Four types of noise in data for PAC learning

Robert H. Sloan

Department of Electrical Engineering & Computer Science (M/C 154), 851 S. Morgan St, Rm 1120 SEO,
University of Illinois at Chicago, Chicago, IL 60607-7053, USA

Communicated by L.A. Hemaspaandra; received 4 June 1994; revised 16 December 1994

Keywords: Concept learning; Design of algorithms; Noise; PAC learning; Computational learning theory

1. Introduction

In order to be useful in practice, machine learning algorithms must tolerate noisy inputs. In this paper we compare and contrast the effects of four different types of noise on learning in Valiant's PAC (probably approximately correct), or distribution-free, model of learning [11].

Two previously studied models, malicious noise [12] and random classification noise [1], represent the extremes. Malicious noise is intended to model the worst possible sort of noise, and in general only a very small amount of it can be tolerated [7]. On the other hand, Angluin and Laird [1] have shown that for random misclassification noise - instances are never altered but their labels are reversed with probability \( \nu \) - PAC learning can be achieved for any \( \nu < 1/2 \). They further show that any algorithm that chooses as its output concept some concept that minimizes disagreements with a polynomial size set of examples meets this bound.

Here we extend Angluin and Laird's result to malicious misclassification noise - the noisy label is chosen adversarially instead of randomly. We also show that if one considers only algorithms that work by minimizing disagreements, then random attribute noise is nearly as harmful as malicious noise.

2. Definitions and notation

2.1. PAC, learning of concepts

In this subsection, we briefly define the PAC learning model [11]. Haussler et al. [4] provide an excellent discussion of the subtleties and technical details of the definitions.

Formally, a concept \( c \) is a subset of a set called the domain or instance space. For convenience, we refer to concepts interchangeably as sets of instances and as \( \{0, 1\} \)-functions on the domain. We assume a fixed but unknown probability distribution \( \mathcal{P} \) on the domain.

The length of concept \( c \), denoted \( |c| \), is the number of bits it takes to write down \( c \) in some agreed-upon encoding scheme. The length of an instance is defined similarly.

The goal of a learner is to infer an unknown target concept from a given fixed set of concepts called the concept class. We usually parameterize our problems, and write \( C = \{X_n, C_n\}_{n \geq 1} \), for concept class \( C = \bigcup_{n \geq 1} C_n \) on domain \( X = \bigcup_{n \geq 1} X_n \). Each \( C_n \subset 2^X \), and each instance in \( X_n \) and concept in \( C_n \) has length polynomial in \( n \). A typical case is boolean function...
PAC learning is a model of learning from examples, and the learner is provided with an oracle \( \text{Ex} \) such that each call to \( \text{Ex} \) returns a labeled instance, or example, \((x, c(x))\), where \( x \) is drawn randomly from the domain according to \( P \), and \( c \) is the target concept.

**Definition 1.** A concept class \( \mathcal{C} = \{X_n, C_n\}_{n \geq 1} \) is PAC learnable by class \( H \) if there exists an algorithm \( A \) such that for every \( n \geq 1 \), for every probability distribution \( P \) on \( X_n \) and every \( c \in C_n \), when given access to the \( \text{Ex} \) oracle for \( P \) and \( c \), and inputs \( 0 < \varepsilon, \delta \leq 1 \) and \( n \), Algorithm \( A \) makes at most a polynomial number of calls to \( \text{Ex} \) and outputs a hypothesis \( h \in H \) such that

\[
\Pr[P(c \Delta h) > \varepsilon] \leq \delta,
\]

where the probability is over the calls to \( \text{Ex} \) and any coin flips used by the learning algorithm, and \( \Delta \) denotes symmetric difference.

In this setting "polynomial number" means polynomial in \( 1/\varepsilon, 1/\delta, \) and \( n \). If, additionally, Algorithm \( A \) has polynomial running time, then we say that \( \mathcal{C} \) is polynomially PAC learnable.

The class \( H \subset 2^X \) is called the hypothesis class.

### 2.2. Noise

The definition of PAC learning (from noiseless data) assumes that the oracle \( \text{Ex} \) returns correct data. We will now introduce other oracles which we will use to model the case where the labeled examples given to our learner are somehow corrupted by noise.

To obtain the malicious noisy model we will use the malicious error oracle, \( \text{MAL}_\nu \) [12]. To obtain the most benign error model, the random misclassification model, we will use the random misclassification oracle, \( \text{RMC}_\nu \) [1]. We also introduce two new oracles, the malicious misclassification oracle, \( \text{MMC}_\nu \), and the random attribute error oracle, \( \text{RAT}_\nu \). Each of these four oracles is defined with respect to a fixed target concept and a fixed probability distribution \( P \) on the domain.

Each of these oracles represents some sort of noisy version of \( \text{Ex} \). The "desired," noiseless, output of these oracles is a correctly labeled example \((x, c(x))\), where \( x \) is drawn according to \( P \). The actual outputs are as follows:

- With probability \( 1 - \nu \), oracle \( \text{MAL}_\nu \) returns a correctly labeled example \((x, c(x))\) where \( x \) is drawn according to \( P \). With probability \( \nu \), oracle \( \text{MAL}_\nu \) returns an example about which no assumptions whatsoever may be made. In particular, this example may be maliciously selected by an adversary that has arbitrary computing power, and that knows \( c, P, \nu \), and the internal state of the algorithm calling this oracle. This oracle models situations where the learner usually gets a correct example, but some fraction \( \nu \) of the time the learner gets noisy examples, and the nature of the noise is unknown or unpredictable.
- When oracle \( \text{RMC}_\nu \) is called, it calls \( \text{Ex} \) to obtain some (noiseless) \((x, c(x))\), and, with probability \( 1 - \nu \), it returns \((x, c(x))\). With probability \( \nu \), oracle \( \text{RMC}_\nu \) returns \((x, 1)\) (i.e., \( x \) with the wrong label). In this model, the only source of noise is random misclassification.
- When oracle \( \text{MMC}_\nu \) is called, it calls \( \text{Ex} \) to obtain some (noiseless) \((x, c(x))\), and, with probability \( 1 - \nu \), it returns \((x, c(x))\). With probability \( \nu \), oracle \( \text{MMC}_\nu \) returns \((x, 1)\) where \( 1 \) is a label about which no assumptions whatsoever may be made. As with \( \text{MAL}_\nu \), we assume an omnipotent, omniscient adversary, but now the adversary only gets to choose the label of the example. This oracle models situations where the only source of noise is misclassification, but the nature of the misclassification is unknown or unpredictable.
- We consider the oracle \( \text{RAT}_\nu \) only when learning boolean functions. This oracle calls \( \text{Ex} \) and obtains some \((x, c(x))\). Oracle \( \text{RAT}_\nu \) then adds noise to this example by independently flipping each bit \( x_i \) of instance \( x \) to \( \bar{x}_i \) with probability \( \nu \) for \( 1 \leq i \leq n \). The oracle returns the altered instance and the original label \( c(x) \). This oracle models situations where the attributes of the examples are subject to noise, but this noise is rather benign.

**Definition 2.** We say that algorithm \( A \) PAC learns \( \mathcal{C} = \{X_n, C_n\}_{n \geq 1} \) from noisy examples of type \( O_\nu \) if and only if algorithm \( A \), given \( \varepsilon, \delta, n \), an upper bound \( \nu_b \) on \( \nu \), and access to oracle \( O_\nu \) for \( c \in C_n \), makes at most a polynomial number (in \( 1/\varepsilon, 1/\delta, 1/\nu_b \), and \( n \)) of calls to oracle \( O_\nu \), and otherwise meets the definition of PAC learning \( \mathcal{C} \) by \( H \) with the oracle \( \text{Ex} \) replaced by oracle \( O_\nu \).

Again, we say that Algorithm \( A \) polynomially PAC
learns if its running time is also polynomially bounded.

Notice that this definition requires only that the algorithm be given a bound on the noise rate. In fact, the real source of examples for the algorithm may be $O_\nu$ for any $0 \leq \nu \leq \nu_b$.

**Definition 3.** The optimal error rate for concept class $C$ given errors of type $O$, $E(C, O)$ is the largest $\nu$ such that there is an algorithm that PAC learns $C$ from noisy examples of type $O_\nu$.

### 2.3. Minimizing disagreements

In some cases we will consider only those learning algorithms that minimize disagreements. Fix a hypothesis class $H$. Let $S$ be a set of (noisy) examples of some concept. For $h \in H$, the disagreement number of $h$ on $S$, $d_h$, is the number of labeled instances $(x, s)$ in $S$ for which $h(x) \neq s$. The disagreement rate of $h$ is $d_h/|S|$. A learning algorithm that minimizes disagreements must output a concept from its hypothesis class that has a minimal disagreement rate on the set of all examples the algorithm drew from its oracle. (The interest in such algorithms is seen in Theorems 7 and 9 below.)

**Definition 4.** The optimal error rate for algorithms that work by minimizing disagreements on concept class $C$ given errors of type $O$, $EMD(C, O)$, is defined to be the largest $\nu$ such that there is an algorithm that PAC learns $C$ from noisy examples of type $O_\nu$ by minimizing disagreements.

The optimal polynomial time error rate for concept class $C$ given errors of type $O$, $E^p(C, O)$, is the largest $\nu$ such that there exists an algorithm that PAC learns $C$ from noisy examples of type $O_\nu$.

### 3. Attribute noise results

A concept class $C$ is distinct if there are concepts $c_1, c_2 \in C$ and instances $u$ and $v$ in the domain of $C$ satisfying $u, v \in c_1, u \not\in c_2$, and $v \in c_2$ [7]. Restricting our attention to the domain $\{0, 1\}^n$, we say $C = \{\{0, 1\}^n, C_n\}_{n \geq 1}$ is instance distinct if for all sufficiently large $n$ there are concepts $c_1, c_2 \in C_n$ and an instance $u$ in $\{0, 1\}^n$ satisfying $c_1(u) \neq c_2(u)$ and for all $v \neq u$, $c_1(v) = c_2(v)$. Thus for a concept class to be instance distinct, all that is required is the existence of two concepts which agree on every instance in the domain save one. If we can always satisfy the definition of instance distinct with a $u$ and $v$ that have Hamming distance one, then we say $C$ is strongly instance distinct. Most common boolean concept classes, including $k$DNF, DNF, CNF, etc., are in fact strongly instance distinct.

For malicious noise, for any distinct concept class $C$, Kearns and Li showed that $E(C, \text{MAL}) < \varepsilon/(1 + \varepsilon)$, where $\varepsilon$ is the accuracy parameter [7]. This bound holds for both the one-oracle and two-oracle models of PAC learning [4], but the definition of distinct is slightly different for the two-oracle model.

For instance-distinct concept classes, restricting our attention to algorithms that work by minimizing disagreements, we get only slightly weaker results when we substitute the much less extreme oracle $\text{RAT}_\nu$ for $\text{MAL}_\nu$:

**Theorem 5.** For instance-distinct concept class $C$ and accuracy parameter $\varepsilon$, $EMD(C, \text{RAT}_\nu) < \varepsilon$.

**Proof.** We use the technique of induced distributions [7].

Fix $n$, and let $c_1$ and $c_2$ be the two concepts that cause $C$ to be instance distinct. Let $u$ be the instance in the domain on which $c_1$ and $c_2$ differ, and let $v$ be any instance in the domain that differs from $u$ in only one bit position. Now assume our algorithm is trying to PAC learn to accuracy $\varepsilon$, and fix the probability distribution $P$ on the domain that assigns probability $\varepsilon$ to $u$ and $1 - \varepsilon$ to $v$.

Say, without loss of generality, that $c_1(u) = c_1(v) = c_2(u) = 1$, and $c_2(u) = 0$. In the absence of noise, our algorithm sees the following distribution on examples from the oracle:

<table>
<thead>
<tr>
<th>Concept</th>
<th>$(u, 1)$</th>
<th>$(u, 0)$</th>
<th>$(v, 1)$</th>
<th>$(v, 0)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_1$</td>
<td>$\varepsilon$</td>
<td>$0$</td>
<td>$1 - \varepsilon$</td>
<td>$0$</td>
</tr>
<tr>
<td>$c_2$</td>
<td>$0$</td>
<td>$\varepsilon$</td>
<td>$1 - \varepsilon$</td>
<td>$0$</td>
</tr>
</tbody>
</table>

If, however, the examples come from the noise oracle $\text{RAT}_\nu$, then the examples will have the distribution (ignoring instances other than $u$ and $v$) shown in Fig. 1(a) where $\tilde{\nu} = 1 - \nu$. 

When $c_2$ is the correct concept, it must be that the "observed mistake rate" of $c_2$ is strictly less than the observed mistake rate of $c_1$. Since $c_1$ and $c_2$ classify all instances other than $u$ identically, this amounts to:

$$\text{(1 - \epsilon)}v(1 - \nu)^{n-1} < \epsilon(1 - \nu)^n$$

which simplifies to $\nu < \epsilon$ as desired. \(\square\)

We can define an attribute noise oracle that is gentler than $\text{RAT}_\nu$, and obtain a somewhat weaker bound. Oracle $\text{RAT}_\nu'$ will obtain a correct example $(x_1 \cdots x_n; s)$ from $\text{Ex}$, and with probability $1 - \nu$ output that example unaltered. With probability $\nu$, oracle $\text{RAT}_\nu'$ instead picks some $1 \leq i \leq n$ uniformly at random, and returns $(x_1x_2\cdots \hat{x}_i\cdots x_{n-1}x_n; s)$ (i.e., the correct example with exactly one of its attribute bits flipped).

**Theorem 6.** For strongly instance-distinct concept class $C$ and accuracy parameter $\epsilon$, $\text{EMD}(C, \text{RAT}_\nu') < \frac{n\epsilon}{1 + (n-1)\epsilon}$.

**Proof (sketch).** The proof is very similar to the proof of Theorem 5. This time we pick $u$ and $v$ to have Hamming distance one. Now the table showing the probability distribution for examples from $\text{RAT}_\nu'$ will be as in Fig. 1(b).

Now Eq. (1) becomes $(1 - \epsilon)\nu/n < \epsilon(1 - \nu)$, which simplifies to $\nu < \epsilon$ as desired. \(\square\)

**Remark.** It is easy to convert Theorems 5 and 6 from the one-oracle model of PAC learning to the two-oracle model of PAC learning by using the techniques of Haussler et al. \[4\]. The same bounds are obtained.

Theorems 5 and 6 are somewhat surprising, especially since we will see below that much larger amounts of malicious classification noise can be tolerated. These results are sharply different from those of Quinlan \[9\], who found empirically that attribute noise was less harmful than classification noise.

Note that the above results depend on restricting attention to algorithms that work by minimizing disagreements. The simple concept class of monomials is PAC learnable from examples from $\text{RAT}_\nu$ for $\nu < 1/2$ \[3\], and so is $k$DNF if the learning algorithm is given the exact noise rate instead of merely a bound on it \[10\].

### 3.1 Misclassification noise

For noisy examples generated by $\text{RMC}_\nu$, Angluin and Laird have shown that PAC learning on discrete domains is possible whenever $\nu < 1/2$ \[1\]. In particular, they showed that a modified version of the "Occam's razor" algorithm \[2\] will work. (See Fig. 2.)

There are two key modifications. First, the sample size must depend on some bound $\nu_b$ on the noise rate $\nu$ in addition to the usual parameters. Also, we cannot simply return any concept $c$ that agrees with all of the labels of the instances in the sample. Since those labels are noisy, there may not be any such $c$. Instead, we return a concept with minimal disagreement rate.

The following theorem shows that this algorithm indeed PAC learns:

**Algorithm** Simple-Occam

(PAC learns concept class $C$.)

- **Inputs:** $\epsilon$, $\delta$, and access to $\text{Ex}$.
- **Set** $m := \epsilon^{-1} \ln(|C|/\delta)$.
- **Obtain** a set of $m$ labeled examples from $\text{Ex}$.
- **Output** any $c \in C$ consistent with the set of examples.

![Fig. 2. Simple Occam algorithm.](image-url)
Theorem 7 (Angluin and Laird). Let $C$ be any finite concept class. If we draw a sample of size
\begin{equation}
m = \frac{2}{\varepsilon^2(1 - 2\nu_b)^2} \ln \left( \frac{|C| + 1}{\delta} \right)
\end{equation}
from $RMC_\nu$ for any $\nu \leq \nu_b < 1/2$, and find any hypothesis $h$ with a minimal disagreement number, then
\[\Pr[h \text{ is an } \varepsilon\text{-approximation of the target concept}] \geq 1 - \delta.\]

Remark. Laird [8] has shown that in fact $m = O(1/\varepsilon)$ rather than $m = O(1/\varepsilon)^2$ is sufficient.

We will give the proof of Theorem 7 here, since its extension to Theorem 9 is most easily proved by an additional argument added to the proof of Theorem 7. First, however, we state a lemma from probability theory that we will need for this proof.

Lemma 8 (Hoeffding's Inequality [5]). Let $S = \sum_1^n X_i$, where $X_1, \ldots, X_n$ are independent 0-1 random variables each with probability $p$ of being 1. For any positive constant $t$,
\[\Pr[S \geq pm + tm] \leq e^{-2tm^2}.\]

Proof of Theorem 7. Our examples have random misclassification noise with noise rate $\nu$. (The rate $\nu$ is fixed, but all the learning algorithm knows is that the noise rate is between 0 and $\nu_b$.) Consider a concept $c$ which disagrees with the target concept $c_t$ on probability weight $p$ of instances in the domain. The expected disagreement rate of $c$ with a sample from $RMC_\nu$ is
\begin{equation}
\text{E}[\text{disagreement rate}] = (1 - \nu)p + \nu(1 - p).
\end{equation}

For the target concept the expected disagreement rate is simply $\nu$. Let us call a concept $c$ $\varepsilon$-good if $\text{P}(c \triangle c_t) < \varepsilon$, where $c_t$ is the target concept, and $\varepsilon$-bad otherwise. For any $\varepsilon$-bad concept, the expected disagreement rate is at least $\nu + \varepsilon(1 - 2\nu)$. The gap between those two rates, which we will denote by $g$ is
\[g = \varepsilon(1 - 2\nu) \geq \varepsilon(1 - 2\nu_b).\]

As long as the measured disagreement rate of the target concept is less than its expectation plus $g/2$, and the measured disagreement rate of every $\varepsilon$-bad concept is greater than its expectation minus $g/2$, then
\begin{equation}
e^{-2(\varepsilon/2)^2m} \leq \delta/(|C| + 1).
\end{equation}

Similarly, the probability that any one particular $\varepsilon$-bad rule has a disagreement rate as much as $g/2$ less than its expectation is at most $\delta/(|C| + 1)$, so the probability that any $\varepsilon$-bad concept does so is at most $|C|\delta/(|C| + 1)$. Thus, as desired, the probability of outputting an $\varepsilon$-bad concept is at most $\delta$. \Box

In fact, malicious misclassification is no more harmful:

Theorem 9. Theorem 7 holds with $RMC_\nu$ replaced by $MMC_\nu$.

Proof. The argument is similar to the proof of Theorem 7, except that now we must consider what happens if our examples are maliciously misclassified.

The difference between $RMC_\nu$ and $MMC_\nu$ is that in $MMC_\nu$, the fraction $\nu$ of the time that "the coin toss comes up heads", the adversarial oracle $MMC_\nu$ may choose not to misclassify the example. The oracle may thus make some incorrect concept appear better to the learner than it would if the noise were purely random.

Let $E$ be the event that the disagreement number of the target concept is less than the disagreement number of any $\varepsilon$-bad concept.

In the proof of Theorem 7, the way we showed that we get an $\varepsilon$-good concept with probability at least $1 - \delta$ in the case of $RMC_\nu$ was to show that $\Pr[E] \geq 1 - \delta$. If event $E$ occurs when the examples are randomly misclassified, then it will still occur even if the oracle chooses not to misclassify some (or all) of the examples misclassified by $RMC_\nu$. Not mislabeling may cause the disagreement number of some $\varepsilon$-bad concepts to decrease by 1 (or to increase by 1), but it always causes the disagreement number of the target concept to decrease by 1, so if event $E$ occurs with mislabeling, then it still occurs without mislabeling. \Box

In fact, this reasoning still holds even if the oracle can change the particular instances (but not the num-
ber of instances) that get bad labels. That is to say, if the sample returned by RMC, caused event $E$ to occur, and that sample contained the two labeled instances $(x_1, \text{true label of } x_1)$ and $(x_2, \text{wrong label for } x_2)$, then if those two labeled instances were replaced in the sample by $(x_1, \text{wrong label for } x_1)$ and $(x_2, \text{true label of } x_2)$, then event $E$ would still occur.

Thus we also have:

**Corollary 10.** Theorem 9 still holds for PAC learning even if MMC, is replaced by a noise model where first the entire sample is drawn from $EX$ and then the adversary is allowed to pick any subset of the sample up to a fraction $\nu$ of the total sample to mislabel.

This noise model is the label-noise-only analogue of the "malicious burst noise" model briefly alluded to at the end of Kearns and Li's article [7], and can model much worse label noise than the "variable classification noise" model [6]. For instance, the model of Corollary 10 can accurately model situations where "borderline" instances from the domain are misclassified much more often than "obvious" instances.

**References**


