Machine Learning, 41, 123–152, 2000

Maximizing Theory Accuracy Through Selective Reinterpretation

SHLOMO ARGAMON-ENGELSON argamon@mail.jct.ac.il
Department of Computer Science, Jerusalem College of Technology, Machon Lev, P.O.B. 16031, 91160 Jerusalem, Israel

MOSHE KOPPEL koppel@cs.biu.ac.il
HILLEL WALTERS
Department of Mathematics and Computer Science, Bar-Ilan University, 52900 Ramat Gan, Israel

Editor: Raymond J. Mooney

Abstract. Existing methods for exploiting flawed domain theories depend on the use of a sufficiently large set of training examples for diagnosing and repairing flaws in the theory. In this paper, we offer a method of theory reinterpretation that makes only marginal use of training examples. The idea is as follows: Often a small number of flaws in a theory can completely destroy the theory's classification accuracy. Yet it is clear that valuable information is available even from such flawed theories. For example, an instance with several independent proofs in a slightly flawed theory is certainly more likely to be correctly classified as positive than an instance with only a single proof.

This idea can be generalized to a numerical notion of "degree of provedness" which measures the robustness of proofs or refutations for a given instance. This "degree of provedness" can be easily computed using a "soft" interpretation of the theory. Given a ranking of instances based on the values so obtained, all that is required to classify instances is to determine some cutoff threshold above which instances are classified as positive. Such a threshold can be determined on the basis of a small set of training examples.

For theories with a few localized flaws, we improve the method by "rehardening": interpreting only parts of the theory softly, while interpreting the rest of the theory in the usual manner. Isolating those parts of the theory that should be interpreted softly can be done on the basis of a small number of training examples.

Softening, with or without rehardening, can be used by itself as a quick way of handling theories with suspected flaws where few training examples are available. Additionally softening and rehardening can be used in conjunction with other methods as a meta-algorithm for determining which theory revision methods are appropriate for a given theory.

Keywords: logical theories, theory revision, probabilistic theories, flawed domain theories, approximate reasoning, machine learning

1. Introduction

A central concern of machine learning research is how to use prior knowledge effectively to provide a useful learning bias. An important type of prior knowledge that may thus be used is a flawed domain theory, obtained from some domain expert by knowledge engineering. One of the main methods for using such a theory has been to attempt to *revise* it in order to improve its classification accuracy (Saitta, Botta, & Neri, 1993; Towell

& Shavlik, 1993; Cohen, 1994; Koppel, Feldman, & Segre, 1994a; Ourston & Mooney, 1994). Although this idea has great intuitive appeal, revision is not always the best way to use a given theory. Another class of methods does not attempt to repair the given theory, but to *reinterpret* it in a more profitable manner. This can be done by using the theory as a resource for constructive induction (Pazzani & Kibler, 1992; Donoho & Rendell, 1995; Ortega & Fisher, 1995; Koppel & Engelson, 1996), or by numerical refinement of probabilistic theories (Mahoney & Mooney, 1994; Mahoney, 1996; Buntine, 1991; Lam & Bacchus, 1994; Russell et al., 1995; Ramachandran & Mooney, 1998), or, most relevant to this paper, by interpreting a logical theory in a probabilistic manner (Towell & Shavlik, 1993; Koppel, Feldman, & Segre, 1994b; Ortega, 1995).

All of these methods depend on the use of a sufficiently large set of training examples for diagnosing and repairing flaws in the theory. In this paper, we offer a method of theory reinterpretation that makes only marginal use of training examples. In the simplest version of the method, examples are required only in order to approximate the number of positive and negative instances. In a more sophisticated version of our method, which selectively reinterprets the theory on the basis of training examples, empirical evidence indicates that a very small training set is sufficient.

The idea is to squeeze out as much reliable information as possible from an unreliable theory prior to invoking the information contained in training examples. The central observation is as follows: Often a small number of flaws in a theory can completely destroy the theory's classification accuracy. For example, one easily satisfied extra clause near the root of a theory can render all instances positive, ostensibly destroying the theory. Yet it is clear that valuable information is available even from such flawed theories. For example, an instance with several independent proofs in the theory is certainly more likely to be correctly classified as positive than an instance with only a single proof. This idea can be generalized to an easily computed numerical notion of "degree of provedness" which measures the robustness of proofs or refutations for a given instance. That is, instead of interpreting a theory in the usual Boolean manner, we interpret it "softly", assigning each instance a "degree of provedness" value between 0 and 1. Given a ranking of instances based on the values so obtained, all that is required to classify is to determine some cutoff threshold above which instances are classified as positive. Such a threshold can be determined on the basis of a small set of training examples. (In fact, it might even be enough for this purpose to know the approximate number of positive examples in some set of examples without actually knowing the correct classification of any single example.)

We will see that interpreting a theory softly is a remarkably effective method for classifying examples despite the presence of flaws. Moreover, the method is benign in that in the case of an unflawed theory it does no harm. For theories with a few localized flaws, we improve the method by "rehardening": interpreting only parts of the theory softly, while interpreting the rest of the theory in the usual manner. Isolating those parts of the theory that should be interpreted softly can be done on the basis of a small number of training examples.

Softening, with or without rehardening, can be used by itself as a quick way of handling theories with suspected flaws where few training examples are available. Additionally softening and rehardening may be used in conjunction with other methods as a meta-algorithm for determining which theory revision methods are appropriate for a given theory. In particular, this method can be used to determine whether the theory has localized flaws which should be revised, distributed flaws requiring reinterpretation, or whether the theory contains no useful information and should not be used at all as a learning bias. When revision is deemed appropriate, rehardening can offer suggestions as to which components of the theory ought to be the focus of repair.

The outline of this paper is as follows: In Section 2, we explain and justify the soft interpretation of theories and in Section 3 we show how to use softening to classify instances. In Section 4 we explain and justify the technique of rehardening. In Section 5, we illustrate how the methods work on several well-known theories and in Section 6 we give the results of tests of these methods on a large testbed of synthetically generated flawed theories. In the appendix, we offer proofs of some analytic claims concerning the connection between our measure of degree of provedness and the actual robustness of proofs and refutations.

2. Softening logical theories

2.1. *Logical provedness*

We consider here the case of propositional theories expressed in definite-clause form, with negation-as-failure. Each clause's head is a positive literal, and its body is a conjunction of positive and negative literals. We assume the concept to be learned is represented by a unique 'root' proposition, which does not appear in the body of any clause.

In this section we review the theory probabilization method described in Koppel, Feldman, and Segre (1994b), which serves as the basis for the current work. We first review the standard method for computing a function which is 1 if an example E is proved in the propositional theory Γ and 0 otherwise. In the next section we will extend the function to take on values between 0 and 1, measuring a relative notion of example 'provedness'.

For each observable proposition P, define

$$u'(E, P, \Gamma) = \begin{cases} 0 & \text{if } P \text{ is false in } E \\ 1 & \text{if } P \text{ is true in } E \end{cases}$$

For each clause C with antecedents $l_1, \ldots, l_n$, let

$$u'(E, C, \Gamma) = \prod_{i=1}^{n} u'(E, l_i, \Gamma)$$

Similarly, for each non-observable proposition P which is the head of clauses $C_1, \ldots, C_n$, let

$$u'(E, P, \Gamma) = 1 - \prod_{i=1}^{n} (1 - u'(E, C_i, \Gamma))$$

And finally, for each negated proposition $\neg P$, let

$$u'(E, \neg P, \Gamma) = 1 - u'(E, P, \Gamma)$$

These formulae are simply arithmetic forms of the boolean functions AND, OR, and NOT, respectively.

This formulation can be simplified by reformulating the theory in terms of NAND relations. Define the children of a proposition to be the clauses for which it is a head, the children of a clause to be its antecedent literals, and the children of a negated proposition to be the unnegated proposition. For each primitive proposition k, we define $u(E, k, \Gamma) = u'(E, k, \Gamma)$. For a component (proposition, clause, or negative literal) k with children $c_1, \ldots, c_n$, define

$$u(E, k, \Gamma) = 1 - \prod_{i=1}^{n} u(E, c_i, \Gamma)$$

Since ANDs and ORs strictly alternate, we have for every proposition P that $u(E, P, \Gamma) = u'(E, P, \Gamma)$ for every example E. In particular, if r is the root proposition of Γ, $u(E, r, \Gamma) = u'(E, r, \Gamma)$. Thus E is proved in Γ exactly when $u(E, r, \Gamma) = 1$.

2.2. *Soft provedness*

As defined, $u(E, r, \Gamma)$ can only assume the values 0 or 1, Γ either proves or refutes r given E. However, since Γ is assumed to be flawed, we would like to evaluate more precisely *to what degree* an example is proved in the theory.

Consider, for example, the theory:

$$\begin{aligned} r &\leftarrow a \\ r &\leftarrow b, c \\ r &\leftarrow d, e \end{aligned}$$

and three examples proved in the theory: $E_1 = \{a, b, c, d, e\}$, $E_2 = \{a, b, d\}$, and $E_3 = \{a\}$. Although $u(E_1, r, \Gamma) = u(E_2, r, \Gamma) = u(E_3, r, \Gamma) = 1$, their intuitive 'degree of provedness' varies. E_1 can be considered 'proved to a greater degree' than E_2, since it has three proofs to E_2's one. Furthermore, although both E_2 and E_3 have one proof each, E_2 also has two 'near proofs' and so can be thought of as 'proved to a greater degree' than E_3. That is, if we had reason to believe that the theory might be slightly flawed and that the classification as positive of some of these examples might therefore be mistaken, suspicion ought to fall most readily on E_3, since reclassifying E_3 would require doing the least violence to the theory.

What we want, therefore, is a relative measure of 'degree of proof'. We extend here the definition of u in such a way that it can assume values between 0 and 1 that correspond to our intuitive notion of 'degree of proof'. This has the effect of *softening* Γ's classifications.

Let ϵ be some small value greater than 0. Now, similar to the development above, for each observable proposition P define the *softening function* u_ϵ by:

$$u_\epsilon(E, P, \Gamma) = \begin{cases} \epsilon & \text{if } P \text{ is false in } E \\ 1 & \text{if } P \text{ is true in } E \end{cases}$$

For each component k of Γ with children $c_1, \ldots, c_n$, define

$$u_\epsilon(E, k, \Gamma) = 1 - (1 - \epsilon) \prod_{i=1}^{n} u_\epsilon(E, c_i, \Gamma)$$

The term $(1 - \epsilon)$ can be thought of as introducing uncertainty into the theory by placing a probability measure on subtheories of Γ, such that each component has independent probability of ϵ to be deleted. (Thus the asymmetry in $u_\epsilon(E, P, \Gamma)$; deleting a false proposition may cause its clause to become true, but not vice versa.) The computation of $u_\epsilon(E, r, \Gamma)$ approximates the expected classification of E over this measure (see the appendix). (Note that these component 'weights' represent a meta-theory concept, giving a probability measure over possible *theories*, and not, as in Bayesian networks (Pearl, 1988), conditional probabilities of results given premises.)

In this way, $u_\epsilon(E, r, \Gamma)$ provides a useful measure of the resilience of E's classification to changes in the theory. In particular, as discussed in Section 4.4 below, for sufficiently small ϵ, $u_\epsilon(E, r, \Gamma)$ reflects the minimal number of components in Γ which would need to be revised in order to change E's classification (E's *revision distance*, defined more precisely below).

In the example above, E_1 has the highest revision distance at 3, whereas E_2 and E_3 both have revision distance 1, reflecting our intuitive notion that E_1 is more strongly classified as positive by the theory than the other two examples. A more fine-grained measure is given by u_ϵ, however. Indeed, $u_{0.1}(E_1, r) = 0.999$, $u_{0.1}(E_2, r) = 0.925$, and $u_{0.1}(E_3, r) = 0.916$, i.e., r is 'more proved' given E_1 than it is given E_2 and it is more 'more proved' given E_2 than it is given E_3, more completely reflecting our intuitions about the relative degree-of-proof of the three examples. (In the experiments reported in this paper, we set $\epsilon = 0.1$ without tuning. Although our results to date show little sensitivity to this choice, properly tuning ϵ is a matter for further investigation.)

3. Soft classification

Given a flawed theory Γ (for concept r), we can now consider how to use the softening function u_ϵ to classify examples. The idea is that examples which are proved to a greater degree according to u_ϵ are more likely to be truly positive, and vice versa, regardless of whether or not the example actually is proved in Γ. Thus, we can rank a set of unclassified examples E_i according to $u_\epsilon(E_i, r, \Gamma)$. Then by choosing a good threshold θ, we classify an example E as positive if $u_\epsilon(E, r, \Gamma) > \theta$ and negative if $u_\epsilon(E, r, \Gamma) \leq \theta$.

For example, consider the well-known domain theory for identifying *E. coli* promoter gene sequences (the 'promoter theory' (Merz, Murphy, & Aha, 1996)). The theory consists

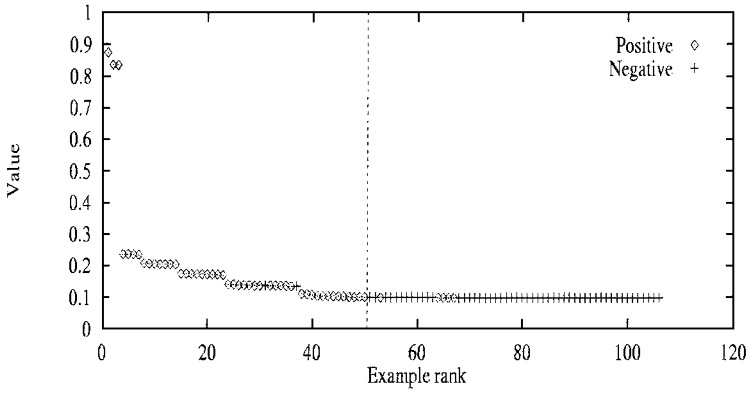


Figure 1. Softened truth values ($u_{0.1}$) for the 106 examples of the promoter theory in rank order. The optimal classification threshold for these examples is depicted by the vertical line.

of 10 rules and a single top-level proposition indicating whether a particular gene sequence is a promoter or not.

The theory as given has a classification accuracy of only 50%; every example is classified as negative, when in fact only half of them should be. However, when we sort by u_ϵ, we can distinguish nearly perfectly between positive and negative examples (as shown in figure 1). In fact, by choosing the optimal threshold for u_ϵ, where examples scoring above the threshold are taken as proved and those scoring below as unproved, we get a classification accuracy of 93.4%. This example illustrates how by softening a theory we may dramatically improve its classification accuracy (50% to 93.4%) without doing any revision whatsoever.[1] Naturally, the classification threshold must be chosen properly. In practice, the right threshold can be estimated from a very small set of preclassified training examples, as we will see below.

More precisely, given a theory Γ with root r, a training set $\mathcal{E}$, a softening function u_ϵ, and a threshold θ, define $\mathsf{Acc}(\Gamma, \mathcal{E}, u_\epsilon, \theta)$, as the fraction of examples in $\mathcal{E}$ accurately classified by using θ as a classification threshold for u_ϵ. Then, we can classify a new example E using the algorithm **SoftClassify**.

```
SoftClassify(Γ, $\mathcal{E}$, $u_\epsilon$, $E$):

1. Let $\theta$ be the threshold maximizing $\mathsf{Acc}(\Gamma, \mathcal{E}, u_\epsilon, \theta)$;
2. If $u_\epsilon(E, r, \Gamma) > \theta$, classify it as positive;
3. Else, classify it as negative.
```

The promoter theory requires very little training to reach respectable classification accuracy when using **SoftClassify**. For example, when choosing the optimal threshold based on only 20 training examples we get an average classification accuracy of 91% (using 5-fold cross-validation, withholding all but 20 training examples each time).

Figure 2 shows corresponding learning curves for softening and several other learning techniques. We performed 5-fold cross-validation, withholding different amounts of training

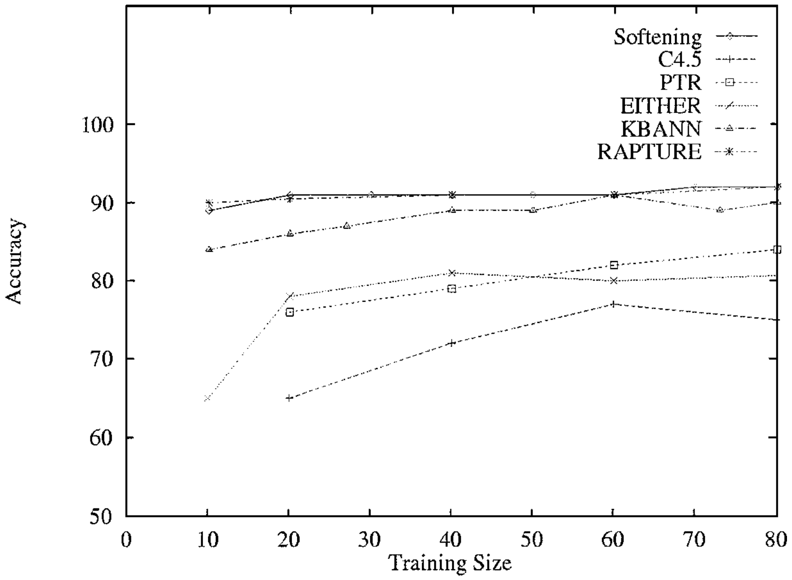


Figure 2. Softening for the promoter theory. Accuracy of the original theory and learning curves for C4.5, softening, PTR, EITHER, and KBANN.

to generate each data point. We compare softening with the accuracy of the original theory, with the example-based learning system C4.5 (Quinlan, 1993), and with the theory revision systems[2] PTR (Koppel, Feldman, & Segre, 1994a), EITHER (Ourston & Mooney, 1994), KBANN (Towell & Shavlik, 1993), and RAPTURE (Mahoney & Mooney, 1994). As the figure shows, softening on the promoter theory is better than most of the alternatives.[3] The only alternatives that are competitive with softening are KBANN and RAPTURE, both theory revision systems that use numerical representations of the theory in the course of revision. (In fact, the results of RAPTURE are virtually identical to those of **SoftClassify**.) It is interesting to note that softening still performs as well as those systems here, despite the fact that the only learning it does is to estimate a single threshold.

4. Partial rehardening

4.1. The problem with softening

Although softening works remarkably well on a theory like promoter, where errors are distributed throughout the theory (Koppel, Feldman, & Segre, 1994b; Ortega, 1995), we should not expect it to work as well on a theory where flaws are highly localized. This is because softening treats all components of the theory in the same way. Since both flawed and correct components are softened equally, **SoftClassify** cannot always distinguish correctly classified examples from incorrectly classified examples. Softening those parts of the theory that are correct cannot be expected to improve classification accuracy.

illegal	:-	same-loc-ab-cd, **adj-bf**
illegal	:-	same-loc-ab-ef
illegal	:-	same-loc-cd-ef
illegal	:-	king-attack-king
illegal	:-	rook-attack-king
king-attack-king	:-	adj-ae, adj-bf
king-attack-king	:-	adj-ae, b=f
king-attack-king	:-	a=e, *adj-bf*
king-attack-king	**:-**	**knight-move-ab-ef**
rook-attack-king	*:-*	*c=e, king-not-b-file*
rook-attack-king	:-	d=f, king-not-b-rank
king-not-b-rank	:-	¬b=d
king-not-b-rank	:-	b=d, ¬between-cae
king-not-b-file	:-	¬a=c
king-not-b-file	:-	a=c, ¬between-dbf

Figure 3. A flawed version of the chess endgame theory. Added antecedents and clauses are shown in boldface, while deleted components (not actually in the flawed theory) are shown in italics. Low-level propositions such as a=b are defined in terms of primitive attributes a through f, each of which takes on values from 1 to 8 (not shown).

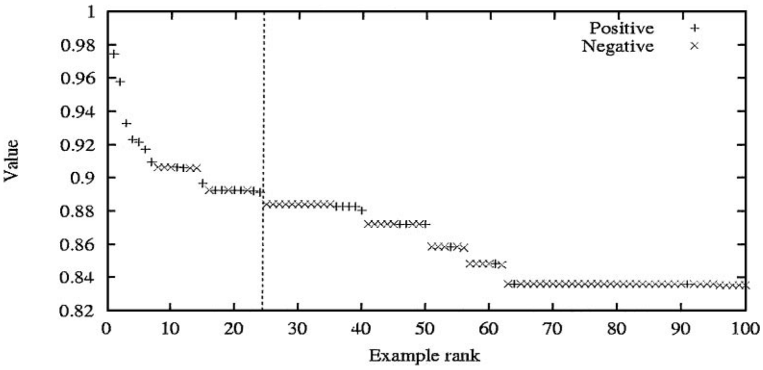


Figure 4. Sorting graph (as per figure 1) for 100 randomly chosen examples for the softened chess endgame theory; the vertical dashed line shows the optimal classification threshold.

Consider, for example, the flawed domain theory for categorizing king-rook-king chess endgames (Merz, Murphy, & Aha, 1996) depicted in figure 3, with root $\mathtt{r} = \mathtt{illegal}$. For this theory, softening neither improves nor harms classification accuracy, as shown in figure 4. Using any number of training examples between 10 and 100, **SoftClassify** gives a maximum accuracy of 79% on a separate set of 100 test examples (34% positive, 66% negative), which is the same as the theory's raw accuracy. Soft classification doesn't help for the chess endgame theory because most of the theory should not be softened at all.

To illustrate this point, consider two examples for the theory: E_1 for which `adj-bf` is true, and E_2 for which `same-loc-ab-cd` is true, where the examples are otherwise identical and have no proofs in the theory as given. Since `adj-bf` is incorrectly added to the first `illegal` clause, E_2 is truly positive. Despite this, $u_\epsilon(E_1, r, \Gamma)$ in the flawed theory will be greater than $u_\epsilon(E_2, r, \Gamma)$, due to the greater number of occurrences of `adj-bf` in the theory.

Surprisingly, though, we will find that the fact that softening on this theory does not reduce classification accuracy is no fluke. On the whole, even for locally flawed theories softening almost never does harm and often improves classification accuracy significantly. Nevertheless, we shall see below that for theories with localized flaws we can generally obtain improved classification accuracy by softening in a more selective fashion.

4.2. Partially rehardened theories

As the above example illustrates, a given theory may contain regions which should be interpreted in a soft manner (i.e., like promoter) and regions which should be interpreted in a non-soft manner (i.e., are correct as is). For theories with localized flaws, such as the chess endgame theory, classification by the **SoftClassify** algorithm would greatly improve if we could somehow soften only the flawed portions of the theory. In this section we will describe a simple algorithm which finds those components in a theory which should not be interpreted in a soft manner. First, though, we will define more precisely what it means to interpret a theory as partially soft, i.e., with some components defined as *hard*. Note that here we consider each appearance of a proposition in an antecedent literal as a separate component of the theory, so that one appearance of a proposition can be hard, while another is soft.

Formally, given a set H of theory components defined as *hard*, for each component k with children $c_1, \ldots, c_n$ we define

$$u_\epsilon^H(E, k, \Gamma) = \begin{cases} 1 - \prod_{i=1}^{n} u_\epsilon^H(E, c_i, \Gamma) & \text{if } k \in H \\ 1 - (1-\epsilon)\prod_{i=1}^{n} u_\epsilon^H(E, c_i, \Gamma) & \text{otherwise} \end{cases}$$

whereas, for each appearance l of a primitive proposition P, we have

$$u_\epsilon^H(E, l, \Gamma) = \begin{cases} u(E, P, \Gamma) & \text{if } l \in H \\ u_\epsilon(E, P, \Gamma) & \text{otherwise} \end{cases}.$$

Intuitively, u_ϵ^H introduces uncertainty only into the soft components of the theory; all others are assumed to be correct. (For example, evaluating an example in a theory with all components hardened simply gives 1 or 0, according as the example is positive or negative in the theory.)

4.3. The rehardening algorithm

We now wish to exploit a given set of training examples in order to determine which components in Γ should be hardened and which softened. The idea is to harden those components whose hardening improves classification accuracy using **SoftClassify**. We can then use **SoftClassify** to classify new examples using the partially rehardened theory thus obtained.

The idea is to iteratively harden components of the theory, each time evaluating the optimal accuracy of the theory on the training set. Thus, given a theory Γ, a training set $\mathcal{E}$, and softening function u_ϵ, we define the *soft accuracy* of Γ as: $\mathsf{SoftAcc}(\Gamma, \mathcal{E}, u_\epsilon^H) = \max_\theta \mathsf{Acc}(\Gamma, \mathcal{E}, u_\epsilon^H, \theta)$. The **Reharden** algorithm greedily hardens components of Γ until doing so would reduce the accuracy of the theory on the training set $\mathcal{E}$, i.e. $\mathsf{SoftAcc}(\Gamma, \mathcal{E}, u_\epsilon^H)$. Note that evaluating a theory with n components for a given set of hardened components takes $O(n|\mathcal{E}|)$ time. Since such an evaluation is performed n times for each component that is hardened, in the worst case, a straightforward implementation of **Reharden** takes $O(n^3|\mathcal{E}|)$ time. The method is usually much faster than this in practice, and efficiency can be further improved by caching of intermediate results.

```
Reharden($\Gamma$,$\mathcal{E}$):
1. $H \leftarrow \emptyset$;
2. Evaluate $a_0 = \mathsf{SoftAcc}(\Gamma, \mathcal{E}, u_\epsilon^H)$;
3. Evaluate, for every component $c_i \in \Gamma \backslash H$, its hardening accuracy
   $a_i = \mathsf{SoftAcc}(\Gamma, \mathcal{E}, u_\epsilon^{H \cup \{c_i\}})$;
4. Let $c^*$ be the component closest to the root whose accuracy $a^* > a_0$
   (breaking ties arbitrarily);
5. If such a component exists:
   a) $H \leftarrow H \cup \{c^*\}$,
   b) Goto Step 2;
6. Else, let $c^*$ be the component closest to the root whose hardening accuracy $a^* = a_0$
   (breaking ties arbitrarily);
7. If such a component exists:
   a) $H \leftarrow H \cup \{c^*\}$,
   b) Goto Step 2;
8. Else return $H$.
```

4.4. Justification: Rehardening and revision distance

The reason the rehardening procedure works is that (i) hardening *flawed* components tends to *reduce* the accuracy of **SoftClassify**, and (ii) hardening *unflawed* components of the theory tends to *increase* the accuracy of **SoftClassify**. Since hardening flawed components usually reduces accuracy, our greedy rehardening algorithm will, in general, harden only unflawed components, which in turn will tend to increase **SoftClassify**'s accuracy using the theory.

The connection between the choice of components to be hardened and the resulting accuracy of **SoftClassify** is a consequence of a fundamental property of the u_ϵ^H function which forms the basis for **SoftClassify**. The property is that u_ϵ^H sorts examples primarily by how many revisions (that is, deletions) to non-hardened components in the theory would suffice to change the examples' classifications (an example's *revision distance*). Classifying using revision distance is obviously correlated with hardening unflawed components, in that hardening unflawed components causes the revision distance of correctly classified examples to increase as the number of possible revision sites decreases. For incorrectly classified examples, however, it is always sufficient to revise only flawed components. Thus, if flawed components remain unhardened, the revision distance of an incorrectly classified example cannot increase beyond the minimum number of flawed components which need to be revised in order to change the example's class.

We make this intuitive notion more precise by considering the *subtheories* of the given theory Γ, obtained by deleting some of Γ's *non-hard* components (i.e., components not in H), where the distance of a subtheory Γ' from Γ, $\mathsf{dist}(\Gamma, \Gamma')$, is the number of components deleted. In order to quantify how robust an example E's classification is, with respect to possible flaws in Γ, we measure the distance of the nearest subtheory Γ' which classifies E differently from Γ. We define the *revision distance*, $D^H(\Gamma, E)$, of E in Γ with respect to the set of hardened components H as the number of deletions required to change an example's classification. In particular:

– A positive revision distance gives the number of component deletions needed to make an unproved example proved,
– A negative revision distance gives the number needed to make a proved example unproved.

(In a theory without negation, the former deletions are of antecedents, while the latter are of clauses.)

The key idea here is that there exists a close relationship between u_ϵ^H and revision distance. For the case of *tree-structured* theories, where each non-primitive proposition appears (possibly negated) as an antecedent of no more than one clause, the relationship can be neatly formulated as follows:

Theorem 1. *Given a tree-structured theory Γ with root r, a set H of components of Γ, and examples E_1 and E_2 for Γ, such that $D^H(\Gamma, E_1) > D^H(\Gamma, E_2)$, then for all sufficiently small ϵ, we have that $u_\epsilon^H(E_1, r, \Gamma) < u_\epsilon^H(E_2, r, \Gamma)$.*

This theorem states that, in the limit, larger values of D^H lead to smaller values of u_ϵ^H. That is, sorting according to u_ϵ^H is consistent with sorting according to how much the theory would have to change in order to change each example's classification. Sorting by u_ϵ^H, however, provides a more fine-grained measure which gives useful information even when revision distances are identical.

For example, if all components of a theory are soft, revision distance is nearly useless, since any example's classification can be changed by revising a few components at the root. Revision distance becomes meaningful, however, as more and more components in a theory are hardened. The more components are hardened, the more precise a measure it becomes for distinguishing the degree to which examples are proved.

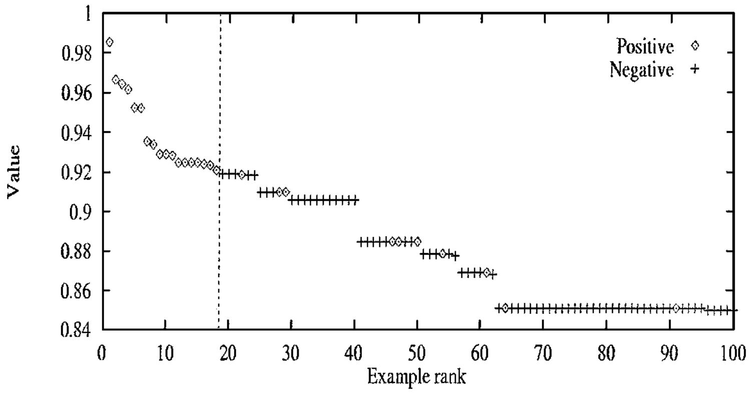
See the appendix for a more formal treatment of these ideas and a proof of the theorem.

5. Illustrations of softening and rehardening

In the section following we will show the results of a systematic set of experiments designed to test our hypotheses about the effectiveness of softening and rehardening flawed theories.

```
          illegal   :-  same-loc-ab-cd, adj-bf
          illegal   :-  same-loc-ab-ef
          illegal   :-  same-loc-cd-ef
          illegal   :-  king-attack-king
          illegal   :-  rook-attack-king
king-attack-king    :-  adj-ae, adj-bf
king-attack-king    :-  adj-ae, b=f
king-attack-king    :-  a=e
king-attack-king    :-  knight-move-ab-ef
rook-attack-king    :-  d=f, king-not-b-rank
 king-not-b-rank    :-  ¬b=d
 king-not-b-rank    :-  b=d, ¬between-cae
```

(a)



(b)

Figure 5. Results of rehardening: (a) A hardened version of the chess-endgame theory with softened clauses in boldface and softened antecedents underlined. (b) The sorting graph of the rehardened theory (using the same 100 examples as in figure 4); the vertical dashed line shows the optimal classification threshold.

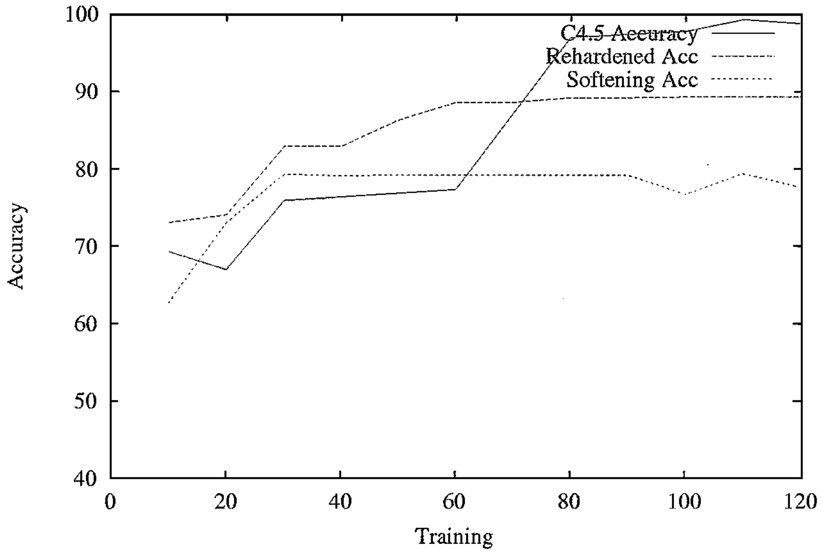


Figure 6. Rehardening for the chess endgame theory. Averaged learning curves for C4.5 and **SoftClassify** with and without rehardening.

Before getting to that, though, let us consider a few illustrations of the method on some familiar theories.

5.1. Rehardening the chess theory

Let us now reconsider Γ, the flawed version of the chess endgame theory shown in figure 3. Figure 5(a) shows the results of rehardening that theory on a training set $\mathcal{E}$ of just 30 examples. Note that the components which remain soft are almost exactly those which contain flaws in the theory. Figure 5(b) shows how **SoftClassify** sorts 100 test examples (not in $\mathcal{E}$) using the rehardened theory shown. With rehardening, $\mathsf{SoftAcc}(\Gamma, \mathcal{E}, u_{\epsilon}^{H})$ increases to 90%, as opposed to $\mathsf{SoftAcc}(\Gamma, \mathcal{E}, u_{\epsilon})$=79% without rehardening.

In figure 6 we compare the accuracies obtained on a test set using C4.5, softening alone, and rehardening, respectively, with varying amounts of training data. We evaluated the methods over 10 trials. Each trial trained on each of 10, 20, 30, up to 120 examples, and accuracy was tested on a disjoint 100-example test set (separate for each trial). We then averaged the accuracies from all trials. It is evident that rehardening significantly improves over softening (which never exceeds the original theory accuracy here). Furthermore, for few training examples (less than 70), rehardening is better than learning directly from examples. (We performed t-tests on the paired data for the 10 trials, giving a significance of $p < 0.05$ to the difference in accuracy in all cases tested.) The rehardening curve flattens very quickly, however, even as C4.5 continues to improve. Since rehardening is somewhat crude (for example, it cannot fine-tune theories by adding components) its potential is limited. Thus when sufficient examples are available, it may be preferable to use inductive methods.

5.2. Comparing types of flawed theories

Here we examine our proposed explanation for the difference in softening performance between the promoter and chess endgame theories. In addition to the "locally flawed" theory presented above, therefore, we also created a flawed theory with synthetic "distributed flaws", to show how we can distinguish between these types of theories based on their performance under softening and rehardening. In a theory with distributed flaws, many or most components of the theory are flawed, but each flaw does not change the meaning of the theory very much. For example, if the antecedents of two clauses for a proposition *p* are randomly redistributed between the clauses, many flaws are introduced (for each antecedent wrongly placed), but still, the same set of components influences the truth of *p* in the flawed theory.

Thus, we consider here three flawed versions of the chess endgame theory:

1. the theory considered above with four localized flaws (Chess-1),
2. a theory with distributed flaws, created by repeatedly merging and randomly splitting the antecedent sets of clauses in the theory (Chess-2, figure 7), and
3. a randomly generated theory with the same primitives as the correct theory (Chess-3, figure 7).

We performed a set of trials, each using a different 40-example training set and 200-example testing set. In each trial, for each theory we evaluated the accuracy on the test set of (a) the original flawed theory, (b) the softened theory using a threshold evaluated from the training set, and (c) the theory rehardened based on the training set. We averaged the results of 5 random trials; results are shown in Table 1.

Table 1 shows the initial accuracies and the results of using softening and partial rehardening on the three test theories. Note that just using classification accuracy tells us nothing about the relative merits of the theories; in fact, the random Chess-3 is better than Chess-2 with distributed flaws. Nevertheless, we see that the accuracy of the random theory using **SoftClassify**, even with rehardening, is little better than its raw accuracy—and, more significantly, little better than simply classifying all examples as negative—indicating clearly that the theory is essentially useless for distinguishing between positive and negative examples, and thus should be discarded. On the other hand, both Chess-1 and Chess-2 show significant improvement in accuracy using **SoftClassify** with rehardening. These theories

Table 1. Accuracies for different interpretation methods for chess theories with different types of flaws. The last line shows the fraction of the total example set classified by each theory as positive. Chess-1 has localized flaws, Chess-2 has distributed flaws, and Chess-3 is random.

	Chess-1	Chess-2	Chess-3
Theory itself	77%	35%	65%
Softening	77%	87%	68%
Rehardening	89%	94%	72%

```
illegal           :-  king-attack-king, rook-attack-king
illegal           :-  king-attack-king, same-loc-cd-ef
illegal           :-  king-attack-king, same-loc-ab-ef
illegal           :-  rook-attack-king, same-loc-ab-ef, same-loc-ab-cd
rook-attack-king  :-  king-not-b-file, adj-ae, d=f
rook-attack-king  :-  d=f
rook-attack-king  :-  king-not-b-rank, d=f, king-not-b-file, c=e
king-not-b-file   :-  adj-bf, adj-ae, ¬between-dbf
king-not-b-file   :-  c=e, d=f, ¬between-dbf, a=c
king-not-b-file   :-  ¬between-dbf, a=c
king-not-b-file   :-  ¬a=e, adj-ae, ¬a=c
king-not-b-file   :-  ¬d=f, c=e, ¬a=c
king-not-b-rank   :-  ¬between-cae, b=d
king-not-b-rank   :-  ¬between-cae, ¬b=d
king-attack-king  :-  adj-bf, a=e, b=f, adj-ae
king-attack-king  :-  b=f, adj-bf, adj-ae
```

(Chess-2)

```
illegal  :-  Int-b, b=d
illegal  :-  ¬a=c, adj-ae
Int-b    :-  Int-i, c=e
Int-b    :-  ¬c=e, Int-c
Int-i    :-  Int-c, ¬Int-d
Int-i    :-  a=e, b=d
Int-c    :-  ¬a=c, Int-j
Int-c    :-  Int-d, a=c
Int-d    :-  between-cae, same-loc-cd-ef
Int-d    :-  ¬b=d, between-cae
Int-d    :-  same-loc-ab-cd, a=c
Int-d    :-  same-loc-cd-ef, adj-bf
Int-j    :-  adj-ae, b=f
Int-j    :-  ¬adj-bf, ¬between-cae
```

(Chess-3)

Figure 7. Chess-2: Chess theory with distributed flaws. Chess-3: Random ‘chess’ theory.

are distinguishable, however, by the respective differences in the gap between softening and rehardening. Rehardening adds less to the effect of softening in Chess-2, which suggests that its flaws are non-localized, while rehardening improves Chess-1 quite noticeably over softening, which suggests that Chess-1's flaws are localized. Such information could be useful for deciding how to handle each one of the three theories. We should probably revise the theory with localized flaws, interpret the theory with distributed flaws probabilistically, and throw out the random theory. Indeed, C4.5 performs significantly better (81%) than Chess-3 even with rehardening, whereas both Chess-1 and Chess-2 show improvement over C4.5 with rehardening (for a small training set).

More generally, these results suggest how we might decide the proper way to use a given theory based on a training set. First check if the theory contains any useful information, i.e., that positive examples are 'proved to a greater degree' than negative examples. Specifically, we need to check that the accuracy obtained by the softened theory is significantly better than the accuracy expected from optimally partitioning a random ordering of positive and negative examples. (Roughly speaking, this expected accuracy slightly exceeds max(Pos,Neg), where Pos and Neg are the respective percentages of positive and negative examples in our training set. Thus, for example, for the chess theory 66% of our training examples are negative and the softened random theory correctly classes 68%. This is easily achieved by choosing a threshold near 1.0, i.e., by classing almost all examples as negative.) If this is the case, the theory should be revised just when it contains localized errors, i.e., when rehardening obtains significantly better classification on training data than softening does. This could be done, given a small training set, by evaluating the expected accuracy of softening and rehardening by cross-validation. If softening increases accuracy greatly, perhaps the theory should be used as is. However, if softening does little but rehardening helps, the theory should probably be revised. If neither softening nor rehardening helps, the theory should be discarded and pure inductive techniques should be applied. In the next section we will see, though, that such a method is not completely reliable in general: rehardening often helps significantly for theories with non-localized flaws, while not helping at all for some theories with localized flaws.

5.3. Rehardening flawed student-loan theories

We now take a closer look at some rehardened flawed theories in order to compare the set of flawed components with the set of components left soft. We will see that although these two sets are generally similar, they are not always identical. Obviously, those flaws that do not adversely affect the classification of any training examples are not left soft. Additionally, it turns out, surprisingly, that there are subtle ways in which hardening flawed components while leaving related components soft actually leads to better results than leaving the flawed components themselves soft. In order to illustrate this and related phenomena, we consider three arbitrarily chosen flawed versions of the student-loan theory (shown in figure 8), used for determining whether a student must pay back a student loan (Pazzani & Brunk, 1991).

We performed five independent trials for each theory, using disjoint 100-example training and test sets. We compare the flawed theories with the sets of hardened components, as well

```
 1. no-payment-due :- deferment
 2. no-payment-due :- continuous
 3. deferment :- disability-deferment
 4. deferment :- student-deferment
 5. deferment :- financial-deferment
 6. deferment :- peace-corps-deferment
 7. deferment :- military-deferment
 8. continuous :- enrolled-five-years, never-left
 9. military-deferment :- armed-forces-enlist
10. peace-corps-deferment :- peace-corps-enlist
11. financial-deferment :- unemployed
12. financial-deferment :- filed-for-bankruptcy
13. student-deferment :- enrolled-eleven-years
14. disability-deferment :- disabled
```

Figure 8. The correct student-loan theory.

as examining the accuracies of the initial theory, the theory with softening, with rehardening, and with just the flawed components left soft (simulating ideal rehardening).

Although the set of rehardened components for each flawed theory varies with the given training examples, in each case there was one rehardened theory which appeared in a majority of the independent trials (and it is to these rehardened theories that we refer below). The flawed theories are compared with the results of rehardening in Tables 2 and 3.

In theory SL I, flaw 1 does not let through any negative examples and in fact lets through some positive ones incorrectly blocked by flaw 3. Therefore it is not left soft. Flaws 2 and 4 are captured directly. This is an easy theory for softening to handle since the flaws are additions rather than deletions. The rehardened theory perfectly classifies all the test examples.

In Theory SL II, flaws 3, 4, and 5 are captured directly. Flaw 2 does not let through any negative examples. Only the deleted clause is not directly compensated for. As a result the rehardened theory is only 94% accurate. As can be seen in Table 3, even if precisely the flawed components are left soft (i.e., if the antecedent `deferment`, which is rendered overly specific by the deletion of clause 7, is also left soft), the rehardened accuracy does not exceed 94%.

Theory SL III is the most interesting of the three flawed theories considered here. The effect of the added clause (flaw 2) is diminished both by its being softened and by its parent (clause 6) being softened. Softening ¬`fire-department-enlist` in the softened added clause boosts the relative effect of the other antecedent, `never-left`, which was deleted from clause 8 (flaw 3). Softening clause 2 further diminishes the influence of clause 6 which lets through negative examples (as a result of flaw 3). Finally, softening the antecedent `continuous` in clause 2 diminishes the influence of softening the clause itself. Flaw 4 does not let through any negative examples and is not left soft. As seen in Table 3, this

Table 2. Rehardening results for three flawed student-loan theories. The table shows which flaws were introduced into the correct theory (figure 8) and which components were left soft by rehardening. Each component whose softening compensates for a particular flaw is placed near that flaw in the table.

Theory	Flaws	Soft components
SL I	1. Add military-deferment :- filed-for-bankruptcy, ¬disabled	–
	2. In clause 8 add antecedent: financial-deferment	* financial-deferment in clause 8
	3. In clause 12 add antecedents: military-deferment, student-deferment	–
	4. In clause 14 add antecedent: continuous	* continuous in clauses 2 and 14
SL II	1. Delete clause 7	–
	2. Add deferment :- ¬filed-for-bankruptcy, financial-deferment, student-deferment disability-deferment	–
	3. In clause 1 add antecedent: enrolled-eleven	* enrolled-eleven in clause 1
	4. In clause 10 add antecedent: continuous	* continuous in clause 10
	5. In clause 14 add antecedent: continuous	* continuous in clause 14
	–	* continuous in clause 2
SL III	1. Delete clause 5	–
	2. Add peace-corps-deferment :- ¬fire-department-enlist, never-left	* peace-corps-deferment :- ¬fire-department-enlist, never-left * ¬fire-department-enlist in added clause * Clause 6
	3. In clause 8 delete antecedent never-left	* Clause 2
		* continuous in clause 2
	4. Add military-deferment :- disabled, ¬student-deferment, foreign-legion-enlist	–

Table 3. Results of softening and rehardening for three flawed student-loan theories. Shown are the initial accuracy of the flawed theory, as well as the accuracies obtained by softening, by rehardening, and by hardening all components except for those actually flawed.

	SL I	SL II	SL III
Initial	75%	74%	75%
Softened	98%	86%	87%
Rehardened	100%	94%	94%
Flaws soft	100%	94%	90%

results in a more accurate rehardened theory (94%) than the one in which precisely the flawed components are left soft (90%).

6. Experimental results

6.1. Theory accuracy

After the anecdotal results of the previous section, we turn now to several systematic experiments. We synthetically generated five different random propositional theories. The number of distinct propositions in the theories ranged from 15 to 21, and the number of clauses in the theories ranged from 14 to 22, with an average of 2.5 antecedents per clause. For each theory, 20 flawed theories were generated, 10 with five local flaws each and 10 with five distributed flaws each, as follows:

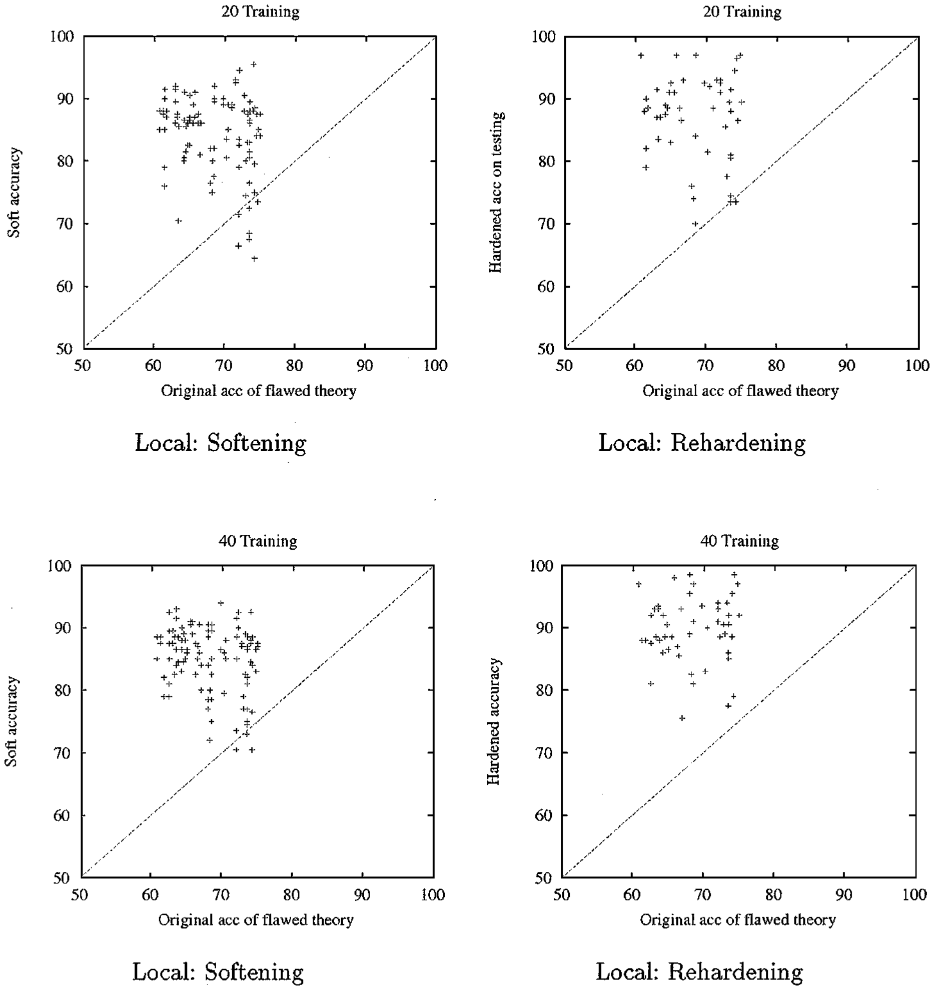
Local: The flaw generator chooses a theory component at random and inserts into it a random flaw (adding a clause with 2–4 random antecedents for a proposition, deleting an antecedent, or either adding a random antecedent for a clause or deleting the clause with equal probability).

Distributed: The flaw generator chooses a proposition at random and replaces two random clauses for the proposition by a new clause C with the union of the two clauses' antecedents, and then splits C randomly into two new clauses by randomly assigning antecedent literals to one (or both) of the new clauses.

These two types of flaws allow us to experimentally evaluate our explanation of the difference between the promoter and chess endgame theories described above.

For each of the five correct theories, 100 examples were randomly generated and divided into two equal sets. Nested subsets of various sizes were selected from one of the sets as training examples for each corresponding flawed theory and the results were tested against all the examples in the other set. The roles of the two sets were then switched, so that for each flawed theory and each number of training examples two data points were generated. These were then averaged.

In the accompanying scatter plots we plot the initial accuracy of each of the fifty locally-flawed theories and each of the fifty distributed-flawed theories against the softened accuracy



Local: Softening

Local: Rehardening

Local: Softening

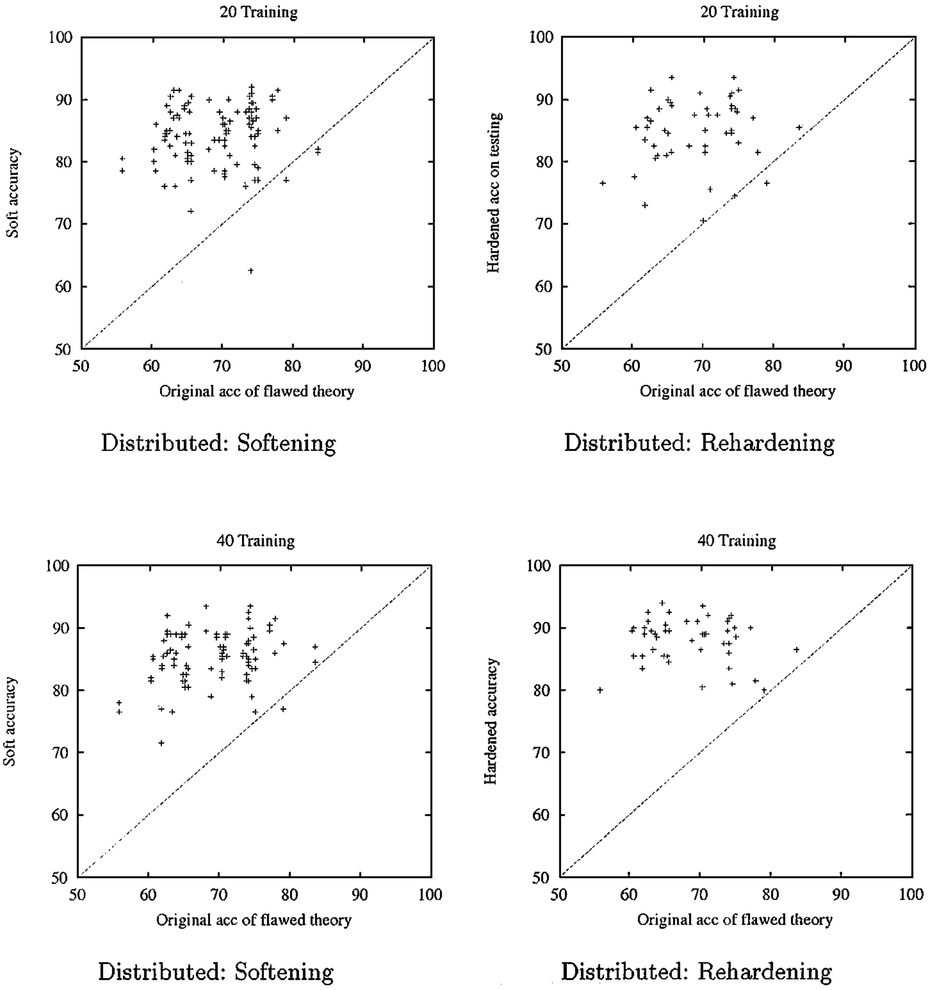
Local: Rehardening

Figure 9. Scatter plots for experiments on synthetic theories with local flaws, using 20 and 40 training examples.

and against the hardened accuracy. We show results here for 20 and for 40 training examples in figures 9 and 10; average results are given in Table 4.

As is evident from the plots, softening is astonishingly effective even using only 20 training examples. It almost never does any harm. When rehardening is used to focus softening, every single one of the 100 flawed theories is improved when only 40 training examples are used. Even when only twenty training examples are used, only two out of one hundred rehardened theories classify less accurately than the original theories (and these only by a tiny margin).

We tested for statistical significance of the improvement of (a) softening over the original theory, and (b) rehardening over softening, by performing t-tests on the paired data. In seven of eight cases the accuracy improvement proved to be significant with $p < 0.002$.



Distributed: Softening

Distributed: Rehardening

Distributed: Softening

Distributed: Rehardening

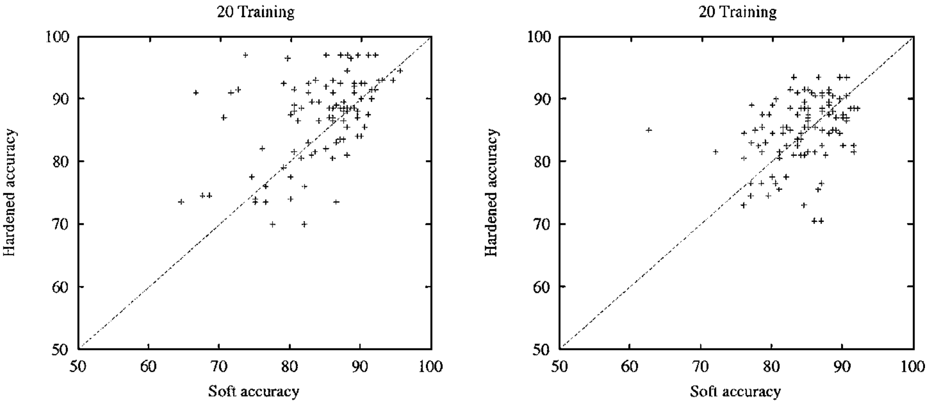
Figure 10. Scatter plots for experiments on synthetic theories with distributed flaws, using 20 and 40 training examples.

Only for the case of distributed flaws with 20 training examples was rehardening accuracy indistinguishable from softening accuracy. This accords with our hypothesis about the lessened effect of rehardening for theories with distributed flaws.

Nevertheless, it is interesting to note that contrary to our expectations, the difference in effectiveness of softening versus rehardening for local versus distributed flaws, although detectable in Table 4, is not very large. Softening works well for both and rehardening generally slightly improves both (with a greater improvement for locally flawed theories). A small difference can be seen, however, in figure 11, which plots accuracy after rehardening as a function of accuracy after softening.

Table 4. Average results for experiments on synthetic theories with local and distributed flaws. Shown are the average original accuracies of the theories, with accuracies after softening and rehardening, with 20 and 40 training examples.

		20 Training		40 Training	
Flaw type	Original	Softening	Rehardening	Softening	Rehardening
Local	68.4%	84.2%	87.0%	84.9%	89.7%
Distributed	69.0%	84.2%	84.9%	85.2%	88.1%



Local

Distributed

Figure 11. Scatter plots for experiments on synthetic theories with local distributed flaws, using 20 training examples, comparing rehardening to softening accuracy.

6.2. *Finding single flaws*

As we saw above, on theories with multiple flaws rehardening often works more effectively by leaving soft theory components other than those which are actually flawed. In the case of theories with single flaws, we would hope that rehardening would be able to isolate the flaws precisely. To the extent that rehardening is successful at isolating flaws in this way it could be effectively used in conjunction with other theory revision algorithms which repair isolated flaws. Accordingly we ran the following experiment to test the ability of rehardening to isolate individual theory flaws.

We generated 10 small random theories (between 5 and 10 clauses each). For each theory we generated 10 flawed versions, each with a single random flaw (as above for local flaws). For each correct theory, we also generated 10 random training sets of 20 examples each, for use in rehardening the flawed theories. We rehardened each flawed theory using each training set for that theory, giving us 1000 data points. After rehardening each flawed theory on a training set, we evaluated which component(s) were soft, as compared with the actual location of the theory flaw.

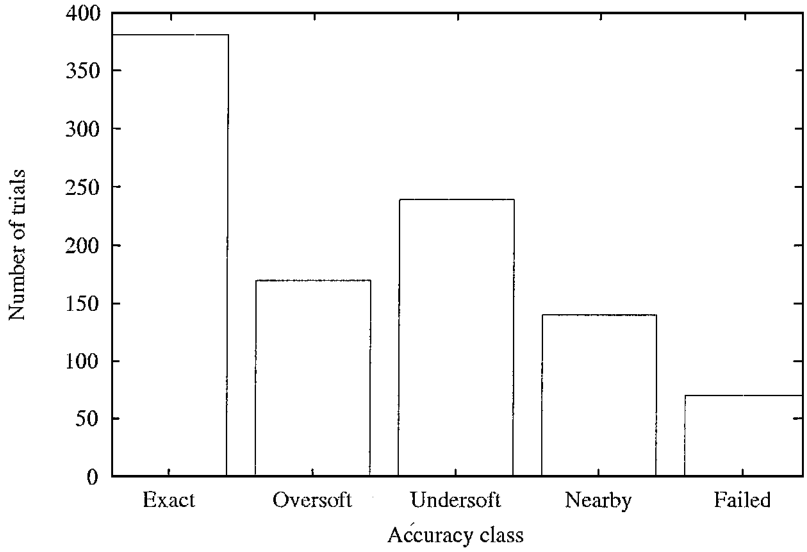


Figure 12. Results of 1000 trials on single-error theories. See text for explanation.

The possible outcomes of each trial are:

Exact the flawed component is the only one left soft,
Oversoft the flawed component is one of several left soft,
Undersoft no components are left soft,
Nearby the flawed component is not left soft but an ancestor/descendant is (and possibly other components as well), or
Failed the flawed component is not left soft but some entirely different component(s) are.

Figure 12 shows the results of the experiment. As is shown, the flaw was left soft in over 55% of all trials, and in 69% of all trials either the flaw or one of its ancestors or descendants was left soft. On average, 1.28 components were left soft in each theory. In only 70 cases (7%) were only components unrelated to the flaw left soft. Out of the 470 cases in which exactly one component was left soft, in 381 cases (81%), the single soft component was the flawed component. It should be noted that of the 239 cases in which no component was left soft, 100 were unavoidable since in them the flawed theory was perfect on the training set (as expected, since the training only contained 20 random examples).

7. Conclusions

We have introduced the notion of “degree of provedness”, which we believe is fundamental for distilling reliable information from unreliable theories. In particular, we have found that softening is an extremely effective method of theory reinterpretation. It increases classification accuracy for almost all flawed theories even when only a handful of training examples are available. Moreover, softening adds no computational expense to ordinary logical methods of classification using theories. Hence, whenever the reliability of a given propositional theory is in doubt and sufficient information for choosing a threshold (e.g., a small number of examples) is available, it is recommended that the theory be interpreted softly.

We have shown formally why rehardening unflawed components of the theory should typically improve soft classification. We have shown empirically that rehardening based on small training sets does in fact improve on softening (sometimes performing even better than rehardening the actual flawed components). However, it is not necessarily the case that the improvement in classification accuracy obtainable by rehardening always justifies its computational expense. Methods for improving the efficiency and efficacy of the rehardening process remain to be explored.

Both softening and rehardening are crude methods which reinterpret theories but do not revise them. Thus their potential for increasing accuracy is limited, especially where theory components have been deleted. The strength of these methods lies primarily in their not requiring large amounts of training examples. When training examples abound, inductive methods may prove superior to these methods.

We have suggested ways in which rehardening can be used as a meta-algorithm for theory revision by determining whether a flawed theory contains useful information, whether its flaws are localized or distributed throughout the theory, and where localized flaws are located. This information can be used to decide whether a theory is a candidate for revision and whether it should be patched or reinterpreted. However, our results do not bear out that the relative effectiveness of softening versus rehardening can reliably distinguish theories with localized flaws from those with the type of distributed flaws that we considered here. We believe that the weakness of these results is an artifact of our method for generating distributed flaws. The precise definition of distributed flaws and methods for generating them are an important topic for future research.

One issue that remains open is the interaction between various parameters. In our softening experiments we always set $\epsilon = 0.1$ and the theories we used had between 25 and 150 components and between 5 and 20 flaws. For these parameters we found 20 to 30 training examples to be sufficient for softening and 30 to 40 sufficient for rehardening. The relationship between ϵ, theory size, the number and type of flaws, and the number of examples required for varying accuracy levels merits further investigation.

Appendix A: SoftClassify and revision distance

In this appendix we formalize the intuition that the more components that get rehardened, the better we expect **SoftClassify** with rehardening to classify, as discussed in Section 4.4.

Definition 1. A theory Γ' is a *subtheory* of Γ (notated $\Gamma' \subset \Gamma$), if Γ' can be obtained by deleting zero or more components from Γ. The *distance* between a theory Γ and a subtheory Γ', $\mathsf{dist}(\Gamma, \Gamma')$, is the number of components that are deleted from Γ to obtain Γ'.

Definition 2. Let H be a subset of the components of Γ.

The *proof distance* of an example E in a theory Γ is defined as $DP^H(\Gamma, E) = \min_i \mathsf{dist}(\Gamma, \Gamma_i)$ such that $H \subset \Gamma_i \subset \Gamma$ and $\Gamma_i(E) = 1$. If no such Γ_i exists, $DP^H(\Gamma, E) = \infty$.

The *refutation distance* of an example E in a theory Γ is defined as $DR^H(\Gamma, E) = \min_i \mathsf{dist}(\Gamma, \Gamma_i)$ such that $H \subset \Gamma_i \subset \Gamma$ and $\Gamma_i(E) = 0$. If no such Γ_i exists, $DR^H(\Gamma, E) = \infty$.

The *revision distance* of an example E in a theory Γ is defined as $D^H(\Gamma, E) = DP^H(\Gamma, E) - DR^H(\Gamma, E)$.

The proof distance of a proved example is zero, since it is proved in the theory as given; the refutation distance of an unproved example is similarly zero, since it is not proved in the theory as given. Hence, a positive revision distance gives the number of deletions of components not in H needed to make an unproved example proved, whereas a negative distance gives the number needed to make a proved example unproved (no example has revision distance zero). For example, in a theory without negation, the former deletions are of antecedents, while the latter are of clauses.

The relationship between u_ϵ^H and revision distance can be neatly formulated in the following theorem for the case of *tree-structured* theories, where each non-primitive proposition appears (possibly negated) as an antecedent of no more than one clause.

Theorem 1. *Given a tree-structured theory* Γ *with root* r, *a set* H *of components of* Γ, *and examples* E_1 *and* E_2 *for* Γ, *such that* $D^H(\Gamma, E_1) > D^H(\Gamma, E_2)$, *then for all sufficiently small* ϵ, *we have that* $u_\epsilon^H(E_1, r, \Gamma) < u_\epsilon^H(E_2, r, \Gamma)$.

This theorem states that, in the limit, sorting according to u_ϵ^H is consistent with sorting according to revision distance. Note that this theorem makes no distributional assumptions, and does not make any direct claims as to the effect of softening or rehardening on the accuracy of the theory.

Define $P_\epsilon^H(\Gamma_i) \stackrel{\Delta}{=} (1-\epsilon)^{(N-d)}\epsilon^d$, where N is the number of components in Γ not in H, and $d = \mathsf{dist}(\Gamma, \Gamma_i)$. For heuristic purposes, it is useful to think of ϵ as the independent probability of deleting any component not in H from Γ. From this point of view, $P_\epsilon^H(\Gamma_i)$ is simply the probability of the subtheory Γ_i. Similarly, for sets of components $S, R \subset \Gamma$, where S, R, and H are pairwise disjoint, we can denote the probability of the class of subtheories which includes all components in S and deletes all components in R as

$$P_\epsilon^H(\mathrm{inc}(S), \mathrm{del}(R)) = \sum_{H \cup S \subset \Gamma_i \subset \Gamma \setminus R} P_\epsilon^H(\Gamma_i) = (1-\epsilon)^{|S|}\epsilon^{|R|} \quad .$$

Finally, define $\mathrm{Exp}_{P_\epsilon^H}[u(E, r, \Gamma)] = \sum_{H \subset \Gamma_i \subset \Gamma} P_\epsilon^H(\Gamma_i) u(E, r, \Gamma_i)$, the expected value of $u(E, r, \Gamma)$ over all subtheories that include H. The theorem then follows from the following lemma.

Lemma 1. *Given a tree-structured theory* Γ *with root* r, *a set* H *of components of* Γ, *and* $0 \leq \epsilon \leq 1$, *we have that for any example* E, $u_\epsilon^H(E, r, \Gamma) = \mathrm{Exp}_{P_\epsilon^H}[u(E, r, \Gamma)]$.

Proof: We prove the lemma by demonstrating inductively the slightly stronger claim:

– For every component k in Γ, $u^H_\epsilon(E, k, \Gamma) = \text{Exp}_{P^H_\epsilon}[u(E, k, \Gamma)]$.

Base case: Let k be a leaf antecedent (k's proposition p is specified by E). If $k \in H$, $\text{Exp}_{P^H_\epsilon}[u(E, k, \Gamma)] = u(E, k, \Gamma) = u^H_\epsilon(E, k, \Gamma)$. If $k \notin H$, the probability that a random subtheory $H \subset \Gamma' \subset \Gamma$ contains k is $1 - \epsilon$. If Γ' does not contain k, then under the NAND interpretation of the theory we can say that $u(E, k, \Gamma') = 1$ (since a deleted antecedent can be treated as always true and a deleted clause as always false). Therefore, with probability ϵ, $u(E, k, \Gamma') = 1$, and with probability $1-\epsilon$, $u(E, k, \Gamma') = u(E, k, \Gamma)$. Therefore

$$\text{Exp}_{P^H_\epsilon}[u(E, k, \Gamma)] = \epsilon + (1 - \epsilon)u(E, k, \Gamma) = \left\{ \begin{array}{ll} \epsilon & \text{if } k \text{ is false in } E \\ 1 & \text{if } k \text{ is true in } E \end{array} \right\} = u^H_\epsilon(E, k, \Gamma)$$

Inductive step: Let k be some internal component of Γ (an antecedent or a clause), with children $\{c_i\}$, denoting the descendants of component k by desc(k). By assumption, for each c_i, $u^H_\epsilon(E, c_i, \Gamma) = \text{Exp}_{P^H_\epsilon}[u(E, c_i, \Gamma)]$. For S a set of descendants of k (possibly including k), let S_i be those components in S which are descendants of c_i. Since Γ is tree-structured, $S_i \cap S_j = \emptyset$, $i \neq j$. Let $\eta = \epsilon$ if $k \in H$, $\eta = 0$ otherwise. Then:

$$\begin{aligned}
u^H_\epsilon(E, k, \Gamma) &= 1 - (1 - \eta) \prod_i u^H_\epsilon(E, c_i, \Gamma) \\
&= 1 - (1 - \eta) \prod_i \sum_{H \subset \Gamma_j \subset \Gamma} P^H_\epsilon(\Gamma_j) u(E, c_i, \Gamma_j) \\
&= 1 - (1 - \eta) \prod_i \sum_{S_j \subset \text{desc}(c_i)} P^H_\epsilon(\text{inc}(S_j), \text{del}(\text{desc}(c_i) \backslash S_j)) \\
&\qquad \times u(E, c_i, (\text{inc}(S_j), \text{del}(\text{desc}(c_i) \backslash S_j))) \\
&= 1 - (1 - \eta) \sum_{S \subset \text{desc}(k)} \prod_i P^H_\epsilon(\text{inc}(S \cap \text{desc}(c_i)), \text{del}(\text{desc}(c_i) \backslash S)) \\
&\qquad \times u(E, c_i, (\text{inc}(S), \text{del}(\text{desc}(c_i) \backslash S))) \\
&= 1 - \sum_{S \subset \text{desc}(k)} P^H_\epsilon(\text{inc}(S), \text{del}(\text{desc}(k) \backslash S)) \\
&\qquad \times \prod_i u(E, c_i, (\text{inc}(S), \text{del}(\text{desc}(c_i) \backslash S))) \\
&= 1 - \sum_{H \subset \Gamma_j \subset \Gamma} P^H_\epsilon(\Gamma_j) \prod_i u(E, c_i, \Gamma_j) \\
&= \sum_{H \subset \Gamma_j \subset \Gamma} P^H_\epsilon(\Gamma_j)[1 - \prod_i u(E, c_i, \Gamma_j)] \\
&= \sum_{H \subset \Gamma_j \subset \Gamma} P^H_\epsilon(\Gamma_j) u(E, k, \Gamma_j) \\
&= \text{Exp}_{P^H_\epsilon}[u(E, k, \Gamma)]
\end{aligned}$$

□

Proof of Theorem 1: First note that by Lemma 1:

$$u_\epsilon^H(E, r, \Gamma) = \sum_{H \subset \Gamma_i \subset \Gamma} P_\epsilon^H(\Gamma_i) u(E, r, \Gamma_i) = \sum_{i=0}^{N} \epsilon^i (1-\epsilon)^{N-i} N_i(E),$$

where N is the number of components in Γ not in H and $N_i(E)$ is the number of theories Γ' such that $H \subset \Gamma' \subset \Gamma$, $\mathsf{dist}(\Gamma, \Gamma') = i$, and $u(E, r, \Gamma') = 1$. We then have that

$$u_\epsilon^H(E, r, \Gamma) = \sum_{i=DP^H(\Gamma, E)}^{N} \epsilon^i (1-\epsilon)^{N-i} N_i(E),$$

by removing zero terms from the sum. In the limit, we can ignore higher-order terms in the sum, and so

$$\lim_{\epsilon \to 0} \frac{u_\epsilon^H(E, r, \Gamma)}{\epsilon^{DP^H(\Gamma,E)}(1-\epsilon)^{N-DP^H(\Gamma,E)} N_{DP^H(\Gamma,E)}(E)} = 1,$$

Similarly we have that

$$\lim_{\epsilon \to 0} \frac{1 - u_\epsilon^H(E, r, \Gamma)}{\epsilon^{DR^H(\Gamma,E)}(1-\epsilon)^{N-DR^H(\Gamma,E)} \bar{N}_{DR^H(\Gamma,E)}(E)} = 1,$$

where $\bar{N}_i(E)$ is the number of theories Γ' such that $H \subset \Gamma' \subset \Gamma$, $\mathsf{dist}(\Gamma, \Gamma') = i$, and $u(E, r, \Gamma') = 0$.

We now divide the theorem into two cases:

1. $DP^H(\Gamma, E_1) > DP^H(\Gamma, E_2)$; E_1 is unproved in Γ (if E_2 is proved in Γ, $DP^H(\Gamma, E_2) = 0$).
2. $DR^H(\Gamma, E_2) > DR^H(\Gamma, E_1)$; E_2 is proved in Γ (if E_1 is unproved in Γ, $DR^H(\Gamma, E_2) = 0$).

Since, for any given example, either its proof distance or refutation distance is zero, if $D^H(\Gamma, E_1) > D^H(\Gamma, E_2)$, one of the two cases must obtain.

Case 1: Let examples E_1 and E_2 be such that $DP^H(E_1) > DP^H(E_2)$. Consider the ratio

$$\lim_{\epsilon \to 0} \frac{u_\epsilon^H(E_1, r, \Gamma)}{u_\epsilon^H(E_2, r, \Gamma)} = \lim_{\epsilon \to 0} \frac{\epsilon^{DP^H(E_1)}(1-\epsilon)^{N-DP^H(E_1)} N_{DP^H(E_1)}(E_1)}{\epsilon^{DP^H(E_2)}(1-\epsilon)^{N-DP^H(E_2)} N_{DP^H(E_2)}(E_2)}$$

Since $DP^H(E_1) > DP^H(E_2)$,

$$\lim_{\epsilon \to 0} \frac{\epsilon^{DP^H(E_1)}(1-\epsilon)^{N-DP^H(E_1)} N_{DP^H(E_1)}(E_1)}{\epsilon^{DP^H(E_2)}(1-\epsilon)^{N-DP^H(E_2)} N_{DP^H(E_2)}(E_2)} < 1,$$

and hence, for sufficiently small ϵ,

$$u_\epsilon^H(E_1, r, \Gamma) < u_\epsilon^H(E_2, r, \Gamma).$$

Case 2: An argument isomorphic to the above holds for Case 2 in the theorem, by considering $1 - u^H_\epsilon(E, r, \Gamma)$ in place of $u^H_\epsilon(E, r, \Gamma)$.

The theorem then follows. □

Theorem 1 leads immediately to the following corollary, which states that using u^H_ϵ for classification with a perfect theory cannot adversely affect classification accuracy.

Corollary 1 (*Perfect theories*). *If* Γ *is a tree-structured theory with set of hardened components* $H \subset \Gamma$ (*possibly empty*), *we have that for any set of examples* $\mathcal{E}$ *and all sufficiently small* $\epsilon > 0$, *if* Γ *classifies all examples in* $\mathcal{E}$ *correctly*, *then* $\mathsf{SoftAcc}(\Gamma, \mathcal{E}, u^H_\epsilon) = 100\%$.

Proof: Suppose Γ classifies all examples in $\mathcal{E}$ correctly. Consider two examples in $\mathcal{E}$: E_1, proved in Γ (and hence positive by assumption), and E_2, not proved in Γ (and hence negative by assumption). We have that $D^H(\Gamma, E_1) > D^H(\Gamma, E_2)$, and hence by Theorem 1, for sufficiently small ϵ, $u^H_\epsilon(E_1, r, \Gamma) < u^H_\epsilon(E_2, r, \Gamma)$. Since this relation holds for any two examples with differing classifications, there exists a threshold θ on u^H_ϵ which separates positive and negative examples perfectly, i.e., $\mathsf{SoftAcc}(\Gamma, \mathcal{E}, u^H_\epsilon) = 100\%$. □

The next corollary is more interesting, in that it shows how using soft classification can dramatically improve classification accuracy. Define Γ to be a *pseudo-perfect* theory if Γ would classify every example correctly but for the presence of a spurious clause for the root proposition, $r \leftarrow$, which results in every example being classified as positive. The classifications provided by Γ provide no information at all regarding the correct classification of the examples. However, Γ does indeed contain much information regarding the correct classification of examples.

Corollary 2 (*Pseudo-perfect theories*). *If* Γ *is a pseudo-perfect tree-structured theory with set of hardened components* $H \subset \Gamma$ (*possibly empty, but not including the spurious clause*), *then for any set of examples* $\mathcal{E}$ *and all sufficiently small* $\epsilon > 0$, $\mathsf{SoftAcc}(\Gamma, \mathcal{E}, u^H_\epsilon) = 100\%$.

Proof: We will show that any negative example for the concept incorrectly classified as positive by Γ has a lower revision distance than all examples correctly classified as positive by Γ; the corollary then follows by reasoning as above. Consider an arbitrary negative example E_1 and an arbitrary positive example E_2. The proof distance of E_1 is clearly -1, since removal of the single invalid rule $r \leftarrow$ from Γ suffices to remove E_1's spurious proof. Refutation of E_2, however, requires removing not only the spurious proof from the invalid rule, but also at least one more proof from the correct portion of the theory. Hence $D^H(\Gamma, E_2) \leq -2 < D^H(\Gamma, E_1)$; the corollary the follows from Theorem 1. □

Acknowledgments

The authors gratefully acknowledge Michael Pazzani, Raymond Mooney and the anonymous reviewers for their helpful comments.

Notes

1. This merely confirms earlier results, such as those of Ortega (1995) that indicate that simple "numerical" generalization strategies are very effective for this particular theory. It is surprising, though, that so simple a scheme achieves as good or better results on this theory than many theory revision techniques (Towell & Shavlik, 1993; Koppel, Feldman, & Segre, 1994a; Ourston & Mooney, 1994).
2. Theory revision results are those presented in Koppel, Feldman, & Segre (1994a).
3. Unfortunately, we do not have the original data from the revision system experiments, and so could not compute the statistical significance of these results.

References

Buntine, W. (1991). Theory refinement on Bayesian networks. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence* (pp. 52–60).

Cohen, W. W. (1994). Grammatically biased learning: Learning logic programs using an explicit antecedent description language. *Artificial Intelligence, 68*, 303–366.

Donoho, S. K. & Rendell, L. A. (1995). Rerepresenting and restructuring domain theories: A constructive induction approach. *Journal of Artificial Intelligence Research, 2*, 411–446.

Koppel, M. & Engelson, S. P. (1996). Integrating multiple classifiers by finding their areas of expertise. In *Proceedings of the AAAI-96 Workshop On Integrating Multiple Learned Models*.

Koppel, M., Feldman, R., & Segre, A. (1994a). Bias-driven revision of logical domain theories. *Journal of Artificial Intelligence Research, 1*, 159–208.

Koppel, M., Feldman, R., & Segre, A. (1994b). Getting the most from a flawed theory. In *Proceedings of the International Conference on Machine Learning* (pp. 139–147).

Lam, W. & Bacchus, F. (1994). Using new data to refine a Bayesian network. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence* (pp. 383–390).

Mahoney, J. J. (1996). Combining Symbolic and Connectionist Learning to Revise Certainty-Factor Rule Bases. Ph.D. Thesis, Department of Computer Sciences, University of Texas, Austin, TX. Also appears as Artificial Intelligence Laboratory Technical Report AI 96-260.

Mahoney, J. J. & Mooney, R. J. (1994). Comparing methods for refining certainty-factor rule bases. In *Proceedings of the International Conference on Machine Learning*, New Brunswick, NJ (pp. 173–180).

Merz, C. J., Murphy, P. M., & Aha, D. W. (1996). UCI repository of machine learning databases. [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Department of Information and Computer Science, University of California, Irvine, CA.

Ortega, J. (1995). On the informativeness of the DNA promoter sequences domain theory. *Journal of Artificial Intelligence Research, 2*, 361–367.

Ortega, J. & Fisher, D. (1995). Flexibly exploiting prior knowledge in empirical learning. In *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 1041–1049).

Ourston, D. & Mooney, R. (1994). Theory refinement combining analytic and empirical methods. *Artificial Intelligence, 66*(2), 273–309.

Pazzani, M. & Brunk, C. (1991). Detecting and correcting errors in rule-based expert systems: An integration of empirical and explanation-based learning. *Knowledge Acquisition, 3*, 157–173.

Pazzani, M. & Kibler, D. (1992). The utility of prior knowledge in inductive learning. *Machine Learning, 9*, 57–94.

Pearl, J. (1988). *Probabilistic Reasoning In Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufman.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Ramachandran, S. & Mooney, R. J. (1998). Theory refinement for Bayesian networks with hidden variables. In *Proceedings of the International Conference on Machine Learning* (pp. 454–462). Madison, WI: Morgan Kaufman.

Russell, S., Binder, J., Koller, D., & Kanazawa, K. (1995). Local learning in probabilistic networks with hidden variables. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Montreal, Canada (pp. 1146–1152).

Saitta, L., Botta, M., & Neri, F. (1993). Multistrategy learning and theory revision. *Machine Learning*, *11*, 153–172.

Towell, G. & Shavlik, J. (1993). Extracting refined rules from knowledge-based neural networks. *Machine Learning*, *13*(1), 71–101.

Received June 4, 1996
Revised September 23, 1999
Final manuscript September 23, 1999