Practical PAC Learning

Dale Schuurmans
Department of Computer Science
University of Toronto
Toronto, Ontario M5S 1A4, Canada
dalefIcs toronto edu

Russell Greiner
Siemens Corporate Research
Princeton, N,J 08540, USA
greinerOscr Siemens com

Abstract

We present new strategies for "probably approximately correct" (par) learning that use fewer training examples than previous approaches The idea is to observe training exam pies one-at-a-time and decide "on-line" when to return a hypothesis, rather than collect a large fixed-size training sample This yields *sequen tial* learning procedures that par-learn by observing a small random number of examples We provide theoretical bounds on the expected training sample size of our procedure — but establish its efficiency primarily by a scries of experiments which show sequential learning ac tually uses *many times* fewer training examples in practice These results demonstrate that pac-learning can be far more efficiently achieved in practice than previously thought

1 Introduction

We consider the standard problem of learning an accurate classifier from examples given a target classification scheme X \ —► *Y* defined on a domain X, we are interested in observing a sequence of training examples *{(T_l,c(x1)), ,(x_t,c(x_t)))* and producing a hypothesis *h X—> Y* that agrees with c on as much of the domain as possible Here we adopt the standard *batch* training protocol, where after a finite number of training examples the learner must produce a hypothesis *h* which is then tested *ad infinitum* on subsequent training examples

In practice, domain objects can be represented in many different ways (e *g* , boolean or real-valued vectors, or structured descriptions like strings, graphs terms, *etc),* and so too can hypotheses *(e g* , decision trees, neural networks, nearest neighbor classifiers, *etc*) How ever, regardless of the specific representation used, the central question is always how best to extrapolate the classifications of a few domain objects to an accurate classification scheme over the entire domain

Motivation *Classification learning* is by far the most studied in machine learning research The immense interest in this problem arises from the fact that *classification* itself is an important subtask in many applications — in fact, comprising the central function of most expert systems [Clancey 1985] The importance of /earning in this context is that w< often lack the requisite knowledge needed to specify an appropriate classifier, and yet have access lo many correctly classified examples In such situations, *we* can attempt to exploit the wealth of available data to overcome inadequate prior knowledge and hence use learning as an effective classifier *synthesis* tool In fact there are numerous examples where learning systems have produced classifiers that outperform the best available "hand-coded' systems, *c q* , [le (un e*t al* , 1989, Weiss and Kulikowski, 1991]

Although empirical research tends to examine the performance properties of particular hypothesis guessing strategies on specific domains, the underlying goal of classification learning research is to uncover whatever *general* principles might underly the effective extrapolation of training object classifications to entire domains However it has often been observed that there really is no such thing as a genera) purpose extrapolation strat egy [Schaffer 1994] — a particular strategy performs Well on a specific application only by *fortuitous predisposition* il just happens to 'guess right on unseen domain objects, whether by prior knowledge or luck lo *guaranttt* success one must supply prior constraints

The current trend towards *theoretical analysis* in machine learning represents a fundamental shift in emphasis away from discovering "universal" extrapolation strate gies towards explicitly acknowledging the role played by prior constraints in yielding successful extrapolat ion The role of a theoretical analysis is not to *prescribe* prior knowledge/constraints hut to determine the best that can be achieved *given* whatever is known beforehand

1 1 Pac-learning theory

The most influential analysis of classification learning is the theory of 'probably approximately correct" (par) learnmg introduced by Valiant [1984] Rather than spec ulate about the mechanisms that might underly gen eral purpose" classification learning Valiant's idea was to characterize those situations where successful learning could be *provably* achieved, and where it is demonstrably impossible

Problem Pac-learning theory adopts an "i i d random examples" model of the learning situation, which assumes domain objects are independently generated by a fixed distribution P_x and labelled according to a fixed target concept c $X \to \{0,1\}$ Under this model, the error of a hypothesis h $X \to \{0,1\}$ with respect to c and P_x is given by $\mathbf{P}_x\{h(x) \neq c(x)\}$ Here we consider the difficulty of meeting the so-called *pac criterion* producing a hypothesis *h* with error at most e, with probability at least 1 — *S,* for specified accuracy and reliability parameters e and *6* Of course, the difficulty of achieving this criterion depends on how much we know about *c* and P_K beforehand Pac-learning theory adopts a model of prior knowledge where we assume the target concept *c* belongs to some known class *C* but nothing is known about the domain distribution P_x Given this model, we naturally consider what, can be achieved in *the* "WORST case distribution-free' sense

Definition 1 (Pac-learning problem) *A learner L solves the pac learning problem $\langle X, C, \epsilon, \delta\rangle$ (or "pac(ϵ,δ) learns C") if for* any *c in C and* P_x, *L produces a hypothesis h such that $P_x\{h(x) \neq c(x)\} \leq \epsilon$ with probability at least $1-\delta$ (over possible training sequences)*

For example, we might be interested in solving the problem $\langle X = \mathbb{R}^{10}, C = \text{halfspace}, \epsilon = 0\ 01, \delta = 0\ 05\rangle$, where domain objects are described by 10 real-valued attributes, the target concept is known to be some linear-halfspacc of *JR.*[10], and we wish to produce a hypothesis with 1% error with probability at least 95% Our goal is to solve these learning problems as *efficiently* as possible - i. e., using a minimum of data and computational resources The primary focus of this paper is on improving the data-efficiency of pac-learncr6, rather than their *computational-* efficiency

Results Some of the most important technical results of pac-learning theory concern the *data* resources needed to solve pac-learning problems Intuitively, it should take more training examples to pac-learn a complex concept class than a simple one, since it is harder to disambiguate possible targel concepts from a complex class The question is how can one measure the representational complexity of a concept class *C* so as to precisely determine the number of training examples needed to pac(ϵ δ) learn C? It turns out that just such a measure is provided by the *Vapnik-Chtrvonenhs* (VC) dimension of *C* [1] Ehrenfeucht *et al* [1989] have shown that, for any concept class *C* with vc(C) — *d* the minimum number of training examples needed by any learner to pac(ϵ,δ)-learn *C* is at least $t_{EHKV}(C\ \epsilon\ \delta) = \max\left\{\frac{d-1}{32\epsilon}, \frac{1-\epsilon}{\epsilon}\ln\frac{1}{\delta}\right\}$ Furthermore there is a simple *fixed-sample-size* learning procedure, F, that always meets this lower bound tc within constant and log factors and hence learns with near-optimal data-efficiency see Figure 1 In particular, Blumer *et al* [1989] have shown that for any[2] concept class C, $T_{BEHW}(C, \epsilon\ \delta) = \max\left\{\frac{8d}{\epsilon}\log_2\frac{13}{\epsilon}, \frac{4}{\epsilon}\log_2\frac{2}{\delta}\right\}$ random training examples are sufficient to ensure F pac(ϵ δ)-learns C, where $d = \text{VC}(C)$ (This result has since been improved by Shawe-Taylor *ft al* [1993] to ' O v e r a l l , t h e s e arepowerful results as they characterize the necessary and sufficient training sample sizes needed to pac-learn *any* concept class *C* in terms of a ' tight" linear function of its VC dimension

Procedure $\mathbf{F}\ (C, \epsilon, \delta)$

COLLECT $T_{\mathbf{F}}(C, \epsilon\ \delta)$ training examples sufficient to eliminate all e-bad concepts from *C* with prob at least $1-\delta$

RETURN an $h \in C$ that correctly classifies every example

Figure 1 Procedure F

1 2 Issue

However, despite these impressive results, pac-learning theory has arguably had little direct impact on the actual practice of machine learning Why? Beyond criticisms of certain modelling assumptions (e *g* noise-free examples bivalent classifications, *etc* — which actually *have* been addressed the pac-framework, *cf* [Haussler, 1992]), the most prevalent criticism of pac-learning theory is that the *actual* numbers of training examples it demands are far too large to be practical

Example Consider the $\langle X = \mathbb{R}^{10}, C = \text{halfspaces}, \epsilon = 0\ 01, \delta = 0\ 05\rangle$ problem mentioned earlier Noting that vc(C) = 11, we can simply use T_{BEHW} to determine a sufficient sample size for Procedure F But here we find T_{BEHW} demands 91,030 training examples' (Even the improved T_{STAB} demands 15,981 examples in this rase) This seems like an outrageous number given the apparently modest parameter settings Moreover, these results compare poorly to the empirical "rule of thumb" that for a concept class defined by to free parameters, roughly $T_{thumb} = \frac{w}{\epsilon}$ training examples are needed to achieve an error of e [Baum and Haussler, 1989] Applied here, T_{thumb} demands only 1 100 training examples — an order of magnitude fewer than T_{STAB} (Of course, this rule of thumb comes with no guarantees, but it does give an indication of how many training examples practitioners would deem "reasonable" for this problem) Furthermore, T_{BEHW} and T_{STAB} r e orders of magnitude larger than the best known lower bound t_{EHKV} which demands only 32 training examples in this case' See Table 1 in Section 3 for a direct comparison

This shows that, although the theoretical upper and lower bounds are tight up to constant and log factors, they give results that are orders of magnitude apart in practice This has drastic consequences for the applicability of the theory, since in practice it is often training data, *not* computation time, that is the critical resource / e , cutting the training sample size in half would be a *significant* improvement in most applications, even if this came with a slight increase in overall computation time

The apparent inefficiency of pac learning has lead to much speculation about the sources of difficulty The

[1] The VCdimension measures how "fine grained" *C is* by the maximum number of domain objects *C* can independently label [Vapnik and Chervonenlas, 197l] This in an abstract combinatorial measure which applies to arbitrary domains and concept classes Moreover, it often gives intuitive results (*eg,* the class of halfspace concepts on $\mathbb{R}^n$ is defined by n +1 "free parameters" and also has a VCdimension of n + 1)

[2] *C* must satisfy certain (benign) measurabibty constraints [Blumer, *et al*, 1989], which we will assume throughout

predominant "folk wisdom" is that the large sample sizes follow from the *worst case* nature of the par-guarantees [Haussler, 1990] — that is, the worst case bounds are inherently unreasonable because they must Lake into account "pathological" domain distributions and target concepts which force large training sample sizes (moreover, the argument continues these pathological situations do not arise in ' typical' applications) In fact, this belief motivates much research that makes distributional assumptions in order to improve data-efficiency, e *g* , [Benedek and Itai, 1988 Aha, e*t al* 1991, Barllett and Williamson, 199l] However, notice that this line of reasoning is actually quite weak First of all, no-one can demonstrate that these "pathological' situations really exist (for this would be tantamount to improving the lower bound t_{EHKV}) Secondly, it is clear from the previous example that the current bounds are loose, and can likely be substantially improved t *g* T_{STAB} a-nd t_{EHKV} differ by roughly a factor of 64 In -

Approach In this paper we investigate an alternative view perhaps the simplistic (collect find) learning procedure F is not particularly data-efficient This raises the obvious question of whether alternative learning strategies might be more data-efficient than F Here we investigate *sequential* learning procedures that observe training examples one-at-a-time and autonomously deride "on-line' when to stop training and return a hypothesis The idea is that we should be able to detect situations where an accurate hypothesis can be reliably returned even before the sufficient sample si7C bounds have been reached *(c g* , we might detect that *C* has been reduced to a single possible target) The hope is that, in this way, we can significantly reduce the number of training examples, observed, while still meeting the *exact same* pac-criterion as before namely, that an e-accurate hypothesis be returned with probability at least $1-\delta$ for any target concept $c \in C$ and distribution P_x An underlying assumption here is that we are willing to incur a slight computational penalty to obtain a significant improvement in data-efficiency This is motivated by the fact that *training data* is usually the most critical resource in practical learning applications

The remainder of this paper develops a few simple sequential learning procedures that *(i)* are correct pat learners, *(n)* are provably data-efficient, and *(III)* use *many ttmes* fewer training examples in empirical case studies

2 Sequential pac-learning

A sequential /earner *L* consists of a *stopping rule* T_L, that maps training sequences to stopping times, and a *hypothesizer HL,* that maps finite training sequences to hypotheses Our basic strategy for constructing sequential pac-learners is to take an arbitrary *consistent hy-pothesizer* *H* for *C* (which produces hypotheses $h \in C$ that correctly classify every observed training example) collect H's hypotheses and test these against subsequent training examples until one proves to have sufficiently small error The main challenge is finding an appropriate stopping rule that guarantees the pac-cntenon, while observing as few training examples as possible

---

Procedure R (C ϵ δ H)

FIX a sequence $\{\delta_i\}_1^\infty$ such that $\sum \delta_i = \delta$ Call H to obtain an initial hypothesis h_0

SEQUENTIALLY observe training examples (x_t, y_t) $t = 1, 2$

IF current hypothesis h_i makes a mistake call H to obtain a consistent h_{i+1} (drop h begin testing h_{i+1})

RETURN current hypothesis h if it correctly classifies $\frac{1}{\epsilon} \ln \frac{1}{\delta}$ consecutive training examples

---

Figure 2 Procedure R

Note that in general a sequential learner observes a *random* rather than fixed number of training examples Thus to compare the data-efficiency of our approach with previous techniques we must comparr a *distribution* of sample sizes to a fixed number There are a number of ways one could do this but we focus on what is arguably the most natural measure comparing the *average (i e* expected) training sample size of a sequential learner with the *fixed* sample size demanded by previous approaches In solve the same pac-learning problem

Obvious approach Perhaps the rnost obvious strat egy for sequential pac learning is based on the ide a of *repeated significance testing*- test a series of hypotheses generated b\ *H* until one (orrettly classifies a sufficient number of consecutive training examples see Procedure R in figure *2* [3] Although this is a plaubible approach (which, in fact, works well in prat ticc) it is hard lo prove reasonable bounds on R s expected sample si7< The problem ii that R r<jecls "good enough hypotheses with high probability and yet takes a long time to do so (i c , R rejer ts hypotheses of error f with prob\bilily 1 $\cdot\delta$, but this takes | expected time) lhus, if *H* produces a series of ' borderline" hypotheses R will take a long time to terminate (expected time about $\frac{1}{\epsilon\delta}$, which is hot very good) fortunately there is a better approach

Better approach Here we introduce a novel learning procedure S (Figure i), which is also based on repeated significance testing but avoids the apparent inefficiency of R s survival testing approach S is based on two ideas First instead of throwing away *H* s hypotheses after a single mistake, S *saves* hypotheses and continues testing them until one proves to have small error Second, S identifies accurate hypotheses by using a se *quential probability ratio test* (sprt) [Wald, 1947] to test each candidate "on-line" (in parallel), Frigure *4* Thus, S never rejects a potentially acceptable hypothesis, and quickly identifies any sufficiently accurate candidate

Procedure S is a correct pac learner in the exac t same sense as F The key property of S is that its call to sprt eventually accepts any $\frac{\epsilon}{\kappa}$ good hypothesis with proba bility 1 (wpl), but only accepts an e-bad hypothesis h, with probability at most δ_i This implies that S even tually halts wpl, and returns an c-good hypothesis with

[3] Variants of Procc dure R have been proposed *by* many authors in the past [Liniual *tt al* 1991, Oblow, 1992], primarily to achieve "nonuniform" pac-learning However, the goals of nonuniform pac-leaming fundamentally differ from what we are trying to accomplish here (see Footnote 6)

Procedure S (C, ϵ, δ, H)

FIX a sequence $\{\delta_i = \frac{6\delta}{\pi^2 i^2}\}_1^\infty$ and a constant $\kappa > 1$. Initialize a list of hypotheses with h_0, obtained by calling H

SEQUENTIALLY observe training examples $\langle x_t, y_t \rangle$, $t = 1, 2, \ldots$

IF the most recent hypothesis h_i makes a mistake, call H to add a new, consistent hypothesis h_{i+1} to the list

TEST all hypotheses in the list (in parallel) by calling sprt$(h_i(x) \neq c(x), \frac{\epsilon}{\kappa}, \epsilon, \delta_i, 0)$ for each h_i (when generated)

RETURN the first generated hypothesis h_j sprt accepts

Figure 3 Procedure S

Procedure sprt $(\phi(x), a, r, \delta_{acc}, \delta_{rej})$

FOR boolean random variable $\phi(x)$, test H_{acc}: $P_X\{\phi(x)=1\} \leq a$ vs H_{rej}: $P_X\{\phi(x)=1\} \geq r$, with
- probability of deciding H_{acc} given H_{rej} bounded by δ_{acc},
- probability of deciding H_{rej} given H_{acc} bounded by δ_{rej}

SEQUENTIALLY observe $\phi_t = \phi(x_t)$, $t = 1, 2, \ldots$, monitoring

$$S_t = \sum_{j=1}^{t} \phi_j \ln \frac{a}{r} + (1 - \phi_j) \ln \frac{1-a}{1-r}$$

RETURN "accept" if ever $S_t \geq \ln 1/\delta_{acc}$

RETURN "reject" if ever $S_t \leq \ln \delta_{rej}$

Figure 4 Procedure sprt

probability at least $1 - \delta$, for *any* target concept $c \in C$ and domain distribution P_x (thug, achieving the exact same *worst case* pac-guarantees as F) [4] This property also allows US to prove a reasonable *upper bound* on the average number of training examples S observes for any target concept *cEC* and domain distribution P_x

Theorem 1 *For $\epsilon > 0$, $\delta > 0$, and any (well behaved) concept class C with* vc(C) = *d using a consistent hypothesizer H for C and any constant K> l, Procedure S observes an average training sample size of at most*

$$\mathrm{E}T_S(C, \epsilon, \delta) \leq \left(\frac{\kappa}{\kappa - 1 - \ln \kappa}\right) \frac{1}{\epsilon} \left([2\ 12\kappa d + 3] \ln \frac{14\kappa}{\epsilon} + \ln \frac{1}{\delta}\right)$$

for any target concept $c \in C$ and distribution P_x

Although this is a crude bound, it is interesting to note that it *scales* the same as T_{BEHW} and T_{STAB} Moreover, this bound actually *beats* T_{BEHW} and T_{STAB} for small values of *S* [Schuurmans, 1995J However, as shown below S actually performs *much* better in practice than any bounds we can prove about its performance Since this is *not* a possibility for fixed-sample-sized approaches, we expect S to perform much better t h T_{BEHW} n d T_{STAB} in practical applications

Before demonstrating S's advantage in empirical tests, we first note that there are inherent limits to the data-efficiency even of sequential learning

Theorem 2 For *sufficiently small c and 6, and any concept class* C *wtth* $\mathrm{vc}(C) = d \geq 2$ *any learner that always observes (for any fixed c £ C and Y_x) an average training sample size less than*

$$t_{avg}(C, \epsilon, \delta) = \max\left\{\frac{d-1}{480\epsilon}, \frac{1-\delta}{4\epsilon}\right\}$$

cannot meet the pac(ϵ, δ) criterion for all $c \in C$ and P_x

Notice that this lower bound scales the same as t_{EHKV} in terms of (and vc(C) — which shows that no *new* concept classes become pac-learnable merely by considering a sequential over fixed-sample-size approach

[4]Provided $\mathrm{vc}(C) < \infty$ (details omitted) Proofs of all results mentioned in this paper (and more) are outlined in [Schuurmans and Greiner, 1995] Complete details appear in [Schuurmans, 1995]

3 Empirical efficiency

Although the theoretical advantage we can demonstrate for S is only slight, we expect S to perform much better m practice than any bounds we can prove about its perfor mance This is because S's actual data-efficiency in any particular case study is determined by the *specific case at hand,* and not by *the worst case situation* (or, worse yet, what we can *prove* about the worst case situation) In fact, in empirical studies, S proves to be far more efficient than any bounds we can prove about its per formance, and *many times* more efficient than T_{BEHV}, or T_{StAB} This is easily demonstrated by a simple example

Illustration We tested Procedure S on the problem $\langle X = \mathbb{R}^n, C =$ halfspaces, $\epsilon = 0\ 01, \delta = 0\ 05\rangle$ with the following setup Training objects were generated according to a uniform distribution on $[-1, 1]^n$ and labelled by a fixed target halfspace (defined by a "diagonal" hyper plane passing through the origin 0" with norm directed towards $\mathbf{1}^n$) The constant *K* was set to 3 14619 (so thai $\frac{\kappa}{\kappa-1-\ln\kappa} = \kappa$), and we supplied S with a hypothesizer *H* that finds consistent halfspace concepts [5] 'We ran Procedure S 100 times for n = 10 and obtained the results shown in Table 1 Notice that S's average training sample size of 3,402 is about 5 *times* smaller than T_{STAB}, 27 *times* smaller than T_{BEHW}, and only about 3 timet, larger t h T_{thumb} t is important to emphasize that S obtains these empirical sample size improvements while maintaining the *exact same* worst case pac-guarantees as before (that an c-accurate hypothesis is returned with probability at least $1 - \delta$) These results are in fact representative over the entire range of parameter settings S's empirical advantage actually *improves* for increased problem dimension *n* (Figure 5), and is maintained at higher accuracy and reliability levels [Schuurmans, 1995] Overall, S appears to be pac-learning with near-practical data-efficiency in this example

Interestingly, S also outperforms the simplistic procedure R on this problem Figure 6 shows that, R performs nearly as well as S on easy problems (low dimension, accuracy, reliability), but S's advantage grows significantly as these parameters are scaled up

[5]Specifically, we used the BFGS secant optimization procedure [Dennis and Schnabel, 1983] with a "relaxation" objective function [Duda and **Hart,** 1973]

Explanations These results demonstrate a clear advantage for sequential over fixed-sample-size learning we solve the exact same pac-learning problem using far fewer training examples in this case Of course these preceding results are anecdotal, and it *is* tempting to explain away the advantage as a mere artifact of the specific experimental setup However, we have found that these experimental results are, in fact, quite robust

First, the previous experiment only tested a single domain distribution (uniform), which could happen to be a particularly "easy" one for S To counter this claim, we repeated the experiment with various domain distribu tions to see if any could seriously affect S's performance In particular we considered three different transformations of the uniform[—1,1]" distribution *spherical* (nonlinear compression towards origin), *pyramidal* (compression from opposite corners towards hyperplane) and *accretive* (translation towards discrete points m $\{-1\ 1\}^n$) Surprisingly, none *of* these transformations had any noticeable effect on S's performance [Sc huurmans, 1995], as demonstrrted in Figure 7 for the pyramidal case

A second reason for S's advantage might be that the specific target concept (diagonal halfspace) is a particularly "easy" one for S — if, H could somehow be biased to guess similar hypotheses However this is easily shown not to be the case We repeated the original experiment on 10 different target halfspaces each successively closer to "axis-parallel, and found that none of these made any appreciable difference, Figure 8

Third, it could be the case that the class of halfspaces <oncepts happens to be 'easy" among classes with comparable VOdimension This turns out to be partly true We have been able to construct alternative concept classes which force S to observe slightly more training examples, see Figure 9 However, we have yet to devise any roncept class (with the same \(dimension) that ran even double S's original performance on halfspaces In fact, S's performance often *imprnvts* for different concept classes (particularly *finite* ones) Overall, it appears that halfspaces is not a remarkably hard or easy class for a given VCdimension

Another explanation of S s advantage over F is that T_{STAB} might possibly be a gross overestimate of the true worst case situation (which seems likely given the gap between T_{STAB} and t_{SHAV}) Of course, this means that any current advantage enjoyed by S could potentially be overcome by future improvements to T_{STAB} — but notice that we can enjoy S s improved performance *immediately,* without having to wait for theoreticians to improve the bounds (Ensuring the correctness of F requires one to *prove* some bound is sufficient this is not a requirement for Procedure S since its correctness is completely decoupled from its. efficiency)

A final explanation of S s advantage is that sequential learning might be *inherently* more efficient than fixed sample-size learning Clearly since the sequential approach *generalizes* the fixed-sample-size approach, it can be no worse than F The question is how substantial an advantage can be obtained in principle? This is left largely unanswered by our empirical results and remains an interesting open topic for future research

For $\langle X = \mathbb{R}^{10}, C = \text{halfspaces}, \epsilon = 0\ 01, \delta = 0\ 05\rangle$

Sufficient	T_{BEHW}	=	91,030
Improved	T_{STAB}	=	15,981
Folklore	T_{thumb}	≈	1,100
Necessary	t_{SHAV}	=	32

After 100 trials, Procedure S used

avg T_S	=	**3,402**
max T_S	=	5 155
min T_S	=	2 267

Table 1 A direct comparison of training sample sizes for the pac learning problem $\langle \mathbb{R}^{10}$ halfspaces $\epsilon = 0\ 01$ $\delta = 0\ 05\rangle$

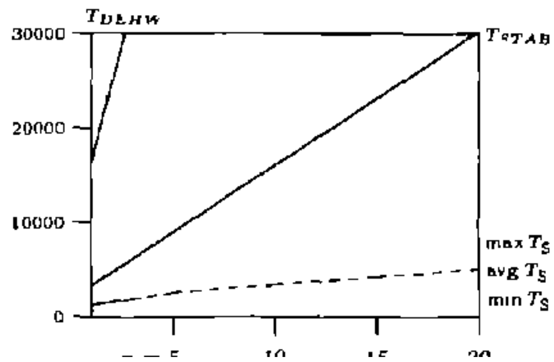

Figure 5 Scaling in input dimension n Number of training examples observed for $\langle R^n$ halfspaces, t = 0 *01,8* = 0 OS) with n = 1 2 3 5 10 15, 20 (Results of 100 runs each)

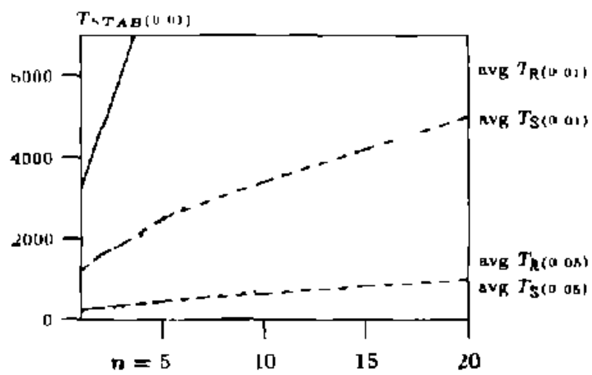

Figure 6 Comparing S versus R Number of training examples observed for $\langle \mathbb{R}^n$ halfspaces $\epsilon, \delta = 0\ 05\rangle$ with $n = 1, 2\ 3, 5, 10, 15, 20$ and $\epsilon = 0\ 01\ \ 0\ 05$ ($T_{STAB(0\ 01)}$ and T_{BEHW} not shown)

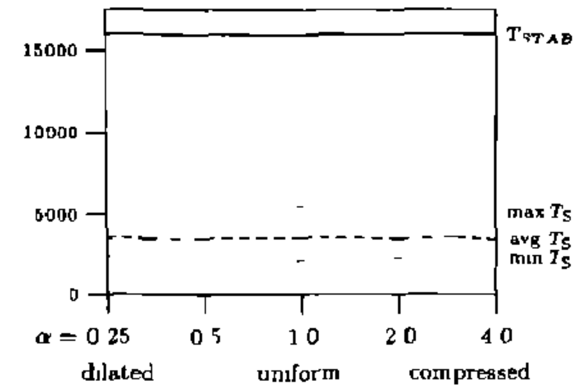

Figure 7 Comparing different domain distributions Results for $\langle \mathbb{R}^{10}$, halfspaces, $\epsilon = 0\ 01, \delta = 0\ 05\rangle$ under *pyramidal* transformations of the uniform$[-1\ 1]^{10}$ distribution \ *axis* power factor of transformed dot products

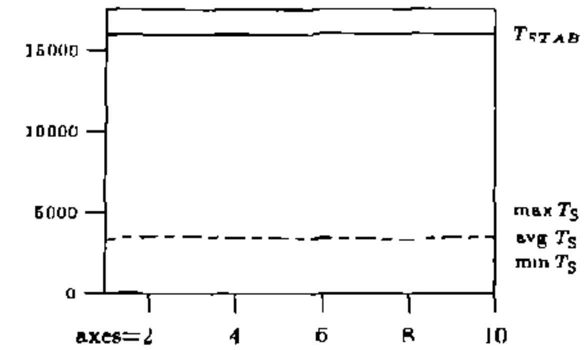

Figure 8 Comparing different target concepts Results for $\langle \mathbb{R}^{10}$, halfspaces $\epsilon = 0\ 01\ \delta = 0\ 05\rangle$ with 'diagonal" target concepts depending on $r = 1, 2$, 10 relevant axes

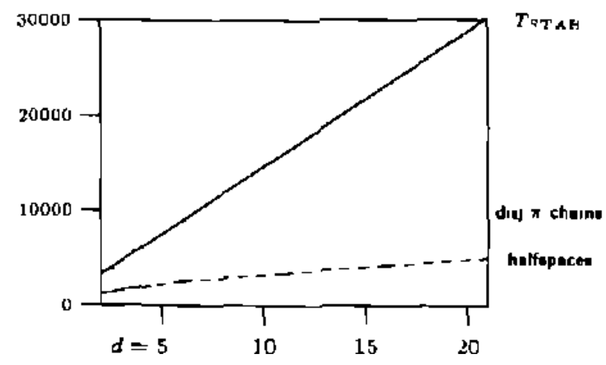

Figure 9 Comparing different concept classes with matching VCdimensions Average T_S for $\langle \mathbb{R}^n, C\ \ \epsilon = 0\ 01, \delta = 0\ 05\rangle$ with C_1 = halfspaces, C_2 = disj-π-chains, and $vc(C_i)$ = 2, 3, 4, 6, 11, 16 21 (The class of disj π-chains is defined by $\frac{1}{d_c}$ copies of a d-dimensional "product chain" of concepts where the concepts in different copies are mutually exclusive [Schuurmans, 1995] Thus class has VCdimension d)

Advantages Despite the empirical nature of these results, sequential learning holds many clear advantages over fixed-sample-size learning for solving pac-learning problems First, the sequential approach *decouples* the actual data-efficiency of a pac-learner from the precise bounds we can prove about its performance *a pnon* Thus, the actual data-efficiency of a sequential learner depends on the specifir case at hand, not on what W(can prove about the worst case situation Consequent!} the sequential approach *automatically* takes advantage of beneficial situations like 'easy" target concepts and domain distributions [Oblow, 1992], or a ' good" hypothcsizer that makes lucky guesses — without the system designer having to explicitly notice that these beneficial situations exist *a pnon* ' More importantly the true worst case data-efficiency of sequential learning depends on the *true* worst case convergence properties of the concept class, not on the particular bounds we happen to be able to prove at the time (i e , if bad concepts are eliminated sooner than proven bounds, then S automatically stops sooner) So in effect we are able to exploit the *optimal* worst case bounds right now, even though we are unable to prove exactly what tbey are

Computation Wc also note that Procedure S only introduces reasonable computational overhead over Procedure F and in fact is often *more* computational efficient than R' Although, at first glance, S appears to be extremely space-inefficient this *rarely* amounts to a significant expense in practical applications The point is that, in practice, it is the task of *finding* consistent hypotheses (calling *H)* that takes most of the work — storing hypotheses once (found (updating statistics, *etc)* does not require much overhead in comparison Consequently, R is often *slower* than S (even though it uses less space) simply because R Lends to call *H* more often

4 Additional results

Special cases We have obtained even stronger results in slightly restricted settings [Schuurmans and Gremer 1905] For example, a variant of Procedure S can serve as a sequential ' mistake bounded to pac" conversion procedure that is *provably* more efficient than Litllestone's fixed-sample-size procedure [Littlestone, 1989] (and which uses 30 dmes fewer training examples in empirical tests) We also obtain stronger improvements for the case of distribution *specific* pac-learmng (where we assume the learner *knows* P_x, but not the target concept $c \in C$) Notice that a *sequential* approach is still possible in this case, and, in fact a variant of Procedure S can pac-learn concept spaces (C, P_x) using 5 *times* fewer training examples than the best known fixed-sample-size procedure developed in [Benedek and Itai, 1988]

Range of applicability Beyond improving data-efficiency, sequential learning is also applicable to a much wider range of pac-learning problems than fixed-sample-size learning For example. Procedure S can be direcLly applied to "nearest neighbor" and "decision-tree" hypothesizers (like *CART* [Breiman, *et al*, 1984]) which implicitly consider concept classes of *infinite* VCdimension *No* fixed-sample-size bound can ever be sufficient in these cases, and yet Procedure S can be applied to pac learn these classes "as is " The only catch is that we can no longer place a uniform upper bound on S s expected training sample size [6]

5 Conclusion

Research directions There are numerous directions for future research First, since our empirical results address "artificial" learning problems, it would be interesting to test these procedures on real world' data sets (e g , as contained in the UCI repository of machine learning databases) to verify that the same empirical advantages can be realized there Another important re search direction is to extend our techniques lo deal with classification noise which remains the main barrier between the results presented here and real applications Finally, one can also consider a slightly different learning sconano which perhaps has more practical applications than pac-learning rather than first fixing the accuracy and reliability parameters and then determining sufficient sample size it is much more natural to take a fixed sample size, fix a reliability parameter and produce an estimate of the accuracy achieved by the learner's final hypothesis In this regard we are currently investigat mg a variant of Procedure S which produces hypotheses with small (but reliable[1]) error estimates

Contributions We have described a novr 1 pac-learning pro<edurc, S, that uses far fewer training examples than previous approaches Procedure S is, in effect generic test, procedure (hat can pac-learn arbitrary ran (ept rlasses C (with finite VC dimension), provided only that we can supply a hypothesizer H that produces consistent concepts from C This procedure introduces little computational overhead and yet substantially reduces the number of training examples needed to pac learn in practice — as demonstrated in numerous case studies where S used many times fewer training examples than the previous best known approaches while still maintaining the exact same worst case pac guarantees

In a way these results exploit the empirical advantage demonstrated by practical learning algorithms over the theoretical bounds, to improve the efficiency of pac-learmng Overall, our results show how pac learning can be far more efficiently achieved in practice than previously thought — countering the claim that pac learning can never be feasibly achieved in real applications

Acknowledgments

Thanks to Steven Shapiro for his help with the implementations

[6] It is important not Lo confuse the idea of sequential with nonuniform pac-learning [Lima] et al 1991, Oblow, 1992] Although nonuniform pac learning procedures also Use ' on-bne" stopping rules very similar to R they do not share the same theoretical advantages shown for S Sequen tial pac-learning, seeks to obtain a uniform improvement in data efficiency for all cases permitted by our prior knowledge whereas nonuniform pac learning sacrifices data-efficiency in some situations to obtain an improvement in others These two concerns are in fact orthogonal

References

[Aha et al , 199l] D Aha el al Instance-based learning al gonthms Machine Learning 6(1) 37-66 1991

[Bartlett and Williamson, 199l] P Bartlett and R William son Investigating the distributional assumptions of the pac learning model In COLT 91, pages 24-32, 1991

[Baum and Haussler 1989] E B aum and D Haussler What size net gives valid generalization[7] Neural Computation 1 151 160 1989

[Benedek and Itai 1988] G Benedek and A Itai Learnabil ity by fixed distributions In (OLT MS pages 80-90 1988

[Blumer et al 19891 A Blumer el al Leamabiht> and the Vapnik Chervonen dimension J A (M 36 929-965 1989

[Breiman, et al , 1984] L Breiman et al Classification and Regression Trees WadsWorth Belmont f A 1984

[Clancey 1985] W Clancev Heuristic classification Artifi cial Intelligence, 27 289-350 1985

[Dennis and Schnabel 1983] J Dennis and R Schnabel Nu men a! Methods for Unconstrained and Nonlinear Equa tions Prentice Hall Englewood ChlTs, NJ 1983

[Duda and Hart 1973] R Duda and P Hart Pattern Clas sification and Scene 4nalysis Wiley NewY\ork 1971

[Fhrenleucht et al 1989] A Ehrenfeuchl et al \ general lower bound on the number of examples needed for learn mg Information and (omputation 82 247-261, 1989

[Haussler, 1990] I) Haussler Probably approximately c or rect learning In A 4 41 90 pages 1101- 1108 1990

[Haussler 1992] D Haussler Decision theoretic gcneraliza tions of the PAC model Inf and (omp 100 78-150, 1992

[le (un el al 1989] Y Cun et al Backpropagation appli(d to handwritten zip code recognition Neural (ompu tation 1 541 551, 1989

[I itual et al 199l] N Linial et al Results on learnability and the Vapmk-Chervonenkis dimension Information and (omputation 90 33 49, 199]

[Littlestone, 1989] N I ittlestone from online to batch learning In (OLT S9, pages 269 284, 1989

[Oblow 1992] E Oblow Implementing Valiant a leanabilitv theory using random sets Mach Learn , 8 45-73, 1992

[Schaffer 1994] C Schaffer A conservation law for general izatiori performance In Proceeding* MI 94 1994

[Schuurmans and Greiner, 1995] D Schuurinaiis and R Grnner Sequential PAC learning In (O i l 95_t 1995

[Schuurmans 1995] D Schuurmans tfficttvc Classification Learning PhD thesis U Toronlo C omputer Sci , 1995

[Shawe Tavlor et al , 199 i] T Shawe Tavlor ct al Bound mg sample size with the Vapnik Chervonenkis dimension Discrete Applied Mathematics 42 65 73 1993

[Valiant, 1984] L Vahanl A theory of the learnable (A(M 27(11) 1134-1142, 1984

[Vapnik and C hervonenkis 1971] V vapnik and A (hervonenkiH On the uniform convergence of relative frequencies of events to their probabilities Theory of Probability and its Application* 16(2) 264 280, 1971

[Wald 1947] A Wald Sequential 4nalysu Wiley 1947

[Weiss and kuhkowski 199l] S Weiss and C Kuhkowski Computer Systems that Learn Morgan Kaufmanri, 1991