Note: More Efficient Conversion of Equivalence-Query Algorithms to PAC Algorithms

Ricard Gavaldà*
Department of Software (LSI)
LARCA Research Group
Universitat Politècnica de Catalunya

May 8th, 2008

Abstract

We present a method for transforming an Equivalence-query algorithm using Q queries into a PAC-algorithm using $\frac{Q}{\epsilon} + O(\frac{Q^{2/3}}{\epsilon} \log \frac{Q}{\delta})$ examples in expectation. The method is a variation of that by Schuurmans and Greiner which provides, for each $\gamma > 0$, an algorithm using $(1+\gamma)\frac{Q}{\epsilon} + O(\frac{1}{\epsilon} \log \frac{Q}{\delta})$ examples in expectation. In other words, we show that the constant in front of the dominating term Q/ϵ can be made $1 + o(1)$.

1 Introduction

In her seminal paper on learning from queries, Angluin [Ang87] showed that algorithms using Equivalence queries can be rewritten as PAC algorithms. Her simulation uses a worst-case sample $O(\frac{Q^2}{\epsilon} \ln \frac{1}{\delta})$ to achieve (ϵ, δ)-confidence from an algorithm using Q Equivalence queries, but it is not difficult to show that in her same simulation, sample size $O(\frac{Q}{\epsilon} \ln \frac{Q}{\delta})$ suffices.

*Partially supported by the EU PASCAL2 Network of Excellence, and by the Spanish Ministry of Education through the MOISES-BAR project, TIN2005-08832-C03-03. gavalda@lsi.upc.edu, http://www.lsi.upc.edu/~gavalda.

It was shown later that, with a diferent algorithm, that the dependence on n can be made linear. Specifically, Littlestone [Lit89] showed that there is a simulation using a worst-case sample size $4\,\frac{Q}{\epsilon} + O(\frac{1}{\epsilon}\,\ln\frac{Q}{\delta}))$ (his simulation was phrased in terms of on-line learning rather than Equivalence queries, but the distinction is irrelevant for our purpose). Schuurmans and Greiner [SG95, Sch96] showed how to build, for every constant $\gamma > 0$, a simulation that uses *expected* sample size $(1+\gamma)\,\frac{Q}{\epsilon} + c(\gamma)\frac{1}{\epsilon}\,\ln\frac{Q}{\delta}$. Here $c(\gamma)$ is constant for each γ, but tends to infinity as γ tends to 0.

In this note we show that the leading constant in front of the Q/ϵ term can be made $1 + o(1)$, that is, arbitrarily close to 1 as Q grows. In fact, our algorithm is essentially the same as the Schuurmans-Greiner one, except that instead of using a fixed value for γ a priori, we let the value of γ decrease at a precisely controlled rate as the algorithm progresses.

2 The Algorithm

We view an Equivalence query algorithm as a particular case of a strategy for generating hypothesis from sequences of labelled examples. Given such an algorithm, we build a new algorithm `S`, given in Figure 1, which reads a sequence of example, uses the Equivalence-query strategy as a black box, and eventually outputs a hypothesis from those generated by the strategy. We will show that `S` is a PAC-learning algorithm.

Procedure `sprt` is Wald's Sequential Probability Ratio Test, discussed below, and also used in the Schuurmans-Greiner approach. The main difference with their method is that we do not fix a constant γ *a priori*, but rather use a different γ_i that varies with i. We will fix one particular setting for the sequence of γ_i to obtain our bound on the sample size used by `S`, but occasionally comment on the effect of using other values for γ_i.

We will argue that procedure `S` satisfies three conditions, which we formulate as theorems: Correctness, Completeness, and Efficiency.

Theorem 1 (Correctness) *The probability that* $\mathtt{S}(\epsilon,\delta)$ *outputs some* $h \in H$ *with* $error(h) > \epsilon$ *is less than* δ*.*

The completeness condition can be stated in many ways, of which the following is but one example:

Theorem 2 (Completeness) *If for some* i *we have that* $error(h_i) = 0$ *with probability* 1*, then* $\mathtt{S}(\epsilon,\delta)$ *stops with probability* 1*.*

Algorithm $\texttt{S}(\epsilon, \delta)$

Generate initial hypothesis h_1;
$i := 1$; $t := 0$;
while TRUE
    do
        $t := t + 1$;
        get a training example $(x_t, c(x_t))$, labelled by the unknown target c;
        if $h_i(x_t) \neq c(x_t)$ (i.e., $(x_t, c(x_t))$ is a counterexample for h_i)
          then
                use (x_t, y_t) to generate h_{i+1};
                start testing $error(h_i)$ on subsequent examples
                        using $\texttt{sprt}(\epsilon/(1+\gamma_i), \epsilon, \delta/(i(i+1)), 0)$;
                $i := i + 1$;
        if for some $j < i$, the $\texttt{sprt}$ test for h_j rejects
          then
                drop h_j from the list of hypothesis being tested
        if for some $j < i$, the $\texttt{sprt}$ test for h_j accepts
          then
                output h_j and stop
end while

Figure 1: Algorithm $\texttt{S}$

Putting both claims together, if the strategy used to generate hypothesis is an exact Equivalence-query algorithm learning with finitely many queries, with probability 1 the algorithm stops, and its output is, with probability $1-\delta$, a hypothesis h having $error(h) < \epsilon$.

Theorem 2 in fact follows from this more general statement:

Theorem 3 (Running time) *Define $\gamma_i = i^{-1/3}$, and let the base Equivalence-query learner learn with at most Q queries. Then*

$$E[\textit{running time of } \mathtt{S}(\epsilon, \delta)] \leq \frac{Q}{\epsilon} + 7\,\frac{Q^{2/3}}{\epsilon} \cdot (\ln \frac{Q(Q+1)}{\delta} + 2).$$

We do not describe here the `sprt` test. We quote, however, some relevant properties from [Sch96], appendix A:

Theorem 4 *[Sch96] Let $k > 1$ and suppose $\mathtt{sprt}(\epsilon/k, \epsilon, \delta_{acc}, \delta_{rej})$ is run on a sequence X_1, X_2, ..., X_i,... of i.i.d. boolean random variables. Then:*

1. *If $E[X_i] > \epsilon$, the probability that `sprt` accepts is at most δ_{acc}.*
2. *If $E[X_i] < \epsilon/k$, the probability that `sprt` rejects is at most δ_{rej}.*
3. *([Sch96], Lemma A.4) If $\delta_{rej} = 0$, the expected running time of `sprt` is*

$$\left(\frac{k}{k-1-\ln k}\right) \frac{1}{\epsilon} \left(\ln \frac{1}{\delta_{acc}} + 1\right).$$

3 Proof of Theorem 1

The proof is as in [SG95, Sch96], but we reproduce it for completeness. We say that a hypothesis $h \in H$ is ϵ-bad iff $error(h) \geq \epsilon$. Observe that the `sprt` instance associated to h_i is fed boolean variables whose expected value is precisely $error(h_i)$. Therefore, by Theorem 4, part (1), we have the following (where probabilities are taken over infinite sequences of independently generated examples).

$$\begin{aligned} & \Pr[\mathtt{S}(\epsilon, \delta) \text{ outputs an } \epsilon\text{-bad hypothesis}] \\ \leq\ & \sum_{i=1}^{\infty} \Pr[h_i \text{ is } \epsilon\text{-bad yet } \mathtt{S}(\epsilon, \delta) \text{ outputs } h_i] \end{aligned}$$

$$\leq \sum_{i=1}^{\infty} \Pr[\texttt{sprt}(\epsilon/(1+\gamma_i), \epsilon, \delta/(i(i+1)), 0) \text{ accepts } h_i \mid h_i \text{ is } \epsilon\text{-bad}]$$

$$\leq \sum_{i=1}^{\infty} \frac{\delta}{i(i+1)} = \delta.$$

4 Proof of Theorem 3

For every i, we define the following random variables and expected values:

- h_i is the random variable representing the ith generated hypothesis,
- ϵ_i is such that $1/\epsilon_i = E[1/\mathit{error}(h_i)]$,
- T_i is the number of examples read from the moment in which h_i is generated until either h_{i+1} is generated (if h_{i+1} is ever generated; otherwise, $T_i = \infty$)
- R_i is the running time of the `sprt` test run on h_i, and
- T is the running time of the algorithm.

Proving Theorem 3 is thus bounding $E[T]$. Let i be the first index such that $\epsilon_i(1+\gamma_i) < \epsilon$. Note that if the base Equivalence learner uses at most Q queries, we have $i \leq Q$. Observe also that

$$T \leq \sum_{j<i} T_j + R_i \tag{1}$$

because, by definition of T_j and R_i, by this time h_i has been generated and the `sprt` test for h_i has stopped. Since the test is run with parameter δ_{rej}, it rejects h_i with probability 0, i.e., it accepts h_i. Therefore, by this time either `S` stops outputting h_i, unless it has stopped before due to another h_j.

Taking expectations in Equation (1), we have

$$E[T] \leq \sum_{j<i} E[T_j] + E[R_i]. \tag{2}$$

We first bound $E[T_j]$; the proof of the lemma is given later.

Lemma 1 $E[T_j] = 1/\epsilon_j$.

Taking $k = (1+\gamma_i)$ in Theorem 4, part (3), provides the following bound on $E[R_i]$:

$$E[R_i] \le \frac{1+\gamma_i}{\gamma_i - \ln(1+\gamma_i)} \, \frac{1}{\epsilon} \, (\ln \frac{i(i+1)}{\delta} + 1). \tag{3}$$

As a detour, let us note how to get the result in [SG95, Sch96]. Since i is the first index such that $\epsilon_i(1+\gamma_i) < \epsilon$, for $j < i$ we have $\epsilon_j \ge \epsilon/(1+\gamma_j)$, that is, $E[T_j] = 1/\epsilon_j \le (1+\gamma_j)/\epsilon$. Fix $\gamma_i = \gamma$ for every i. Then from Equation (2) we get

$$\begin{aligned} E[T] &\le \sum_{j<i} \frac{1+\gamma}{\epsilon} + \frac{1+\gamma}{\gamma - \ln(1+\gamma)} \, \frac{1}{\epsilon} \, (\ln \frac{i(i+1)}{\delta} + 1) \\ &= (1+\gamma)\frac{i}{\epsilon} + c(\gamma) \, \frac{1}{\epsilon} \, (\ln \frac{i(i+1)}{\delta} + 1). \end{aligned}$$

Now, take take instead $\gamma_i = i^{-1/3}$. We have the following two lemmas, whose proofs are given later:

Lemma 2 *For* $\gamma_j = j^{-1/3}$,

$$\sum_{j<i} (1+\gamma_j) \le i + \frac{3}{2} \, i^{2/3}.$$

Lemma 3 *Define* $c(\gamma) = (1+\gamma)/(\gamma - \ln(1+\gamma))$. *Then* $c(\gamma) \le 7/\gamma^2$ *for every* $\gamma \in (0,1]$, *and* $c(\gamma)$ *tends to* $2/\gamma^2$ *as* γ *tends to* 0.

From Equations (2) and (3) and Lemmas 2 and 3, and using again that for all $j < i$ we have $E[T_j] = 1/\epsilon_j \le (1+\gamma_j)/\epsilon$, we obtain

$$\begin{aligned} E[T] &\le \sum_{j<i} \frac{1+\gamma_j}{\epsilon} + \frac{7}{\gamma_i^2} \, \frac{1}{\epsilon} \, (\ln \frac{i(i+1)}{\delta} + 1) \\ &\le \frac{1}{\epsilon} \, (i + \frac{3}{2} \, i^{2/3}) + 7 \, \frac{i^{2/3}}{\epsilon} \, (\ln \frac{i(i+1)}{\delta} + 1) \\ &\le \frac{i}{\epsilon} + 7 \, \frac{i^{2/3}}{\epsilon} \, (\ln \frac{i(i+1)}{\delta} + 2) \end{aligned}$$

i.e., the statement of Theorem 3.

Proof of Lemma 1. Suppose that in a particular run of the algorithm the random variable h_j takes a particular value $h \in H$. Conditioned to $h_j = h$, the expected number of examples that have to be read to produce a counterexample for h_j is an exponential distribution with base $\mathit{error}(h)$, and therefore,

$$E[T_j | h_j = h] = \sum_{\ell=1}^{\infty} (1 - \mathit{error}(h))^{\ell-1} \cdot \mathit{error}(h) \cdot \ell = 1/\,\mathit{error}(h).$$

So $E[T_j] = E[1/\,\mathit{error}(h_j)]$ (where the expectation is taken over h on the right-hand side), which is $1/\epsilon_j$ by definition of ϵ_j. ∎ (Lemma 1)

Proof of Lemma 2. We show by induction on i the following inequality, which implies the lemma:

$$\sum_{j \le i} (1 + j^{-1/3}) \le \frac{i}{\epsilon} + \frac{3}{2} \frac{i^{2/3}}{\epsilon}.$$

For $i = 1$ it is obvious. Assume true for i, then

$$\sum_{j=1}^{i+1} j^{-1/3} \le \frac{3}{2} i^{2/3} + (i+1)^{-1/3}$$

and observe that

$$\frac{3}{2} i^{2/3} + (i+1)^{-1/3} \le \frac{3}{2} (i+1)^{2/3}$$

iff (multiplying on both sides by $(i+1)^{1/3}$)

$$\frac{3}{2} (i^2(i+1))^{1/3} + 1 \le \frac{3}{2} (i+1)$$

iff (taking cubes on both sides)

$$\left(\frac{3}{2}\right)^3 (i^2(i+1)) \le (\frac{3}{2}(i+1) - 1)^3$$

which is verified to be true by simple algebra. ∎ (Lemma 2)

Proof of Lemma 3. We have $c(1)1^2 = 2/(1 - \ln(2)) < 7$, and studying the Taylor expansion of $c(\gamma)\gamma^2$ shows that it is strictly increasing with γ, so $c(\gamma)\gamma^2 < 7$ for all $\gamma < 1$. Also, for small enough γ we have $\ln(1+\gamma) \cong \gamma - \gamma^2/2$, from which $c(\gamma) \cong 2/\gamma^2$ follows. ∎ (Lemma 3)

5 Final Remarks

Observe that Theorem 3 does not strictly require that the algorithm produces an hypothesis with 0 error within the first Q queries. It is enough to assume that within the first Q queries it generates a hypothesis h_i with $\epsilon_i(1+\gamma_i) < \epsilon$.

Note also that a variety of bounds on the sample size are possible by taking other definitions for γ_i. In particular, with essentially the same proof, if we take $\gamma_i = 1/i^\beta$ for $\beta < 1$, we obtain (approximately)

$$E[T] \leq \frac{Q}{\epsilon} + \frac{1}{1-\beta}\,\frac{Q^{1-\beta}}{\epsilon} + 7\,\frac{Q^{2\beta}}{\epsilon}\,\ln\frac{Q(Q+1)}{\delta}.$$

We just chose $\beta = 1/3$ to make $1 - \beta = 2\beta$, but if the values of Q and δ are known in advance, other values of β may give better bounds.

Finally, as indicated by Lemma 3, the factor 7 in front of the second term is actually a decreasing function of Q that tends to 2 as Q grows.

References

[Ang87] Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1987.

[Lit89] Nick Littlestone. From on-line to batch learning. In *COLT*, pages 269–284, 1989.

[Sch96] Dale Schuurmans. *Effective Classification Learning*. PhD thesis, Department of Computer Science, University of Toronto, 1996.

[SG95] Dale Schuurmans and Russell Greiner. Practical PAC learning. In *IJCAI*, pages 1169–1177, 1995.