Theory revision with queries: results and problems

Judy Goldsmith[1]*, Robert H. Sloan[2]**, Balázs Szörényi[3]***, and György Turán[3,4]†

[1] Computer Science Department, University of Kentucky, Lexington, KY 40506-0046, USA
[2] Department of Computer Science, University of Illinois at Chicago, Chicago, IL 60607-7053, USA
[3] Hungarian Academy of Sciences and University of Szeged, Research Group on Artificial Intelligence, H-6720 Szeged, Aradi vértanúk tere 1, Hungary
[4] Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, Chicago, IL 60607-7045, USA

Abstract. A brief overview is given of recent results on theory revision with queries for propositional formulas, such as monotone and unate DNF, Horn formulas, read-once formulas, and threshold functions. Several open problems are formulated for revision in propositional and predicate logic, and finite automata.

1 Introduction

Theory revision is a branch of machine learning where it is assumed that the learning process starts with an initial concept that is an approximation of the target concept. A typical example is an initial version of an expert system provided by an expert, which needs to be refined using further examples or other information available. It is argued that in order to learn a complex classification rule efficiently, one needs such an approximation. Descriptions of theory revision systems are given, for example, in Ourston and Mooney [15], Richards and Mooney [16] and Wrobel [21, 22]. One of the first papers studying revision from a theoretical aspect is due to Mooney [14]. He assumed that the target theory can be obtained from the initial theory by syntactic modifications, such as the deletion or the addition of a literal, and gave bounds for the the number of random examples needed in the PAC model for revision in terms of the number of these modifications necessary. Greiner [11] considered the computational complexity of hypothesis finding in a related framework.

In this note we describe some results from our work on theory revision, given in the papers [8–10, 17, 18]. We also formulate several open problems suggested by these results.

2 Definitions

We primarily use the model of learning with equivalence and membership queries (see, e.g., Angluin [2]). In an equivalence query the learner proposes a hypothesis h; if h classifies every instance as the target c then the answer is "correct", otherwise the answer is a counterexample, that is., any instance x such that $h(x) \neq c(x)$. In this paper we assume that equivalence queries are proper, that is, they belong to the target class. The only exception is the case of Horn formulas, where we use "almost proper" equivalence queries (see below). In a membership query, the learner presents an instance x, and receives the classification of that instance.

This learning model is of interest in computational learning theory partly because many important learning problems, such as propositional Horn formulas and finite automata, have efficient learning algorithms in this model, but do not have efficient algorithms in less powerful models. Furthermore, in some applications, such as the construction of expert systems and natural language applications, it seems reasonable to assume that the learning algorithm has the option of asking membership queries. Equivalence queries can be simulated by random examples; alternatively, such an algorithm can be used to find a hypothesis which is consistent with a set of counterexamples to the initial theory, given in advance.

Some of the results use another standard learning model: mistake bounded learning [13], which proceeds in a sequence of rounds. In each round the learner receives an instance, and produces a prediction

* Email: goldsmit@cs.uky.edu. Partially supported by NSF grants CCR-0100040 and ITR-0325063.
** Email sloan@uic.edu. Partially supported by NSF grant CCF-0431059.
*** Email: szorenyi@rgai.inf.u-szeged.hu.
† Email: gyt@uic.edu. Partially supported by NSF grant CCF-0431059.

$\hat{y}$ of its classification. Then the correct classification y is revealed. If $\hat{y} \neq y$ then the learner made a mistake. The mistake bound of the learning algorithm is the maximal number of mistakes, taken over all possible runs, that is, sequences of instances.

A mistake-bounded learning algorithm can be thought of as an equivalence query learning algorithm, where the equivalence queries correspond to the predictions at each stage of the algorithm. These queries are usually improper. In the example considered below, the target class is monotone disjunctions and the hypothesis class is monotone threshold functions. Thus, proper equivalence and membership query algorithms and mistake-bounded algorithms are incomparable in general.

The syntactic revision operators can, in general, be either deletion or addition type. The definition of these operators may depend on the target class. A *deletion* operation is fixing a literal occurrence to 0 or 1 (in a DNF or CNF expression the effect of such an operation is the deletion of a literal, a term, or a clause). An *addition* operation for DNF expressions is the addition of a literal to a term, and for propositional Horn formulas it is the addition of a new literal to the *body* of a clause. It appears to be the case that addition type revisions are more difficult to handle than deletion type revisions (this is intuitively to be expected, and is also borne out by experience in most, though not all, cases).

The *revision distance* between the initial theory and the target theory is the minimal number of revision operations needed to transform the initial theory to a theory equivalent to the target. A revision algorithm is considered to be *efficient* if the number of queries is polynomial in the revision distance, and only *polylogarithmic* in the total number of variables. The concept classes may have additional parameters (such as bounds on the number of terms or on the size of the terms), which are considered to be constants appearing in the bounds. We use d for the revision distance, n for the total number of variables, m for the number of terms, and k for the maximal size of the terms. There is an interesting connection between efficient revision algorithms and *attribute efficient learning* [5, 6].

For certain types of revision problems, such as revising finite automata, revision operators unique to the problem domain must be used.

3 Results

The following two theorems summarize most of our positive results in the query model. The first theorem contains revision algorithms for the deletion model, and the second theorem contains results for the general model of deletions and additions. The revision algorithms for Horn formulas use CNF hypotheses with each clause being a revision of a clause from the initial formula, but one initial clause can have multiple revisions in a hypothesis.

Theorem 1. *In the deletion model of revisions*

(a) monotone k-DNF formulas can be revised with $O(k \cdot d \cdot \log n)$ queries,
(b) m-term monotone DNF formulas can be revised with $O(m \cdot d + m^2)$ queries,
(c) m-clause Horn formulas can be revised with $O(m^3 \cdot d + m^4)$ queries,
(d) read-once formulas can be revised with $O(d \cdot \log n)$ queries.

The *graph* of a Horn formula has the variables as its vertices, and has an edge from every variable in the body of a rule to the variable in the head (with extra vertices T and F defined in the natural way). A Horn formula is *depth-1 acyclic*, if its graph is acyclic and has depth 1. A Horn formula is *definite with unique heads* if the heads of the clauses exist and are all different. A *0–1 threshold function* is 1 if and only if at least t of its relevant variables are set to 1 (for this class revision means deleting or adding a relevant variable, or changing the threshold by any amount).

Theorem 2. *In the deletion and addition model of revisions*

(a) m-term monotone DNF formulas can be revised with $O(m^3 \cdot d \cdot \log n)$ queries,
(b) 2-term unate DNF formulas can be revised with $O(d^2 \cdot \log n)$ queries,
(c) depth-1 acyclic m-clause Horn formulas can be revised with $O(m^3 \cdot d \cdot \log n)$ queries,
(d) definite m-clause Horn formulas with unique heads can be revised with $O(m^3 \cdot d + d \cdot \log n + m^5)$ queries,
(e) 0–1 threshold functions can be revised with $O(d \cdot \log n)$ queries.

A DNF formula is *k-projective* if it is of the form

$$\varphi = \rho_1 t_1 \vee \cdots \vee \rho_\ell t_\ell,$$

where every ρ_i is a conjunction of size k, every t_i is a conjunction and for every i it holds that $\rho_i\varphi \equiv \rho_i t_i$. This class of DNF formulas was introduced by Valiant [19] (see [18, 19] for motivation and basic properties). The revision distance of an initial k-projective DNF φ_0 and a target k-projective DNF φ of forms

$$\varphi_0 = \rho_1 t_1 \vee \cdots \vee \rho_\ell t_\ell \vee \rho_{\ell+1} t_{\ell+1} \vee \cdots \vee \rho_{\ell+a} t_{\ell+a}$$
$$\varphi = \rho_1 t_1^* \vee \cdots \vee \rho_\ell t_\ell^* \vee \rho_1' t_1' \vee \cdots \vee \rho_b' t_b'$$

is defined to be (somewhat differently from the previous definition for DNF)

$$a + \sum_{i=1}^{\ell} |t_i \oplus t_i^*| + \sum_{i=1}^{b} \max(|t_i'|, 1),$$

where $|t|$ is the size of t, and $|t \oplus t'|$ is the number of literals in the symmetric difference of t and t'.

The following theorem gives an efficient revision algorithm for k-projective DNF. It uses a slight modification of Valiant's attribute efficient learning algorithm [19], which is based on Littlestone's Winnow algorithm [13]. A generalization of this result to the case of noisy examples is presented in [17].

Theorem 3. *In the deletion and addition model of revisions k-projective DNF can be revised with $O(k \cdot d \cdot \log n)$ mistakes.*

As a counterpart to the previous positive results, we mention a negative result. Consider the variables $x_1, \ldots, x_n, y_1, \ldots, y_n$, let

$$t_i = x_1 \wedge \cdots \wedge x_{i-1} \wedge x_{i+1} \wedge \cdots \wedge x_n \wedge y_i$$

and $\varphi = t_1 \vee \cdots \vee t_n$.

Theorem 4. *In the deletion model of revisions at least $n-1$ queries are needed to revise φ_n, even if it is known that exactly one literal y_i is deleted.*

Theorem 1 shows that in the deletion model a monotone DNF can be revised efficiently, if it has small terms (part *(a)*), few terms (part *(b)*) or few occurrences of each variable (part *(d)* applied to read-once DNF). Theorem 4, on the other hand, shows that if neither of these conditions hold then efficient revision is not always possible.

In computational learning theory one considers the complexity of learning a target concept from a given concept class. For a revision algorithm, the concept class consists of revisions of the initial concept. Thus the complexity of this revision task associates a learning complexity to a single concept. (More precisely, we have several learning complexities, depending on the edit distance bound.) This appears to be a novel feature of revision algorithms. Thus, Theorem 4 refers to the revision complexity of a specific formula. Theorems 1, 2 and 3, in this sense, provide meta-algorithms that work for revising a whole class of formulas.

4 Two simple revision algorithms

In this section we give two simple examples of efficient revision algorithms. The first revises monotone conjunctions with equivalence and membership queries. The second is Winnow, which is shown to revise monotone disjunctions efficiently in the mistake bounded model. Compared to these simple algorithms, the main complication arising in the revision algorithms described in Theorems 1, 2 and 3 above is the ubiquitous credit assignment problem.

Proposition 1. *In the deletion and addition model of revisions monotone conjunctions can be revised with $O(d \cdot \log n)$ queries.*

Proof. Let us assume that the variables are $x_1, \ldots, x_n$, we would like to revise the initial conjunction $t = x_1 \wedge \cdots \wedge x_r$, and the target conjunction is h. The first equivalence query is t. If we receive a *positive* counterexample $a = (a_1, \ldots, a_n)$ then, as $t(a) = 0$, it must be the case that $a_i = 0$ for some i, $1 \le i \le r$. As $h(a) = 1$, this means that x_i does not occur in h. Thus we have identified a variable that is deleted from t. If we receive a *negative* counterexample $a = (a_1, \ldots, a_n)$ then there is an $i > r$ such that $a_i = 0$ and x_i occurs in the target. Let $a^{(j)}$ be the vector obtained from a by switching the first j 0-bits of a to 1. Then $a = a^{(0)} < a^{(1)} < \cdots < \mathbf{1}$ (where $\mathbf{1}$ is the all ones vector and $<$ is the componentwise partial ordering of $\{0,1\}^n$), thus $0 = h(a^{(0)}) < h(\mathbf{1}) = 1$. Using $O(\log n)$ membership queries implementing a binary search, we find a j such that $h(a^{(j)}) = 0$ and $h(a^{(j+1)}) = 1$. The variable switched on in $a^{(j)}$ to get $a^{(j+1)}$ is a variable added to t. Thus in both cases we found a variable deleted from, or added to t, using $O(\log n)$ queries. Repeating this process, h is identified with $O(d \cdot \log n)$ queries. □

Next we give a revision algorithm for monotone disjunctions, in the mistake bound model.

Theorem 5. *In the deletion and addition model of revisions monotone disjunctions can be revised with $O(d \cdot \log n)$ mistakes.*

Proof. We use the Winnow algorithm. Let the set of variables be $x_1, \ldots, x_n$ and the initial disjunction be $x_1 \vee \cdots \vee x_k$. The algorithm maintains a weight w_i for each variable x_i. Initially $w_1 = \cdots = w_k = n$ and $w_{k+1} = \cdots = w_n = 1$. In each round, given weights $w_1, \ldots, w_n$ and an instance $x = (x_1, \ldots, x_n)$, we predict $\hat{y} = 1$ if $w_1 x_1 + \cdots + w_n x_n \geq n$, and we predict $\hat{y} = 0$ otherwise. If a mistake is made, that is, if $\hat{y} \neq y$, then those weights w_i for which $x_i = 1$ are updated to $w_i\, 2^{(y-\hat{y})}$. The case when $\hat{y} = 0$ and $y = 1$ (resp., $\hat{y} = 1$ and $y = 0$) is called a *promotion* (resp., *demotion*).

The analysis of the mistake bound is as follows. Consider first an i such that x_i is in both the initial and the target disjunction. Any time that $x_i = 1$, the hypothesis correctly predicts that the disjunction is 1, so w_i is never updated, it always (correctly) remains at n.

In a promotion step, the weight w_i of some variable x_i that needs to be added to the initial disjunction to get the target disjunction, is doubled. The weight of such a variable is never halved. Once it reaches n, it will never be doubled again. Thus if there are altogether d_1 variables to be added, there can be at most $d_1 \lceil \log_2 n \rceil$ promotion steps. The total weight increase in each promotion step is less than n, as the weights to be doubled add up to less than n before the promotion. Thus the total weight increase over all promotion steps is at most $d_1 \lceil \log_2 n \rceil\, n$.

In a demotion step, none of the variables in the target disjunction have value 1. The variables that are set to 1 must have weights totaling at least n, so we decrease the weights of variables *not in the target disjunction* by a total of at least $n/2$.

Let d_2 be the number of variables that need to be deleted from the initial disjunction to obtain the target disjunction. The total weight to decrease over all demotion steps can be at most $n + d_2 n + d_1 \lceil \log_2 n \rceil\, n$. Here the n term is from the initial weight 1 of the at most n variables in neither the initial nor the target disjunction, the $d_2\, n$ term is from the d_2 variables to be deleted, and the $d_1 \lceil \log_2 n \rceil\, n$ term is from the weight added in the promotion steps. Dividing by $n/2$, we get that the number demotion steps is at most $2 + 2d_2 + 2d_1 \lceil \log 2n \rceil$. Since $d = d_1 + d_2$, this gives the desired mistake bound. □

It is interesting to note that Winnow is *not* an efficient revision algorithm for 0–1 threshold functions, if, as in Theorem 2 *(e)* above, one also allows the revision of the threshold. To see this, consider the initial concept $x_1 \vee \cdots \vee x_n$ (true iff at least one variable is on) and assume that we would like to revise it to the concept which is true iff at least *two* variables are on. Then each unit vector is a negative counterexample to the initial concept, and Winnow will only change the weight of a single component for each such counterexample. Thus, altogether, it will make at least n mistakes.

5 Open problems

We mention several open problems. A general question is to simplify the existing revision algorithms (the description and analysis of the 2-term unate DNF revision algorithm, for example, takes up more than 15 pages).

Problem 1 *In the deletion and addition model, is there an efficient revision algorithm for*

(a) m-term unate DNF formulas,
(b) m-clause Horn formulas?

It would be especially interesting from the practical point of view to be able to handle Horn formulas with deletions and additions, as this is the class used by several revision systems. On the other hand, we note that Horn revision can mean different things in different contexts (see [9] for a detailed discussion of this issue). A technical problem for the revision of Horn formulas is to replace the "almost proper" equivalence queries in Theorems 1 and 2 by proper ones.

Another problem concerns extending the model of revisions with addition.

Problem 2 *Are there efficient revision algorithms when the addition operator also allows for the addition of a literal to start a new term or clause?*

The work described above deals with revision in propositional logic only. It appears to be a very interesting topic for further research to study similar revision algorithms in predicate logic.

Problem 3 *Are there efficient revision algorithms in predicate logic using queries?*

We mention two possible starting points. Learning Horn formulas in predicate logic with queries has been studied in several papers, for example [3, 4, 12]. As a first step for revisions, one could look at the class of universally quantified function-free Horn expressions in the different setups considered by Khardon [12]. Another interesting framework is the *robust logic* of Valiant [20]. Valiant argues for the importance of attribute efficient learning in this model, which is both logic-based and is biologically realistic from certain aspects. Extending Valiant's argument, there appears to be motivation to consider efficient revision here as the modification of previously learned, or innate concepts. Theorem 3 above gives an efficient revision algorithm in a somewhat related framework. Adding to the problems mentioned in Problem 1 above, it may be useful here to extend the results of Theorem 2 *(e)* from threshold functions to conjunctions of threshold functions.

Another domain where revision appears to be interesting is finite automata. Some of the earliest work on learning with equivalence and membership queries showed that finite automata can be learned in this model [1]. Syntactic revision operators for finite automata can be classified by two parameters: Whether they are addition or deletion type operators, and whether they act only on *transitions* or on *states* as well.

Problem 4 *Are there efficient revision algorithms for deterministic finite automata*

(a) in the transition operators model,
(b) in the transition and state operators model?

Goldsmith, Homer, and Klapper [7] conjecture that the answer to these questions is negative. We are not aware of any results on the related *computational* problem of finding the minimal number of revisions of a finite automaton to get an automaton equivalent to a given target automaton.

Similar questions can also be asked about grammars. Here, syntactic revision operators for context-free grammars could be classified by two parameters: Whether they are addition or deletion type operators, and whether they act on productions or on nonterminals.

References

1. D. Angluin. Learning regular sets from queries and counterexamples. *Inform. Comput.*, 75(2):87–106, Nov. 1987.
2. D. Angluin. Queries revisited. *Theoret. Comput. Sci.*, 313(2):175–194, 2004. Special issue for ALT 2001.
3. M. Arias and R. Khardon. Learning closed Horn expressions. *Inform. Comput.*, 178(1):214–240, 2002.
4. H. Arimura. Learning acyclic first-order Horn sentences from entailment. In *Algorithmic Learning Theory, 8th International Workshop, ALT '97, Sendai, Japan, October 1997, Proceedings*, volume 1316 of *Lecture Notes in Artificial Intelligence*, pages 432–445. Springer, 1997.
5. A. Blum, L. Hellerstein, and N. Littlestone. Learning in the presence of finitely or infinitely many irrelevant attributes. *J. of Comput. Syst. Sci.*, 50(1):32–40, 1995. Earlier version in 4th COLT, 1991.
6. N. Bshouty and L. Hellerstein. Attribute-efficient learning in query and mistake-bound models. *J. of Comput. Syst. Sci.*, 56(3):310–319, 1998.
7. J. Goldsmith, S. Homer, and A. Klapper. Personal communication, Spring 2000.
8. J. Goldsmith, R. H. Sloan, B. Szörényi, and G. Turán. New revision algorithms. In *Algorithmic Learning Theory, 15th International Conference, ALT 2004, Padova, Italy, October 2004, Proceedings*, volume 3244 of *Lecture Notes in Artificial Intelligence*, pages 395–409. Springer, 2004.
9. J. Goldsmith, R. H. Sloan, B. Szörényi, and G. Turán. Theory revision with queries: Horn, read-once, and parity formulas. *Artificial Intelligence*, 156:139–176, 2004.
10. J. Goldsmith, R. H. Sloan, and G. Turán. Theory revision with queries: DNF formulas. *Machine Learning*, 47(2/3):257–295, 2002.
11. R. Greiner. The complexity of theory revision. *Artificial Intelligence*, 107:175–217, 1999.
12. R. Khardon. Learning function-free Horn expressions. *Machine Learning*, 37(3):241–275, 1999.
13. N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.

14. R. J. Mooney. A preliminary PAC analysis of theory revision. In T. Petsche, editor, *Computational Learning Theory and Natural Learning Systems*, volume III: Selecting Good Models, chapter 3, pages 43–53. MIT Press, 1995.
15. D. Ourston and R. J. Mooney. Theory refinement combining analytical and empirical methods. *Artificial Intelligence*, 66:273–309, 1994.
16. B. L. Richards and R. J. Mooney. Automated refinement of first-order Horn-clause domain theories. *Machine Learning*, 19:95–131, 1995.
17. R. H. Sloan and B. Szörényi. Revising projective DNF in the presence of noise. In *Proc. Kalmár Workshop on Logic and Computer Science*, pages 143–152, Szeged, Hungary, Oct. 2003. Dept. of Informatics, University of Szeged.
18. R. H. Sloan, B. Szörényi, and G. Turán. Projective DNF formulae and their revision. In *Learning Theory and Kernel Machines, 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003, Proceedings*, volume 2777 of *Lecture Notes in Artificial Intelligence*, pages 625–639. Springer, 2003.
19. L. G. Valiant. Projection learning. *Machine Learning*, 37(2):115–130, 1999.
20. L. G. Valiant. Robust logics. *Artificial Intelligence*, 117:231–253, 2000.
21. S. Wrobel. *Concept Formation and Knowledge Revision.* Kluwer, 1994.
22. S. Wrobel. First order theory refinement. In L. De Raedt, editor, *Advances in ILP*, pages 14–33. IOS Press, Amsterdam, 1995.