Improved Noise-Tolerant Learning and Generalized Statistical Queries

Javed A. Aslam* Scott E. Decatur†

Aiken Computation Laboratory
Harvard University
Cambridge, MA 02138

July 1994

Abstract

The statistical query learning model can be viewed as a tool for creating (or demonstrating the existence of) noise-tolerant learning algorithms in the PAC model. The complexity of a statistical query algorithm, in conjunction with the complexity of simulating SQ algorithms in the PAC model with noise, determine the complexity of the noise-tolerant PAC algorithms produced. Although roughly optimal upper bounds have been shown for the complexity of statistical query learning, the corresponding noise-tolerant PAC algorithms are not optimal due to inefficient simulations. In this paper we provide both improved simulations and a new variant of the statistical query model in order to overcome these inefficiencies.

We improve the time complexity of the classification noise simulation of statistical query algorithms. Our new simulation has a roughly optimal dependence on the noise rate. We also derive a simpler proof that statistical queries can be simulated in the presence of classification noise. This proof makes fewer assumptions on the queries themselves and therefore allows one to simulate more general types of queries.

We also define a new variant of the statistical query model based on *relative error*, and we show that this variant is more natural and strictly more powerful than the standard additive error model. We demonstrate efficient PAC simulations for algorithms in this new model and give general upper bounds on both learning with relative error statistical queries and PAC simulation. We show that *any* statistical query algorithm can be simulated in the PAC model with malicious errors in such a way that the resultant PAC algorithm has a roughly optimal tolerable malicious error rate and sample complexity.

Finally, we generalize the types of queries allowed in the statistical query model. We discuss the advantages of allowing these generalized queries and show that our results on improved simulations also hold for these queries.

This paper is available from the Center for Research in Computing Technology, Division of Applied Sciences, Harvard University as technical report TR-17-94.

*Author was supported by Air Force Contract F49620-92-J-0466. Part of this research was conducted while the author was at MIT and supported by DARPA Contract N00014-87-K-825 and by NSF Grant CCR-89-14428. Author's net address: `jaa@das.harvard.edu`

†Author was supported by an NDSEG Doctoral Fellowship and by NSF Grant CCR-92-00884. Author's net address: `sed@das.harvard.edu`

1 Introduction

The statistical query model of learning was created so that algorithm designers could construct noise-tolerant PAC learning algorithms in a natural way. Ideally, such a model of robust learning should restrict the algorithm designer as little as possible while maintaining the ability to efficiently simulate these new algorithms in the PAC model with noise. In this paper, we both extend and improve the current statistical query model in ways which both increase the power of the algorithm designer and decrease the complexity of simulating these new algorithms. We begin by introducing the various models of learning required for the exposition that follows.

Since Valiant's introduction of the Probably Approximately Correct model of learning [19], PAC learning has proven to be an interesting and well studied model of machine learning. In an instance of PAC learning, a learner is given the task of determining a close approximation of an unknown $\{0,1\}$-valued *target function* f from *labelled examples* of that function. The learner is given access to an *example oracle* and accuracy and confidence parameters. When polled, the oracle draws an example according to a distribution D and returns the example along with its label according to f. The error rate of an hypothesis output by the learner is the probability that an example chosen according to D will be mislabelled by the hypothesis. The learner is required to output an hypothesis such that, with high confidence, the error rate of the hypothesis is less then the accuracy parameter. Two standard complexity measures studied in the PAC model are *sample complexity* and *time complexity*. Efficient PAC learning algorithms have been developed for many function classes [1], and PAC learning continues to be a popular model of machine learning.

One criticism of the PAC model is that the data presented to the learner is assumed to be *noise-free*. In fact, most of the standard PAC learning algorithms would fail if even a small number of the labelled examples given to the learning algorithm were "noisy." Two popular noise models for both theoretical and experimental research are the *classification noise* model introduced by Angluin and Laird [2, 14] and the *malicious error* model introduced by Valiant [20] and further studied by Kearns and Li [13]. In the classification noise model, each example received by the learner is mislabelled randomly and independently with some fixed probability. In the malicious error model, an adversary is allowed, with some fixed probability, to substitute a labelled example of his choosing for the labelled example the learner would ordinarily see.

While a limited number of efficient PAC algorithms had been developed which tolerate classification noise [2, 11, 16], no general framework for efficient learning[1] in the presence of classification noise was known until Kearns introduced the Statistical Query model [12].

In the SQ model, the example oracle of the standard PAC model is replaced by a statistics oracle. An SQ algorithm queries this new oracle for the values of various statistics on the distribution of labelled examples, and the oracle returns the requested statistics to within some specified additive error. Upon gathering a sufficient number of statistics, the SQ algorithm returns an hypothesis of the desired accuracy. Since calls to the statistics oracle can be *simulated* with high probability by drawing a sufficiently large sample from the example oracle, one can view this new oracle as an intermediary which effectively limits the way in which a learning algorithm can make use of labelled examples. Two standard complexity measures of SQ algorithms are *query complexity*, the maximum number of statistics required, and *tolerance*, the minimum additive error required.

Kearns [12] has demonstrated two important properties of the SQ model which make it worthy of study. First, he has shown that nearly every PAC learning algorithm can be cast within the SQ model, thus demonstrating that the SQ model is quite general and imposes a rather weak restriction on learning algorithms. Second, he has shown that calls to the statistics oracle can be simulated (with high probability) by a procedure which draws a sufficiently large sample from a *classification noise oracle*. An immediate consequence of these two properties is that nearly every PAC learning

[1] Angluin and Laird [2] introduced a general framework for learning in the presence of classification noise. However, their methods do not yield computationally efficient algorithms in most cases.

algorithm can be transformed into one which tolerates arbitrary amounts of classification noise.

Decatur [7] has demonstrated that calls to the statistics oracle can also be simulated (with high probability) by a procedure which draws a sufficiently large sample from a *malicious error oracle*. The amount of malicious error tolerable in such a simulation is proportional to the tolerance of the SQ algorithm.

The complexity of a statistical query algorithm in conjunction with the complexity of simulating SQ algorithms in the various noise models determine the complexity of the noise-tolerant PAC learning algorithms obtained. Kearns [12] has derived bounds on the minimum complexity of SQ algorithms, and Aslam and Decatur [4] have demonstrated a general technique for constructing SQ algorithms which are nearly optimal with respect to these bounds. In spite of this, the robust PAC learning algorithms obtained by simulating SQ algorithms in the presence of noise are inefficient when compared to known lower bounds for PAC learning in the presence of noise [8, 13, 18]. In fact, the PAC learning algorithms obtained by simulating SQ algorithms in the *absence* of noise are inefficient when compared to the tight bounds known for noise-free PAC learning [6, 8]. These shortcomings could be consequences of either inefficient simulations or a deficiency in the model itself. In this paper, we show that both of these explanations are true, and we provide both new simulations and a variant of the SQ model which combat the current inefficiencies of PAC learning via statistical queries.

We improve the complexity of simulating SQ algorithms in the presence of classification noise by providing a more efficient simulation. If τ_* is a lower bound on the minimum additive error requested by an SQ algorithm and $\eta_b < 1/2$ is an upper bound on the unknown noise rate, then Kearns' original simulation essentially runs $\Theta(\frac{1}{\tau_*(1-2\eta_b)^2})$ different copies of the SQ algorithm and processes the results of these runs to obtain an output. We show that this "branching factor" can be reduced to $\Theta(\frac{1}{\tau_*} \log \frac{1}{1-2\eta_b})$, thus reducing the time complexity of the simulation. We also provide a new and simpler proof that statistical queries can be estimated in the presence of classification noise, and we show that our formulation can easily be generalized to accommodate a strictly larger class of statistical queries.

We improve the complexity of simulating SQ algorithms in the absence of noise and in the presence of malicious errors by proposing a natural variant of the SQ model and providing efficient simulations for this variant. In the *relative error* SQ model, we allow SQ algorithms to submit statistical queries whose estimates are required within some specified relative error. We show that a class is learnable with relative error statistical queries if and only if it is learnable with (standard) additive error statistical queries. Thus, known learnability and hardness results for statistical queries [5, 12] also hold in this variant.

We demonstrate general bounds on the complexity of SQ learning with relative error statistical queries, and we show that many learning algorithms can naturally be written as highly efficient, relative error SQ algorithms. We further provide simulations of relative error SQ algorithms in both the absence and presence of noise. These simulations in the absence of noise and in the presence of malicious errors are more efficient than the simulations of additive error statistical queries and yield PAC learning algorithms using roughly optimal number of examples. These results hold for *all* classes which are learnable from statistical queries.

Finally, we show that our simulations of SQ algorithms in the absence of noise, in the presence of classification noise, and in the presence malicious errors can all be modified to accommodate a strictly larger class of statistical queries. In particular, we show that our simulations can accommodate real-valued and probabilistic statistical queries. Probabilistic queries arise naturally when applying boosting techniques to algorithms which output probabilistic hypotheses [4], while real-valued queries allow an algorithm to query the *expected value* of a real-valued function of labelled examples. Our results on improved simulations hold for these generalized queries in both the absence and presence of noise.

The remainder of the paper is organized as follows. In Section 2, we formally define the learning models of interest. In Section 3, we describe the improved simulation of statistical query algorithms in the presence of classification noise. We introduce the model of statistical queries with relative errors in Section 4 and give results relating to this model. In Section 5, we generalize the types of queries permitted in a statistical query algorithm. We conclude the paper with some open questions in Section 6.

2 Learning Models

In this section, we formally define the relevant models of learning necessary for the exposition that follows. We begin by defining the example-based PAC learning model as well as the classification noise and malicious error variants. We then define the standard statistical query model which we later generalize.

2.1 Example-Based PAC Learning

In an instance of PAC learning, a learner is given the task of determining a close approximation of an unknown $\{0,1\}$-valued target function from labelled examples of that function. The unknown target function f is assumed to be an element of a known function class $\mathcal{F}$ defined over an example space X. The example space X is typically either the Boolean hypercube $\{0,1\}^n$ or n-dimensional Euclidean space $\Re^n$. We use the parameter n to denote the common length of each example $x \in X$.

We assume that the examples are distributed according to some unknown probability distribution D on X. The learner is given access to an example oracle $EX(f,D)$ as its source of data. A call to $EX(f,D)$ returns a labelled example $\langle x, l\rangle$ where the example $x \in X$ is drawn randomly and independently according to the unknown distribution D, and the label $l = f(x)$. We often refer to a sequence of labelled examples drawn from an example oracle as a sample.

A learning algorithm draws a sample from $EX(f,D)$ and eventually outputs an hypothesis h from some hypothesis class $\mathcal{H}$ defined over X. For any hypothesis h, the *error rate* of h is defined to be the distribution weight of those examples in X where h and f differ. By using the notation $\Pr_D[P(x)]$ to denote the probability of drawing an example in X according to D which satisfies the predicate P, we may define $\mathit{error}(h) = \Pr_D[h(x) \neq f(x)]$. We often think of $\mathcal{H}$ as a class of representations of functions in $\mathcal{F}$, and as such we define $\mathit{size}(f)$ to be the size of the smallest representation in $\mathcal{H}$ of the target function f.

The learner's goal is to output, with probability at least $1-\delta$, an hypothesis h whose error rate is at most ϵ, for the given *error parameter* ϵ and *confidence parameter* δ. A learning algorithm is said to be *polynomially efficient* if its running time is polynomial in $1/\epsilon$, $1/\delta$, n and $\mathit{size}(f)$.

2.1.1 Classification Noise

In the *classification noise model*, the labelled example oracle $EX(f,D)$ is replaced by a noisy example oracle $EX^{\eta}_{\mathrm{CN}}(f,D)$. Each time this noisy example oracle is called, an example $x \in X$ is drawn according to D. The oracle then outputs $\langle x, f(x)\rangle$ with probability $1-\eta$ or $\langle x, \neg f(x)\rangle$ with probability η, randomly and independently for each example drawn. Despite the noise in the labelled examples, the learner's goal remains to output an hypothesis h which, with probability at least $1-\delta$, has error rate $\mathit{error}(h) = \Pr_D[h(x) \neq f(x)]$ at most ϵ.

While the learner does not typically know the exact value of the *noise rate* η, the learner is given an upper bound η_b on the noise rate, $0 \le \eta \le \eta_b < 1/2$, and the learner is said to be polynomially efficient if its running time is polynomial in the usual PAC learning parameters as well as $\frac{1}{1-2\eta_b}$.

2.1.2 Malicious Errors

In the *malicious error model*, the labelled example oracle $EX(f,D)$ is replaced by a noisy example oracle $EX^{\beta}_{\mathrm{MAL}}(f,D)$. When a labelled example is requested from this oracle, with probability $1-\beta$, an example x is chosen according to D and $\langle x, f(x)\rangle$ is returned to the learner. With probability β, a malicious adversary selects any example x, selects a label $l \in \{0,1\}$, and returns $\langle x, l\rangle$. Again, the learner's goal is to output an hypothesis h which, with probability at least $1-\delta$, has error rate $error(h) = \Pr_D[h(x) \neq f(x)]$ at most ϵ.

2.2 Statistical Query Based Learning

In the SQ model, the example oracle $EX(f,D)$ from the standard PAC model is replaced by a statistics oracle $STAT(f,D)$. An SQ algorithm queries the $STAT$ oracle for the values of various statistics on the distribution of labelled examples (*e.g.* "What is the probability that a randomly chosen labelled example $\langle x, l\rangle$ has variable $x_i = 0$ and $l = 1$?"), and the $STAT$ oracle returns the requested statistics to within some specified additive error. Formally, a statistical query is of the form $[\chi, \tau]$. Here χ is a mapping from labelled examples to $\{0,1\}$ (*i.e.* $\chi : X \times \{0,1\} \to \{0,1\}$) corresponding to an indicator function for those labelled examples about which statistics are to be gathered, while τ is an additive error parameter. A call $[\chi, \tau]$ to $STAT(f,D)$ returns an *estimate* $\hat{P}_\chi$ of $P_\chi = \Pr_D[\chi(x, f(x))]$ which satisfies $|\hat{P}_\chi - P_\chi| \leq \tau$.

A call to $STAT(f,D)$ can be simulated, with high probability, by drawing a sufficiently large sample from $EX(f,D)$ and outputting the fraction of labelled examples which satisfy $\chi(x, f(x))$ as the estimate $\hat{P}_\chi$. Since the required sample size depends polynomially on $1/\tau$ and the simulation time additionally depends on the time required to evaluate χ, an SQ learning algorithm is said to be polynomially efficient if $1/\tau$, the time required to evaluate each χ, and the running time of the SQ algorithm are all bounded by polynomials in $1/\epsilon$, n and $size(f)$.

An SQ learning algorithm is said to use *query space* $\mathcal{Q}$ if it only makes queries of the form $[\chi, \tau]$ where $\chi \in \mathcal{Q}$. Let τ_* be the lower bound on the additive error of every query made by an SQ algorithm.

3 Classification Noise Simulations of SQ Algorithms

In this section, we describe an improved method for efficiently simulating a statistical query algorithm using a classification noise oracle. The advantages of this new method are twofold. First, our simulation employs a new technique which significantly reduces the running time of simulating SQ algorithms. Second, our formulation for estimating individual queries is simpler and more easily generalized.

Kearns' procedure for simulating SQ algorithms works in the following way. Kearns shows that given a query χ, P_χ can be written as an expression involving the unknown noise rate η and other probabilities which can be estimated from the noisy example oracle $EX^{\eta}_{\mathrm{CN}}(f,D)$. We note that the derivation of this expression relies on χ being $\{0,1\}$-valued. The actual expression obtained is given below.

$$P_\chi = \tfrac{1}{1-2\eta} P^{\eta}_{\chi} + \left(1 - \tfrac{1}{1-2\eta}\right) p_2 P^2_\chi - \tfrac{\eta}{1-2\eta} p_1 \tag{1}$$

In order to estimate P_χ with additive error τ, a sensitivity analysis is employed to determine how accurately each of the components on the right-hand side of Equation 1 must be known. Kearns shows that for some constants c_1 and c_2, if η is estimated within additive error $c_1\tau(1-2\eta)^2$ and each of the probabilities is estimated within additive error $c_2\tau(1-2\eta_b)$, then the estimate obtained for P_χ from Equation 1 will be sufficiently accurate. Since the value of η is not known, the procedure for simulating SQ algorithms essentially guesses a set of values for η, $\{\eta_0, \eta_1, \ldots, \eta_i\}$, such that at least one η_j satisfies $|\eta_j - \eta| \le c_1\tau_*(1-2\eta)^2$ where τ_* is the minimum tolerance of the SQ algorithm. Since $c_1\tau_*(1-2\eta_b)^2 \le c_1\tau_*(1-2\eta)^2$, the simulation uniformly guesses $\Theta(\frac{1}{\tau_*(1-2\eta_b)^2})$ values of η between 0 and η_b. For each guess of η, the simulation runs a separate copy of the SQ algorithm and estimates the various P_χ's using the formula given above. Since some guess at η was good, at least one of the runs will have produced a good hypothesis with high probability. The various hypotheses are then tested to find a good hypothesis, of which at least one exists. Note that the η-guessing has a significant impact on the running time of the simulation.

In what follows, we show a new derivation of P_χ which is simpler and more easily generalizable than Kearns' original version. We also show that to estimate individual P_χ's, it is only necessary to have an estimate of η within additive error $c\tau(1-2\eta)$ for some constant c. We further show that the number of η-guesses need only be $O(\frac{1}{\tau_*}\log\frac{1}{1-2\eta_b})$, thus significantly reducing the time complexity of the SQ simulation.

3.1 A New Derivation for P_χ

In this section, we present a simpler derivation of an expression for P_χ. In previous sections, it was convenient to view a $\{0,1\}$-valued χ as a predicate so that $P_\chi = \Pr_D[\chi(x, f(x))]$. In this section, it will be more convenient to view χ as a function so that $P_\chi = \mathbf{E}_D[\chi(x, f(x))]$. Further, by making no assumptions on the range of χ, the results obtained herein can easily be generalized; these generalizations will be discussed in Section 5.

Let X be the example space, and let $Y = X \times \{0,1\}$ be the labelled example space. We consider a number of different examples oracles and the distributions these example oracles impose on the space of labelled examples. For a given target function f and distribution D over X, let $EX(f, D)$ be the standard, noise-free example oracle. In addition, we define the following example oracles: Let $EX(\bar{f}, D)$ be the *anti-example oracle*, $EX^{\eta}_{\mathrm{CN}}(f, D)$ be the *noisy example oracle* and $EX^{\eta}_{\mathrm{CN}}(\bar{f}, D)$ be the *noisy anti-example oracle*. Note that we have access to $EX^{\eta}_{\mathrm{CN}}(f, D)$ and we can easily construct $EX^{\eta}_{\mathrm{CN}}(\bar{f}, D)$ by simply flipping the label of each example drawn from $EX^{\eta}_{\mathrm{CN}}(f, D)$.

Each of these oracles imposes a distribution over labelled examples. Let D_f, $D_{\bar{f}}$, D^{η}_f and $D^{\eta}_{\bar{f}}$ be these distributions, respectively. Note that $P_\chi = \mathbf{E}_D[\chi(x, f(x))] = \mathbf{E}_{D_f}[\chi]$.

Finally, for a labelled example $y = \langle x, l\rangle$, let $\overline{y} = \langle x, \overline{l}\rangle$. We define $\overline{\chi}(y) = \chi(\overline{y})$. Note that $\overline{\chi}$ is a new function which, on input $\langle x, l\rangle$, simply outputs $\chi(x, \overline{l})$. The function $\overline{\chi}$ is easily constructed from χ.

Theorem 1

$$P_\chi = \mathbf{E}_{D_f}[\chi] = \frac{(1-\eta)\mathbf{E}_{D^{\eta}_f}[\chi] - \eta\mathbf{E}_{D^{\eta}_{\bar{f}}}[\overline{\chi}]}{1-2\eta} \tag{2}$$

Proof: For simplicity of exposition, we assume that X is a finite, discrete space (*e.g.* the Boolean hypercube $\{0,1\}^n$) so that $D(x)$ is well defined. In general, the theorem holds for any probability space (X, Σ, D) where D is a probability measure on Σ, a σ-algebra of subsets of X.

We first make some observations regarding the relationship between the various distributions defined above.

$$\begin{aligned}
D_{\bar{f}}(y) &= D_f(\overline{y}) \\
D^{\eta}_f(y) &= (1-\eta)D_f(y) + \eta D_f(\overline{y}) \\
&= (1-\eta)D_f(y) + \eta D_{\bar{f}}(y) && (3) \\
D^{\eta}_{\bar{f}}(y) &= (1-\eta)D_{\bar{f}}(y) + \eta D_{\bar{f}}(\overline{y}) \\
&= (1-\eta)D_{\bar{f}}(y) + \eta D_f(y) && (4) \\
&= D^{\eta}_f(\overline{y}) && (5)
\end{aligned}$$

Multiplying Equation 3 by $(1-\eta)$ and Equation 4 by η, we obtain:

$$(1-\eta)D_f^\eta(y) = (1-\eta)^2 D_f(y) + \eta(1-\eta)D_{\bar{f}}(y) \quad (6)$$
$$\eta D_{\bar{f}}^\eta(y) = \eta(1-\eta)D_{\bar{f}}(y) + \eta^2 D_f(y) \quad (7)$$

Subtracting Equation 7 from Equation 6 and solving for $D_f(y)$, we finally obtain:

$$D_f(y) = \frac{(1-\eta)D_f^\eta(y) - \eta D_{\bar{f}}^\eta(y)}{1-2\eta}$$

This implies that

$$\mathbf{E}_{D_f}[\chi] = \frac{(1-\eta)\mathbf{E}_{D_f^\eta}[\chi] - \eta \mathbf{E}_{D_{\bar{f}}^\eta}[\chi]}{1-2\eta}$$

and since $\mathbf{E}_{D_{\bar{f}}^\eta}[\chi] = \mathbf{E}_{D_f^\eta}[\overline{\chi}]$ (by Equation 5 and the definition of $\overline{\chi}$), we obtain Equation 2. □

Note that in the derivation given above, we have not assumed that χ is $\{0,1\}$-valued. This derivation is quite general and can be applied to estimating the expectations of probabilistic and real-valued χ's. These results are given in Section 5.

Finally, note that if we define $\chi^\eta(y) = \frac{(1-\eta)\chi(y)-\eta\overline{\chi}(y)}{1-2\eta}$, then $P_\chi = \mathbf{E}_{D_f}[\chi] = \mathbf{E}_{D_f^\eta}[\chi^\eta]$. Thus, given a χ whose expectation we require with respect to the noise-free oracle, we can construct a new χ whose expectation with respect to the noisy oracle is identical to the answer we require. This formulation may even be more convenient if one has the capability of estimating the expectation of real-valued functions; we discuss this generalization in Section 5.

3.2 Sensitivity Analysis

In this section, we provide a sensitivity analysis of Equation 2 to determine the accuracy with which various quantities must be estimated. We make use of the following claim which can easily be shown.

Claim 1 *If $0 \le a, b, c, \tau \le 1$ and $\{a = b/c,\ a = b \cdot c,\ a = b - c\}$, then to obtain an estimate of a within additive error τ, it is sufficient to obtain estimates of b and c within additive error $\{c\tau/3,\ \tau/3,\ \tau/2\}$, respectively.*

Lemma 1 *Let $\hat{\eta}$, $\hat{\mathbf{E}}_{D_f^\eta}[\chi]$ and $\hat{\mathbf{E}}_{D_f^\eta}[\overline{\chi}]$ be estimates of η, $\mathbf{E}_{D_f^\eta}[\chi]$ and $\mathbf{E}_{D_f^\eta}[\chi]$ each within additive error $\tau(1-2\eta)/18$. Then the quantity*

$$\frac{(1-\hat{\eta})\hat{\mathbf{E}}_{D_f^\eta}[\chi] - \hat{\eta}\hat{\mathbf{E}}_{D_f^\eta}[\overline{\chi}]}{1-2\hat{\eta}}$$

is within additive error τ of $P_\chi = \mathbf{E}_{D_f}[\chi]$.

Proof: To obtain an estimate of the right-hand side of Equation 2 within additive error τ, it is sufficient to obtain estimates of the numerator and denominator with additive error $(1-2\eta)\tau/3$. This condition holds for the denominator if η is estimated with additive error $(1-2\eta)\tau/6$.

To obtain an estimate of the numerator within additive error $(1-2\eta)\tau/3$, it is sufficient to estimate the summands of the numerator with additive error $(1-2\eta)\tau/6$. Similarly, to obtain accurate estimates of these summands, it is sufficient to estimate η, $\mathbf{E}_{D_f^\eta}[\chi]$ and $\mathbf{E}_{D_f^\eta}[\overline{\chi}]$ each with additive error $(1-2\eta)\tau/18$. □

Estimates for $\mathbf{E}_{D_f^\eta}[\chi]$ and $\mathbf{E}_{D_f^\eta}[\chi]$ are obtained by sampling, and an "estimate" for η is obtained by guessing. We address these issues in the following sections.

3.3 Estimating $\mathbf{E}_{D_f^\eta}[\chi]$ and $\mathbf{E}_{D_f^\eta}[\overline{\chi}]$

One can estimate the expected values of all queries submitted by drawing separate samples for each of the corresponding χ and $\overline{\chi}$'s and applying Lemma 1. However, better results are obtained by appealing to uniform convergence.

Let $\mathcal{Q}$ be the query space of the SQ algorithm and let $\overline{\mathcal{Q}} = \{\overline{\chi} : \chi \in \mathcal{Q}\}$. The query space of our simulation is $\mathcal{Q}' = \mathcal{Q} \cup \overline{\mathcal{Q}}$. Note that for finite $\mathcal{Q}$, $|\mathcal{Q}'| \leq 2|\mathcal{Q}|$, and for all $\mathcal{Q}$, $\textit{VC-dim}(\mathcal{Q}') = \Theta(\textit{VC-dim}(\mathcal{Q}))$.

If τ_* is a lower bound on the minimum additive error requested by the SQ algorithm and η_b is an upper bound on the noise rate, then by Lemma 1, $(1-2\eta_b)\tau_*/18$ is a sufficient additive error with which to estimate all expectations. Standard uniform convergence results can be applied to show that all expectations can be estimated within the given additive error using a single noisy sample of size

$$m_1 = O\left(\frac{1}{\tau_*^2(1-2\eta_b)^2}\log\frac{|\mathcal{Q}|}{\delta}\right)$$

in the case of a finite query space or a single noisy sample of size

$$m_1 = O\left(\frac{\textit{VC-dim}(\mathcal{Q})}{\tau_*^2(1-2\eta_b)^2}\log\frac{1}{\tau_*(1-2\eta_b)} + \frac{1}{\tau_*^2(1-2\eta_b)^2}\log\frac{1}{\delta}\right)$$

in the case of an infinite query space of finite VC-dimension.

3.4 Guessing the Noise Rate η

By Lemma 1, to obtain estimates for P_χ, it is sufficient to have an estimate of the noise rate η within additive error $(1-2\eta)\tau_*/18$. Since the noise rate is unknown, the simulation guesses various values of the noise rate and runs the SQ algorithm for each guess. If one of the noise rate guesses is sufficiently accurate, then the corresponding run of the SQ algorithm will produce the desired accurate hypothesis.

To guarantee that an accurate η-guess is used, one could simply guess $\Theta(\frac{1}{\tau_*(1-2\eta_b)})$ values of η spaced uniformly between 0 and η_b. This is essentially the approach adopted by Kearns. Note that this would cause the simulation to run the SQ algorithm $\Theta(\frac{1}{\tau_*(1-2\eta_b)})$ times.

We now show that this "branching factor" can be reduced to $O(\frac{1}{\tau_*}\log\frac{1}{1-2\eta_b})$ by constructing our η-guesses in a much better way. The result follows immediately from the following lemma when $\gamma = \tau_*/18$.

Lemma 2 *For all $\gamma, \eta_b < 1/2$, there exists a sequence of η-guesses $\{\eta_0, \eta_1, \ldots, \eta_i\}$ where $i = O(\frac{1}{\gamma}\log\frac{1}{1-2\eta_b})$ such that for all $\eta \in [0, \eta_b]$, there exists an η_j which satisfies $|\eta - \eta_j| \leq \gamma(1-2\eta)$.*

Proof: The sequence is constructed as follows. Let $\eta_0 = 0$ and consider how to determine η_j from η_{j-1}. The value η_{j-1} is a valid estimate for all $\eta \geq \eta_{j-1}$ which satisfy $\eta - \gamma(1-2\eta) \leq \eta_{j-1}$. Solving for η, we find that η_{j-1} is a valid estimate for all $\eta \in [\eta_{j-1}, \frac{\eta_{j-1}+\gamma}{1+2\gamma}]$. Consider an $\eta_j > \frac{\eta_{j-1}+\gamma}{1+2\gamma}$. The value η_j is a valid estimate for all $\eta \leq \eta_j$ which satisfy $\eta + \gamma(1-2\eta) \geq \eta_j$. Solving for η, we find that η_j is a valid estimate for all $\eta \in [\frac{\eta_j-\gamma}{1-2\gamma}, \eta_j]$. To ensure that either η_{j-1} or η_j is a valid estimate for any $\eta \in [\eta_{j-1}, \eta_j]$, we set

$$\frac{\eta_{j-1}+\gamma}{1+2\gamma} = \frac{\eta_j-\gamma}{1-2\gamma}.$$

Solving for η_j in terms of η_{j-1}, we obtain

$$\eta_j = \frac{1-2\gamma}{1+2\gamma}\eta_{j-1} + \frac{2\gamma}{1+2\gamma}.$$

Substituting $\gamma' = 2\gamma/(1+2\gamma)$, we obtain the following recurrence:

$$\eta_j = (1-2\gamma')\eta_{j-1} + \gamma'$$

Note that if $\gamma < 1/2$, then $\gamma' < 1/2$ as well.

By constructing η-guesses using this recurrence, we ensure that for all $\eta \in [0, \eta_i]$, at least one of $\{\eta_0, \ldots, \eta_i\}$ is a valid estimate. Solving this recurrence, we find that

$$\eta_i = \gamma' \sum_{j=0}^{i-1} (1-2\gamma')^j + \eta_0 (1-2\gamma')^i.$$

Since $\eta_0 = 0$ and we are only concerned with $\eta \le \eta_b$, we may bound the number of guesses required by finding the smallest i which satisfies

$$\gamma' \sum_{j=0}^{i-1} (1-2\gamma')^j \ge \eta_b.$$

Given that

$$\gamma' \sum_{j=0}^{i-1} (1-2\gamma')^j = \gamma' \tfrac{1-(1-2\gamma')^i}{1-(1-2\gamma')} = \tfrac{1-(1-2\gamma')^i}{2}$$

we need $(1-2\gamma')^i \le 1 - 2\eta_b$. Solving for i, we find that any $i \ge \ln \frac{1}{1-2\eta_b} / \ln \frac{1}{1-2\gamma'}$ is sufficient. Using the fact that $1/x > 1/\ln \frac{1}{1-x}$ for all $x \in (0,1)$, we find that

$$i = \tfrac{1}{2\gamma'} \ln \tfrac{1}{1-2\eta_b} = \tfrac{1+2\gamma}{4\gamma} \ln \tfrac{1}{1-2\eta_b}$$

is an upper bound on the number of guesses required. □

3.5 The Overall Simulation

We now combine the results of the previous sections to obtain an overall simulation as follows:

1. Draw m_1 samples from $EX_{\mathrm{CN}}^{\eta}(f, D)$ in order to estimate expectations in Step 2.
2. Run the SQ algorithm once for each of the $O(\frac{1}{\tau_*} \log \frac{1}{1-2\eta_b})$ η-guesses, estimating the various queries by applying Lemma 1 and using the sample drawn.
3. Draw m_2 samples and test the $O(\frac{1}{\tau_*} \log \frac{1}{1-2\eta_b})$ hypotheses obtained in Step 2. Output one of these hypotheses whose error rate is at most ϵ.

Step 3 can be accomplished by a generalization of a technique due to Laird [14]. The sample size required is

$$m_2 = O\left(\tfrac{1}{\epsilon(1-2\eta_b)^2} \log(\tfrac{1}{\delta\tau_*} \log \tfrac{1}{1-2\eta_b}) \right).$$

Since $1/\tau_* = \Omega(1/\epsilon)$ for all SQ algorithms [12], the sample complexity of the overall simulation is

$$O\left(\tfrac{1}{\tau_*^2(1-2\eta_b)^2} \log \tfrac{|\mathcal{Q}|}{\delta} + \tfrac{1}{\epsilon(1-2\eta_b)^2} \log\log \tfrac{1}{1-2\eta_b} \right)$$

in the case of a finite query space or

$$O\left(\tfrac{VC\text{-}dim(\mathcal{Q})}{\tau_*^2(1-2\eta_b)^2} \log \tfrac{1}{\tau_*(1-2\eta_b)} + \tfrac{1}{\tau_*^2(1-2\eta_b)^2} \log \tfrac{1}{\delta} \right)$$

in the case of an infinite query space of finite VC-dimension.

To determine the running time of our simulation, one must distinguish between two different types of SQ algorithms. Some SQ algorithms submit a fixed set of queries independent of the estimates they receive for previous queries. We refer to these algorithms as "batch" SQ algorithms. Other SQ algorithms may submit various queries based upon the estimates they receive for previous queries. We refer to these algorithms as "dynamic" SQ algorithms.[2] Note that multiple runs of a dynamic SQ algorithm may produce many more queries which need to be estimated.

With respect to η, Simon [18] has shown a sample complexity, and therefore time complexity, lower bound of $\Omega(\frac{1}{(1-2\eta)^2})$ for PAC learning in the presence of classification noise. We therefore note that by reducing the "branching factor" of the simulation from $\Theta(\frac{1}{\tau_*(1-2\eta_b)^2})$ to $\Theta(\frac{1}{\tau_*}\log\frac{1}{1-2\eta_b})$, the running time of our simulation is optimal with respect to the noise rate (modulo lower order logarithmic factors) for both dynamic and batch algorithms. For dynamic algorithms, the time complexity of our new simulation is in fact a $\tilde{\Theta}(\frac{1}{(1-2\eta_b)^2})$ factor better than the current simulation.[3]

4 Statistical Queries with Relative Error Estimates

In the standard model of statistical query learning, a learning algorithm asks for an estimate of the probability that a predicate χ is true. The required accuracy of this estimate is specified by the learner in the form of an additive error parameter. The limitation of this model is clearly evident in even the standard, *noise-free* statistical query simulation [12]. This simulation uses $\Omega(1/\tau_*^2)$ examples. Since $1/\tau_* = \Omega(1/\epsilon)$ for all SQ algorithms [12], this simulation effectively uses $\Omega(1/\epsilon^2)$ examples. However, the ϵ-dependence of the general bound on sample complexity for PAC learning is $\tilde{\Theta}(1/\epsilon)$ [6, 8].

This $\Omega(1/\tau_*^2) = \Omega(1/\epsilon^2)$ sample complexity results from the worst case assumption that large probabilities may need to be estimated with small additive error. Either the nature of statistical query learning is such that learning sometimes requires the estimation of large probabilities with small additive error, or it is always sufficient to estimate each probability with an additive error comparable to the probability. If the former were the case, then the present model and simulations would be the best that one could hope for. We show that the latter is true, and that a model in which queries are specified with *relative error* is a more natural and strictly more powerful tool.

We define such a model of relative error statistical query learning and we show how this new model relates to the standard additive error model. We also show general upper bounds on learning in this new model which demonstrate that for *all* classes learnable by statistical queries, it is sufficient to make estimates with relative error independent of ϵ. We then give roughly optimal PAC simulations for relative error SQ algorithms. Finally, we demonstrate natural problems which only require estimates with *constant* relative error.

4.1 The Relative Error SQ Model

Given the motivation above, we modify the standard model of statistical query learning to allow for estimates to be requested with relative error. We replace the additive error $STAT(f, D)$ oracle with a relative error *Rel-STAT*(f, D) oracle which accepts a query χ, a relative error parameter μ, and a threshold parameter θ. The value $P_\chi = \Pr_D[\chi(x, f(x))]$ is defined as before. If P_χ is less than the threshold θ, then the oracle may return the symbol $\perp$. If the oracle does not return $\perp$, then it must return an estimate $\hat{P}_\chi$ such that

$$P_\chi(1-\mu) \leq \hat{P}_\chi \leq P_\chi(1+\mu)$$

[2] Note that we consider any SQ algorithm with a polynomially sized query space to be a "batch" algorithm since all queries may be processed in advance.

[3] When $b > 1$, we define $\tilde{O}(b)$ to mean $O(b\log^c b)$ for some constant $c > 0$. When $b < 1$, we define $\tilde{O}(b)$ to mean $O(b\log^c(1/b))$ for some constant $c > 0$. We define $\tilde{\Omega}$ similarly for some constant $c < 0$ and define $\tilde{\Theta}$ to mean $\tilde{O}$ *and* $\tilde{\Omega}$.

Note that the oracle may chose to return an accurate estimate even if $P_\chi < \theta$. A class is said to be learnable by relative error statistical queries if it satisfies the same conditions of additive error statistical query learning except we instead require that $1/\mu$ and $1/\theta$ are polynomially bounded. Let μ_* and θ_* be the lower bounds on the relative error and threshold of every query made by an SQ algorithm. Given this definition of relative error statistical query learning, we show the following desirable equivalence.

Theorem 2 *$\mathcal{F}$ is learnable by additive error statistical queries if and only if $\mathcal{F}$ is learnable by relative error statistical queries.*

Proof: One can take any query χ to the additive error oracle which requires additive tolerance τ and simulate it by calling the relative error oracle with relative error τ and threshold τ. Similarly, one can take any query to the relative error oracle which requires relative error μ and threshold θ and simulate it by calling the additive error oracle with tolerance $\mu\theta$. In each direction, the simulation uses polynomially bounded parameters if and only if the original algorithm uses polynomially bounded parameters. $\square$

Kearns [12] shows that almost all classes known to be PAC learnable are learnable with additive error statistical queries. By the above theorem, these classes are also learnable with relative error statistical queries. In addition, the hardness results of Kearns [12] for learning parity functions and the general hardness results of Blum *et al.* [5] based on Fourier analysis also hold for relative error statistical query learning.

One can convert an additive error SQ algorithm to a relative error SQ algorithm in a more efficient way than described in the proof of Theorem 2. The key idea is that for each query $[\chi, \tau]$, we search for P_χ starting at τ by successive doubling.

Theorem 3 *An additive error query $[\chi, \tau]$ can be simulated by $O(\log(P_\chi/\tau))$ relative error queries $[\chi, \mu_i, \theta_i]$ where, for each i, $\mu_i\theta_i = \Omega(\tau)$ and $\mu_i^2\theta_i = \Omega(\tau^2/P_\chi)$.*

Proof: We first give the search algorithm, then prove it correctness, and finally prove bounds on the number and complexity of the queries made.

Let $\mu_0 = 1/4$ and $\theta_0 = \tau$. Let $\mu_{i+1} = \mu_i/2$ and $\theta_{i+1} = 2\theta_i$. If the response $\hat{P}_\chi^i$ to the query $[\chi, \mu_i, \theta_i]$ is $\perp$ then 0 is returned. If $\hat{P}_\chi^i \leq \tau(1-\mu_i)/\mu_i$ then $\hat{P}_\chi^i$ is returned. Otherwise, the next query $[\chi, \mu_{i+1}, \theta_{i+1}]$ is asked.

Note that for all $i > 0$, $\theta_i = \tau/2\mu_{i-1}$. Since $\mu_{i-1} \leq 1/4$ we have $(1-\mu_{i-1})/(1+\mu_{i-1}) \geq 3/5 > 1/2$. If we ask query $i > 0$, it must have been true that $\hat{P}_\chi^{i-1} > \tau(1-\mu_{i-1})/\mu_{i-1}$, and since $\hat{P}_\chi^{i-1} \leq P_\chi(1+\mu_{i-1})$, we can show that $P_\chi > \theta_i$ as follows:

$$\begin{aligned}
P_\chi &\geq \hat{P}_\chi^{i-1}/(1+\mu_{i-1}) \\
&> \tau/\mu_{i-1} \cdot (1-\mu_{i-1})/(1+\mu_{i-1}) \\
&> \tau/2\mu_{i-1} \\
&= \theta_i
\end{aligned}$$

Thus, after the first query, no query will be asked which could be answered $\perp$. If the first query is answered $\perp$, then returning 0 is correct since $P_\chi \leq \tau$.

When $\hat{P}_\chi^i$ is returned, we know that $\hat{P}_\chi^i \leq \tau(1-\mu_i)/\mu_i$ which implies that $P_\chi \leq \tau/\mu_i$. But since the additive difference between P_χ and $\hat{P}_\chi^i$ is guaranteed to be no more than $\mu_i P_\chi$, this gap is no more than τ, and therefore the returned value is sufficiently accurate.

It is trivial to verify that for all i, $\mu_i\theta_i = \tau/4$ and $\mu_i^2\theta_i = 2^{-i}\mu_0^2\theta_0 = 2^{-i}\tau/16$. The search will end when μ_i is small enough to force $\hat{P}_\chi^i \leq \tau(1-\mu_i)/\mu_i$. This condition is guaranteed when $P_\chi \leq \tau/\mu_i \cdot (1-\mu_i)/(1+\mu_i)$ and therefore when $P_\chi \leq \tau/2\mu_i$. Since $\mu_i = \mu_0/2^i$, this holds for the first i such that $2^i \geq 2\mu_0 P_\chi/\tau$. Therefore, the maximum number of queries made is $O(\log(P_\chi/\tau))$ and the worst case $\mu_i^2\theta_i = 2^{-i}\tau/16 = \Omega(\tau^2/P_\chi)$. $\square$

4.2 A Natural Example of Relative Error SQ

In this section we examine a learning problem which has both a simple additive error SQ algorithm and a simple relative error SQ algorithm. We consider the problem of learning a monotone conjunction of Boolean variables in which the learning algorithm must determine which subset of the variables $\{x_1, \ldots, x_n\}$ are contained in the unknown target conjunction f.

We construct an hypothesis h which contains all the variables in the target function f, and thus h will not misclassify any negative examples. We further guarantee that for each variable x_i in h, the distribution weight of examples which satisfy "$x_i = 0$ and $f(x) = 1$" is less than ϵ/n. Therefore, the distribution weight of positive examples which h will misclassify is at most ϵ. Such an hypothesis has error rate at most ϵ.

Consider the following query: $\chi_i(x, l) = (x_i = 0 \wedge l = 1)$. P_{χ_i} is simply the probability that x_i is false and $f(x)$ is true. If variable x_i is in f, then $P_{\chi_i} = 0$. If we mistakenly include a variable x_i in our hypothesis which is not in f, then the error due to this inclusion is at most P_{χ_i}. We simply construct our hypothesis to contain all target variables, but no variables x_i for which $P_{\chi_i} > \epsilon/n$.

An additive error SQ algorithm queries each χ_i with additive error $\epsilon/2n$ and includes all variables for which the estimate $\hat{P}_{\chi_i} \leq \epsilon/2n$. Even if $P_{\chi_i} = 1/2$, the oracle is constrained to return an estimate with additive error less than $\epsilon/2n$. A relative error SQ algorithm queries each χ_i with relative error $1/2$ and threshold ϵ/n and includes all variables for which the estimate $\hat{P}_{\chi_i} = 0$ or $\perp$.

The sample complexity of the standard, noise-free PAC simulation of additive error SQ algorithms depends linearly on $1/\tau_*^2$ [12], while in Section 4.4, we show that the sample complexity of a noise-free PAC simulation of relative error SQ algorithms depends linearly on $1/\mu_*^2\theta_*$. Note that in the above algorithms for learning conjunctions, $1/\tau_*^2 = \Theta(n^2/\epsilon^2)$ while $1/\mu_*^2\theta_* = \Theta(n/\epsilon)$. We further note note that the μ_* is *constant* for learning conjunctions. We show in Section 4.3 that *no* learning problem requires μ_* to depend on ϵ and in Section 4.5 that μ_* is actually a constant in many algorithms.

4.3 General Bounds on Relative Error SQ Learning

In this section we prove general upper bounds on the complexity of relative error statistical query learning. We do so by applying boosting techniques [9, 10, 17] and specifically, these techniques as applied in the statistical query model [4]. We first prove some useful lemmas which allow us to decompose relative estimates of ratios and sums.

Lemma 3 *Let $a = b/c$ where $0 \leq a, b, c \leq 1$. If an estimate of a is desired with (μ, θ) error provided that $c \geq \Phi$, then it is sufficient to estimate c with $(\mu/3, \Phi)$ error and b with $(\mu/3, \theta\Phi/2)$ error.*

Proof: If the estimate $\hat{c}$ is $\perp$ or less than $\Phi(1-\mu/3)$, then $c < \Phi$. Therefore an estimate for a is not required, and we may halt. Otherwise $\hat{c} \geq \Phi(1-\mu/3)$, and therefore $c \geq \Phi(1-\mu/3)/(1+\mu/3) \geq \Phi/2$ since $\mu \leq 1$. If the estimate $\hat{b}$ is $\perp$, then $b < \theta\Phi/2$. Therefore $a = b/c < \theta$, so we may answer $\hat{a} = \perp$. Otherwise, $\hat{b}$ and $\hat{c}$ are estimates of b and c, each within a $1 \pm \mu/3$ factor. It is easy to show that $\hat{b}/\hat{c}$ is within a $1 \pm \mu$ factor of a. □

Lemma 4 *Let $s = \sum p_i z_i$ where the $\{p_i\}$ are known, $0 \leq s, p_i, z_i \leq 1$ and $\sum_i p_i \leq 1$. If an estimate of s is desired with (μ, θ) error, then it is sufficient to estimate each z_i with $(\mu/3, \mu\theta/3)$ error.*

Proof: Let $B = \{i : \text{estimate of } z_i \text{ is } \perp\}$, $E = \{i : \text{estimate of } z_i \text{ is } \hat{z}_i\}$, $s_B = \sum_B p_i z_i$ and $s_E = \sum_E p_i z_i$. Note that $s_B < \theta\mu/3$. Let $\hat{s}_E = \sum_E p_i \hat{z}_i$. If $\hat{s}_E < \theta(1-\mu/3)^2$ then we return $\perp$, otherwise we return $\hat{s}_E$.

If $\hat{s}_E < \theta(1-\mu/3)^2$, then $s_E < \theta(1-\mu/3)$. But in this case $s = s_E + s_B < \theta(1-\mu/3) + \theta\mu/3 = \theta$, so we are correct in returning $\perp$.

Otherwise we return $\hat{s}_E$ which is at least $\theta(1-\mu/3)^2$. If $B=\emptyset$, then it is easy to see that $\hat{s}_E$ is within a $1\pm\mu/3$ (and therefore $1\pm\mu$) factor of s. Otherwise, we are implicitly setting $\hat{z}_i=0$ for each $i\in B$, and therefore it is enough to show that $\hat{s}_E\geq s(1-\mu)$.

Since $\hat{s}_E\geq\theta(1-\mu/3)^2$, we have $s_E\geq\theta(1-\mu/3)^2/(1+\mu/3)$. Using the fact that for all $\mu\leq 1$, $(1-\mu/3)/(1+\mu/3)\geq 1/2$, we have $s_E\geq\theta(1-\mu/3)/2$. If $\hat{s}_E\geq(\theta\mu/3+s_E)(1-\mu)$, then $\hat{s}_E\geq s(1-\mu)$ since $s_B<\theta\mu/3$ and $s=s_B+s_E$. But since $\hat{s}_E\geq s_E(1-\mu/3)$, this condition holds when $s_E(1-\mu/3)\geq(\theta\mu/3+s_E)(1-\mu)$. Solving for s_E, this final condition holds when $s_E\geq\theta(1-\mu/3)/2$ which we have shown to be true whenever an estimate is returned. □

Theorem 4 *If a concept class $\mathcal{F}$ is SQ learnable by algorithm A, then $\mathcal{F}$ is SQ learnable with $O(N_0\log^2\frac{1}{\epsilon})$ queries each of relative error $\Omega(\mu_0)$ and threshold $\Omega(\mu_0\theta_0\epsilon/\log\frac{1}{\epsilon})$. Here N_0, μ_0 and θ_0 are the number of queries, worst case relative error, and worst case threshold, respectively, of algorithm A run with a constant accuracy parameter. Note that N_0, μ_0 and θ_0 are independent of ϵ.*

Proof: Aslam and Decatur [4] show that given an SQ learning algorithm A, one can construct a very efficient SQ algorithm by combining the output of $O(\log\frac{1}{\epsilon})$ runs of A. Each run of A is made with respect to a different distribution and uses accuracy parameter $\epsilon=1/4$. Each run makes at most N_0 queries, each with relative error no smaller than μ_0 and threshold no smaller than θ_0. In run $i+1$, the algorithm makes queries of the form $STAT(f,D_{i+1})[\chi(x,f(x))]$ where D_{i+1} is a distribution based on D. Since the learning algorithm only has access to a statistics oracle for D, they show that a query with respect to D_{i+1} may be written in terms of new queries with respect to D as follows:

$$STAT(f,D_{i+1})[\chi(x,f(x))]=\frac{\sum_{j=0}^{w}\lambda_j^w\cdot STAT(f,D)[\chi(x,f(x))\wedge\chi_j^w(x,f(x))]}{\sum_{j=0}^{w}\lambda_j^w\cdot STAT(f,D)[\chi_j^w(x,f(x))]}\tag{8}$$

In the above equation $w\leq i$, the values $\lambda_j^w\in[0,1]$ are known, and $\sum_j\lambda_j^w\leq 1$. It is also the case that if the denominator of Equation 8 is less than $\Phi=\Omega(\epsilon/\log\frac{1}{\epsilon})$, then the query need not be estimated. Using Lemmas 3 and 4, it is easy to show that a query to $STAT(f,D_{i+1})$ can be estimated by making queries to $STAT(f,D)$ with relative error at least $\mu_0/9$ and threshold at least $\mu_0\theta_0\Phi/18$. Since a query with respect to D_{i+1} requires $O(i)$ queries with respect to D, the total number of queries made is $O(N_0\log^2\frac{1}{\epsilon})$. □

4.4 Simulating Relative Error SQ Algorithms

In this section we give the complexity of simulating relative error SQ algorithms in the PAC model, both in the absence and presence of noise. We also give general upper bounds on the complexity of PAC algorithms derived from SQ algorithms, based on the simulations and the general upper bounds of Theorem 4.

The simulation of relative error SQ algorithms in the noise-free PAC model is based on a standard Chernoff bound analysis. We obtain the following theorem on the sample complexity of such a simulation. The proof of this theorem is identical to the proof of Theorem 7 below if in the latter proof the malicious error rate is set to 0.

Theorem 5 *If $\mathcal{F}$ is learnable by a statistical query algorithm which makes N queries from query space $\mathcal{Q}$ with worst case relative error μ_* and worst case threshold θ_*, then $\mathcal{F}$ is PAC learnable with sample complexity $O(\frac{1}{\mu_*^2\theta_*}\log\frac{|\mathcal{Q}|}{\delta})$ when $\mathcal{Q}$ is finite or $O(\frac{N}{\mu_*^2\theta_*}\log\frac{N}{\delta})$ when drawing a separate sample for each query.*

Corollary 1 *If $\mathcal{F}$ is SQ learnable, then $\mathcal{F}$ is PAC learnable with a sample complexity whose dependence on ϵ is $\tilde{O}(1/\epsilon)$.*

Although one could use boosting techniques in the PAC model to achieve this nearly optimal sample complexity, these boosting techniques would result in a more complicated algorithm and output hypothesis (a circuit whose inputs were hypotheses from the original hypothesis class). If instead we have a relative error SQ algorithm meeting the bounds of Theorem 4, then we achieve this PAC sample complexity directly.

For SQ simulations in the classification noise model, we achieve the sample complexity given in Theorem 6 below. This sample complexity is essentially identical to the simulation of an additive error SQ algorithm for which $\tau = \mu\theta$, and, in fact, this is one way of proving the result. Although this result does not improve the sample complexity of SQ simulations in the presence of classification noise, we believe that to improve upon this bound requires the use of relative error statistical queries for the reasons discussed in the introduction to Section 4.

Theorem 6 *If $\mathcal{F}$ is learnable by a statistical query algorithm which makes N queries from query space $\mathcal{Q}$ with worst case relative error μ_* and worst case threshold θ_*, then $\mathcal{F}$ is PAC learnable in the presence of classification noise. If $\eta_b < 1/2$ is an upper bound on the noise rate, then the sample complexity required is*

$$O\left(\frac{1}{\mu_*^2\theta_*^2(1-2\eta_b)^2}\log\frac{|\mathcal{Q}|}{\delta} + \frac{1}{\epsilon(1-2\eta_b)^2}\log\log\frac{1}{1-2\eta_b}\right)$$

when $\mathcal{Q}$ is finite or

$$O\left(\frac{N}{\mu_*^2\theta_*^2(1-2\eta_b)^2}\log\frac{N}{\delta} + \frac{1}{\epsilon(1-2\eta_b)^2}\log\log\frac{1}{1-2\eta_b}\right)$$

when drawing a separate sample for each query.

Corollary 2 *If $\mathcal{F}$ is SQ learnable, then $\mathcal{F}$ is PAC learnable in the presence of classification noise. The dependence on ϵ and η_b of the required sample complexity is $\tilde{O}(\frac{1}{\epsilon^2(1-2\eta_b)^2})$.*

We next consider the simulation of relative error SQ algorithms in the presence of malicious errors. Decatur [7] showed that an SQ algorithm can be simulated in the presence of malicious errors with a maximum allowable error rate which depends on τ_*, the smallest additive error required by the SQ algorithm. In Theorem 7, we show that an SQ algorithm can be simulated in the presence of malicious errors with a maximum allowable error rate and sample complexity which depend on μ_* and θ_*, the minimum *relative error* and *threshold* required by the SQ algorithm.

The key to this malicious error tolerant simulation is to draw a large enough sample such that for each query, the combined error in an estimate due to both the adversary and the statistical fluctuation on error-free examples is less than the accuracy required for this query. We make use of the following lemmas.

Lemma 5 (Chernoff Bounds [3]) *Let $X_1, \ldots, X_m$ be independent Bernoulli random variables, each of whose expectation is p. Let Y be the random variable $\frac{1}{m}\sum_i X_i$. If $0 < \alpha \leq 1$ and $m \geq \frac{3}{\alpha^2 p}\ln\frac{2}{\delta}$, then $p(1-\alpha) \leq Y \leq p(1+\alpha)$ with probability at least $1-\delta$.*

Lemma 6 *Let P_χ^* be the fraction of examples satisfying χ in a noise-free sample of size m, and let $\hat{P}_\chi$ be the fraction of examples satisfying χ in a sample of size m drawn from $EX_{\mathrm{MAL}}^{\beta}(f, D)$. Then to ensure, with high probability, $|P_\chi - \hat{P}_\chi| \leq \tau_1 + \tau_2$, it is sufficient to draw a sample of size m which, with high probability, simultaneously ensures both:*
(1) The adversary corrupts at most a τ_1 fraction of examples drawn from $EX_{\mathrm{MAL}}^{\beta}(f, D)$.
(2) $|P_\chi - P_\chi^*| \leq \tau_2$.

Proof: Let m be a sample size large enough to ensure, with high probability, both conditions hold. Consider a sample of size m drawn from a malicious error oracle in which the adversary decided not to corrupt any examples for which it was given the opportunity. Then by Condition (2), with high probability, the fraction of examples satisfying χ on this sample is within τ_2 of P_χ. But by Condition (1), with high probability, the adversary may change the empirical fraction of examples satisfying χ by no more than τ_1. Thus the empirical fraction of examples is within $\tau_1 + \tau_2$ of P_χ. □

Theorem 7 *If $\mathcal{F}$ is learnable by a statistical query algorithm which makes N queries from query space $\mathcal{Q}$ with worst case relative error μ_* and worst case threshold θ_*, then $\mathcal{F}$ is PAC learnable in the presence of malicious errors. The maximum allowable error rate is $\beta_* = \Omega(\mu_* \theta_*)$ and the sample complexity required is $O(\frac{1}{\mu_*^2 \theta_*} \log \frac{|\mathcal{Q}|}{\delta})$ when $\mathcal{Q}$ is finite or $O(\frac{N}{\mu_*^2 \theta_*} \log \frac{N}{\delta})$ when drawing a separate sample for each query.*

Proof: We first analyze the tolerable error and sample complexity for simulating a single query and then determine these values for simulating the entire algorithm.

We assume that $\beta \leq \beta_* = \mu_* \theta_* / 32$. For a given query $[\chi, \mu, \theta]$, P_χ is the probability with respect to the noise-free example oracle which needs to be estimated with error (μ, θ). Let S be a sample of size $m = \frac{c_1}{\mu_*^2 \theta_*} \ln \frac{c_2}{\delta}$ for which constants c_1 and c_2 are appropriately chosen. Note that this sample size is sufficient to ensure with high probability that the adversary does not corrupt more than a $2\beta_*$ fraction of the examples.

We first show that this sample is large enough to confidently say that either $P_\chi < \theta$ or $P_\chi \geq \theta/8$. Let P_χ^* be the fraction of examples satisfying χ in sample of size m drawn from $EX(f, D)$. With high probability

$$P_\chi < \theta/8 \Rightarrow P_\chi^* < \theta/4 \quad \text{and} \quad P_\chi \geq \theta \Rightarrow P_\chi^* \geq \theta/2.$$

Let $\hat{P}_\chi$ be the fraction of examples in a sample of size m drawn from $EX_{\mathrm{MAL}}^{\beta}(f, D)$. With high probability, the adversary corrupts no more than $2\beta_* \leq \mu_* \theta_* / 16 < \theta/8$ fraction of the examples and therefore by Lemma 6, with high probability

$$P_\chi < \theta/8 \Rightarrow \hat{P}_\chi < 3\theta/8 \quad \text{and} \quad P_\chi \geq \theta \Rightarrow \hat{P}_\chi \geq 3\theta/8.$$

Thus if $\hat{P}_\chi < 3\theta/8$, then $P_\chi < \theta$ and $\perp$ can be returned. Otherwise, $\hat{P}_\chi \geq 3\theta/8$, and therefore $P_\chi \geq \theta/8$. In this case, $\hat{P}_\chi$ is returned, and we must show that $\hat{P}_\chi$ is within a $1 \pm \mu$ factor of P_χ.

Since $P_\chi \geq \theta/8$, a sample of size m ensures with high probability that P_χ^* is within a $1 \pm \mu/2$ factor of P_χ, or equivalently that P_χ^* is within $\pm P_\chi \mu/2$ of P_χ. The sample also ensures with high probability that the adversary corrupts no more than a $2\beta_* = \mu_* \theta_* / 16 \leq P_\chi \mu / 2$ fraction of the examples. Therefore, Lemma 6 ensures with high probability that $\hat{P}_\chi$ is within $\pm P_\chi \mu$ of P_χ. Thus, $\hat{P}_\chi$ is within a $1 \pm \mu$ factor of P_χ.

The total sample complexity required when drawing a separate sample for each query follows directly from the above analysis. The total sample complexity required when using a finite query space follows by noting that on such a sample, the noise-free empirical estimates of *all* queries converge to their true probabilities and the adversary corrupts no more than a $2\beta_*$ fraction of the examples. □

Corollary 3 *If $\mathcal{F}$ is SQ learnable, then $\mathcal{F}$ is PAC learnable in the presence of malicious errors. The dependence on ϵ of the maximum allowable error rate is $\tilde{\Omega}(\epsilon)$, while the dependence on ϵ of the required sample complexity is $\tilde{O}(1/\epsilon)$.*

Note that we are within logarithmic factors of both the $O(\epsilon)$ maximum allowable malicious error rate [13] and the $\Omega(1/\epsilon)$ lower bound on the sample complexity for *noise-free* PAC learning [8]. In this malicious error tolerant PAC simulation, the sample, time, space and hypothesis size complexities are asymptotically identical to the corresponding complexities in our noise-free PAC simulation.

4.5 Very Efficient Malicious Error Learning

In previous sections, we showed general upper bounds on relative error SQ algorithms and the efficiency of PAC algorithms derived from them. In this section, we describe relative error SQ algorithms which actually achieve these bounds and therefore have very efficient, malicious error tolerant PAC simulations. We first present a very efficient algorithm for learning conjunctions[4] in the presence of malicious errors when there are many irrelevant attributes. We then highlight a property of this SQ algorithm which allows for its efficiency, and we further show that many other SQ algorithms naturally exhibit this property as well. We can simulate these SQ algorithms in the PAC model with malicious errors with roughly optimal malicious error tolerance *and* sample complexity.

Decatur [7] gives an algorithm for learning conjunctions which tolerates a malicious error rate *independent* of the number of irrelevant attributes, thus depending only on the number of *relevant* attributes and the desired accuracy. This algorithm, while reasonably efficient, is based on an additive error SQ algorithm of Kearns [12] and therefore does not have an optimal sample complexity.

We present an algorithm based on relative error statistical queries which tolerates the *same* malicious error rate and has a sample complexity whose dependence on ϵ roughly matches the general lower bound for *noise-free* PAC learning.

Theorem 8 *The class of conjunctions of size k over n variables is PAC learnable with malicious errors. The maximum allowable malicious error rate is $\Omega(\frac{\epsilon}{k\log\frac{1}{\epsilon}})$, and the sample complexity required is $O\left(\frac{k^2}{\epsilon}\log^2\frac{1}{\epsilon}\log n + \frac{k}{\epsilon}\log\frac{1}{\epsilon}\log\frac{1}{\delta}\right)$.*

Proof: We present a proof for learning *monotone* conjunctions of size k, and we note that this proof can easily be extended for learning non-monotone conjunctions of size k.

The target function f is a conjunction of k variables. We construct an hypothesis h which is a conjunction of $r = O(k\log\frac{1}{\epsilon})$ variables such that the distribution weight of misclassified positive examples is at most $\epsilon/2$ and the distribution weight of misclassified negative examples is also at most $\epsilon/2$.

First, all variables which could contribute more than $\epsilon/2r$ error on the positive examples are eliminated from consideration. This is accomplished by using the same queries that the monotone conjunction SQ algorithm of Section 4.2 uses. The queries are asked with relative error $1/2$ and threshold $\epsilon/2r$.

Next, the negative examples are greedily "covered" so that the distribution weight of misclassified negative examples is no more than $\epsilon/2$. We say that a variable covers all negative examples for which this variable is false. We know that the set of variables of f is a cover of size k for the entire space of negative examples. We iteratively construct h by conjoining new variables such that the distribution weight of negative examples covered by each new variable is at least a $\frac{1}{2k}$ fraction of the distribution weight of negative examples remaining to be covered.

Given a partially constructed hypothesis $h_j = x_{i_1} \wedge x_{i_2} \wedge \cdots \wedge x_{i_j}$, let X_j^- be the set of negative examples not covered by h_j, *i.e.* $X_j^- = \{x : f(x) = 0 \wedge h_j(x) = 1\}$. Let D_j^- be the *conditional distribution* on X_j^- induced by D, *i.e.* for any $x \in X_j^-$, $D_j^-(x) = D(x)/D(X_j^-)$. By definition, X_0^- is the space of negative examples and D_0^- is the conditional distribution on X_0^-. We know that the target variables not yet in h_j cover the remaining examples in X_j^-, and therefore there exists a cover of X_j^- of size at most k. Thus there exists at least one variable which covers a set of negative examples in X_j^- whose distribution weight with respect to D_j^- is at least $\frac{1}{k}$.

Given h_j, for each x_i, let $\chi_{j,i}(x,l) = [A|B] = [x_i = 0 | l = 0 \wedge h_j(x) = 1]$. Note that $P_{\chi_{j,i}}$ is the distribution weight, with respect to D_j^-, of negative examples in X_j^- covered by x_i. Thus there

[4] By duality, identical results also hold for learning *disjunctions*.

exists a variable x_i such that $P_{\chi_{j,i}}$ is at least $\frac{1}{k}$. To find such a variable, we ask queries of the above form with relative error $\frac{1}{3}$ and threshold $\frac{2}{3k}$. [Note that this is a query for a *conditional* probability, which must be determined by the ratio of two *unconditional* probabilities. We show how to do this below.] Since there exists a variable x_i such that $P_{\chi_{j,i}} \geq \frac{1}{k}$, we are guaranteed to find some variable $x_{i'}$ such that the estimate $\hat{P}_{\chi_{j,i'}}$ is at least $\frac{1}{k}(1-\frac{1}{3}) = \frac{2}{3k}$. Note that if $\hat{P}_{\chi_{j,i'}} \geq \frac{2}{3k}$, then $P_{\chi_{j,i'}} \geq \frac{2}{3k}/(1+\frac{1}{3}) = \frac{1}{2k}$. Thus, by conjoining $x_{i'}$ to h_j, we are guaranteed to cover a set of negative examples in X_j^- whose distribution weight with respect to D_j^- is at least $\frac{1}{2k}$. Since the distribution weight, with respect to D_0^-, of uncovered negative examples is reduced by at least a $(1-\frac{1}{2k})$ factor in each iteration, it is easy to show that this method requires no more than $r = O(k \log\frac{1}{\epsilon})$ iterations to cover all but a set of negative examples whose distribution weight, with respect to D_0^- (and therefore with respect to D) is at most $\epsilon/2$.

We now show how to estimate the conditional probability query $[A|B]$ with relative error $\mu = \frac{1}{3}$ and threshold $\theta = \frac{2}{3k}$. We estimate both queries which constitute the standard expansion of the conditional probability. Appealing to Lemma 3, we first estimate $[B]$, the probability that a negative example is not covered by h, using relative error $\mu/3 = 1/9$ and threshold $\epsilon/2$. If this estimate is $\perp$ or less than $\epsilon/2 \cdot (1-1/9) = \frac{4\epsilon}{9}$, then the weight of negative examples misclassified by h is at most $\epsilon/2$ so we halt and output h. Otherwise we have a lower bound of $\frac{4\epsilon}{9}/(1+1/9) = \frac{2\epsilon}{5}$ on this probability and can therefore estimate $[A \wedge B]$ with relative error $\mu/3 = 1/9$ and threshold $\mu/2 \cdot \epsilon/2 = \frac{\epsilon}{6k}$.

For this algorithm, the worst case relative error is $\Omega(1)$, the worst case threshold is $\Omega(\frac{\epsilon}{k \log\frac{1}{\epsilon}})$ and $\log|\mathcal{Q}| = O(k \log\frac{1}{\epsilon} \log n)$. Therefore, the theorem follows from Theorem 7. □

An important property of this statistical query algorithm is that for every query, we need only to determine whether P_χ falls below some threshold or above some constant fraction of this threshold. This allows the relative error parameter μ to be a constant. The learning algorithm described in Section 4.2 for monotone conjunctions has this property, and we note that many other learning problems which involve "covering," such as learning axis parallel rectangles and decision lists, also have this property. In all these cases we obtain very efficient malicious error tolerant algorithms.

5 Probabilistic and Real-Valued Queries

Throughout this paper, we have assumed that queries submitted to the statistical query oracle were restricted to being deterministic, $\{0,1\}$-valued function of labelled examples. In this case, the oracle returned an estimate of the probability that $\chi(x, f(x)) = 1$ on an example x chosen randomly according to D.

We now generalize the SQ model to allow algorithms to submit queries which are probabilistic and real-valued. Formally, we define a probabilistic real-valued query to be a χ such that for every labelled example $\langle x, l\rangle$, $\chi(x, l)$ is a random variable whose range is $[0, M]$ and whose expectation exists.[5] We define P_χ to be the expected value of χ where the expectation is taken over the draw of the example and the random value of χ given the labelled example. Note that under these conditions, P_χ always exists.

This generalization can be quite useful. If the learning algorithm requires the expected value of some function of labelled examples, it may simply specify this using a real-valued query. Probabilistic queries are required when applying boosting techniques to weak learning algorithms[6] which output probabilistic hypotheses. When applying boosting techniques, the queries constructed are functions of weak hypotheses. Thus if these weak hypotheses are probabilistic, the correspond-

[5]The range $[0, M]$ is used so that we can show relative error simulations. For additive error, one may consider any interval $[a, a+M]$ and simply translate χ.

[6]A weak learning algorithm is one which outputs an hypothesis whose accuracy is just slightly better than random guessing.

ing queries are also be probabilistic. Goldman, Kearns and Schapire [11] show that by allowing a weak learning algorithm to output a probabilistic hypothesis, the complexity of learning is reduced. Therefore this this generalization gives the algorithm designer more freedom and power. Furthermore, the ability to efficiently simulate these algorithms in the PAC model in both the absence and presence of noise is retained, as shown below.

Since the expectation P_χ always exists, the results given below (Theorems 9–14) may be proven identically to the respective deterministic, $\{0,1\}$-valued cases by simply applying Hoeffding and Chernoff style bounds for bounded real random variables (Lemmas 7 and 8). Note that when the range of the queries is a unit sized interval, *i.e.* $M = 1$, these sample complexities and noise tolerances are identical to those for deterministic, $\{0,1\}$-valued queries.

Lemma 7 (Lemma 1.2 in [15]) *Let $X_1, \ldots, X_m$ be independent, identically distributed random variables taking real values in the range $[a, a+M]$. Let Y be the random variable $\frac{1}{m}\sum_{i=1}^m X_i$. Then for any $t > 0$,*

$$\Pr[|Y - E(Y)| \geq t] \leq 2e^{-2mt^2/M^2}.$$

Lemma 8 (Corollary 5.2 in [15]) *Let $X_1, \ldots, X_m$ be independent, identically distributed random variables taking real values in the range $[0,1]$. Let Y be the random variable $\frac{1}{m}\sum_{i=1}^m X_i$ and $p = E(Y)$. Then for any $0 < \alpha < 1$,*

$$\Pr[Y - p \geq \alpha p] \leq e^{-\alpha^2 mp/3}$$

$$\Pr[p - Y \leq -\alpha p] \leq e^{-\alpha^2 mp/2}.$$

Theorem 9 *If $\mathcal{F}$ is learnable by a statistical query algorithm which makes N probabilistic, $[0,M]$-valued queries from query space $\mathcal{Q}$ with worst case additive error τ_*, then $\mathcal{F}$ is PAC learnable with sample complexity $O(\frac{M^2}{\tau_*^2}\log\frac{|\mathcal{Q}|}{\delta})$ when $\mathcal{Q}$ is finite or $O(\frac{NM^2}{\tau_*^2}\log\frac{N}{\delta})$ when drawing a separate sample for each query.*

Theorem 10 *If $\mathcal{F}$ is learnable by a statistical query algorithm which makes N probabilistic, $[0,M]$-valued queries from query space $\mathcal{Q}$ with worst case relative error μ_* and worst case threshold θ_*, then $\mathcal{F}$ is PAC learnable with sample complexity $O(\frac{M}{\mu_*^2\theta_*}\log\frac{|\mathcal{Q}|}{\delta})$ when $\mathcal{Q}$ is finite or $O(\frac{NM}{\mu_*^2\theta_*}\log\frac{N}{\delta})$ when drawing a separate sample for each query.*

Theorem 11 *If $\mathcal{F}$ is learnable by a statistical query algorithm which makes N probabilistic, $[0,M]$-valued queries from query space $\mathcal{Q}$ with worst case additive error τ_*, then $\mathcal{F}$ is PAC learnable in the presence of classification noise. If $\eta_b < 1/2$ is an upper bound on the noise rate, then the sample complexity required is*

$$O\left(\tfrac{M^2}{\tau_*^2(1-2\eta_b)^2}\log\tfrac{|\mathcal{Q}|}{\delta} + \tfrac{1}{\epsilon(1-2\eta_b)^2}\log\log\tfrac{1}{1-2\eta_b}\right)$$

when $\mathcal{Q}$ is finite or

$$O\left(\tfrac{NM^2}{\tau_*^2(1-2\eta_b)^2}\log\tfrac{N}{\delta} + \tfrac{1}{\epsilon(1-2\eta_b)^2}\log\log\tfrac{1}{1-2\eta_b}\right)$$

when drawing a separate sample for each query.

Theorem 12 *If $\mathcal{F}$ is learnable by a statistical query algorithm which makes N probabilistic, $[0,M]$-valued queries from query space $\mathcal{Q}$ with worst case relative error μ_* and worst case threshold θ_*, then $\mathcal{F}$ is PAC learnable in the presence of classification noise. If $\eta_b < 1/2$ is an upper bound on the noise rate, then the sample complexity required is*

$$O\left(\tfrac{M^2}{\mu_*^2\theta_*^2(1-2\eta_b)^2}\log\tfrac{|\mathcal{Q}|}{\delta} + \tfrac{1}{\epsilon(1-2\eta_b)^2}\log\log\tfrac{1}{1-2\eta_b}\right)$$

when $\mathcal{Q}$ is finite or

$$O\left(\frac{NM^2}{\mu_*^2\theta_*^2(1-2\eta_b)^2}\log\frac{N}{\delta}+\frac{1}{\epsilon(1-2\eta_b)^2}\log\log\frac{1}{1-2\eta_b}\right)$$

when drawing a separate sample for each query.

Theorem 13 *If $\mathcal{F}$ is learnable by a statistical query algorithm which makes N probabilistic, $[0,M]$-valued queries from query space $\mathcal{Q}$ with worst case additive error τ_*, then $\mathcal{F}$ is PAC learnable in the presence of malicious errors. The maximum allowable error rate is $\Omega(\tau_*/M)$ and the sample complexity required is $O(\frac{M^2}{\tau_*^2}\log\frac{|\mathcal{Q}|}{\delta})$ when $\mathcal{Q}$ is finite or $O(\frac{NM^2}{\tau_*^2}\log\frac{N}{\delta})$ when drawing a separate sample for each query.*

Theorem 14 *If $\mathcal{F}$ is learnable by a statistical query algorithm which makes N probabilistic, $[0,M]$-valued queries from query space $\mathcal{Q}$ with worst case relative error μ_* and worst case threshold θ_*, then $\mathcal{F}$ is PAC learnable in the presence of malicious errors. The maximum allowable error rate is $\Omega(\mu_*\theta_*/M)$ and the sample complexity required is $O(\frac{M}{\mu_*^2\theta_*}\log\frac{|\mathcal{Q}|}{\delta})$ when $\mathcal{Q}$ is finite or $O(\frac{NM}{\mu_*^2\theta_*}\log\frac{N}{\delta})$ when drawing a separate sample for each query.*

6 Open Questions

The question of what sample complexity is required to simulate statistical query algorithms in the presence of classification noise remains open. The current simulations of both additive and relative error SQ algorithms yield PAC algorithms whose sample complexities depend quadraticly on $1/\epsilon$. However, in the absence of computational restrictions, all finite concept classes can be learned in the presence of classification noise using a sample complexity which depends linearly on $1/\epsilon$ [14].

As discussed in Section 4.5, many classes which are SQ learnable have algorithms with a constant worst case relative error μ_*. Can one show that *all* classes which are SQ learnable have algorithms with this property, or instead characterize exactly which classes do?

References

[1] Dana Angluin. Computational learning theory: Survey and selected bibliography. In *Proceedings of the 24^{th} Annual ACM Symposium on the Theory of Computing*, 1992.

[2] Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

[3] Dana Angluin and Leslie G. Valiant. Fast probabilistic algorithms for Hamiltonian circuits and matchings. *Journal of Computer and System Sciences*, 18(2):155–193, April 1979.

[4] Javed Aslam and Scott Decatur. General bounds on statistical query learning and PAC learning with noise via hypothesis boosting. In *Proceedings of the 34^{th} Annual Symposium on Foundations of Computer Science*, pages 282–291, November 1993. To appear in *Information and Computation.*

[5] Avrim Blum, Merrick Furst, Jeffery Jackson, Michael Kearns, Yishay Mansour, and Steven Rudich. Weakly learning DNF and characterizing statistical query learning using Fourier analysis. In *Proceedings of the 26^{th} Annual ACM Symposium on the Theory of Computing*, May 1994.

[6] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–865, 1989.

[7] Scott Decatur. Statistical queries and faulty PAC oracles. In *Proceedings of the Sixth Annual ACM Workshop on Computational Learning Theory*, pages 262–268. ACM Press, July 1993.

[8] Andrzej Ehrenfeucht, David Haussler, Michael Kearns, and Leslie Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3):247–251, September 1989.

[9] Yoav Freund. Boosting a weak learning algorithm by majority. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 202–216. Morgan Kaufmann, 1990.

[10] Yoav Freund. An improved boosting algorithm and its implications on learning complexity. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 391–398. ACM Press, 1992.

[11] Sally A. Goldman, Michael J. Kearns, and Robert E. Schapire. On the sample complexity of weak learning. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 217–231. Morgan Kaufmann, 1990.

[12] Michael Kearns. Efficient noise-tolerant learning from statistical queries. In *Proceedings of the 25^{th} Annual ACM Symposium on the Theory of Computing*, pages 392–401, San Diego, 1993.

[13] Michael Kearns and Ming Li. Learning in the presence of malicious errors. In *Proceedings of the 20^{th} Annual ACM Symposium on Theory of Computing*, Chicago, Illinois, May 1988.

[14] Philip D. Laird. *Learning from Good and Bad Data*. Kluwer international series in engineering and computer science. Kluwer Academic Publishers, Boston, 1988.

[15] Colin McDiarmid. On the method of bounded differences. In J. Siemons, editor, *Surveys in Combinatorics*, pages 149–188. Cambridge University Press, Cambridge, 1989. London Mathematical Society LNS 141.

[16] Yasubumi Sakakibara. *Algorithmic Learning of Formal Languages and Decision Trees*. PhD thesis, Tokyo Institute of Technology, October 1991. (International Institute for Advanced Study of Social Information Science, Fujitsu Laboratories Ltd, Research Report IIAS-RR-91-22E).

[17] Robert Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–226, 1990.

[18] Hans Ulrich Simon. General bounds on the number of examples needed for learning probabilistic concepts. In *Proceedings of the Sixth Annual ACM Workshop on Computational Learning Theory*, pages 402–411. ACM Press, 1993.

[19] Leslie Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.

[20] Leslie Valiant. Learning disjunctions of conjunctions. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, 1985.