is lower for theory refinement than that for pure induction.

5 Computational Complexity of Theory Revision

Unfortunately, the above analysis of sample complexity cannot be directly applied to current theory-revision systems since it does not address the problem of computational complexity. Because of computational issues, existing systems are not guaranteed to produce the closest theory to the initial theory that is consistent with the training data. The problem of finding a minimally-revised theory is a difficult optimization problem for which there is no known polynomial-time algorithm. As previously mentioned, by using an empty initial theory, the problem of finding a minimum propositional Horn-clause theory for a set of data is easily reduced to propositional theory revision. Consequently, implemented theory revision systems use various heuristic methods to minimize change but do not guarantee optimal results.

Of course, if we assume that the initial theory is always within a fixed distance of the correct theory, i.e. d is a constant, then exhaustive search of the space H_d can produce the minimally revised theory in time polynomial in the size of the initial theory (s_0) and the number of features (n) (see equation 4). However, this is only practical for very small values of d.

In addition, it is well known result that one does not need to find the absolute minimum hypothesis in order to guarantee polynomial sample complexity. If an algorithm is guaranteed to find a consistent hypothesis that is within a polynomial factor of the simplest one, its sample complexity is still polynomial (Blumer et al., 1987). In particular, a greedy algorithm for simple conjunctive concepts is known to find a hypothesis that is within a logarithmic factor of optimal (Haussler, 1988). Consequently, its sample and computational complexity are both $O(s\log(n))$ where s is the size of the concept to be learned. The sample complexity of such approximation algorithms is determined by the size of their *effective hypothesis space*. The effective hypothesis space for a learning algorithm L for a concept class C, denoted $H^L_C(m)$, is the set of all hypotheses produced by L from samples of size m of target concepts in C (Haussler, 1988). The effective hypothesis space can be used in place of H in Equation 1 to obtain a bound on the sample complexity of algorithms that use a preference bias instead of a language bias.

Consequently, if a theory revision algorithm was guaranteed to produced a consistent theory that was within a distance d^k of the initial theory, where k is some constant and d is the distance to the closest consistent theory (i.e. the theory is within a polynomial factor of optimal), then its effective hypothesis space would be bounded by:

$$|H_C^L(m)| \leq d^k[(s_0 + d^k)^2 + (n+2)(s_0 + d^k) + 1]^{d^k}, \tag{8}$$

and it would therefore have a sample complexity of:

$$S_H^L(\epsilon, \delta) = O(\frac{1}{\epsilon}[\ln \frac{1}{\delta} + d^k \ln(s_0 + d + n)]). \tag{9}$$

Unfortunately, there is no known polynomial-time algorithm that guarantees a revised theory within a polynomial factor of optimal. This is a common problem in machine learning – polynomial approximation algorithms for minimum decision trees and minimum three-layer neural networks are also open problems (Shavlik and Dietterich, 1990). Until such algorithms can be found, the primary support for the efficacy of real systems will probably remain empirical in nature.

6 Syntactic versus Semantic Criteria

This paper has assumed that the "goodness" of an initial theory or a proposed revision is based purely on syntactic criteria. However, minimally modifying the syntax of a theory to fit the training data is not always the best approach. For example, assume the training set contains only positive examples of the target concept C. In this case, adding the empty rule $C \leftarrow$, stating that everything is positive, is normally the syntactically minimal revision; however, it completely destroys the semantics of the theory and is therefore probably not the best revision. Note that this situation violates one of the basic assumptions of PAC analysis; namely, that the distribution of examples is the same during training and testing. However, in many situations (e.g. language acquisition), learning must take place primarily from positive data.

On the other hand, minimally modifying the semantics of a theory is also inadequate. If a theory is revised to minimally alter its extension, it never generalizes and simply memorizes exceptions. For example, if the theory does not cover a positive example whose complete feature description is $f_1, \ldots f_n$, then the minimal semantic change is to add the rule $C \leftarrow f_1 \wedge \cdots \wedge f_n$, which changes the extension of C just enough to cover this specific example. Further evidence that purely semantic criteria do not adequately measure the goodness of a theory is the observation that many useful initial theories have very poor accuracy. For example, the DNA promoter theory used in many recent experiments in theory revision (Towell et al., 1990) has an accuracy no better than random chance. Despite this fact, it is syntactically close to the correct theory and revising it produces a more accurate theory than pure induction.

Consequently, it seems the characterization of the ideal revision must incorporate both semantic and syntactic criteria. Current systems use various heuristics to resolve this trade-off, but a completely satisfactory formal characterization of the properties of an optimally revised theory is still an elusive goal.

7 Conclusions

In this paper, we have presented a general approach to analyzing the sample complexity of theory revision algorithms. In this approach, the notion that the initial theory is "close" to the correct one is formalized in terms of syntactic distance, the number of primitive modifications needed to transform one theory into another. It was shown that an optimal propositional Horn-clause theory revision algorithm, i.e. one that produces the closest consistent theory, has a sample complexity that is $O(d \log d)$ in the syntactic distance d between the initial and correct theories. Unfortunately, optimal theory revision is computationally intractable and polynomial approximation algorithms are needed.

References

Birnbaum, L. A., and Collins, G. C., editors (1991). *Proceedings of the Eighth International Workshop on Machine Learning: Section on Learning From Theory and Data*. Evanston, IL.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. (1987). Occam's razor. *Information Processing Letters*, 24:377–380.

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

Clark, K. (1978). Negation as failure. In Gallaire, H., and Minker, J., editors, *Logic and Data Bases*. New York, NY: Plenum Press.

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

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

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

Haussler, D. (1988). Quantifying inductive bias: Artificial intelligence learning algorithms and Valiant's learning framework. *Artificial Intelligence*, 26:177–221.

Matwin, S., and Plante, B. (1991). A deductive-inductive method for theory revision. In *Proceedings of the International Workshop on Multistrategy Learning*, 160–174. Harper's Ferry, W.Va.

Mooney, R., and Ourston, D. (in press). A multistrategy approach to theory refinement. In Michalski, R. S., and Teccuci, G., editors, *Machine Learning: A Multistrategy Approach, Vol. IV.* San Mateo, CA: Morgan Kaufman.

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

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

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

Segre, A., editor (1989). *Proceedings of the Sixth International Workshop on Machine Learning: Section on Combining Empirical and Explanation-Based Learning.* Ithaca, NY.

Shavlik, J., and Dietterich, T. (1990). Inductive learning from preclassified examples: Introduction. In Shavlik, J., and Dietterich, T., editors, *Readings in Machine Learning*, 45–56. San Mateo, CA: Morgan Kaufman.

Towell, G. G. (1991). *Symbolic Knowledge and Neural Networks: Insertion, Refinement, and Extraction.* PhD thesis, University of Wisconsin, Madison, WI.

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

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

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

Footnotes

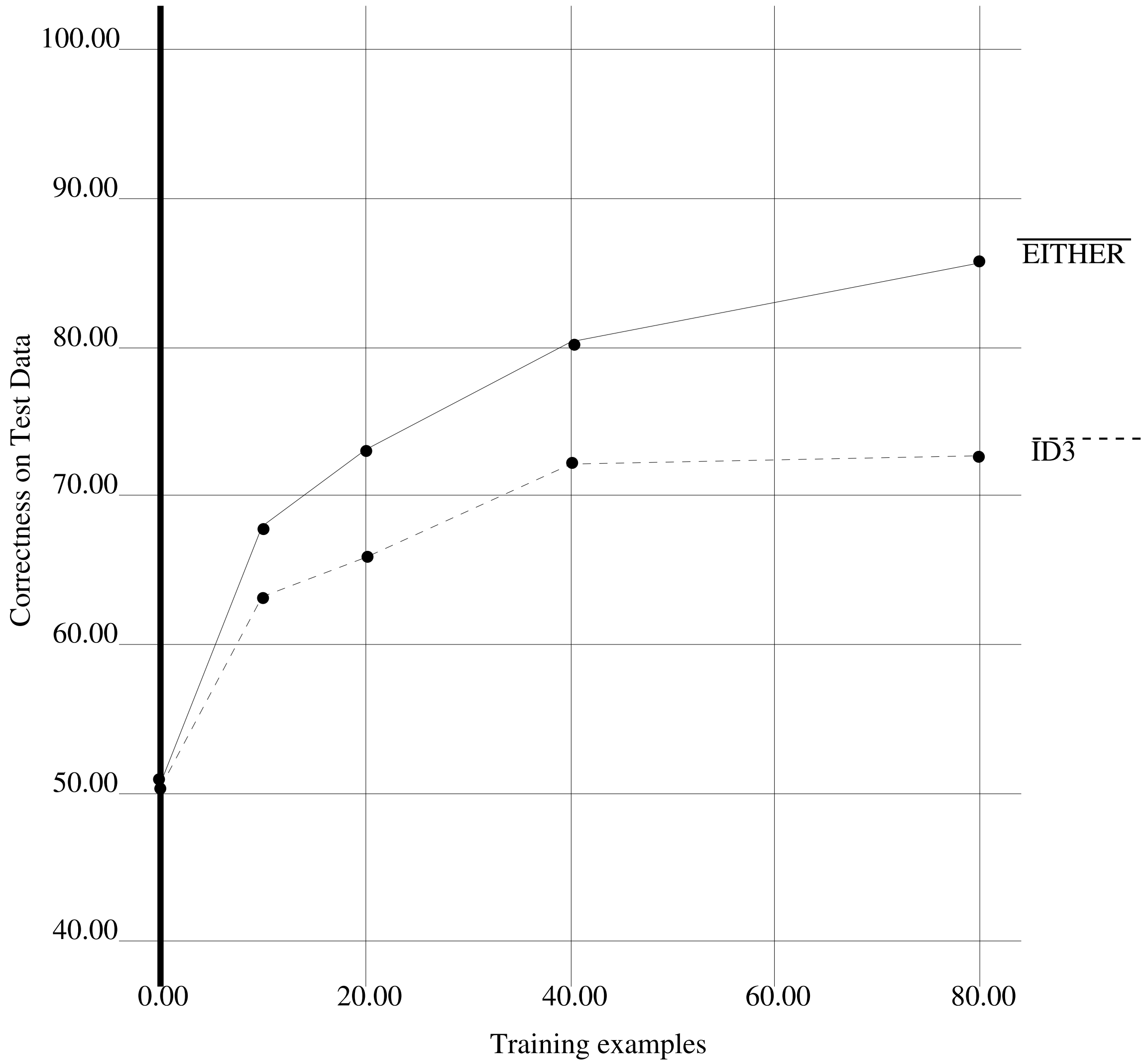
1. A hypothesis space is also frequently referred to as a *concept class*. Typical examples are pure conjunctive, DNF, k-DNF, etc..

2. This is a direct consequence of a result in Blumer et al. (1987).

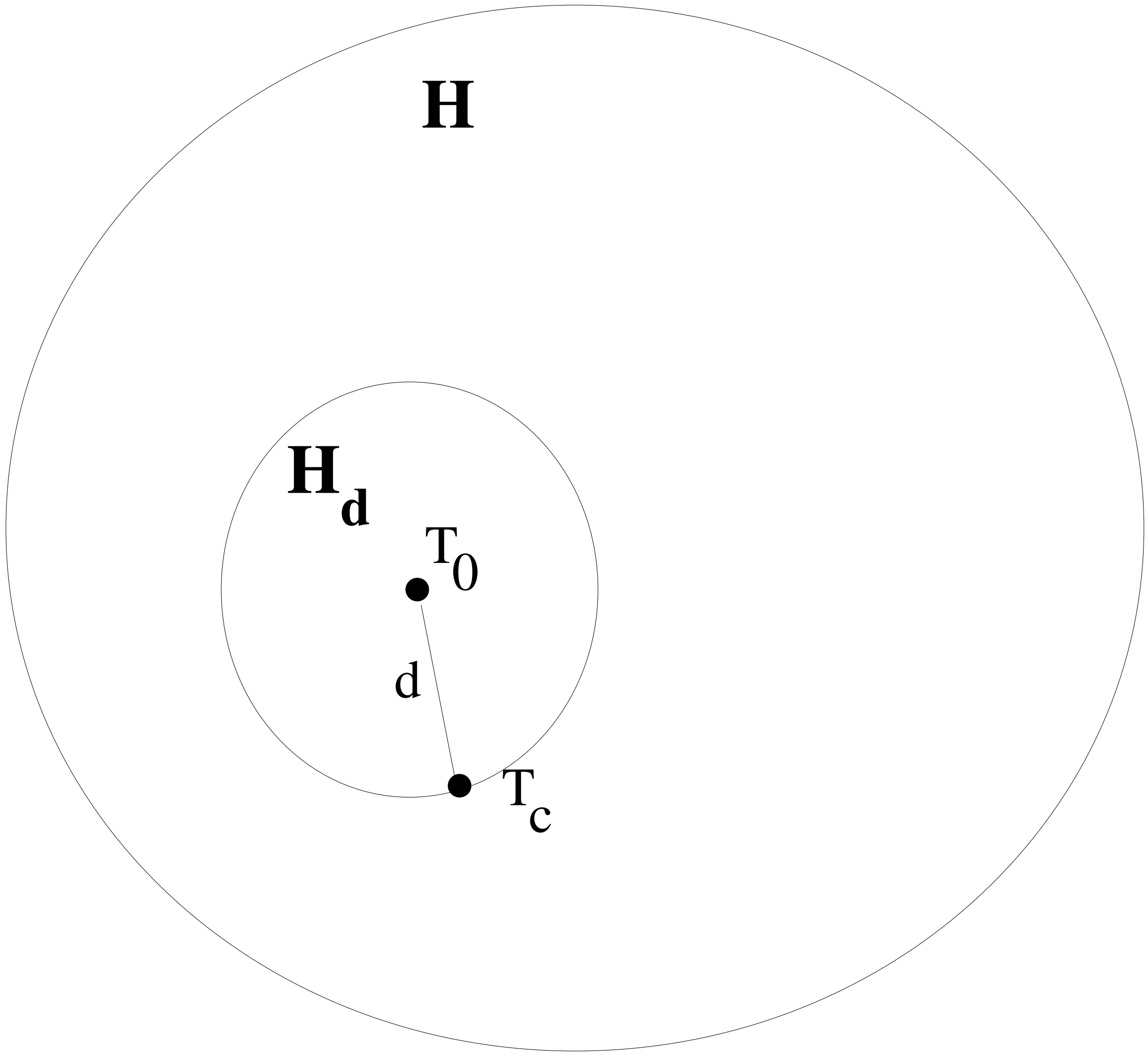
Figure Captions

- Figure 1: Learning Curves for the DNA Promoter Problem.

- Figure 2: Restricted Hypothesis Space for Theory Revision.



100.00
90.00
80.00
70.00
60.00
50.00
40.00
Correctness on Test Data
0.00
20.00
40.00
60.00
80.00
Training examples
EITHER
ID3


H
H
d
T
0
d
T
c