PAC Learning with Nasty Noise

Nader H. Bshouty[*]
Department of Computer Science,
Technion – Israel Institute of Technology,
Haifa 32000, Israel.
email: `bshouty@cs.technion.ac.il`

Nadav Eiron[†]
IBM Almaden Research Center, 650 Harry Road,
San Jose, CA 95120, USA.
email: `nadav@us.ibm.com`

Eyal Kushilevitz
Department of Computer Science,
Technion – Israel Institute of Technology,
Haifa 32000, Israel.
email: `eyalk@cs.technion.ac.il`

Abstract

We introduce a new model for learning in the presence of noise, which we call the *Nasty Noise* model. This model generalizes previously considered models of learning with noise. The learning process in this model, which is a variant of the PAC model, proceeds as follows: Suppose that the learning algorithm during its execution asks for m examples. The examples that the algorithm gets are generated by a nasty adversary that works according to a following steps. First, the adversary chooses m examples (independently) according to the fixed (but unknown to the learning algorithm) distribution D as in the PAC-model. Then the powerful adversary, upon seeing the specific m examples that were chosen (and using his knowledge of the target function, the distribution D and the learning algorithm), is allowed to remove a fraction of the examples at its choice, and replace these examples by the same number of arbitrary examples of its choice; the m modified examples are then given to the learning algorithm. The only restriction on the adversary is that the number of examples that the adversary is allowed to modify should be distributed according to a binomial distribution with parameters η (the noise rate) and m.

On the negative side, we prove that no algorithm can achieve accuracy of $\epsilon < 2\eta$ in learning any non-trivial class of functions. We also give some lower bounds on the sample complexity required to achieve accuracy $\epsilon = 2\eta + \Delta$. On the positive side, we show that a polynomial (in the usual parameters, and in $1/(\epsilon - 2\eta)$) number of examples suffice for learning any class of finite VC-dimension with accuracy $\epsilon > 2\eta$. This algorithm may not be efficient; however, we also show that a fairly wide family of concept classes can be *efficiently* learned in the presence of nasty noise.

[*]This research was supported by the fund for the promotion of research at the Technion. Part of this research was done at the University of Calgary, Calgary, Alberta, Canada.

[†]Some of this research was done while this author was a graduate student in the Department of Computer Science, Technion, Haifa, Israel

1 Introduction

Valiant's *PAC model* of learning [22] is one of the most important models for learning from examples. Although being an extremely elegant model, the PAC model has some drawbacks. In particular, it assumes that the learning algorithm has access to a *perfect* source of random examples. Namely, upon request, the learning algorithm can ask for random examples and in return gets pairs $(x, c_t(x))$ where all the x's are points in the input space distributed *identically and independently* according to some fixed probability distribution D, and $c_t(x)$ is the *correct* classification of x according to the target function c_t that the algorithm tries to learn.

Since Valiant's seminal work, there were several attempts to relax these assumptions, by introducing models of *noise*. The first such noise model, called the *Random Classification Noise* model, was introduced in [2] and was extensively studied, e.g., in [1, 6, 9, 12, 13, 16]. In this model the adversary, before providing each example $(x, c_t(x))$ to the learning algorithm tosses a biased coin; whenever the coin shows "H", which happens with probability η, the classification of the example is flipped and so the algorithm is provided with the, wrongly classified, example $(x, 1-c_t(x))$. Another (stronger) model, called the *Malicious Noise* model, was introduced in [23], revisited in [17], and was further studied in [8, 10, 11, 20]. In this model the adversary, whenever the η-biased coin shows "H", can replace the example $(x, c_t(x))$ by some arbitrary pair (x', b) where x' is any point in the input space and b is a boolean value. (Note that this in particular gives the adversary the power to "distort" the distribution D.)

In this work, we present a new model which we call the *Nasty (Sample) Noise* model. In this model, the adversary gets to see the whole sample of examples requested by the learning algorithm before giving it to the algorithm and then modify E of the examples, *at its choice*, where E is a random variable distributed by the binomial distribution with parameters η and m, where m is the size of the sample. This distribution makes the number of examples modified be the same as if it were determined by m independent tosses of an η-biased coin. However, we allow the adversary's choice to be dependent on the sample drawn. The modification applied by the adversary can be arbitrary (as in the Malicious Noise model).[1] Intuitively speaking, the new adversary is more powerful than the previous ones – it can examine the whole sample and then remove from it the most "informative" examples and replace them by less useful and even misleading examples (whereas in the Malicious Noise Model for instance, the adversary also may insert to the sample misleading examples but does not have the freedom to choose which examples to remove). The relationships between the various models are shown in Table 1.

	Random Noise-Location	Adversarial Noise-Location
Label Noise Only	Random Classification Noise	Nasty Classification Noise
Point and Label Noise	Malicious Noise	Nasty Sample Noise

Table 1: Summary of models for PAC-learning from noisy data

We argue that the newly introduced model, not only generalizes the previous noise models, including variants such as Decatur's CAM model [11] and CPCN model [12], but also applies to real-life situations more appropriately. For example, when training data is the result of some physical experiment, noise may tend to be stronger in boundary areas rather than being uniformly distributed over all inputs. While special models were devised to describe this situation in the exact-learning

[1]We also consider a weaker variant of this model, called the *Nasty Classification Noise* model, where the adversary may modify only the classification of the chosen points (as in the Random Classification Noise model).

setting (for example, the incomplete boundary query model of Blum et al., [5]), it may be regarded as a special case of Nasty Noise, where the adversary chooses to provide unreliable answers on sample points that are near the boundary of the target concept (or to remove such points from the sample). Another situation to which our model is related is the setting of Agnostic Learning. In this model, a concept class is not given. Instead, the learning algorithm needs to minimize the empirical error while using a hypothesis from a predefined hypotheses class (see, for example, [18] for a definition of the model). Assuming the best hypothesis misclassifies the input up to an η fraction, we may alternatively see the problem as that of learning the hypotheses class under nasty noise of rate η. However, we note that the success criterion in the agnostic learning literature is different from the one used in our PAC-based setting.

We show two types of results. Sections 3 and 4 show information theoretic results, and Sect. 5 shows algorithmic results. The first result, presented in Section 3, is a lower bound on the accuracy achievable when learning with a nasty adversary. This result shows that any learning algorithm cannot learn any non-trivial concept class with accuracy better than 2η when the sample contains nasty noise of rate η. We further show that learning a concept class of VC dimension d with accuracy $\epsilon = \Delta + 2\eta$ requires $\Omega(\eta/\Delta^2 + d/\Delta)$ examples (assuming a constant confidence parameter δ). It is complemented by a matching positive result in Section 4 that shows that any class of finite VC-dimension can be learned by using a sample of polynomial size, with any accuracy $\epsilon > 2\eta$. The size of the sample required is polynomial in the usual PAC parameters and in $1/\Delta$ where $\Delta = \epsilon - 2\eta$ is the margin between the requested accuracy ϵ and the above mentioned lower bound.

The main, quite surprising, result (presented in Section 5) is another positive result showing that *efficient* learning algorithms are still possible in spite of the powerful adversary. More specifically, we present a composition theorem (analogous to [3, 8] but for the nasty-noise learning model) that shows that any concept class that is constructed by composing concept classes that are PAC-learnable from a hypothesis class of fixed VC-dimension, is efficiently learnable when using a sample subject to nasty noise. This includes, for instance, the class of all concepts formed by any boolean combination of half-spaces in a constant dimensional Euclidean space. The complexity here is, again, polynomial in the usual parameters and in $1/\Delta$. The algorithm used in the proof of this result is an adaptation to our model of the PAC algorithm presented in [8].

Our results may be compared to similar results available for the Malicious Noise model. For this model, Cesa-Bianchi et al. [10] show that the accuracy of learning with malicious noise of rate η is lower bounded by $\eta/(1-\eta)$. A matching algorithm for learning classes similar to those presented here with malicious noise is presented in [8]. As for the Random Classification Noise model, learning with arbitrarily small accuracy, even when the noise rate is close to a half, is possible. Again, the techniques presented in [8] may be used to learn the same type of classes we examine in this work with Random Classification Noise.

2 Preliminaries

In this section we provide basic definitions related to learning in the PAC model, with and without noise. A learning task is specified using a *concept class*, denoted $\mathcal{C}$, of boolean concepts defined over an *instance space*, denoted $\mathcal{X}$. A boolean concept c is a function $c : \mathcal{X} \mapsto \{0,1\}$. The concept class $\mathcal{C}$ is a set of boolean concepts: $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$.

Throughout this paper we sometimes treat a concept as a set of points instead of as a boolean function. The set that corresponds to a concept c is simply $\{x | c(x) = 1\}$. We use c to denote both the function and the corresponding set interchangeably. Specifically, when a probability distribution D is defined over $\mathcal{X}$, we use the notation $D(c)$ to refer to the probability that a point x drawn from

$\mathcal{X}$ according to D will have $c(x) = 1$.

2.1 The Classical PAC Model

The *Probably Approximately Correct (PAC)* model was originally presented by Valiant [22]. In this model, the learning algorithm has access to an oracle PAC that returns on each call a labeled example $(x, c_t(x))$ where $x \in \mathcal{X}$ is drawn (independently) according to a fixed distribution D over $\mathcal{X}$, unknown to the learning algorithm, and $c_t \in \mathcal{C}$ is the target function the learning algorithm should "learn".

Definition 1: A class $\mathcal{C}$ of boolean functions is *PAC-learnable using hypothesis class $\mathcal{H}$* in polynomial time if there exists an algorithm that, for any $c_t \in \mathcal{C}$, any input parameters $0 < \epsilon < 1/2$ and $0 < \delta < 1$ and any distribution D on $\mathcal{X}$, when given access to the PAC oracle, runs in time polynomial in the representation size of the target, $1/\delta$, $1/\epsilon$ and with probability at least $1-\delta$ outputs a function $h \in \mathcal{H}$ for which:

$$\Pr_D[c_t(x) \neq h(x)] < \epsilon.$$

2.2 Models for Learning in the Presence of Noise

Next, we define the model of PAC-learning in the presence of *Nasty Sample Noise* (NSN for short). In this model, a learning algorithm for the concept class $\mathcal{C}$ is given access to an (adversarial) oracle $\mathrm{NSN}_{\mathcal{C},\eta}(m)$. The learning algorithm is allowed to call this oracle *once* during a single run. The learning algorithm passes a single natural number m to the oracle, specifying the size of the sample it needs, and gets in return a labeled sample $S \in (\mathcal{X} \times \{0, 1\})^m$. (It is assumed, for simplicity, that the algorithm knows in advance the number of examples it needs; the extension of the model for scenarios where such a bound is not available in advance is given in Section 6.)

The sample required by the learning algorithm is constructed as follows: As in the PAC model, a distribution D over the instance space $\mathcal{X}$ is defined, and a target concept $c_t \in \mathcal{C}$ is chosen. The adversary then draws a sample S_g of m points from $\mathcal{X}$ according to the distribution D. Having full knowledge of the learning algorithm, the target function c_t, the distribution D, and the sample drawn, the adversary chooses $E = E(S_g)$ points from the sample, where $E(S_g)$ is a random variable (which may also depend on the adversary's internal randomness). The E points chosen by the adversary are removed from the sample and replaced by any other E point-and-label pairs by the adversary. The $m - E$ points not chosen by the adversary remain unchanged and are labeled by their correct labels according to c_t. The modified sample of m points, denoted S, is then given to the learning algorithm. The only limitation that the adversary has on the number of examples that it may modify is that it should be distributed according to the binomial distribution with parameters m and η, namely:

$$\Pr[E = n] = \binom{m}{n} \eta^n (1 - \eta)^{m-n},$$

where the probability is taken by first choosing $S_g \in D^m$ and then choosing E according to the corresponding random variable $E(S_g)$.

Definition 2: An algorithm $\mathcal{A}$ is said to learn a class $\mathcal{C}$ with nasty sample noise of rate $\eta \geq 0$ with accuracy parameter $\epsilon > 0$ and confidence parameter $\delta < 1$ if, given access to any oracle $\mathrm{NSN}_{\mathcal{C},\eta}(m)$, for any distribution D and any target $c_t \in \mathcal{C}$ it outputs a hypothesis $h : \mathcal{X} \mapsto \{0, 1\}$ such that, with probability at least $1 - \delta$

$$\Pr_D[h \triangle c_t] < \epsilon.$$

We are also interested in a restriction of this model, which we call the *Nasty Classification Noise* learning model (NCN for short). The only difference between the NCN and NSN models is that the NCN adversary is only allowed to modify the labels of the E chosen sample-points, but it cannot modify the E points themselves. Previous models of learning in the presence of noise can also be readily shown to be restrictions of the Nasty Sample Noise model: The Malicious Noise model corresponds to the Nasty Noise model with the adversary restricted to introducing noise into points that are chosen uniformly at random, with probability η, from the original sample. The Random Classification Noise model corresponds to the Nasty Classification Noise model with the adversary restricted so that noise is introduced into points chosen uniformly at random, with probability η, from the original sample, and each point that is chosen gets its label flipped.

2.3 VC Theory Basics

The VC-dimension [24], is widely used in learning theory to measure the complexity of concept classes. We say a set $Y \subseteq \mathcal{X}$ is shattered by $\mathcal{C}$ if, for each of the $2^{|Y|}$ possible labelings of the points in Y, there exists some function in $\mathcal{C}$ consistent with that labeling. The VC-dimension of a class $\mathcal{C}$, denoted $\mathrm{VCdim}(\mathcal{C})$, is the maximal integer d such that there exists a subset $Y \subseteq \mathcal{X}$ of cardinality d that is shattered by the class $\mathcal{C}$. We say $\mathrm{VCdim}(\mathcal{C}) = \infty$ if such a subset exists for any natural d. It is well known (e.g., [4]) that, for any two classes $\mathcal{C}$ and $\mathcal{H}$ (over $\mathcal{X}$), the class of negations $\{c | \mathcal{X} \setminus c \in \mathcal{C}\}$ has the same VC-dimension as the class $\mathcal{C}$, and the class of unions $\{c \cup h | c \in \mathcal{C},\ h \in \mathcal{H}\}$ has VC-dimension at most $\mathrm{VCdim}(\mathcal{C}) + \mathrm{VCdim}(\mathcal{H}) + 1$. Following [3] we define the dual of a concept class:

Definition 3: The *dual* $\mathcal{H}^{\perp} \subseteq \{0,1\}^{\mathcal{H}}$ of a class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ is defined to be the set $\left\{x^{\perp} | x \in \mathcal{X}\right\}$ where $x^{\perp}$ is defined by $x^{\perp}(h) = h(x)$ for all $h \in \mathcal{H}$.

If we view a concept class $\mathcal{H}$ as a boolean matrix where each row represents a concept and each column a point from the instance space, $\mathcal{X}$, then the matrix corresponding to $\mathcal{H}^{\perp}$ is the transpose of the matrix corresponding to $\mathcal{H}$. The following claim, from [3], gives a tight bound on the VC dimension of the dual class:

Claim 1: For every class $\mathcal{H}$,

$$\mathrm{VCdim}(\mathcal{H}) \geq \left\lfloor \log \mathrm{VCdim}(\mathcal{H}^{\perp}) \right\rfloor .$$

In the following discussion we limit ourselves to concept classes of finite concepts (i.e., $\forall c \in \mathcal{C}$, $|c|$ is finite). The main use we make of the VC-dimension is in constructing α-nets. The following definition and theorem are from [7]:

Definition 4: A set of points $Y \subseteq \mathcal{X}$ is an *α-net* for the concept class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ under the distribution D over $\mathcal{X}$, if for every $h \in \mathcal{H}$ such that $D(h) \geq \alpha$, $Y \cap h \neq \emptyset$.

Theorem 1: For any class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ of VC-dimension d, any distribution D over $\mathcal{X}$, and any $\alpha > 0$, $\delta > 0$, if

$$m \geq \max\left\{\frac{4}{\alpha}\log\frac{2}{\delta}, \frac{8d}{\alpha}\log\frac{13}{\alpha}\right\}$$

examples are drawn i.i.d. from $\mathcal{X}$ according to the distribution D, they constitute an α-net for $\mathcal{H}$ with probability at least $1 - \delta$.

In [21], Talagrand proved a similar result:

Definition 5: A set of points $Y \subseteq \mathcal{X}$ is an α*-sample* for the concept class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ under the distribution D over $\mathcal{X}$, if it holds that for every $h \in \mathcal{H}$:

$$\left| D(h) - \frac{|Y \cap h|}{|Y|} \right| \leq \alpha$$

Theorem 2: There is a constant c_1, such that for any class $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ of VC-dimension d, and distribution D over $\mathcal{X}$, and any $\alpha > 0$, $\delta > 0$, if

$$m \geq \frac{c_1}{\alpha^2}\left(d + \log\frac{1}{\delta}\right)$$

examples are drawn i.i.d. from $\mathcal{X}$ according to the distribution D, they constitute an α-sample for $\mathcal{H}$ with probability at least $1 - \delta$.

3 Information Theoretic Lower Bound

In this section we show that no learning algorithm (not even inefficient ones) can learn a "non-trivial" concept class with accuracy ϵ better than 2η under the NSN model; in fact, we prove that this impossibility result holds even for the NCN model. We also give some results on the size of samples required to learn in the NSN model with accuracy $\epsilon > 2\eta$.

Definition 6: A concept class $\mathcal{C}$ over an instance space $\mathcal{X}$ is called *non-trivial* if there exist two points $x_1, x_2 \in \mathcal{X}$ and two concepts $c_1, c_2 \in \mathcal{C}$, such that $c_1(x_1) = c_2(x_1)$ and $c_1(x_2) \neq c_2(x_2)$.

Theorem 3: Let $\mathcal{C}$ be a non-trivial concept class, η be a noise rate and $\epsilon < 2\eta$ be an accuracy parameter. Then, there is no algorithm that learns the concept class $\mathcal{C}$ with accuracy ϵ and confidence parameter $\delta < \frac{1}{2}$ under the NCN model (with rate η).

Proof: We base our proof on the method of induced distributions introduced in [17, Theorem 1]. We show that there are two concepts $c_1, c_2 \in \mathcal{C}$ and a probability distribution D such that $\Pr_D[c_1 \triangle c_2] = 2\eta$ and an adversary can force the labeled examples shown to the learning algorithm to be distributed identically both when c_1 is the target and when c_2 is the target.

Let c_1 and c_2 be the two concepts whose existence is guaranteed by the fact that $\mathcal{C}$ is a non-trivial class, and let $x_1, x_2 \in \mathcal{X}$ be the two points that satisfy $c_1(x_1) = c_2(x_1)$ and $c_1(x_2) \neq c_2(x_2)$. We define the probability distribution D to be $D(x_1) = 1 - 2\eta$, $D(x_2) = 2\eta$, and $D(x) = 0$ for all $x \in \mathcal{X} \setminus \{x_1, x_2\}$. Clearly, we indeed have $\Pr_D[c_1 \triangle c_2] = \Pr_D[x_2] = 2\eta$.

Now, we define the nasty adversary strategy (with respect to the above probability distribution D). Let m be the size of the sample asked by the learning algorithm. The adversary starts by drawing a sample S_g of m points according to the above distribution. Then, for each occurrence of x_1 in the sample, the adversary labels it correctly according to c_t, while for each occurrence of x_2 the adversary tosses a coin and with probability 1/2 it labels the point correctly (i.e., $c_t(x_2)$) and with probability 1/2 it flips the label (to $1 - c_t(x_2)$). The resulted sample of m examples is given by the adversary to the learning algorithm. First, we argue that the number of examples modified by the adversary is indeed distributed according to the binomial distribution with parameters η and m. For this, we view the above adversary as if it picks independently m points and for each of them decides (as above) whether to flip its label. Hence, it suffices to show that each example is labeled incorrectly with probability η independently of other examples. Indeed, for each example independently, its probability of being labeled incorrectly equals the probability of choosing x_2 according to D times the probability that the adversary chooses to flip the label on an x_2 example; i.e. $2\eta \cdot (1/2) = \eta$ as needed. (We emphasize that the binomial distribution is obtained because D is known to the adversary.)

Next observe that, no matter whether the target is c_1 or c_2, the examples given to the learning algorithm (after being modified by the above nasty adversary) are distributed according to the following probability distribution:

$$\begin{aligned}\Pr[x_1, c_1(x_1)] &= 1 - 2\eta \\ \Pr[x_2, c_1(x_2)] &= \Pr[x_2, c_2(x_2)] = \eta\end{aligned}$$

Therefore, according to the sample that the learning algorithm sees, it is impossible to differentiate between the case where the target function is c_1 and the case where the target function is c_2, hence no algorithm can do any better than a random guess, which gives confidence $\frac{1}{2}$. □

Note that in the above proof we indeed take advantage of the "nastiness" of the adversary. Unlike the malicious adversary, our adversary can focus all its "power" on just the point x_2, causing it to suffer a relatively high error rate, while examples in which the point is x_1 do not suffer any noise. We also took advantage of the fact that E (the number of modified examples) is allowed to depend on the sample (in our case, it depends on the number of times x_2 appears in the original sample). This allows the adversary to further focus its destructive power on samples which were otherwise "good" for the learning algorithm. Finally, since any NCN adversary is also a NSN adversary, Theorem 3 implies the following:

Corollary 4: Let $\mathcal{C}$ be a non-trivial concept class, $\eta > 0$ be the noise rate, and $\epsilon < 2\eta$ be an accuracy parameter. There is no algorithm that learns the concept class $\mathcal{C}$ with accuracy ϵ and confidence parameter $\delta < \frac{1}{2}$ under the NSN model, with noise rate η.

Once we have settled 2η as the lower bound on the accuracy possible with a nasty adversary with error rate η, we turn to the question of the number of examples that are necessary to learn a concept class with some accuracy $\epsilon = 2\eta + \Delta$, where $\Delta > 0$. Again, in this section we are only considering information-theoretic issues. These results are similar to those presented by Cesa-Bianchi et al. [10] for the Malicious Noise model. Note, however, that the definition of the margin Δ used here is relative to a lower bound different than the one used in [10]. In the proofs of these results, we use the following claim (see [10]) that provides a lower bound on the probability that a random variable of binomial distribution deviates from its expectation by more than the standard deviation:

Claim 2: [10, Fact 3.2] Let $S_{N,p}$ be a random variable distributed by the binomial distribution with parameters N and p, and let $q = 1 - p$. For all $N > 37/(pq)$:

$$\Pr\left[S_{N,p} \geq \lfloor Np \rfloor + \left\lfloor \sqrt{Npq - 1} \right\rfloor\right] > \frac{1}{19} \quad (1)$$

$$\Pr\left[S_{N,p} \leq \lceil Np \rceil - \left\lfloor \sqrt{Npq - 1} \right\rfloor\right] > \frac{1}{19} \quad (2)$$

Theorem 5: For any nontrivial concept class $\mathcal{C}$, any noise rate $\eta > 0$, confidence parameter $0 < \delta < 1/342$ and $0 < \Delta < 0.084065 \cdot \eta$, the sample size needed for PAC learning $\mathcal{C}$ with accuracy $\epsilon = 2\eta + \Delta$ and confidence $1 - \delta$ tolerating nasty classification noise of rate η is at least

$$m \geq \frac{17\eta(1-\eta)}{37\Delta^2} = \Omega\left(\frac{\eta}{\Delta^2}\right).$$

Proof: Let c_1 and c_2 be the two concepts whose existence is guaranteed by the fact that $\mathcal{C}$ is a non-trivial class, and let $x_1, x_2 \in \mathcal{X}$ be the two points that satisfy $c_1(x_1) = c_2(x_1)$ and $c_1(x_2) \neq c_2(x_2)$. Let us define a distribution D that gives weight ϵ to the point x_2 and weight $1-\epsilon$ to x_1, making $\Pr_D(c_1 \triangle c_2) = \epsilon$. Denote by c_t the target function (either c_1 or c_2).

The Nasty Classification Adversary will use the following strategy: for each pair of the form $(x_2, c_t(x_2))$ in the sample, with probability η/ϵ reverse the label (i.e., present to the learning algorithm the pair $(x_2, 1-c_t(x_2))$ instead). The rest of the sample (all the pairs of the form $(x_1, c_t(x_1))$) is left unmodified. Note that for each of the m examples the probability that its classification is changed is therefore exactly $\epsilon \cdot \frac{\eta}{\epsilon} = \eta$ and so the number of points that suffer noise is indeed distributed according to the binomial distribution with parameters η and m. The induced probability distribution on the sample that the learning algorithm sees is:

$$\begin{aligned}
\Pr(x_1, c_t(x_1)) &= 1-\epsilon \\
\Pr(x_2, c_t(x_2)) &= \epsilon \cdot (1 - \frac{\eta}{\epsilon}) = \epsilon - \eta \quad = \quad \eta + \Delta \\
\Pr(x_2, 1-c_t(x_2)) &= \epsilon \cdot \frac{\eta}{\epsilon} \quad = \quad \eta
\end{aligned}$$

For contradiction, let A be a (possibly randomized) algorithm that learns $\mathcal{C}$ with accuracy ϵ using a sample generated by the above oracle and whose size is $m < (17\eta(1-\eta))/(37\Delta^2)$. We denote by $p_A(m)$ the expected error of the hypothesis h that A outputs when using m examples. Let B be the Bayes strategy of outputting c_1 if the majority instances of x_2 are labeled $c_1(x_2)$, and c_2 otherwise. Clearly, this strategy minimizes the probability of choosing the wrong hypothesis. This implies $p_B(m) \leq p_A(m)$ for all m.

Define the following two events over runs of B on samples of size m: Let N denote the number of examples in m showing the point x_2. BAD_1 is the event that at least $\lceil N/2 \rceil + 1$ points are corrupted, and BAD_2 is the event that $N \leq 36\eta(\eta+\Delta)/(37\Delta^2)$. Clearly, BAD_1 implies that B will answer incorrectly, as there will be more examples showing x_2 with the wrong label than examples showing x_2 with the correct label. To prove our theorem we will give two lower bounds: One for $\Pr[\text{BAD}_1 | \text{BAD}_2]$, and the other for $\Pr[\text{BAD}_2]$.

Examine now the probability that BAD_2 will occur. Note that N is a random variable distributed by the binomial distribution with parameters m and ϵ (and recall that $\epsilon = 2\eta + \Delta$). We are interested in:

$$\Pr[\text{BAD}_2] = \Pr\left[N \leq \frac{36\eta(\eta+\Delta)}{37\Delta^2}\right] \geq \Pr\left[N \leq \frac{18\eta\epsilon}{37\Delta^2}\right].$$

Since the probability that N is large is higher when m is larger, and m is upper bounded by $(17\eta(1-\eta))/(37\Delta^2)$, we have that:

$$\Pr[\text{BAD}_2] \geq \Pr\left[N \leq \frac{18}{17} m\epsilon\right] = \Pr\left[\frac{N}{E[N]} - 1 \leq \frac{1}{17}\right].$$

Let us assume for now, that $m \geq 17/\epsilon$ (we will later see that this assumption is superfluous). Thus, using Chernoff's inequality, we get:

$$\Pr[\text{BAD}_2] \geq 1 - e^{-1/17}.$$

On the other hand, if we assume that BAD_2 holds, namely that $N \leq 36\eta(\eta+\Delta)/(37\Delta^2)$, and we additionally assume that $N \geq 37(2\eta+\Delta)^2/(\eta(\eta+\Delta))$ then, by Claim 2 (with $p = \eta/\epsilon = \eta/(2\eta+\Delta)$)

and using the following inequality

$$\left\lfloor N\frac{\eta}{2\eta+\Delta}\right\rfloor + \left\lfloor\sqrt{N\frac{\eta(\eta+\Delta)}{(2\eta+\Delta)^2}-1}\right\rfloor \;\; \geq \;\; \left\lceil \frac{1}{2}N\right\rceil + 1 \tag{3}$$

it follows that $\Pr[\mathrm{BAD}_1|\mathrm{BAD}_2] > 1/19$. To see that the inequality of Equation (3) indeed holds when BAD_2 holds, note that (3) is implied by:

$$N\frac{\eta}{2\eta+\Delta} + \sqrt{N\frac{\eta(\eta+\Delta)}{(2\eta+\Delta)^2}-1} \geq \;\; \frac{1}{2}N + 3$$

which is, in turn, implied by the two conditions:

$$\begin{aligned} \frac{1}{2}\sqrt{N\frac{\eta(\eta+\Delta)}{(2\eta+\Delta)^2}-1} \;\; &\geq \;\; 3 \\ \frac{1}{2}\sqrt{N\frac{\eta(\eta+\Delta)}{(2\eta+\Delta)^2}-1} \;\; &\geq \;\; \frac{1}{2}N\frac{\Delta}{2\eta+\Delta} \end{aligned}$$

It can be verified that these two conditions hold if we take N to be in the range we assume:[2] $36\eta(\eta+\Delta)/(37\Delta^2) \geq N \geq 37(2\eta+\Delta)^2/(\eta(\eta+\Delta))$. Since B is an optimal strategy, and hence no worse than a strategy that ignores some of the sample points, its error can only decrease if more points are shown for x_2 (otherwise, assume B achieves better error bounds for lower values of N. Now, an algorithm C that modifies some of the sample points that show x_2 to examples showing x_1, then run B on the resulting sample, will perform better than B, in contradiction with the optimality of the Bayes strategy). Therefore, the same results will hold if we remove the lower bound on N. We thus have that $\Pr[\mathrm{BAD}_1] \geq \Pr[\mathrm{BAD}_1|\mathrm{BAD}_2]\cdot\Pr[\mathrm{BAD}_2] \geq 1/19\cdot(1-e^{-1/17}) > 1/342$, under the assumption that $m \geq 17/\epsilon$.

We now argue that the lower bound on m may be removed. Assume that this assumption on m is indeed required. This means that there exists a learning algorithm A that can learn with noise rate η and confidence δ to within accuracy $\eta + 2\Delta$ using $m_A < 17/\epsilon$ examples. Note further, that by our conditions on Δ, we have that $17/\epsilon < 17\eta(1-\eta)/(37\Delta^2)$. Consider an algorithm B that uses $17/\epsilon < m_B < 17\eta(1-\eta)/(37\Delta^2)$ examples and works as follows: B randomly draws m_A examples from its sample, and feeds them to A. Clearly, B is a learning algorithm with the required parameters (since we assumed A is), but, B contradicts our proof, since it uses $m_B > 17/\epsilon$ examples, but less than $17\eta(1-\eta)/(37\Delta^2)$ examples. Thus, no such algorithm A may exist, and the lower bound on m may be safely removed. □

A second type of a lower bound on the number of required examples is based on the VC dimension of the class to be learned, and is similar to the results (and the proof techniques) of [7] for the standard PAC model:

Theorem 6: For any concept class $\mathcal{C}$ with VC-dimension $d \geq 3$, and for any $0 < \epsilon \leq 1/8$, $0 < \Delta < \epsilon$ and $0 < \delta \leq 1/12$, the sample size required to learn $\mathcal{C}$ with accuracy ϵ and confidence δ when using samples generated by a nasty classification adversary with error rate $\eta = 1/2(\epsilon - \Delta)$ is greater than

$$\frac{d-2}{32\Delta} = \Omega\left(\frac{d}{\Delta}\right).$$

[2]By our conditions on Δ there must be at least one integer in the range we assume for N.

Proof: Let $X = \{x_0, \ldots, x_{d-1}\}$ be a set of d points shattered by $\mathcal{C}$. Define a probability distribution D as follows:

$$\begin{aligned} D(x_0) &= 1 - 2\eta - 8\Delta \\ D(x_1) = \cdots = D(x_{d-2}) &= \frac{8\Delta}{d-2} \\ D(x_{d-1}) &= 2\eta \end{aligned}$$

Assume for contradiction that at most $(d-2)/32\Delta$ examples are used by the learning algorithm. We let the nasty adversary behave as follows: it reverses the label on each example x_{d-1} with probability 1/2 (independent of any other sample points), making the labels of x_{d-1} appear as if they are just random noise (also note that the probability of each example to be corrupted by the adversary is exactly $2\eta \cdot 0.5 = \eta$). Thus, with probability 1/2 the point x_{d-1} is misclassified by the learner's hypothesis. The rest of the sample is left unmodified. Denote by BAD_1 the event that at least half of the points $x_1, \ldots, x_{d-2}$ are not seen by the learning algorithm. Given BAD_1, we denote UP the set of $(d-2)/2$ unseen points with lowest indices, and define BAD_2 as the event that the algorithm's hypothesis misclassifies at least $(d-2)/8$ points from UP. Finally, let BAD_3 denote the event that x_{d-1} is misclassified. It is easy to see that $\text{BAD}_1 \wedge \text{BAD}_2 \wedge \text{BAD}_3$ imply that the hypothesis has error of at least ϵ, as it implies that the hypothesis errs on $(d-2)/8$ points where each of these points has weight $8\Delta/(d-2)$ and on the point x_{d-1} whose weight is 2η, making the total error at least $\Delta + 2\eta = \epsilon$. Therefore, if an algorithm A can learn the class with confidence δ, it must hold that $\Pr[\text{BAD}_1 \wedge \text{BAD}_2 \wedge \text{BAD}_3] < \delta$. As noted before, x_{d-1} appears to be labeled by random noise, and hence $\Pr[\text{BAD}_3] = \frac{1}{2}$, no matter what the true classification of x_{d-1} is; moreover, BAD_3 is independent of BAD_1 and BAD_2, thus $\Pr[\text{BAD}_3|\text{BAD}_1, \text{BAD}_2] = \frac{1}{2}$.

As for the other events, since at most $(d-2)/32\Delta$ examples are seen, the expected number of points from $x_1, \ldots, x_{d-2}$ that the learning algorithm sees is at most $(d-2)/4$. From the Markov inequality it follows that, with probability at least $\frac{1}{2}$, no more than $(d-2)/2$ points are seen. Hence, $\Pr[\text{BAD}_1] > \frac{1}{2}$. Every unseen point will be misclassified by the learning algorithm with probability at least half (since for each such point, the adversary could have chosen a target that labels the point with the label that has lower probability to be given by the algorithm). Thus $\Pr[\text{BAD}_2|\text{BAD}_1]$ is the probability that a fair coin flipped $(d-2)/2$ times shows heads at least $(d-2)/8$ times. Using [10, Fact 3.3] this probability can be shown to be at least 1/3. We thus have:

$$\begin{aligned} \Pr[\text{BAD}_1 \wedge \text{BAD}_2 \wedge \text{BAD}_3] &= \Pr[\text{BAD}_1] \cdot \Pr[\text{BAD}_2|\text{BAD}_1] \cdot \Pr[\text{BAD}_3|\text{BAD}_1, \text{BAD}_2] \\ &> \frac{1}{2} \cdot \frac{1}{3} \cdot \frac{1}{2} = \frac{1}{12} \\ &\geq \delta. \end{aligned}$$

This completes the proof. □

Since learning with Nasty Sample Noise is not easier than learning with Nasty Classification Noise, the results of Theorems 5 and 6 also hold for learning from a Nasty Sample Noise oracle.

4 Information Theoretic Upper Bound

In this section we provide a positive result that complements the negative result of Section 3. This result shows that, given a sufficiently large sample, any hypothesis that performs sufficiently well on the sample (even when this sample is subject to nasty noise) satisfies the PAC learning condition.

Formally, we analyze the following generic algorithm for learning any class $\mathcal{C}$ of VC-dimension d, whose inputs are a certainty parameter $\delta > 0$, the nasty error rate parameter $\eta < \frac{1}{2}$ and the required accuracy $\epsilon = 2\eta + \Delta$:

Algorithm NastyConsistent:

1. Request a sample $S = \{(x, b_x)\}$ of size
$$m \geq \frac{c}{\Delta^2}\left(d + \log\frac{2}{\delta}\right)$$
2. Output any $h \in \mathcal{C}$ such that
$$|\{x \in S \,:\, h(x) \neq b_x\}| \leq m\left(\eta + \frac{\Delta}{4}\right)$$
(if no such h exists, choose any $h \in \mathcal{C}$ arbitrarily).

Theorem 7: Let $\mathcal{C}$ be any class of VC-dimension d. Then, (for some constant c) algorithm NastyConsistent is a PAC learning algorithm under nasty sample noise of rate η.

In our proof of this theorem (as well as in the analysis of the algorithm in the next section), we use, for convenience, a slightly weaker definition of PAC learnability than the one used in Definition 1. We require the algorithm to output, with probability at least $1 - \delta$, a function h for which:
$$\Pr_D[c_t(x) \neq h(x)] \leq \epsilon$$
(rather than a strict inequality). However, if we use the same algorithm but give it a slightly smaller accuracy parameter (e.g., $\epsilon' = 2\eta + \Delta/2 < \epsilon$), we will get an algorithm that learns using the original criterion of Definition 1.

Proof: First, we argue that with "high probability" the number of sample points that are modified by the adversary is at most $m(\eta + \Delta/4)$. As the random variable E is distributed according to the binomial distribution with expectation ηm, we may use Hoeffding's inequality [14] to get:
$$\Pr\left[E > \left(\eta + \frac{\Delta}{4}\right)m\right] \leq \exp\left(-\frac{2m\Delta^2}{16}\right).$$
Since $m \geq \frac{8}{\Delta^2}\ln\frac{2}{\delta}$ (by the choice of c), this event happens with probability at most $\delta/2$.

Now, we note that the target function c_t errs on at most E points of the sample shown to the learning algorithm (as it is completely accurate on the non-modified sample S_g). Thus, with probability at least $1 - \delta/2$, Algorithm NastyConsistent will be able to choose a function $h \in \mathcal{C}$ that errs on no more that $(\eta + \Delta/4)m$ points of the sample shown to it. However, in the worst case, these errors of the function h occur in points that were not modified by the adversary. In addition, h may be erroneous for all the points that the adversary did modify. Therefore, all we are guaranteed in this case, is that the hypothesis h errs on no more that $2(\eta + \Delta/4)m$ points of the original sample S_g. By Theorem 2, there exists a constant c such that, with probability $1 - \delta/2$, by taking S_g to be of size at least $\frac{c}{\Delta^2}\left(d + \log\frac{2}{\delta}\right)$ the resulting sample S_g is a $\frac{\Delta}{2}$-sample for the class of symmetric differences between functions from $\mathcal{C}$ and the target $\{h \triangle c_t \,:\, h \in \mathcal{C}\}$. By the union bound we therefore have that, with probability at least $1-\delta$, $E \le (\eta+\Delta/4)m$, meaning that $|S_g \cap (c_t \triangle h)| \le (2\eta+\Delta/2)m$, and that S_g is a $\Delta/2$-sample for the class of symmetric differences, and so:

$$\Pr_D[(c_t \triangle h)] \le 2\eta + \Delta = \epsilon$$

as required. □

5 Composition Theorem for Learning with Nasty Noise

Following [3] and [8], we define the notion of "composition class": Let $\mathcal{C}$ be a class of boolean functions $g : \mathcal{X} \mapsto \{0,1\}^n$. Define the class $\mathcal{C}^\star$ to be the set of all boolean functions $F(x)$ that can be represented as $f(g_1(x), \ldots, g_k(x))$ where f is any boolean function, and $g_i \in \mathcal{C}$ for $i = 1, \ldots, k$. We define the size of $f(g_1, \ldots, g_k)$ to be k. Given a vector of hypotheses $(h_1, \ldots, h_t) \in \mathcal{H}^t$ we define, following [8], the set $\mathcal{W}(h_1, \ldots, h_t)$ to be the set of sub-domains $W_a = \{x | (h_1(x), \ldots, h_t(x)) = a\}$ for all possible vectors $a \in \{0,1\}^t$.

The remainder of this section presents a learning algorithm that can learn composition classes even when the training sample suffers from nasty noise. Section 5.1 describes results of [8] relating to consistency algorithms that we use in the construction of our learning algorithm. Our new algorithm and its analysis are presented in Section 5.2.

5.1 Consistency Algorithms

Let P and N be subsets of points from $\mathcal{X}$. We say that a function $h : \mathcal{X} \mapsto \{0,1\}$ is consistent on (P,N) if $h(x) = 1$ for every "positive point" $x \in P$ and $h(x) = 0$ for every "negative point" $x \in N$. A *consistency algorithm* (see [8]) for a pair of classes $(\mathcal{C}, \mathcal{H})$ (both over the same instance space $\mathcal{X}$, with $\mathcal{C} \subseteq \mathcal{H}$), receives as input two subsets of the instance space, (P,N), runs in time polynomial in $|P \cup N|$, and satisfies the following. If there is a function in $\mathcal{C}$ that is consistent with (P,N), the algorithm outputs "YES" and some $h \in \mathcal{H}$ that is consistent with (P,N); the algorithm outputs "NO" if no consistent $h \in \mathcal{H}$ exist (there is no restriction on the output in the case that there is a consistent function in $\mathcal{H}$ but not in $\mathcal{C}$).

Given a subset of points of the instance space $Q \subseteq \mathcal{X}$, we will be interested in the set of all possible partitions of Q into positive and negative examples, such that there is a function $h \in \mathcal{H}$ and a function $c \in \mathcal{C}$ that are both consistent with this partition. This may be formulated as: $S_{\mathrm{CON}}(Q) = \{P \mid \mathrm{CON}(P, Q \setminus P) = \text{"YES"}\}$ where CON is a consistency algorithm for $(\mathcal{C}, \mathcal{H})$. The following is based on Sauer's Lemma [19]:

Lemma 1: For any set of points Q,

$$|S_{\mathrm{CON}}(Q)| \le |Q|^{\text{VC-dim}(\mathcal{H})}.$$

Bshouty [8] describes how to construct an efficient algorithm for generating this set of partitions (along with the corresponding functions $h \in \mathcal{H}$), assuming that $\mathcal{C}$ is PAC-learnable from $\mathcal{H}$ of constant VC dimension. The algorithm's output is denoted

$$\hat{S}_{\mathrm{CON}}(Q) \overset{\triangle}{=} \{((P, Q \setminus P), h) \mid P \in S_{\mathrm{CON}}(Q) \text{ and } h \text{ is consistent with } (P, Q \setminus P)\}.$$

5.2 An Efficient Algorithm for Learning with Nasty Noise

We now show a variation of the algorithm presented in [8] that can learn the class $\mathcal{C}^\star$ with a nasty sample adversary, assuming that the class $\mathcal{C}$ is PAC-learnable from a class $\mathcal{H}$ of constant VC dimension d. Our algorithm builds on the fact that a consistency algorithm CON for $(\mathcal{C}, \mathcal{H})$ can be constructed, given an algorithm that PAC learns $\mathcal{C}$ from $\mathcal{H}$ [8]. This algorithm can learn the concept class $\mathcal{C}^\star$ with any confidence parameter δ and with accuracy ϵ that is arbitrarily close to the lower bound of 2η. Its sample complexity and computational complexity are both polynomial in the size of the target, $1/\delta$ and $1/\Delta$, where $\Delta = \epsilon - 2\eta$.

Our algorithm is based on the following idea: Request a large sample from the oracle. Randomly pick a smaller sub-sample from the sample retrieved. By randomly picking this sub-sample, the algorithm neutralize some of the power the adversary has, since the adversary cannot know which examples are the ones that will be most "informative" for us. Then use the consistency algorithm for $(\mathcal{C}, \mathcal{H})$ to find one representative from $\mathcal{H}$ for any possible behavior on the smaller sub-sample. These hypotheses from $\mathcal{H}$ now define a division of the instance space into "cells", where each cell is characterized by a specific behavior of all the hypotheses picked. The final hypotheses is simply based on taking a majority vote among the complete sample inside each such cell.

To demonstrate the algorithm, let us consider (informally) the specific, relatively simple, case where the class to be learned is the class of k intervals[3] on the straight line (see Figure 1). The algorithm, given a sample as input, proceeds as follows:

1. The algorithm uses a relatively small, random sub-sample to divide the line into sub-intervals. Each two adjacent points in the sub-sample define such a sub-interval.
2. For each such sub-interval the algorithm calculates a majority vote on the complete sample. The result is our hypothesis.

The number of points (which in this specific case is the number of sub-intervals) that the algorithm chooses in the first step depends on k. Intuitively, we want the total weight of the sub-intervals containing the target's end-points to be relatively small (this is what is called the "bad part" in the formal analysis that follows). Naturally, there will be $2k$ such "bad" sub-intervals, so the larger k is, the larger the sub-sample needed. Except for these "bad" sub-intervals, all other subintervals on which the algorithm errs have to have at least half of their points modified by the adversary. Thus the total error will be roughly 2η, plus the weight of the "bad" sub-intervals.

Now, we proceed to a formal description of the learning algorithm. Given the constant d, the size k of the target function, the bound on the error rate η, the parameters δ and Δ, and two additional parameters M, N (to be specified below), the algorithm proceeds as follows:

Algorithm NastyLearn:

1. Request a sample S of size N.
2. Choose uniformly at random a sub-sample $R \subseteq S$ of size M.
3. Use the consistency algorithm for $(\mathcal{C}, \mathcal{H})$ to compute

$$\hat{S}_{\mathrm{CON}}(R) = \{((P_1, R \setminus P_1), h_1), \ldots, ((P_t, R \setminus P_t), h_t)\} .$$

[3]For simplicity, we consider the case where the concept class is of intervals that are semi-open, i.e. intervals of the form $[a, b)$. The use of other types of intervals would have required that we handle the boundary points between them separately.

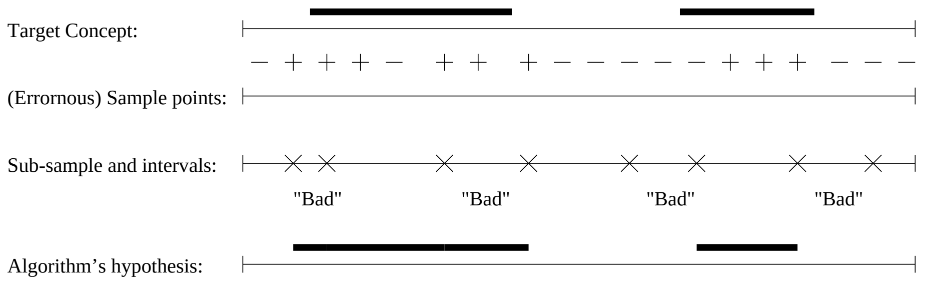

Figure 1: Example of NastyLearn for intervals.

4. Output the hypotheses $H(h_1,\ldots,h_t)$, computed as follows: For all $W_a \in \mathcal{W}(h_1,\ldots,h_t)$ such that $S \cap W_a$ is not empty, set $H(x)$ to be the majority of labels in $S \cap W_a$, for all $x \in W_a$. If $S \cap W_a = \emptyset$, set H to be 0 on any $x \in W_a$.

Theorem 8: Let

$$\begin{aligned} M &= \max\left(\frac{24k}{\Delta}\log\frac{8}{\delta}, \frac{c_2 dk}{\Delta}\log\frac{78k}{\Delta}\right) \\ N &= \frac{M^{c_1 d 2^d} k^2}{\Delta^2}\left(2^d d + \log\frac{4}{\delta}\right) \end{aligned}$$

where c_1 and c_2 are constants. Then, Algorithm NastyLearn learns the class $\mathcal{C}^\star$ with accuracy $\epsilon = 2\eta + \Delta$ and confidence δ in time polynomial in k, $\frac{1}{\delta}$, and $\frac{1}{\Delta}$.

As in Theorem 7, this Theorem refers to the modified PAC criterion that require the algorithm to output, with probability at least $1-\delta$, a function h for which:

$$\Pr_D[c_t(x) \neq h(x)] \leq \epsilon.$$

The same technique we mentioned for Algorithm NastyConsistent may be used to modify this algorithm to be a PAC learning algorithm in the sense of Definition 1.

Before commencing with the actual proof, we present a technical lemma:

Lemma 2: Assuming N is set as in the statement of Theorem 8, with probability at least $1-\frac{\delta}{4}$ the number of points in which errors are introduced, E, is at most $(\eta + \Delta/12)N$.

Proof of Lemma 2: Note that, by the definition of the model, E is distributed according to a binomial distribution with parameters η and N. Thus, E behaves as the number of successes in N independent Bernoulli experiments, and the Hoeffding inequality [14] may be used to bound its value:

$$\Pr\left[E > (\eta + \frac{\Delta}{12})N\right] \leq \exp\left(-\frac{2N\Delta^2}{144}\right).$$

Therefore, if we take $N > \frac{72}{\Delta^2}\ln(4/\delta)$ we have, with probability at least $1-\frac{\delta}{4}$, that E is at most $(\eta + \Delta/12)N$. Note that the value we have chosen for N in the statement of Theorem 8 is clearly large enough. $\square$

We are now ready to present the proof of Theorem 8:

Proof: To analyze the error made by the hypothesis that the algorithm generates, let us denote the adversary's strategy as follows:

1. Generate a sample of the requested size N according to the distribution D, and label it by the target concept F. Denote this sample by S_g.

2. Choose a subset $S_{\text{out}} \subseteq S_g$ of size $E = E(S_g)$, where $E(S_g)$ is a random variable (as defined in Section 2.2).

3. Choose (maliciously) some other set of points $S_{\text{in}} \subseteq \mathcal{X} \times \{0,1\}$ of size E.

4. Hand to the learning algorithm the sample $S = (S_g \setminus S_{\text{out}}) \cup S_{\text{in}}$.

Assume the target function F is of the form $F = f(g_1, \ldots, g_k)$. For all $i \in \{1, \ldots, k\}$, denote by h_{j_i}, where $j_i \in \{1, \ldots, t\}$, the hypothesis chosen in step 3 of the algorithm that exhibits the same behavior g_i has over the points of R (from the definition of $\hat{S}_{\text{CON}}$ we are guaranteed that such a hypothesis exists). By definition, there are no points from R in $h_{j_i} \triangle g_i$, so:

$$R \cap (h_{j_i} \triangle g_i) = \emptyset. \tag{4}$$

As the VC-dimension of both the class $\mathcal{C}$ of all g_i's and the class $\mathcal{H}$ of all h_i's is d, the class of all their possible symmetric differences also has VC-dimension $O(d)$ (see Section 2.3). By applying Theorem 1, when viewing R as a sample taken from S according to the uniform distribution, and by choosing M to be as in the statement of the theorem, R will be an α-net (with respect to the uniform distribution over S) for the class of symmetric differences, with $\alpha = \Delta/6k$, with probability at least $1 - \delta/4$. Note that there may still be points in S which are in $h_{j_i} \triangle g_i$. Hence, we let $S^{(i)} = S \cap (h_{j_i} \triangle g_i)$. Now, by using (4) we get:

$$\frac{\left|S^{(i)}\right|}{|S|} \leq \frac{\Delta}{6k} \tag{5}$$

with probability at least $1 - \delta/4$, simultaneously for all i.

For every sub-domain $B \in \mathcal{W}(h_1, \ldots, h_t)$ we define:

$$\begin{aligned}
N_B &\triangleq |S_g \cap B| \\
N_B^{\text{in}} &\triangleq |S_{\text{in}} \cap B| \\
N_B^{\text{out,g}} &\triangleq \left| S_{\text{out}} \cap B \cap \overline{\bigcup_i (h_{j_i} \triangle g_i)} \right| \\
N_B^{\text{out,b}} &\triangleq \left| S_{\text{out}} \cap B \cap \bigcup_i (h_{j_i} \triangle g_i) \right| \\
N_B^{\alpha} &\triangleq \left| S \cap B \cap \bigcup_i (h_{j_i} \triangle g_i) \right|
\end{aligned}$$

In words, N_B and N_B^{in} simply stand for the size of the restriction of the original (noise-free) sample S_g and the noisy examples S_{in} introduced by the adversary to the sub-domain B. As for the rest of the definitions, they are based on the distinction between the "good" part of B, where the g_is and the h_{j_i}s behave the same, and the "bad" part, which is present due to the fact that the g_is and the h_{j_i}s exhibit the same behavior only on the smaller sub-sample R, rather than on the complete sample S. Note that the target function is constant on the good part of every sub-domain B. We use N_B^α to denote the number of sample points in the bad part of B, and $N_B^{\text{out,g}}$ and $N_B^{\text{out,b}}$ to denote the number of sample points that were removed by the adversary from the good and bad parts of B, respectively.

Since our learning algorithm decides on the classification in each sub-domain by a majority vote, the hypothesis will err on the domain $B \cap (\overline{\bigcup_i (h_{j_i} \triangle g_i)})$ (the "good" part of B) if the number of examples left untouched in the good part of B is less than the number of examples in B that were modified by the adversary, plus those that were misclassified by the h_{j_i}s (with respect to the g_is). This may be formulated as the following condition: $N_B^{\text{in}} + N_B^\alpha \geq N_B - N_B^{\text{out,g}} - N_B^\alpha$, or: $N_B \leq N_B^{\text{in}} + N_B^{\text{out,g}} + 2N_B^\alpha$. Therefore, the total error the algorithm may experience is at most:

$$D\left[\bigcup_i h_{j_i} \triangle g_i\right] + \sum_{B:\ N_B \leq N_B^{\text{in}} + N_B^{\text{out,g}} + 2N_B^\alpha} D(B).$$

We now calculate a bound for each of the two terms above separately. To bound the second term, note that by Theorem 2 our choice of N guarantees S_g to be a $\frac{\Delta}{6|\mathcal{W}(h_1,\ldots,h_t)|}$-sample for our domain with probability at least $1 - \delta/4$. Note that from the definition of $\mathcal{W}(h_1, \ldots, h_t)$ and from the Sauer Lemma [19] (applied to the dual class $\mathcal{H}^\perp$) we have that $|\mathcal{W}(h_1, \ldots, h_t)| \leq t^{\text{VCdim}(\mathcal{H}^\perp)}$, which, with Claim 1 and Lemma 1 yields:

$$|\mathcal{W}(h_1, \ldots, h_t)| \leq M^{\text{VCdim}(\mathcal{H})\text{VCdim}(\mathcal{H}^\perp)} \leq M^{d2^d}$$

Our choice of N indeed guarantees, with probability at least $1 - \delta/4$ that:

$$\begin{aligned} \sum_{B:\ N_B \leq N_B^{\text{in}} + N_B^{\text{out,g}} + 2N_B^\alpha} D(B) &\leq \sum_{B:\ N_B \leq N_B^{\text{in}} + N_B^{\text{out,g}} + 2N_B^\alpha} \left(\frac{N_B}{N} + \frac{\Delta}{6\,|\mathcal{W}(h_1, \ldots, h_t)|}\right) \\ &\leq \frac{\Delta}{6} + \sum_{B \in \mathcal{W}(h_1,\ldots,h_t)} \frac{N_B^{\text{in}} + N_B^{\text{out,g}} + 2N_B^\alpha}{N} \end{aligned}$$

From the above choice of N, it follows that S_g is also a $\frac{\Delta}{6k}$-sample for the class of symmetric differences of the form $h_{j_i} \triangle g_i$, with probability at least $1 - \delta/4$. Furthermore, this implies that S_g is a $\frac{\Delta}{6}$-sample for the class of unions of k such symmetric differences (see, e.g., [4]), with probabilit at least $1 - \delta/4$. Thus, with probability at least $1 - \delta/2$, we have:

$$\begin{aligned} D\left(\bigcup_i (h_{j_i} \triangle g_i)\right) &\leq \frac{|S_g \cap (\bigcup_i (h_{j_i} \triangle g_i))|}{|S_g|} \\ &\leq \frac{\Delta}{6} + \frac{|(S \cup S_{\text{out}}) \cap \bigcup_i (h_{j_i} \triangle g_i)|}{|S|} && \text{(Because } S_g \text{ is a } \tfrac{\Delta}{6} \text{ sample)} \\ &\leq \frac{\Delta}{6} + \frac{\Delta}{6} + \frac{|S_{\text{out}} \cap \bigcup_i (h_{j_i} \triangle g_i)|}{|S|} && \text{(Because (5) holds, w.h.p.)} \\ &\leq \frac{2\Delta}{6} + \sum_{B \in \mathcal{W}(h_1,\ldots,h_t)} \frac{N_B^{\text{out,b}}}{N} \end{aligned}$$

The total error made by the hypothesis (assuming that none of the four bad events happen) is therefore bounded by:

$$\Pr_D[H \triangle F] \leq \frac{3\Delta}{6} + \sum_{B \in \mathcal{W}(h_1,\ldots,h_t)} \frac{N_B^{\text{in}} + N_B^{\text{out,g}} + N_B^{\text{out,b}} + 2N_B^\alpha}{N}$$

$$
\begin{aligned}
&\leq \frac{5\Delta}{6} + \frac{|S_{\text{out}}|}{N} + \frac{|S_{\text{in}}|}{N} && \text{(From (5), } \textstyle\sum N_B^\alpha \leq N/6\text{)} \\
&= 2\frac{E}{N} + \frac{5\Delta}{6} \\
&\leq 2\eta + \Delta = \epsilon \qquad \text{(Using Lemma 2.)}
\end{aligned}
$$

as required. This bound holds with certainty at least $1-\delta$. □

6 Conclusion

We have presented the model of PAC learning with nasty noise, generalizing on previous models. We have proved a negative information-theoretic result, showing that there is no learning algorithm that can learn any non-trivial class with accuracy better than 2η, paired with a positive result showing this bound to be tight. We complemented these results with lower bounds on the sample size required for learning with accuracy $2\eta + \Delta$. We have also shown that for a wide variety of "interesting" concept classes, an efficient learning algorithm in this model exists. Our negative result can be generalized for the case where the learning algorithm uses randomized hypotheses, or coin rules (as defined in [10]); in such a case we get an information theoretic lower bound of η for the achievable accuracy, compared to a lower bound of $\eta/2(1+\eta)$ proved in [10] for learning with Malicious Noise of rate η.

While the partition into two separate variants: the NSN and the NCN models seem intuitive and well-motivated, it remains an open problem to come up with any results that actually separate the two models. Both the negative and positive results we presented in this work apply equally to both the NSN and the NCN models.

Finally, note that the definition of the nasty noise model requires the learning algorithm to know in advance the sample size m (or an upper bound on it). The model however can be extended so as to deal with scenarios where no such bound is known to the learning algorithm. There are several scenarios of this kind. For example, the sample complexity may depend on certain parameters (such as the "size" of the target function) which are not known to the algorithm[4]. The adversary, who knows the learning algorithm and knows the target function (and in general can know all the parameters hidden from the learning algorithm) can thus "plan ahead" and draw in advance a sample S_g of size which is sufficiently large to satisfy, with high probability, all the requests the learning algorithm will make. It then modifies S_g as defined above and reorders the resulted sample S randomly[5]. Now, the learning algorithm simply asks for one example at a time (as in the PAC model) and the adversary supplies the next example in its (randomly-ordered) set S. If the sample is exhausted (which may happen in those cases where we have only an expected sample-complexity guarantee), we say that the learning algorithm has failed; however, when using a large enough sample (with respect to $1/\delta$), this will happen with sufficiently small probability.

References

[1] J. A. Aslam and S. E. Decatur, "General Bounds on Statistical Query Learning and PAC Learning with Noise via Hypothesis Boosting", *FOCS93*, pp. 282-291, 1993.

[4]Other examples include scenarios where the learning algorithm only guarantees a bound on the *expected* sample complexity (either because the algorithm is randomized or because the number of examples used depends on the previous examples).

[5]The sample must be reordered randomly, so that the adversary cannot introduce an effective error rate much greater than η in case the learning algorithm does not ask for all the examples drawn.

[2] D. Angluin and P. Laird, "Learning from Noisy Examples", Machine Learning, Vol. 2, pp. 343–370, 1988.

[3] S. Ben-David, N. H. Bshouty and E. Kushilevitz, "A Composition Theorem for Learning Algorithms with Applications to Geometric Concept Classes", *STOC97*, pp. 324–333, 1997.

[4] S. Ben-David and A. Litman, "Combinatorial Variability of Vapnik–Chervonenkis Classes with Applications to Sample Compression Schemes", *Discrete Applied Math* 86, pp. 3–25, 1998.

[5] A. Blum, P. Chalasani, S. A. Goldman, and D. K. Slonim, "Learning with Unreliable Boundary Queries", *COLT95*, pp. 98–107, 1995.

[6] A. Blum, M. Furst, J. Jackson, M. J. Kearns, Y. Mansour, and S. Rudich, "Weakly Learning DNF and Characterizing Statistical Query Learning Using Fourier Analysis", *STOC94*, 1994.

[7] A. Blumer, A. Ehrenfeucht, D. Haussler and M. K. Warmuth, "Learnability and the Vapnik-Chervonenkis dimension", *J. of the ACM*, 36(4), pp. 929–965, 1989.

[8] N. H. Bshouty, "A New Composition Theorem for Learning Algorithms", *STOC98*, pp. 583–589, 1998.

[9] N. H. Bshouty, S. A. Goldman, H. D. Mathias, S. Suri, and H. Tamaki, "Noise-Tolerant Distribution-Free Learning of General Geometric Concepts", *STOC96*, pp. 151–160, 1996.

[10] N. Cesa-Bianchi, E. Dichterman, P. Fischer, E. Shamir and H. U. Simon, "Sample-efficient strategies for learning in the presence of noise", *J. of the ACM*, 46(5), pp. 684–719, 1999.

[11] S. E. Decatur, "Learning in Hybrid Noise Environments Using Statistical Queries", in *Learning from Data: Artificial Intelligence and Statistics V*, D. Fisher and H. J. Lenz, (Eds.), 1996.

[12] S. E. Decatur, "PAC Learning with Constant-Partition Classification Noise and Applications to Decision Tree Induction", *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics*, pp. 147–156, 1997.

[13] S. E. Decatur and R. Gennaro, "On Learning from Noisy and Incomplete Examples", *COLT95*, pp. 353–360, 1995.

[14] W. Hoeffding, "Probability Inequalities for Sums of Bounded Random Variables", J. of the American Statistical Association, 58(301), pp. 13–30, 1963.

[15] K. Jogdeo and S. M. Samuels, "Monotone Convergence of Binomial Probabilities and a Generalization of Ramanujan's Equation". *The Annals of Mathematical Statistics*, 39(4):1191–1195, 1968.

[16] M. Kearns, "Efficient Noise-Tolerant Learning from Statistical Queries", Proc. 25th ACM Symposium on the Theory of Computing, pp. 392–401, 1993.

[17] M. J. Kearns and M. Li, "Learning in the Presence of Malicious Errors", SIAM J. on Computing, 22:807-837, 1993.

[18] M. J. Kearns, R. E. Schapire, and L. M. Sellie, "Toward Efficient Agnostic Learning", *Machine Learning*, vol. 17(2), pp. 115–142, 1994.

[19] N. Sauer, "On the Density of Families of sets", *Journal of Combinatorial Theory*, 13:145–147, 1972.

[20] R. E. Schapire, "The Design and Analysis of Efficient Learning Algorithms", MIT Press, 1991.

[21] M. Talagrand, "Sharper Bounds for Gaussian and Empirical Processes", *Annals of Probability*, 22:28–76, 1994.

[22] L. G. Valiant, "A Theory of the Learnable", Comm. ACM, 27(11):1134-1142, 1984.

[23] L. G. Valiant, "Learning Disjunctions of Conjunctions", *IJCAI85*, pp. 560–566, 1985.

[24] V. N. Vapnik and A. Y. Chervonenkis, "On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities", Theory Probab. Appl. 16, no. 2 pp. 264–280, 1971.