Reducing Adversarially Robust Learning to Non-Robust PAC Learning

Omar Montasser
omar@ttic.edu

Steve Hanneke
steve.hanneke@gmail.com

Nathan Srebro
nati@ttic.edu

Toyota Technological Institute at Chicago

Abstract

We study the problem of reducing adversarially robust learning to standard PAC learning, i.e. the complexity of learning adversarially robust predictors using access to only a black-box non-robust learner. We give a reduction that can robustly learn any hypothesis class $\mathcal{C}$ using any non-robust learner $\mathcal{A}$ for $\mathcal{C}$. The number of calls to $\mathcal{A}$ depends logarithmically on the number of allowed adversarial perturbations per example, and we give a lower bound showing this is unavoidable.

1 Introduction

We consider the problem of learning predictors that are *robust* to adversarial examples at test time. That is, we would like to be robust against an adversary $\mathcal{U} : \mathcal{X} \to 2^{\mathcal{X}}$ that can perturb examples at test-time, where $\mathcal{U}(x) \subseteq \mathcal{X}$ is the set of allowed corruptions the adversary might replace x with, as measured by the *robust risk*:

$$\mathrm{R}_{\mathcal{U}}(\hat{h}; \mathcal{D}) \triangleq \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \left[\sup_{z \in \mathcal{U}(x)} \mathbb{1}[\hat{h}(z) \neq y] \right]. \tag{1}$$

For example, $\mathcal{U}$ could be perturbations of bounded ℓ_p-norms [Goodfellow et al., 2015].

We ask whether we can adversarialy robustly learn a given target hypothesis class $\mathcal{C} \subseteq \mathcal{Y}^{\mathcal{X}}$ (e.g. neural networks)—that is, whether, if there exists a predictor in $\mathcal{C}$ with zero robust risk w.r.t. some unknown distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, can we find a predictor with (arbitrarily small) robust risk using m i.i.d. (uncorrupted) samples $S = \{(x_i, y_i)\}_{i=1}^m$ from $\mathcal{D}$. Recently, Montasser et al. [2019] showed that if $\mathcal{C}$ is PAC-learnable non-robustly, then $\mathcal{C}$ is also adversarially robustly learnable. However, their result is not constructive and the robust learning algorithm given is inefficient, complex, and does not actually directly use a non-robust learner. In this paper, we ask a more constructive version of this question:

Can we learn adversarially robust predictors given only black-box access to a non-robust learner?

That is, we are asking whether it is possible to reduce adversarially robust learning to standard non-robust learning. Since we have a plethora of algorithms devised for standard non-robust learning, it would be useful if we could design efficient *reduction* algorithms that leverage such non-robust learning algorithms in a black-box manner to learn *robustly*. That is, design generic wrapper methods that take as input a learning algorithm $\mathcal{A}$ and a specification of the adversary $\mathcal{U}$, and robustly learn by calling $\mathcal{A}$. Many systems in practice perform standard learning but with no robustness guarantees, and therefore, it would be beneficial to provide wrapper procedures that can guarantee adversarial robustness in a black-box manner without needing to modify current learning systems internally.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

Related Work Recent work [Mansour et al., 2015, Feige et al., 2015, 2018, Attias et al., 2019] can be interpreted as giving reduction algorithms for adversarially robust learning. Specifically, Feige et al. [2015] gave a reduction algorithm that can robustly learn a *finite* hypothesis class $\mathcal{C}$ using black-box access to an ERM for $\mathcal{C}$. Later, Attias et al. [2019] improved this to handle *infinite* hypothesis classes $\mathcal{C}$. But their complexity and the number of calls to ERM depend super-linearly on the number of possible perturbations $|\mathcal{U}| = \sup_x |\mathcal{U}(x)|$, which is undesirable for most types of perturbations—we completely avoid a sample complexity dependence on $|\mathcal{U}|$, and reduce the oracle complexity to at most a poly-logarithmic dependence. Furthermore, their work assumes access specifically to an ERM procedure, which is a very specific type of learner, while we only require access to any method that PAC-learns $\mathcal{C}$ and whose image has bounded VC-dimension.

A related goal was explored by Salman et al. [2020]: They proposed a method to *robustify pre-trained predictors*. Their method takes as input a black-box *predictor* (not a learning algorithm) and a point x, and outputs a label prediction y for x and a radius r such that the label y is robust to ℓ_2 perturbations of radius r. But this doesn't guarantee that the predictions y are correct, nor that the radius r would be what we desire, and even if the predictor was returned by a learning algorithm and has a very small non-robust error, we do not end up with any gurantee on the robust risk of the robustified predictor. In this paper, we require black-box access to a *learning algorithm* (not just to a single predictor), but we output a predictor that *is* guaranteed to have *small* robust risk (if one exists in the class, see Definition 2.2). We also provide a general treatment for arbitrary adversaries $\mathcal{U}$, not just ℓ_p perturbations.

Finally, we note that the approach of Montasser et al. [2019] can be interpreted as using black-box access to an oracle $\mathrm{RERM}_{\mathcal{C}}$ minimizing the robust *empirical* risk:

$$\hat{h} \in \mathrm{RERM}_{\mathcal{C}}(S) \triangleq \operatorname*{argmin}_{h \in \mathcal{C}} \frac{1}{m} \sum_{i=1}^{m} \sup_{z \in \mathcal{U}(x)} \mathbb{1}[h(z) \neq y]. \tag{2}$$

But this goes well beyond just a *non-robust* learning algorithm, or even ERM.

Efficient Reductions From a computational perspective, the relationship between standard non-robust learning and adversarially robust learning is not well-understood. It is natural to wonder whether there is a general efficient reduction for adversarially robust learning, using only non-robust learners. Recent work has provided strong evidence that this is not the case in general. Specifically, Bubeck et al. [2019] showed that there exists a learning problem that can be learned efficiently non-robustly, but is computationally intractable to learn robustly (under plausible complexity-theoretic assumptions). In this paper, we aim to understand when such efficient reductions *are* possible.

Main Results When studying reductions of adversarially robust learning to non-robust learning, an important aspect emerges regarding the form of access that the reduction algorithm has to the adversary $\mathcal{U}$. How should we model access to the sets of adversarial perturbations represented by $\mathcal{U}$?

In Section 3, we study the setting where the reduction algorithm has explicit access/knowledge of the possible adversarial perturbations induced by the adversary $\mathcal{U}$ on the *training examples*. We first show that there is an algorithm that can learn adversarially robust predictors with black-box oracle access to a non-robust algorithm:

Theorem 3.1 (Informal)**.** *For any adversary $\mathcal{U}$, Algorithm 1 robustly learns any target class $\mathcal{C}$ using any black-box non-robust PAC learner $\mathcal{A}$ for $\mathcal{C}$, with $O(\log^2 |\mathcal{U}|)$ oracle calls to $\mathcal{A}$ and sample complexity independent of $|\mathcal{U}|$.*

The oracle complexity dependence on $|\mathcal{U}|$, even if only logarithmic, might be disappointing, but we show it is unavoidable:

Theorem 3.2 (Informal)**.** *There exists an adversary $\mathcal{U}$ such that for any reduction algorithm $\mathcal{B}$, there exists a target class $\mathcal{C}$ and a PAC learner $\mathcal{A}$ for $\mathcal{C}$ such that $\Omega(\log |\mathcal{U}|)$ oracle queries to $\mathcal{A}$ are necessary to robustly learn $\mathcal{C}$.*

This tells us that only requiring a non-robust PAC learner $\mathcal{A}$ is not enough to avoid the $\log |\mathcal{U}|$ dependence, even with explicit knowledge of $\mathcal{U}$. In Section 4, we show that having an *online* learner $\mathcal{A}$ for $\mathcal{C}$, allows us to robustly learn $\mathcal{C}$ with access to a mistake oracle for $\mathcal{U}$ (see Definition 4.1) where no explicit knowledge of $\mathcal{U}$ is assumed and no dependence on $|\mathcal{U}|$ is incurred:

Theorem 4.2 (Informal)**.** *There exists an algorithm $\mathcal{B}$ that can robustly learn any target class $\mathcal{C}$ w.r.t. any adversary $\mathcal{U}$ when given access to a mistake oracle $\mathsf{O}_{\mathcal{U}}$ and a black-box online learner $\mathcal{A}$ for $\mathcal{C}$. The sample complexity, number of calls to $\mathcal{A}$, and number of calls to $\mathsf{O}_{\mathcal{U}}$ are independent of $|\mathcal{U}|$.*

2 Preliminaries

Let $\mathcal{X}$ denote the instance space and $\mathcal{Y} = \{\pm 1\}$ denote the label space. Let $\mathcal{U} : \mathcal{X} \to 2^{\mathcal{X}}$ denote an arbitrary adversary. For any adversary $\mathcal{U}$, denote by $|\mathcal{U}| \triangleq \sup_x |\mathcal{U}(x)|$ the number of allowed adversarial perturbations. We start with formalizing the notions of non-robust (standard) PAC learning and robust PAC learning:

Definition 2.1 (PAC Learnability)**.** *A target hypothesis class $\mathcal{C} \subseteq \mathcal{Y}^{\mathcal{X}}$ is said to be PAC learnable if there exists a learning algorithm $\mathcal{A} : (\mathcal{X} \times \mathcal{Y})^* \to \mathcal{Y}^{\mathcal{X}}$ with sample complexity $m(\varepsilon, \delta) : (0,1)^2 \to \mathbb{N}$ such that: for any $\varepsilon, \delta \in (0,1)$, for any distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, and any target concept $c \in \mathcal{C}$ with zero risk, $\mathrm{err}_{\mathcal{D}}(c) = 0$, with probability at least $1 - \delta$ over $S \sim \mathcal{D}^{m(\varepsilon,\delta)}$,*

$$\mathrm{err}_{\mathcal{D}}(\mathcal{A}(S)) \triangleq \Pr_{(x,y)\sim\mathcal{D}}[\mathcal{A}(S)(x) \neq y] \leq \varepsilon.$$

Definition 2.2 (Robust PAC Learnability)**.** *A target hypothesis class $\mathcal{C} \subseteq \mathcal{Y}^{\mathcal{X}}$ is said to be* robustly *PAC learnable with respect to adversary $\mathcal{U}$ if there exists a learning algorithm $\mathcal{B} : (\mathcal{X} \times \mathcal{Y})^* \to \mathcal{Y}^{\mathcal{X}}$ with sample complexity $m(\varepsilon, \delta) : (0,1)^2 \to \mathbb{N}$ such that: for any $\varepsilon, \delta \in (0,1)$, for any distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, and any target concept $c \in \mathcal{C}$ with zero robust risk, $\mathrm{R}_{\mathcal{U}}(c; \mathcal{D}) = 0$, with probability at least $1 - \delta$ over $S \sim \mathcal{D}^{m(\varepsilon,\delta)}$,*

$$\mathrm{R}_{\mathcal{U}}(\mathcal{B}(S); \mathcal{D}) \leq \epsilon.$$

We recall the Vapnik-Chervonenkis dimension (VC dimension) is defined as follows:

Definition 2.3 (VC dimension)**.** *We say that a sequence $\{x_1, \ldots, x_k\} \in \mathcal{X}$ is shattered by $\mathcal{C}$ if $\forall y_1, \ldots, y_k \in \mathcal{Y}, \exists h \in \mathcal{C}$ such that $\forall i \in [k], h(x_i) = y_i$. The VC dimension of $\mathcal{C}$ (denoted $\mathrm{vc}(\mathcal{C})$) is then defined as the largest integer k for which there exists $\{x_1, \ldots, x_k\} \in \mathcal{X}$ that is shattered by $\mathcal{C}$. If no such k exists, then $\mathrm{vc}(\mathcal{C})$ is said to be infinite.*

Another important complexity measure that is utilized in the study of robust PAC learning is the notion of *dual* VC dimension, which we define below:

Definition 2.4 (Dual VC dimension)**.** *Consider a* dual space *$\mathcal{G}$: a set of functions $g_x : \mathcal{C} \to \mathcal{Y}$ defined as $g_x(h) = h(x)$, for each $h \in \mathcal{C}$ and each $x \in \mathcal{X}$. Then, the dual VC dimesion of $\mathcal{C}$ (denoted $\mathrm{vc}^*(\mathcal{C})$) is defined as the VC dimension of $\mathcal{G}$. In other words, $\mathrm{vc}^*(\mathcal{C}) = \mathrm{vc}(\mathcal{G})$ and it represents the largest set $\{h_1, \ldots, h_k\}$ that is shattered by points in $\mathcal{X}$.*

If the VC dimension is finite, then so is the dual VC dimesion, and it can be bounded as $\mathrm{vc}^*(\mathcal{C}) < 2^{\mathrm{vc}(\mathcal{C})+1}$ [Assouad, 1983]. Although this exponential dependence is tight for some classes, for many natural classes, such as linear predictors and some neural networks (see, e.g. Lemma 3.2), the primal and dual VC dimensions are equal, or at least polynomially related.

We also formally define what we mean by a *reduction* algorithm:

Definition 2.5 (Reduction Algorithm)**.** *For an adversary $\mathcal{U}$, a reduction algorithm $\mathcal{B}_{\mathcal{U}}$ takes as input a black-box learning algorithm $\mathcal{A}$ and a training set $S \subseteq \mathcal{X} \times \mathcal{Y}$, and can use $\mathcal{A}$ by calling it T times on inputs $\mathcal{B}_{\mathcal{U}}$ constructs each of size $m_0 \in \mathbb{N}$, and outputs a predictor $f \in \mathcal{Y}^{\mathcal{X}}$.*

We emphasize that $\mathcal{B}_{\mathcal{U}}$ is allowed to be adaptive in its calls to $\mathcal{A}$. That is, it can call $\mathcal{A}$ on one constructed data set, then construct another data set depending on the returned predictor, and call $\mathcal{A}$ on this new data set. Such adaptive use of the base learner $\mathcal{A}$ is central to boosting-type constructions.

We know that a hypothesis class $\mathcal{C}$ is PAC learnable if and only if its VC dimension is finite [Vapnik and Chervonenkis, 1971, 1974, Blumer et al., 1989, Ehrenfeucht et al., 1989]. And in this case, $\mathcal{C}$ is properly PAC learnable with $\mathrm{ERM}_{\mathcal{C}}$. Montasser et al. [2019, Theorem 4] showed that if $\mathcal{C}$ is PAC learnable, then $\mathcal{C}$ is adversarially robustly PAC learnable with an improper learning rule that required a $\mathrm{RERM}_{\mathcal{C}}$ oracle (see Equation 2) and sample complexity of $\tilde{O}\left(\frac{\mathrm{vc}(\mathcal{C})\mathrm{vc}^*(\mathcal{C})}{\varepsilon}\right)$. In this paper, we study whether it is possible to adversarially robustly PAC learn $\mathcal{C}$ using only a black-nox non-robust PAC learner $\mathcal{A}$ for $\mathcal{C}$. We will not require $\mathcal{A}$ is "proper" (i.e. returns a predictor in $\mathcal{C}$), but we will rely on it returning a predictor in some, possibly much larger, class which still has finite VC-dimension. To this end, we denote by $\mathrm{vc}(\mathcal{A}) = \mathrm{vc}(\mathrm{im}(\mathcal{A}))$ and $\mathrm{vc}^*(\mathcal{A}) = \mathrm{vc}^*(\mathrm{im}(\mathcal{A}))$ the primal and dual VC dimension of the image of $\mathcal{A}$, i.e. the class $\mathrm{im}(\mathcal{A}) = \{\mathcal{A}(S) | S \in (\mathcal{X} \times \mathcal{Y})^*\}$ of the possible hypothesis $\mathcal{A}$ might return. For ERM, or any other proper learner, $\mathrm{im}(\mathcal{A}) \subseteq \mathcal{C}$ and so $\mathrm{vc}(\mathcal{A}) \leq \mathrm{vc}(\mathcal{C})$ and $\mathrm{vc}^*(\mathcal{A}) \leq \mathrm{vc}^*(\mathcal{C})$.

3 Learning with Explicitly Specified Adversarial Perturbations

When studying reductions of adversarially robust PAC learning to non-robust PAC learning, an important aspect emerges regarding the form of access that the reduction algorithm has to the adversary $\mathcal{U}$. How should we model access to the sets of adversarial perturbations represented by $\mathcal{U}$?

In this section, we explore the setting where the reduction algorithm has explicit knowledge of the adversary $\mathcal{U}$. That is, the reduction algorithm knows the set of possible adversarial perturbations for each example in the training set. This is in accordance with what is typically considered in practice, where the adversary $\mathcal{U}$ (e.g. ℓ_∞ perturbations) is known to the algorithm, and this knowledge is used in adversarial training (see e.g. Madry et al. [2018]). Formally, we consider the following question:

For any adversary $\mathcal{U}$, does there exist an algorithm that can learn a target class $\mathcal{C}$ *robustly* w.r.t $\mathcal{U}$ given only a black-box non-robust PAC learner $\mathcal{A}$ for $\mathcal{C}$?

We give a positive answer to this question. In Theorem 3.1, we present an algorithm (see Algorithm 1)—based on the α-Boost algorithm [Schapire and Freund, 2012, Section 6.4.2] and recent work of Montasser et al. [2019, Theorem 4]— that can adversarially robustly PAC learn a target class $\mathcal{C}$ with only black-box oracle access to a PAC learner $\mathcal{A}$ for $\mathcal{C}$.

Algorithm 1: Robustify The Non-Robust

Input: Training dataset $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, black-box non-robust learner $\mathcal{A}$

Inflate dataset S to $S_{\mathcal{U}} = \bigcup_{i \le m} \{(z, y_i) : z \in \mathcal{U}(x_i)\}$. // $S_{\mathcal{U}}$ contains all possible perturbations of S.

Set $m_0 = O(\mathrm{vc}(\mathcal{A})\mathrm{vc}(\mathcal{A})^* \log \mathrm{vc}(\mathcal{A})^*)$, and $T = O(\log |S_{\mathcal{U}}|)$.

for $1 \le t \le T$ **do**

- Set distribution D_t on $S_{\mathcal{U}}$ as in the α-Boost algorithm.
- Sample $S' \sim D_t^{m_0}$, and project S' to dataset $L \subseteq S$ by replacing each perturbation z with its corresponding example x.
- Call `ZeroRobustLoss` on L, and denote by f_t its output predictor.

Sample $N_{\mathrm{co}} = O\left(\mathrm{vc}^*(\mathcal{A}) \log \mathrm{vc}^*(\mathcal{A})\right)$ i.i.d. indices $i_1, \ldots, i_{N_{\mathrm{co}}} \sim \mathrm{Uniform}(\{1, \ldots, T\})$.
(repeat previous step until $f = \mathrm{MAJ}(f_{i_1}, \ldots, f_{i_{N_{\mathrm{co}}}})$ has $R_{\mathcal{U}}(f; S) = 0$)

Output: A majority-vote $\mathrm{MAJ}(f_{i_1}, \ldots, f_{i_{N_{\mathrm{co}}}})$ predictor.

`ZeroRobustLoss`(*Dataset L, Learner $\mathcal{A}$*)**:**

- Inflate dataset L to $L_{\mathcal{U}} = \bigcup_{(x,y) \in L} \{(z, y) : z \in \mathcal{U}(x)\}$, and set $T_L = O(\log |L_{\mathcal{U}}|)$
- Run α-Boost with black-box access to $\mathcal{A}$ on $L_{\mathcal{U}}$ for T_L rounds.
- Let $h_1, \ldots, h_{T_L}$ denote the hypotheses produced by α-Boost with T_L oracle queries to $\mathcal{A}$.
- Sample $N = O\left(\mathrm{vc}^*(\mathcal{A})\right)$ i.i.d. indices $i_1, \ldots, i_N \sim \mathrm{Uniform}(\{1, \ldots, T_L\})$.
  (repeat previous step until $f = \mathrm{MAJ}(h_{i_1}, \ldots, h_{i_N})$ has $R_{\mathcal{U}}(f; L) = 0$)
- **return** $f = \mathrm{MAJ}(h_{i_1}, \ldots, h_{i_N})$

α-`Boost`(*Dataset L, Learner $\mathcal{A}$*)**:**

- Initialize D_1 to be uniform over L, and set $T_L = O(\log |L|)$.
- **for** $1 \le t \le T_L$ **do**
  - Run $\mathcal{A}$ on $S' \sim D_t^{m_0}$, and denote by h_t its output. (repeat until $\mathrm{err}_{D_t}(h_t) \le 1/3$)
  - Compute a new distribution D_{t+1} by applying the following update for each $(x, y) \in L$:

$$D_{t+1}(x) = \frac{D_t(x)}{Z_t} \times \begin{cases} e^{-2\alpha} & \text{if } h_t(x) = y \\ 1 & \text{otherwise} \end{cases}$$

  - where Z_t is a normalization factor and α is set as in Lemma 3.3
- **return** $h_1, \ldots, h_{T_L}$.

Theorem 3.1. *For any adversary $\mathcal{U}$, Algorithm 1 can* robustly PAC learn *any target class $\mathcal{C}$ using black-box oracle calls to any PAC learner $\mathcal{A}$ for $\mathcal{C}$ with:*

1. *Sample Complexity* $m = O\left(\frac{dd^{*2}\log^2 d^*}{\varepsilon} \log\left(\frac{dd^{*2}\log^2 d^*}{\varepsilon}\right) + \frac{\log(1/\delta)}{\varepsilon}\right)$,
2. *Oracle Complexity* $T = O\left(\left(\log m + \log |\mathcal{U}|\right)^2 + \log(1/\delta)\right)$,

where $d = \mathrm{vc}(\mathcal{A})$ and $d^* = \mathrm{vc}^*(\mathcal{A})$ are the primal and dual VC dimension of $\mathcal{A}$.

Importantly, the sample complexity of Algorithm 1 is independent of the number of allowed perturbations $|\mathcal{U}|$, in contrast to work by Attias et al. [2019], that can be interpreted as giving a reduction with sample complexity $m \propto |\mathcal{U}| \log |\mathcal{U}|$, and oracle complexity $T \propto |\mathcal{U}| \log^2 |\mathcal{U}|$.

Before proceeding with the proof of Theorem 3.1, we briefly describe our strategy and its main ingredients. Given a dataset S that is robustly realizable by some target concept $c \in \mathcal{C}$, we show that we can use the non-robust learner $\mathcal{A}$ to implement a RERM oracle that guarantees zero *empirical* robust loss on S using `ZeroRobustLoss` in Algorithm 1. But what about the *population* robust loss? Our main goal to is adversarially robustly learn $\mathcal{C}$ and not just minimize the empirical robust loss. Fortunately, we show that the arguments on *robust* generalization based on sample compression in [Montasser et al., 2019, Theorem 4] will still go through when we replace the $\mathrm{RERM}_{\mathcal{C}}$ oracle they used with our `ZeroRobustLoss` procedure in Algorithm 1. This is achieved by showing that the image of `ZeroRobustLoss` has bounded VC dimension and *dual* VC dimension. The following lemma, whose proof is provided in Appendix A, bounds the *dual* VC dimension of the convex-hull of a class $\mathcal{H}$. This result might be of independent interest.

Lemma 3.2. *Let $\mathrm{co}^k(\mathcal{H}) = \{x \mapsto \mathrm{MAJ}(h_1, \ldots, h_k)(x) : h_1, \ldots, h_k \in \mathcal{H}\}$. Then, the dual VC dimension of $\mathrm{co}^k(\mathcal{H})$ satisfies $\mathrm{vc}^*(\mathrm{co}^k(\mathcal{H})) \leq O(d^* \log k)$.*

In addition, we state two extra key lemmas that will be useful for us in the proof. First, Lemma 3.3 states that running α-Boost on a dataset for enough rounds produces a sequence of predictors that achieve zero loss on the dataset (with a margin).

Lemma 3.3 (see, e.g., Corollary 6.4 and Section 6.4.3 in Schapire and Freund [2012])**.** *Let $S = \{(x_i, c(x_i))\}_{i=1}^m$ be a dataset where $c \in \mathcal{C}$ is some target concept, and $\mathcal{A}$ an arbitrary PAC learner for $\mathcal{C}$ (for $\varepsilon = 1/3$, $\delta = 1/3$). Then, running α-Boost (see description in Algorithm 1) on S with black-box oracle access to $\mathcal{A}$ with $\alpha = \frac{1}{2}\ln\left(1 + \sqrt{\frac{2\ln m}{T}}\right)$ for $T = \lceil 112 \ln(m) \rceil = O(\log m)$ rounds suffices to produce a sequence of hypotheses $h_1, \ldots, h_T \in \mathrm{im}(\mathcal{A})$ such that*

$$\forall (x, y) \in S, \frac{1}{T}\sum_{i=1}^{T} \mathbb{1}[h_i(x) = y] \geq \frac{5}{9}.$$

In particular, this implies that the majority-vote $\mathrm{MAJ}(h_1, \ldots, h_T)$ achieves zero error on S.

Second, Lemma 3.4 describes a sparsification technique due to Moran and Yehudayoff [2016] which allows us to control the complexity of the majority-vote predictors that we use in Algorithm 1.

Lemma 3.4 (Sparsification of Majority Votes, Moran and Yehudayoff [2016])**.** *Let $\mathcal{H}$ be a hypothesis class with finite primal and dual VC dimension, and $h_1, \ldots, h_T$ be predictors in $\mathcal{H}$. Then, for any $(\varepsilon, \delta) \in (0, 1)$, with probability at least $1 - \delta$ over $N = O\left(\frac{\mathrm{vc}^*(\mathcal{H}) + \log(1/\delta)}{\varepsilon^2}\right)$ independent random indices $i_1, \ldots, i_N \sim \mathrm{Uniform}(\{1, \ldots, T\})$, we have:*

$$\forall (x, y) \in \mathcal{X} \times \mathcal{Y}, \left| \frac{1}{N}\sum_{j=1}^{N} \mathbb{1}[h_{i_j}(x) = y] - \frac{1}{T}\sum_{i=1}^{T} \mathbb{1}[h_i(x) = y] \right| < \varepsilon.$$

We are now ready to proceed with the proof of Theorem 3.1.

Proof of Theorem 3.1. Let $\mathcal{U}$ be an arbitrary adversary. Let $\mathcal{C}$ be a target class that is PAC learnable with some PAC learner $\mathcal{A}$. Let $\mathcal{H}$ denote the base class of hypotheses of learner $\mathcal{A}$. Let d denote the VC dimension of $\mathcal{H}$, and d^* denote the dual VC dimension of $\mathcal{H}$. Our proof is divided into two parts.

Zero Empirical Robust Loss. Let $L = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ be a dataset that is *robustly* realizable with some target concept $c \in \mathcal{C}$; in other words, for each $(x, y) \in L$ and each $z \in \mathcal{U}(x)$, $c(z) = y$. We will show that we can use the non-robust learner $\mathcal{A}$ to guarantee zero *empirical* robust loss on L. This procedure is described in `ZeroRobustLoss` in Algorithm 1. We inflate dataset L to include all possible perturbations under the adversary $\mathcal{U}$. Let $L_{\mathcal{U}} = \bigcup_{i \leq m} \{(z, y_i) : z \in \mathcal{U}(x_i)\}$ denote the inflated dataset. Observe that $|L_{\mathcal{U}}| \leq m|\mathcal{U}|$, since each point $x \in \mathcal{X}$ has at most $|\mathcal{U}|$

possible perturbations. We run the α-Boost algorithm on the inflated dataset $L_{\mathcal{U}}$ with *black-box* access to PAC learner $\mathcal{A}$, where in each round of boosting m_0 samples are fed to $\mathcal{A}$ (where m_0 is chosen Step 2). By Lemma 3.3, running α-Boost with $T = O\left(\log(|L_{\mathcal{U}}|)\right)$ oracle calls to $\mathcal{A}$ suffices to produce a sequence of hypotheses $h_1, \ldots, h_T \in \mathcal{H}$ such that

$$\forall (z,y) \in L_{\mathcal{U}}, \frac{1}{T}\sum_{i=1}^{T} \mathbb{1}[h_i(z) = y] \geq \frac{5}{9}.$$

Specifically, the above implies that a majority-vote over hypotheses $h_1, \ldots, h_T$ achieves zero *robust* loss on dataset L, $\mathrm{R}_{\mathcal{U}}(\mathrm{MAJ}(h_1, \ldots, h_T); L) = 0$. By Step 13 in `ZeroRobustLoss` in Algorithm 1 and Lemma 3.4 (with $\varepsilon = 1/18, \delta = 1/3$), we have that for $N = O(d^*)$, the sampled predictors $h_{i_1}, \ldots, h_{i_N}$ satisfy

$$\forall (z,y) \in L_{\mathcal{U}}, \frac{1}{N}\sum_{j=1}^{N} \mathbb{1}[h_{i_j}(z) = y] > \frac{1}{T}\sum_{i=1}^{T} \mathbb{1}[h_i(z) = y] - \frac{1}{18} > \frac{5}{9} - \frac{1}{18} = \frac{1}{2}.$$

Therefore, the majority-vote over the sampled hypotheses $\mathrm{MAJ}(h_{i_1}, \ldots, h_{i_N})$ achieves zero robust loss on L, $\mathrm{R}_{\mathcal{U}}(\mathrm{MAJ}(h_{i_1}, \ldots, h_{i_N}); S) = 0$. Thus, we can implement a RERM oracle (see Equation 2) using the procedure `ZeroRobustLoss` in Algorithm 1. The sparsification step (Step 12) controls the complexity of the image of `ZeroRobustLoss`, i.e., the hypothesis class that is being implicitly used. Specifically, observe that the sparsified predictor $f = \mathrm{MAJ}(h_{i_1}, \ldots, h_{i_N})$ lives in $\mathrm{co}^{O(d^*)}(\mathcal{H})$, which is the convex-hull of $\mathcal{H}$ that combines at most $O(d^*)$ predictors. To guarantee *robust* generalization in the next part, it suffices to bound the VC dimension and dual VC dimension of $\mathrm{co}^{O(d^*)}(\mathcal{H})$. By [Blumer et al., 1989], the VC dimension of $\mathrm{co}^{O(d^*)}(\mathcal{H})$ is at most $O(dd^* \log d^*)$, and by Lemma 3.2, the dual VC dimension of $\mathrm{co}^{O(d^*)}(\mathcal{H})$ is at most $O(d^* \log d^*)$.

Robust Generalization through Sample Compression. This part builds on the approach of Montasser et al. [2019, Theorem 4]. Specifically, we observe that their proof works even if we replace the $\mathrm{RERM}_{\mathcal{C}}$ oracle they used, with our `ZeroRobustLoss` procedure in Algorithm 1 that is described above. We provide a self-contained analysis below.

Let $\mathcal{D}$ be an arbitrary distribution over $\mathcal{X} \times \mathcal{Y}$ that is robustly realizable with some concept $c \in \mathcal{C}$, i.e., $\mathrm{R}_{\mathcal{U}}(c; \mathcal{D}) = 0$. Fix $\varepsilon, \delta \in (0,1)$ and a sample size m that will be determined later. Let $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ be an i.i.d. sample from $\mathcal{D}$. We run the α-Boost algorithm (see Algorithm 1) on the inflated dataset $S_{\mathcal{U}}$, this time with `ZeroRobustLoss` as the subprocedure. Specifically, on each round of boosting, α-Boost computes an empirical distribution D_t over $S_{\mathcal{U}}$ (according to Step 18). We draw $m_0 = O(dd^* \log d^*)$ samples S' from D_t, and *project* S' to a dataset $L_t \subset S$ by replacing each perturbation $(z,y) \in S'$ with its corresponding original point $(x,y) \in S$, and then we run `ZeroRobustLoss` on dataset L_t. The projection step is crucial for the proof to work, since we use a *sample compression* argument to argue about *robust* generalization, and the sample compression must be done on the *original* points that appeared in S rather than the perturbations in $S_{\mathcal{U}}$.

By classic PAC learning guarantees [Vapnik and Chervonenkis, 1974, Blumer et al., 1989], with $m_0 = O(\mathrm{vc}(\mathrm{co}^{O(d^*)}(\mathcal{H}))) = O(dd^* \log d^*)$, we are guaranteed uniform convergence of 0-1 risk over predictors in $\mathrm{co}^{O(d^*)}(\mathcal{H})$ (the effective hypothesis class used by `ZeroRobustLoss`). So, for any distribution D over $\mathcal{X} \times \mathcal{Y}$ with $\inf_{c \in \mathcal{C}} \mathrm{err}(c; \mathcal{D}) = 0$, with nonzero probability over $S' \sim \mathcal{D}^{m_0}$, every $f \in \mathrm{co}^{O(d^*)}(\mathcal{H})$ satisfying $\mathrm{err}_{S'}(f) = 0$, also has $\mathrm{err}_D(f) < 1/3$. By the guarantee of `ZeroRobustLoss` (established above), we know that $f_t = \texttt{ZeroRobustLoss}(L_t, \mathcal{A})$ achieves zero robust loss on L_t, $\mathrm{R}_{\mathcal{U}}(f_t; L_t) = 0$, which by definition of the projection means that $\mathrm{err}_{S'}(f_t) = 0$, and thus $\mathrm{err}_{D_t}(f_t) < 1/3$. This allows us to use `ZeroRobustLoss` with α-Boost to establish a *robust* generalization guarantee. Specifically, Lemma 3.3 implies that running the α-Boost algorithm with $S_{\mathcal{U}}$ as its dataset for $T = O(\log(|S_{\mathcal{U}}|))$ rounds, using `ZeroRobustLoss` to produce the hypotheses $f_t \in \mathrm{co}^{O(d^*)}(\mathcal{H})$ for the distributions D_t produced on each round of the algorithm, will produce a sequence of hypotheses $f_1, \ldots, f_T \in \mathrm{co}^{O(d^*)}(\mathcal{H})$ such that:

$$\forall (z,y) \in S_{\mathcal{U}}, \frac{1}{T}\sum_{i=1}^{T} \mathbb{1}[f_i(z) = y] \geq \frac{5}{9}.$$

Specifically, this implies that the majority-vote over hypotheses $f_1, \ldots, f_T$ achieves zero *robust* loss on dataset S, $\mathrm{R}_{\mathcal{U}}(\mathrm{MAJ}(f_1, \ldots, f_T); S) = 0$. Note that each of these classifiers f_t is equal to $\texttt{ZeroRobustLoss}(L_t, \mathcal{A})$ for some $L_t \subseteq S$ with $|L_t| = m_0$. Thus, the classifier $\mathrm{MAJ}(f_1, \ldots, f_T)$ is representable as the value of an (order-dependent) reconstruction function ϕ with a compression set size

$$m_0 T = O(\mathrm{vc}(\mathrm{co}^{O(d^*)}(\mathcal{H})) \log(|S_{\mathcal{U}}|)) = O(dd^* \log d^* \, (\log m + \log |\mathcal{U}|)).$$

This is not enough, however, to obtain a sample complexity bound that is independent of $|\mathcal{U}|$. For that, we will sparsify the majority-vote as in Step 7 in Algorithm 1. Lemma 3.4 (with $\varepsilon = 1/18, \delta = 1/3$) guarantees that for $N_{\mathrm{co}} = O(d^* \log d^*)$, the sampled predictors $f_{i_1}, \ldots, f_{i_{N_{\mathrm{co}}}}$ satisfy:

$$\forall (z, y) \in S_{\mathcal{U}}, \frac{1}{N_{\mathrm{co}}} \sum_{j=1}^{N_{\mathrm{co}}} \mathbb{1}[f_{i_j}(z) = y] > \frac{1}{T} \sum_{i=1}^{T} \mathbb{1}[f_i(z) = y] - \frac{1}{18} > \frac{5}{9} - \frac{1}{18} = \frac{1}{2},$$

so that the majority-vote achieves zero robust loss on S, $\mathrm{R}_{\mathcal{U}}(\mathrm{MAJ}(f_{i_1}, \ldots, f_{i_{N_{\mathrm{co}}}}); S) = 0$. Since again, each f_{i_j} is the result of $\texttt{ZeroRobustLoss}(L_t, \mathcal{A})$ for some $L_t \subseteq S$ with $|L_t| = m_0$, we have that the classifier $\mathrm{MAJ}(f_{i_1}, \ldots, f_{i_{N_{\mathrm{co}}}})$ can be represented as the value of an (order-dependent) reconstruction function ϕ with a compression set size $m_0 N_{\mathrm{co}} = O(dd^* \log d^* \cdot d^* \log d^*) = O(dd^{*2} \log^2(d^*))$. Lemma A.1 (Montasser et al. [2019]) which extends to the robust loss the classic compression-based generalization guarantees from the 0-1 loss, implies that for $m \geq cdd^{*2} \log^2(d^*)$ (for an appropriately large numerical constant c), with probability at least $1 - \delta$ over $S \sim \mathcal{D}^m$,

$$R_{\mathcal{U}}(\mathrm{MAJ}(f_{i_1}, \ldots, f_{i_{N_{\mathrm{co}}}}); \mathcal{D}) \leq O\left(\frac{dd^{*2} \log^2(d^*)}{m} \log(m) + \frac{1}{m} \log(1/\delta) \right).$$

Setting this less than ε and solving for a sufficient size of m to achieve this yields the stated sample complexity bound.

Our oracle complexity T (number of calls to $\mathcal{A}$) is at most $O((\log |S_{\mathcal{U}}|)^2 + \log(1/\delta)) \leq O\left((\log m + \log |\mathcal{U}|)^2 + \log(1/\delta) \right)$, since `ZeroRobustLoss` in Algorithm 1 terminates in at most $O(\log |S_{\mathcal{U}}|)$ rounds each time it is invoked, and it is invoked at most $O(\log |S_{\mathcal{U}}|)$ times by the outer α-Boost algorithm in Algorithm 1. Therefore, we have at most $O\left((\log m + \log |\mathcal{U}|)^2 \right)$ geometric random variables that represent that number of times Step 19 is invoked, which is the step where learner $\mathcal{A}$ is called. The success probability of Step 19 is a constant (say $2/3$), therefore the mean of the sum of the geometric random variables is $O\left((\log m + \log |\mathcal{U}|)^2 \right)$. Since sums of geometric random variables concentrate around the mean [Brown], we get that with probability at least $1 - \delta$, the total number of times Step 19 is executed is at most $O\left((\log m + \log |\mathcal{U}|)^2 + \log(1/\delta) \right)$. This concludes the proof. □

3.1 A Lowerbound on the Oracle Complexity

The oracle complexity of Algorithm 1 depends on $\log |\mathcal{U}|$. Can this dependence be reduced or avoided? Unfortunately, we show in Theorem 3.5 that the dependence on $\log |\mathcal{U}|$ is unavoidable and *no* reduction algorithm can do better.

It is relatively easy to show, by relying on lower bounds from the boosting literature [Schapire and Freund, 2012, Section 13.2.2], that for any reduction algorithm, there exists a target class $\mathcal{C}$ with $\mathrm{vc}(\mathcal{C}) \leq 1$ and a PAC learner $\mathcal{A}$ for $\mathcal{C}$ such that $\Omega(\log |\mathcal{U}|)$ oracle calls to this PAC learner are necessary to achieve zero *empirical* robust loss. But this is done by constructing a "crazy" improper learner with $\mathrm{vc}(\mathcal{A}) \propto |\mathcal{U}| \gg \mathrm{vc}(\mathcal{C})$.

Perhaps we can ensure better oracle complexity by requiring a more constrained PAC learner $\mathcal{A}$, e.g. with low VC dimension (as in our upper bound), or perhaps even a proper learner, or an ERM, or maybe where $\mathrm{im}(\mathcal{A})$ (and so also $\mathcal{C}$) is finite. We next present a lower bound to show that none of these help improve the oracle complexity. Specifically, we will present a construction showing that for any reduction algorithm $\mathcal{B}$ there is a randomized target class $\mathcal{C}$ and a PAC learner $\mathcal{A}$ for $\mathcal{C}$ with

$\mathrm{vc}(\mathcal{A}) = 1$ where $\mathcal{B}$ needs to make $\Omega(\log|\mathcal{U}|)$ oracle calls to $\mathcal{A}$ to robustly learn $\mathcal{C}$. The idea here is that the target class $\mathcal{C}$ is chosen randomly after $\mathcal{B}$, and so $\mathcal{B}$ essentially knows nothing about $\mathcal{C}$ and needs to communicate with $\mathcal{A}$ in order to learn. As a reminder, a reduction algorithm has a budget of T oracle calls to a non-robust learner $\mathcal{A}$, where each oracle call is constructed with m_0 points, or more generally a *distribution* over $\mathcal{X} \times \mathcal{Y}$. We next show that any successful reduction requires $T = \Omega(\log|\mathcal{U}|)$ for some non-robust learner $\mathcal{A}$.

Theorem 3.5. *For any sufficiently large integer u, if $|\mathcal{X}| \geq u^{10u}$, there exists an adversary $\mathcal{U}$ with $|\mathcal{U}| = u$ such that for any reduction algorithm $\mathcal{B}$ and for any $\varepsilon > 0$, there exists a target class $\mathcal{C}$ and a PAC learner $\mathcal{A}$ for $\mathcal{C}$ with $\mathrm{vc}(\mathcal{A}) = 1$ such that, if the training sample has size at most $(1/8)|\mathcal{U}|^9$, then $\mathcal{B}$ needs to make $T \geq \frac{\log|\mathcal{U}|}{\log(2/\varepsilon)}$ oracle calls to $\mathcal{A}$ in order to robustly learn $\mathcal{C}$.*

Proof. We begin with describing the construction of the adversary $\mathcal{U}$. Let $m \in \mathbb{N}$; we will construct $\mathcal{U}$ with $|\mathcal{U}| = 2^m$, supposing $|\mathcal{X}| \geq 2\binom{2^{10m}}{2^m} + 2^{10m}$. Let $Z = \{z_1, \ldots, z_{2^{10m}}\} \subset \mathcal{X}$ be a set of 2^{10m} unique points from $\mathcal{X}$. For each subset $L \subset Z$ where $|L| = 2^m$, pick a unique pair $x_L^+, x_L^- \in \mathcal{X} \setminus Z$ and define $\mathcal{U}(x_L^+) = \mathcal{U}(x_L^-) = L$. That is, for every choice L of 2^m perturbations from Z, there is a corresponding pair x_L^+, x_L^- where $\mathcal{U}(x_L^+) = \mathcal{U}(x_L^-) = L$. For any point $x \in \mathcal{X} \setminus Z$ that is remaining, define $\mathcal{U}(x) = \{\}$.

Let $\mathcal{B}$ be an arbitrary reduction algorithm, and let $\varepsilon > 0$ be the error requirement. We will now describe the construction of the target class $\mathcal{C}$. The target class $\mathcal{C}$ will be constructed randomly. Namely, we will first define a labeling $\tilde{h} : Z \to \mathcal{Y}$ on the perturbations in Z that is positive on the first half of Z and negative on the second half of Z: $\tilde{h}(z_i) = +1$ if $i \leq \frac{2^{10m}}{2}$, and $\tilde{h}(z_i) = -1$ if $i > \frac{2^{10m}}{2}$. Divide the positive/negative halves into groups of size 2^m:

$$\underbrace{\{\text{first } 2^m \text{ positives}\}}_{G_1^+}, \ldots, \underbrace{\{\text{last } 2^m \text{ positives}\}}_{G_{2^{9m-1}}^+} \Big| \underbrace{\{\text{first } 2^m \text{ negatives}\}}_{G_1^-}, \ldots, \underbrace{\{\text{last } 2^m \text{ negatives}\}}_{G_{2^{9m-1}}^-}.$$

Let $\varepsilon' = \varepsilon/2$. The target concept $h^* : \mathcal{X} \to \mathcal{Y}$ is generated by randomly flipping the labels of an ε' fraction of the points in each group $G_1^+, \ldots, G_{2^{9m-1}}^+$ from positive to negative and randomly flipping the labels of an ε' fraction of the points in each group $G_1^-, \ldots, G_{2^{9m-1}}^-$ from negative to positive. This defines h^* on Z; then for every pair $x^+, x^- \in \mathcal{X} \setminus Z$ where $\mathcal{U}(x^+) = \mathcal{U}(x^-) \neq \{\}$, define $h^*(x^+) = +1$ and $h^*(x^-) = -1$. Once h^* is generated, we define the distribution D_{h^*} over $\mathcal{X} \times \mathcal{Y}$ that will be used in the lower bound by swapping the ε' fractions of points with the flipped labels in each pair $(G_1^+, G_1^-), \ldots, (G_{2^{9m-1}}^+, G_{2^{9m-1}}^-)$ which defines new positive/negative pairs: $(G(h^*)_1^+, G(h^*)_1^-), \ldots, (G(h^*)_{2^{9m-1}}^+, G(h^*)_{2^{9m-1}}^-)$. Let $x_i^+, \_ = \mathcal{U}^{-1}(G(h^*)_i^+)$ and $\_, x_i^- = \mathcal{U}^{-1}(G(h^*)_i^-)$ for each $i \in [2^{9m-1}]$ ($\mathcal{U}^{-1}$ returns a pair of points). Observe that by definition of h^* on $\mathcal{X} \setminus Z$, we have that $h^*(x_i^+) = +1$ and $h^*(x_i^-) = -1$ since $h^*(z) = +1 \forall z \in G(h^*)_i^+$ and $h^*(z) = -1 \forall z \in G(h^*)_i^-$. Let D_{h^*} be a uniform distribution over $(x_1^+, +1), (x_1^-, -1), \ldots, (x_{2^{9m-1}}^+, +1), (x_{2^{9m-1}}^-, -1)$.

Let $T \leq \frac{\log 2^m}{\log(1/\varepsilon')}$. Define a randomly-constructed target class $\mathcal{C} = \{h_1, \ldots, h_T, h_{T+1}\}$ where $h_{T+1} = h^*$ and $h_1, h_2, \ldots, h_T$ are generated according the following process: If $t = 1$, then $h_1 := \tilde{h}$ (augmented to all of $\mathcal{X}$ by letting $\tilde{h}(x) = h^*(x)$ for all $x \in \mathcal{X} \setminus Z$). For $t \geq 2$, let $\mathrm{DIS}_{t-1} = \{z \in Z : h_{t-1}(z) \neq h^*(z)\}$, and construct h_t by flipping a uniform randomly-selected $1 - \varepsilon'$ fraction of the labels of h_{t-1} in $G_i^+ \cap \mathrm{DIS}_{t-1}$ and $1 - \varepsilon'$ fraction of the labels of h_{t-1} in $G_i^- \cap \mathrm{DIS}_{t-1}$ for each $i \in [2^{9m-1}]$. Observe that by construction, $h_1, \ldots, h_T$ satisfy the property that they agree with h^* on $\mathcal{X} \setminus Z$, i.e. $h_t(x) = h^*(x)$ for each $t \leq T$ and each $x \in \mathcal{X} \setminus Z$.

We now state a few properties of the randomly-constructed target class $\mathcal{C}$ that we will use in the remainder of the proof. First, observe that by definition of DIS_t for $t \leq T$, we have that $G_i^\pm \cap \mathrm{DIS}_T \subseteq G_i^\pm \cap \mathrm{DIS}_{T-1} \subseteq \cdots \subseteq G_i^\pm \cap \mathrm{DIS}_1$ for each $1 \leq i \leq 2^{9m-1}$. In addition,

$$|G_i^\pm \cap \mathrm{DIS}_t| \geq \varepsilon' |G_i^\pm \cap \mathrm{DIS}_{t-1}| \text{ for each } 1 \leq i \leq 2^{9m-1}.$$

By the random process generating h^*, we also know that $|G_i^\pm \cap \mathrm{DIS}_1| \geq \varepsilon' 2^m$. Combined with the above, this implies that:

$$|G_i^\pm \cap \mathrm{DIS}_T| \geq {\varepsilon'}^T 2^m \text{ for each } 1 \leq i \leq 2^{9m-1}.$$

So, for $T \leq \frac{\log 2^m}{\log(2/\varepsilon)}$, we are guaranteed that $|G_i^{\pm} \cap \mathrm{DIS}_T| \geq 1$ for each $1 \leq i \leq 2^{9m-1}$.

We now describe the construction of a PAC learner $\mathcal{A}$ with $\mathrm{vc}(\mathcal{A}) = 1$ for the randomly generated concept h^* above; we assume that $\mathcal{A}$ knows $\mathcal{C}$ (but of course, $\mathcal{B}$ does not know $\mathcal{C}$).

Algorithm 2: Non-Robust PAC Learner $\mathcal{A}$

Input: Distribution P over $\mathcal{X}$.

Output: h_s for the *smallest* $s \in [T]$ with $\mathrm{err}_P(h_s, h^*) \leq \varepsilon$ (or outputting $h_{T+1} = h^*$ if no such s exists).

First, we will show that $\mathrm{vc}(\mathcal{A}) = 1$. By definition of $\mathcal{A}$, it suffices to show that $\mathrm{vc}(\mathcal{C}) = \mathrm{vc}(\{h^*, h_1, \ldots, h_T\}) = 1$. By definition of h^* and h_1, it is easy to see that there is a $z \in Z$ where $h^*(z) \neq h_1(z)$, and thus $\mathrm{vc}(\mathcal{C}) \geq 1$. Observe that by construction, each predictor in $h_1, \ldots, h_T$ operates as a threshold in each group $G_1^+, G_1^-, \ldots, G_{2^{9m-1}}^+, G_{2^{9m-1}}^-$ (ordered according to the order in which the labels are flipped in the $h_1, \ldots, h_T$ sequence). As a result, each $x \in \mathcal{X}$ has its label flipped at most once in the sequence $(h_1(x), \ldots, h_T(x), h^*(x))$. This is because once the ground-truth label of x, $h^*(x)$, is revealed by some h_t (i.e., $h_t(x) = h^*(x)$), all subsequent predictors $h_{t'}$ satisfy $h_{t'}(x) = h^*(x)$. Thus, for any two points $z, z' \in \mathcal{X}$, the number of possible behaviors $|\{(h(z), h(z')) : h \in \mathcal{C}\}| \leq 3$. Therefore, $\mathcal{C}$ cannot shatter two points. This proves that $\mathrm{vc}(\mathcal{C}) \leq 1$.

Analysis Suppose that we run the reduction algorithm $\mathcal{B}$ with non-robust learner $\mathcal{A}$ for T rounds to obtain predictors $h_{s_1} = \mathcal{A}(P_1), \ldots, h_{s_T} = \mathcal{A}(P_T)$. We will show that $\Pr_{h^*}[s_T \leq T | S] > 0$, meaning that with non-zero probability learner $\mathcal{A}$ will not reveal the ground-truth hypothesis h^*. For $t \leq T$, let E_t denote the event that $\mathrm{err}_{P_t}(h_{s_{t-1}+1}, h^*) \leq \varepsilon$. When conditioning on $S, s_1, \ldots, s_{t-1}$, observe that by construction of the randomized hypothesis class $\mathcal{C}$, for each $i \leq 2^{9m-1}$ such that $\{(x_i^-, -1), (x_i^+, +1)\} \cap S = \emptyset$, and each $z \in G_i^{\pm} \cap \mathrm{DIS}_{s_{t-1}}$: $\Pr_{h^*}\left[h^*(z) \neq h_{s_{t-1}+1}(z) | S, s_1, \ldots, s_{t-1}\right] \leq \varepsilon' = \varepsilon/2$. It follows then by the law of total probability that for any distribution P_t constructed by $\mathcal{A}$:

$$\mathop{\mathbb{E}}_{h^*}\left[\mathrm{err}_{P_t}(h_{s_{t-1}+1}, h^*) | S, s_1, \ldots, s_{t-1}\right] \leq \frac{\varepsilon}{2}.$$

By Markov's inequality, it follows that

$$\begin{aligned}\Pr_{h^*}\left[\bar{E}_t | S, s_1, \ldots, s_{t-1}\right] &= \Pr_{h^*}\left[\mathrm{err}_{P_t}(h_{s_{t-1}+1}, h^*) > \varepsilon | S, s_1, \ldots, s_{t-1}\right] \\ &\leq \frac{\mathbb{E}_{h^*}\left[\mathrm{err}_{P_t}(h_{s_{t-1}+1}, h^*) | S, s_1, \ldots, s_{t-1}\right]}{\varepsilon} \leq \frac{1}{2}.\end{aligned}$$

By law of total probability,

$$\Pr_{h^*}[s_T \leq T | S] \geq \Pr_{h^*}[E_1 | S] \times \Pr_{h^*}[E_2 | S, E_1] \times \cdots \times \Pr_{h^*}[E_T | S, E_1, \ldots, E_{T-1}] \geq \left(\frac{1}{2}\right)^T > 0.$$

To conclude the proof, we will show that if the reduction algorithm $\mathcal{B}$ sees at most $1/2$ of the support of distribution D_{h^*} through a training set S and makes only $T \leq \frac{\log 2^m}{\log(2/\varepsilon)}$ oracle calls to $\mathcal{A}$, then it will likely fail in robustly learning h^*. For each $i \leq 2^{9m-1}$, conditioned on the event that $\{(x_i^-, -1), (x_i^+, +1)\} \cap S = \emptyset$, and conditioned on $h_{s_1}, \ldots, h_{s_T}$, there is a $z \in Z$ that is equally likely to be in $\mathcal{U}(x_i^-)$ or $\mathcal{U}(x_i^+)$. To see why such a point exists, we first describe an equivalent distribution generating $h^*, h_1, \ldots, h_T$. For each $i \leq 2^{9m-1}$ randomly select a $2\varepsilon'$ fraction of points from G_i^+ and a $2\varepsilon'$ fraction of points from G_i^-. Then, randomly pair the points in each $2\varepsilon'$ fraction to get $\varepsilon' 2^m$ pairs z_i, z_i' for each $G_i^{\pm}$. For each pair z_i, z_i' flip a fair coin c_i: if $c_i = 1$, z_i's label gets flipped and otherwise if $c_i = 0$ then z_i''s label gets flipped. This is equivalent to generating h^* by flipping the labels of a uniform randomly-selected ε fraction of points in each $G_i^{\pm}$ as originally described, but is helpful book-keeping that simplifies our analysis. In addition, $h_1, \ldots, h_T$ can be generated in a similar fashion. Since $T \leq \frac{\log 2^m}{\log(2/\varepsilon)}$, we are guaranteed that $|G_i^{\pm} \cap \mathrm{DIS}_{s_T}| \geq 1$. By definition of DIS_{s_T}, this implies that that there is a pair of points z_i, z_i' in each $G_i^{\pm}$ where each $h_{s_t}(z_i) = h_{s_t}(z_i')$ for $t \leq T$ but $h^*(z_i) \neq h^*(z_i')$ (i.e., each h_{s_t} never reveals the ground-truth label for at least one pair). And then in the end, if $\{(x_i^-, -1), (x_i^+, +1)\} \cap S = \emptyset$, $\mathcal{B}$ will make some

prediction on z_i, and the posterior probability of it being wrong is $1/2$. More formally, for any training dataset $S \sim D_{h^*}^{|S|}$ where $|S| \leq 2^{9m-3}$, any $h_{s_1}, \ldots, h_{s_T}$ returned by $\mathcal{A}$ where $T \leq \frac{\log 2^m}{\log(2/\varepsilon)}$, and any predictor $f : \mathcal{X} \to \mathcal{Y}$ that is picked by $\mathcal{B}$:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{h^*}\left[\mathrm{R}_{\mathcal{U}}(f; D_{h^*}) | S, h_{s_1}, \ldots, h_{s_T}\right] \geq \mathop{\mathbb{E}}_{h^*}\left[\frac{1}{2^{9m}} \sum_{\substack{(x,y)\notin S,\\ (x,y)\in \mathrm{supp}(D_{h^*})}} \sup_{z\in\mathcal{U}(x)} \mathbb{1}[f(z)\neq y] \middle| S, h_{s_1}, \ldots, h_{s_T}\right] \\
&= \frac{1}{2^{9m}} \sum_{i=1}^{2^{9m-1}} \Pr_{h^*}\left[\left((x_i^+, +1), (x_i^-, -1) \notin S\right) \wedge \right. \\
&\qquad\qquad \left. \left(\exists z \in \mathcal{U}(x_i^+) \text{ s.t. } f(z) \neq +1 \vee \exists z \in \mathcal{U}(x_i^-) \text{ s.t. } f(z) \neq -1\right) \middle| S, h_{s_1}, \ldots, h_{s_T}\right] \\
&\geq \frac{2^{9m-1}}{2^{9m}} \frac{1}{2} = \frac{1}{4}.
\end{aligned}$$

This implies that, for any $\mathcal{B}$ limited to $n \leq 2^{9m-3}$ training examples and $T \leq \frac{m}{\log_2(2/\varepsilon)}$ queries, there exists a *deterministic* choice of h^* and $h_1, \ldots, h_T$, and a corresponding learner $\mathcal{A}$ that is a PAC learner for $\{h^*\}$ using hypothesis class $\{h^*, h_1, \ldots, h_T\}$ of VC dimension 1, such that, for $S \sim D_{h^*}^n$, $\mathbb{E}_S[\mathrm{R}_{\mathcal{U}}(f; D_{h^*})] \geq \frac{1}{4}$. □

Computational Efficiency Although the sample complexity of Algorithm 1 is independent of $|\mathcal{U}|$, we showed that the $\log|\mathcal{U}|$ dependence in oracle complexity is unavoidable. This implies that the runtime of Algorithm 1 will be at best weakly polynomial and have at least a $\log|\mathcal{U}|$ dependence. But maybe this is not so bad, because it is equivalent to the number of bits required to represent the adversarial perturbations. This weak poly-time dependence is common in almost all optimization algorithms (gradient descent, interior point methods, etc). What is more concerning is the linear runtime and memory dependence on $|\mathcal{U}|$ that emerges from the explicit representation of the adversarial perturbations during training. In practice, many of the adversarial perturbations $\mathcal{U}$ are infinite, but specified implicitly, and not by enumerating over all possible perturbations (e.g. ℓ_p perturbations). This motivates the following next steps: What operations do we need to be able to implement efficiently on $\mathcal{U}$ in order to robustly learn? What access (oracle calls, or "interface") do we need to $\mathcal{U}$?

Sampling Oracle Over Perturbations A reasonable form of access to $\mathcal{U}$ that is sufficient for implementing Algorithm 1 is a sampling oracle that takes as input a point x and an energy function $E : \mathcal{X} \to \mathbb{R}$, and does the following:

(a) Samples a perturbation z from a distribution given by $p_x(z) \propto \exp(E(z)) * \mathbb{1}[z \in \mathcal{U}(x)]$. That is, the oracle samples from the set $\mathcal{U}(x)$ based on the weighting encoded in E.
(b) Calculates $\Pr[z \in \mathcal{U}(x)]$ for the distribution given by $p(z) \propto \exp(E(z))$.

With such an oracle, Algorithm 1 can be implemented without the need to do explicit inflation of S to $S_{\mathcal{U}}$, and can avoid the linear dependence on $|\mathcal{U}|$. This is because Algorithm 1 and its subprocedure `ZeroRobustLoss` just need to sample from distributions over the inflated set $S_{\mathcal{U}}$ that are constructed by α-Boost (as required in Steps 5 and 17 in Algorithm 1). This can be simulated via a two-stage process where we maintain a conditional distribution over S (the original points), and then draw perturbations using the sampling oracle. Specifically, to sample from a distribution D_t that is constructed by α-Boost, we use two energy functions $E_t^+(z) = -2\alpha \sum_{i\leq t} \mathbb{1}[g_t(z) = +1]$ and $E_t^-(z) = -2\alpha \sum_{i\leq t} \mathbb{1}[g_t(z) = -1]$, where $g_1, \ldots, g_t$ represent the sequence of predictors produced during the first t rounds of boosting (either h_t's produced by non-robust learner $\mathcal{A}$ or f_t's produced by `ZeroRobustLoss`). Using the sampling oracle, we can sample from D_t, by first sampling (x, y) from S based on the marginal estimates computed by the oracle (operation (b) described above) using energy function E_t^y, and then sampling z's from their $\mathcal{U}(x)$ (operation (a) described above) using energy function E_t^y.

4 Learning with a Mistake Oracle for Adversarial Perturbations

In Section 3, we considered an explicit form of access to the set of adversarial perturbations $\mathcal{U}$, as well as access via a sampling oracle. A more realistic form of access is having a mistake oracle for $\mathcal{U}$:

Definition 4.1 (Mistake Oracle)**.** *Denote by $\mathsf{O}_{\mathcal{U}}$ a mistake oracle for $\mathcal{U}$. $\mathsf{O}_{\mathcal{U}}$ takes as input a predictor $f : \mathcal{X} \to \mathcal{Y}$ and an example (x, y) and either: (a) asserts that f is robust on (x, y) (i.e. $\forall z \in \mathcal{U}(x), f(z) = y$), or (b) returns an adversarial perturbation $z \in \mathcal{U}(x)$ such that $f(z) \neq y$.*

Having only a mistake oracle for $\mathcal{U}$, rather than an explicit representation, is a more realistic form of access. In this case, the reduction algorithm has no explicit knowledge of the set of adversarial perturbations $\mathcal{U}$ and is forced to interact with the mistake oracle $\mathsf{O}_{\mathcal{U}}$ in order to learn an adversarially robust predictor. Furthermore, what is typically referred to as adversarial training in practice fits exactly into this framework [Madry et al., 2018].

Does access to a mistake oracle $\mathsf{O}_{\mathcal{U}}$ suffice to robustly learn a target class $\mathcal{C}$ using a black-box non-robust learner $\mathcal{A}$ for $\mathcal{C}$? First, can we achieve a similar upper bound as in Theorem 3.1 but with a mistake oracle $\mathsf{O}_{\mathcal{U}}$ rather than explicit access to $\mathcal{U}$? Unfortunately, even with a black-box ERM for $\mathcal{C}$, one can show that $|\mathcal{U}|$ oracle calls to $\mathsf{O}_{\mathcal{U}}$ are unavoidable (proof provided in Appendix A):

Claim 4.2. *For any reduction algorithm $\mathcal{B}$, there exists an adversary $\mathcal{U}$, target class $\mathcal{C}$, and an* ERM *for $\mathcal{C}$ with VC dimension 1, such that $\mathcal{B}$ needs to make $T \geq |\mathcal{U}|$ oracle calls to $\mathsf{O}_{\mathcal{U}}$.*

Thus, a non-robust PAC learner $\mathcal{A}$ for $\mathcal{C}$ is not enough to learn $\mathcal{C}$ robustly with a mistake oracle $\mathsf{O}_{\mathcal{U}}$ *without a linear dependence on* $|\mathcal{U}|$. This suggests that a stronger assumption about $\mathcal{A}$ is required. We next show that an *online* learner $\mathcal{A}$ for $\mathcal{C}$ suffices to robustly learn $\mathcal{C}$ with a mistake oracle $\mathsf{O}_{\mathcal{U}}$ *and* without any dependence on $|\mathcal{U}|$. Before proceeding, we include a brief reminder of what it means to learn in an online setting with a finite mistake bound:

Definition 4.3. *(Mistake Bound Model) We say an online learner $\mathcal{A}$ learns a hypothesis class $\mathcal{C}$ with mistake bound $M_{\mathcal{A}}$ if learner $\mathcal{A}$ makes at most $M_{\mathcal{A}}$ mistakes on any sequence of examples that are labeled with some concept $c \in \mathcal{C}$.*

We are now ready to state our main result for this section.

Theorem 4.4. *Algorithm 3 robustly PAC learns any target class $\mathcal{C}$ w.r.t. an adversary $\mathcal{U}$ with black-box access to a mistake oracle $\mathsf{O}_{\mathcal{U}}$ and an online learner $\mathcal{A}$ for $\mathcal{C}$ with sample complexity, number of calls to $\mathcal{A}$, and number of calls to $\mathsf{O}_{\mathcal{U}}$ that is at most $2\frac{M_{\mathcal{A}}}{\varepsilon}\log\left(\frac{M_{\mathcal{A}}}{\delta}\right)$, where $M_{\mathcal{A}}$ is the mistake-bound of online learner $\mathcal{A}$.*

Proof sketch. Run the online learner $\mathcal{A}$ on the sequence of input examples using the mistake oracle $\mathsf{O}_{\mathcal{U}}$ to find mistakes. Details and algorithm are provided in Appendix A. □

Example 4.5. *Let $\mathcal{C}$ be the class of OR functions over the boolean hypercube $\{0,1\}^n$. There is an online learner $\mathcal{A}$ that learns $\mathcal{C}$ with a mistake bound $M_{\mathcal{A}} = n$. Theorem 4.4 implies that we can robustly learn $\mathcal{C}$ using $\mathcal{A}$ with sample complexity, number of calls to $\mathcal{A}$, and number of calls to $\mathsf{O}_{\mathcal{U}}$ that is at most $2\frac{n}{\varepsilon}\log\left(\frac{n}{\delta}\right)$.*

5 Discussion

The main contribution of this paper is in formulating the question of reducing adversarially robust learning to standard non-robust learning and providing answers in some settings. We outline a few directions for furture work below.

Mistake Oracle for $\mathcal{U}$ This is a more challenging setting (but perhaps more realistic) where the reduction algorithm has no knowledge of $\mathcal{U}$ and can only interact with a mistake oracle for $\mathcal{U}$. Theorem 4.4 shows that online learnability is sufficient for robust learning in this model. Beyond this, are there weaker conditions that would enable robust learning under this model? Or is having an online learner essential? What if we consider specific target classes? Montasser et al. [2020] recently gave an algorithm that robustly learns halfspaces in this model. A natural next step is to ask which other classes can be robustly learned in this model, or more ambitiously characterize a necessary and sufficient condition for learning in this model.

Agnostic Setting We focused only on robust PAC learning in the realizable setting, where we assume there is a $c \in \mathcal{C}$ with zero robust error. It would be desirable to extend our results also to the agnostic setting, where we want to compete with the best $c \in \mathcal{C}$. We remark that an agnostic-to-realizable reduction described in Montasser et al. [2019, Theorem 6] can be used in our setting, however, it has runtime that is exponential in $\mathrm{vc}(\mathcal{A})$. Another attempt through the agnostic boosting frameworks [e.g. Kalai and Kanade, 2009] requires a non-robust PAC learner $\mathcal{A}$ with error ε that scales with $|\mathcal{U}|^2$, which results in a sample complexity that depends on $|\mathcal{U}|$, and this is something we would like to avoid.

Boosting and Robustness Boosting has led to many exciting developments in theory and practice of machine learning. It started with asking: Can we boost the accuracy of weak predictors to attain a predictor with high accuracy? Freund and Schapire [1997] showed that boosting the accuracy is possible and can be done efficiently. What we consider in this paper can be viewed as a question of boosting robustness: Can we boost non-robust predictors to attain a *robust* predictor? and can we do this efficiently? Another natural question to consider which we did not study in this paper is: Can we boost *weakly* robust predictors to attain a *robust* predictor?

Broader Impact

Learning predictors that are robust to adversarial perturbations is an important challenge in contemporary machine learning. Current machine learning systems have been shown to be brittle against different notions of robustness such as adversarial perturbations [Szegedy et al., 2013, Biggio et al., 2013, Goodfellow et al., 2014], and there is an ongoing effort to devise methods for learning predictors that *are* adversarially robust. As machine learning systems become increasingly integrated into our everyday lives, it becomes crucial to provide guarantees about their performance, even when they are used outside their intended conditions.

We already have many tools developed for standard learning, and having a universal *wrapper* that can take any standard learning method and turn it into a *robust* learning method could greatly simplify the development and deployment of learning that is *robust* to test-time adversarial perturbations. The results that we present in this paper are still mostly theoretical, and limited to the realizable setting, but we expect and hope they will lead to further theoretical study as well as practical methodological development with direct impact on applications.

In this work we do not deal with training-time adversarial attacks, which is a major, though very different, concern in many cases.

As with any technology, having a more robust technology can have positive and negative societal consequences, and this depends mainly on how such technology is utilized. Our intent from this research is to help with the design of robust machine learning systems for application domains such as healthcare and transportation where its critical to ensure performance guarantees even outside intended conditions. In situations where there is a tradeoff between robustness and accuracy, this work might be harmful in that it would prioritize robustness over accuracy and this may not be ideal in some application domains.

Acknowledgments and Disclosure of Funding

We would like to thank Shay Moran for the insightful discussions that led to the formalization of the question we study in this paper. We also thank the anonymous reviewers for their thoughtful and helpful feedback. This work is partially supported by DARPA[1] cooperative agreement HR00112020003.

References

P. Assouad. Densité et dimension. *Annales de l'Institut Fourier (Grenoble)*, 33(3):233–282, 1983.

[1]This paper does not reflect the position or the policy of the Government, and no endorsement should be inferred.

Idan Attias, Aryeh Kontorovich, and Yishay Mansour. Improved generalization bounds for robust learning. In Aurélien Garivier and Satyen Kale, editors, *Algorithmic Learning Theory, ALT 2019, 22-24 March 2019, Chicago, Illinois, USA*, volume 98 of *Proceedings of Machine Learning Research*, pages 162–183. PMLR, 2019. URL http://proceedings.mlr.press/v98/attias19a.html.

Maria-Florina Balcan. Lecture notes - machine learning theory, January 2010.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery*, 36(4):929–965, 1989.

Daniel G Brown. How I wasted too long finding a concentration inequality for sums of geometric variables.

Sebastien Bubeck, Yin Tat Lee, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. In *International Conference on Machine Learning*, pages 831–840, 2019.

A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3):247–261, 1989.

Uriel Feige, Yishay Mansour, and Robert E. Schapire. Learning and inference in the presence of corrupted inputs. In Peter Grünwald, Elad Hazan, and Satyen Kale, editors, *Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015*, volume 40 of *JMLR Workshop and Conference Proceedings*, pages 637–657. JMLR.org, 2015. URL http://proceedings.mlr.press/v40/Feige15.html.

Uriel Feige, Yishay Mansour, and Robert E. Schapire. Robust inference for multiclass classification. In Firdaus Janoos, Mehryar Mohri, and Karthik Sridharan, editors, *Algorithmic Learning Theory, ALT 2018, 7-9 April 2018, Lanzarote, Canary Islands, Spain*, volume 83 of *Proceedings of Machine Learning Research*, pages 368–386. PMLR, 2018. URL http://proceedings.mlr.press/v83/feige18a.html.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997. doi: 10.1006/jcss.1997.1504. URL https://doi.org/10.1006/jcss.1997.1504.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6572.

Adam Kalai and Varun Kanade. Potential-based agnostic boosting. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, editors, *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*, pages 880–888. Curran Associates, Inc., 2009. URL http://papers.nips.cc/paper/3676-potential-based-agnostic-boosting.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Yishay Mansour, Aviad Rubinstein, and Moshe Tennenholtz. Robust probabilistic inference. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 449–460. SIAM, 2015. doi: 10.1137/1.9781611973730.31. URL https://doi.org/10.1137/1.9781611973730.31.

Omar Montasser, Steve Hanneke, and Nathan Srebro. Vc classes are adversarially robustly learnable, but only improperly. In Alina Beygelzimer and Daniel Hsu, editors, *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 2512–2530, Phoenix, USA, 25–28 Jun 2019. PMLR.

Omar Montasser, Surbhi Goel, Ilias Diakonikolas, and Nati Srebro. Efficiently learning adversarially robust halfspaces with noise. In *Proceedings of Machine Learning and Systems 2020*, pages 10630–10641. 2020.

S. Moran and A. Yehudayoff. Sample compression schemes for VC classes. *Journal of the ACM*, 63 (3):21:1–21:10, 2016.

Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J Zico Kolter. Black-box smoothing: A provable defense for pretrained classifiers. *arXiv preprint arXiv:2003.01908*, 2020.

R. E. Schapire and Y. Freund. *Boosting*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2012.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition*. Nauka, Moscow, 1974.

A Appendix

Proof of Lemma 3.2. Consider a *dual space* $\bar{\mathcal{G}}$: a set of functions $\bar{g}_x : \mathrm{co}^k(\mathcal{H}) \to \mathcal{Y}$ defined as $\bar{g}_x(f) = f(x)$ for each $f = \mathrm{MAJ}(h_1, \ldots, h_k) \in \mathrm{co}^k(\mathcal{H})$ and each $x \in \mathcal{X}$. It follows by definition of dual VC dimension that $\mathrm{vc}(\bar{\mathcal{G}}) = \mathrm{vc}^*(\mathrm{co}^k(\mathcal{H}))$. Similarly, define another dual space $\mathcal{G}$: a set of functions $g : \mathcal{H} \to \mathcal{Y}$ defined as $g(x) = h(x)$ for each $h \in \mathcal{H}$ and each $x \in \mathcal{X}$. We know that $\mathrm{vc}(\mathcal{G}) = \mathrm{vc}^*(\mathcal{H}) = d^*$. Observe that by definition of $\mathcal{G}$ and $\bar{\mathcal{G}}$, we have that for each $x \in \mathcal{X}$ and each $f = \mathrm{MAJ}(h_1, \ldots, h_k) \in \mathrm{co}^k(\mathcal{H})$,

$$\bar{g}_x(f) = f(x) = \mathrm{MAJ}(h_1, \ldots, h_k)(x) = \mathrm{sign}\left(\sum_{i=1}^{k} h_i(x)\right) = \mathrm{sign}\left(\sum_{i=1}^{k} g_x(h_i)\right).$$

By the Sauer-Shelah Lemma applied to dual class $\mathcal{G}$, for any set $H = \{h_1, \ldots, h_n\} \subseteq \mathcal{H}$, the number of possible behaviors

$$|\mathcal{G}|_H| := |\{(g_x(h_1), \ldots, g_x(h_n)) : x \in \mathcal{X}\}| \leq \binom{n}{\leq d^*}. \tag{3}$$

Consider a set $F = \{f_1, \ldots, f_m\} \subseteq \mathrm{co}^k(\mathcal{H})$, the number of possible behaviors can be upperbounded as follows:

$$\begin{aligned}
\left|\bar{\mathcal{G}}|_F\right| &= |\{(\bar{g}_x(f_1), \ldots, \bar{g}_x(f_m)) : x \in \mathcal{X}\}| \\
&= \left|\left\{(\bar{g}_x(\mathrm{MAJ}(h_1^1, \ldots, h_1^k)), \ldots, \bar{g}_x(\mathrm{MAJ}(h_m^1, \ldots, h_m^k))) : x \in \mathcal{X}\right\}\right| \\
&= \left|\left\{\left(\mathrm{sign}\left(\textstyle\sum_{i=1}^k g_x(h_i)\right), \ldots, \mathrm{sign}\left(\textstyle\sum_{i=1}^k g_x(h_i)\right)\right) : x \in \mathcal{X}\right\}\right| \\
&\overset{(i)}{\leq} \left|\left\{(g_x(h_1^1), \ldots, g_x(h_1^k), g_x(h_2^1), \ldots, g_x(h_2^k), \ldots, g_x(h_m^1), \ldots, g_x(h_m^k)) : x \in \mathcal{X}\right\}\right| \\
&\overset{(ii)}{\leq} \binom{mk}{\leq d^*},
\end{aligned}$$

where (i) follows from observing that each expanded vector $(g_x(h_i^1), \ldots, g_x(h_i^k))_{i=1}^m \in \mathcal{Y}^{mk}$ can map to at most one vector $\left(\mathrm{sign}\left(\sum_{i=1}^k g_x(h_i)\right), \ldots, \mathrm{sign}\left(\sum_{i=1}^k g_x(h_i)\right)\right) \in \mathcal{Y}^m$, and (ii) follows from Equation 3. Observe that if $\left|\bar{\mathcal{G}}|_F\right| < 2^m$, then by definition, F is not shattered by $\bar{\mathcal{G}}$, and this implies that $\mathrm{vc}(\bar{\mathcal{G}}) < m$. Thus, to conclude the proof, we need to find the smallest m such that $\binom{mk}{\leq d^*} < 2^m$. It suffices to check that $m = O(d^* \log k)$ satisfies this condition. □

Lemma A.1 (Montasser et al. [2019])**.** *For any $k \in \mathbb{N}$ and fixed function $\phi : (\mathcal{X} \times \mathcal{Y})^k \to \mathcal{Y}^{\mathcal{X}}$, for any distribution P over $\mathcal{X} \times \mathcal{Y}$ and any $m \in \mathbb{N}$, for $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ iid P-distributed random variables, with probability at least $1 - \delta$, if $\exists i_1, \ldots, i_k \in \{1, \ldots, m\}$ s.t. $\hat{R}_{\mathcal{U}}(\phi((x_{i_1}, y_{i_1}), \ldots, (x_{i_k}, y_{i_k})); S) = 0$, then*

$$R_{\mathcal{U}}(\phi((x_{i_1}, y_{i_1}), \ldots, (x_{i_k}, y_{i_k})); P) \leq \frac{1}{m-k}(k \ln(m) + \ln(1/\delta)).$$

Proof sketch of Claim 4.2. Let $\mathcal{B}$ be an arbitrary reduction algorithm. Let $x_0, x_1 \in \mathcal{X}$, and $k \in \mathbb{N}$. Pick arbitrary points $Z = \{z_1, \ldots, z_{2k}\} \subseteq \mathcal{X}$. Let $X = \{x_0, x_1\} \cup Z$. Let $b \in \{0,1\}^{2k}$ be a bit string drawn uniformly at random from the set $\left\{b \in \{0,1\}^{2k} : \sum_i b_i = k\right\}$, think of this as a random partition of Z into two equal sets Z_0 and Z_1. For each $y \in \{0,1\}$, define $\mathcal{U}_b(x_y)$ to include x_y and all perturbations $z \in Z_y$. Also, foreach $z \in Z$ define $\mathcal{U}_b(z) = \{z\}$. Similarly, define target class $\mathcal{C}_b$ to include only a single hypothesis c_b where $c_b(\mathcal{U}(x_0)) = 0$ and $c_b(\mathcal{U}(x_1)) = 1$. We will consider an ERM that uses the set of thresholds $\mathcal{H}_\phi = \{x \mapsto \mathbb{1}[\phi(x) \geq \theta] : \theta \in \mathbb{R}\}$ as its hypothesis class, where ϕ is a random embedding such that for each $z_0 \in \mathcal{U}_b(x_0)$ and each $z_1 \in \mathcal{U}_b(x_1)$: $\phi(z_0) < \phi(z_1)$; this guarantees that the random hypothesis c_b is realized by some $h \in \mathcal{H}_\phi$. On any input $L \subseteq X \times \{0,1\}$, we define the ERM to return the earliest possible threshold that reveals as few 0's as possible.

Since algorithm $\mathcal{B}$ only sees training data $S = \{(x_0, 0), (x_1, 1)\}$ as its input, by picking b uniformly at random, $\mathcal{B}$ has no way of knowing which perturbations belong to $\mathcal{U}(x_0)$ and which belong to

$\mathcal{U}(x_1)$, and therefore its forced to call the mistake oracle $\mathsf{O}_{\mathcal{U}}$ at least k times. The ERM oracle is designed such that it will reveal as little information about this as possible.

Suppose that we run algorithm $\mathcal{B}$ for T rounds, where in each round $t \leq T$, $\mathcal{B}$ maintains a predictor $f_t : X \to \{0,1\}$ that determines that labeling of x_0, x_1 and the set of perturbations Z. We will show that, in expectation over the random choice of b and ϕ, in order for the final predictor f_T outputted by $\mathcal{B}$ to have robust loss zero on S, i.e. $\mathrm{R}_{\mathcal{U}_b}(f_T) = 0$, the number of rounds T needs to be at least k.

On each round $t \leq T$, $\mathcal{B}$ is allowed to:

1. Query the mistake oracle $\mathsf{O}_{\mathcal{U}}$ with a query consisting of some predictor $g_t : X \to \{0,1\}$ and a point $(x, y) \in X \times \{0,1\}$.
2. Query the ERM oracle with a dataset $L_t \subseteq X \times \{0,1\}$.

Let $M_t = \sum_{z \in Z} \mathbb{1}[f_t(z) \neq c_b(z)]$ be the number of mistakes at round t, and let $H_t = \{g_j, (x_j, y_j), L_j\}_{j \leq t}$ denote the history of queries. Then, observe that

$$\mathop{\mathbb{E}}_{b,\phi}[M_t | M_{t-1}, H_{t-1}] \geq M_{t-1} - 1.$$

This is because oracle $\mathsf{O}_{\mathcal{U}}$ reveals the ground truth label of at most 1 point at round t, and the ERM will move the threshold by at most one position. This implies that $\mathbb{E}_{b,\phi}[M_T | M_0, H_0] \geq M_0 - T$. We can further condition on the event that $M_0 \geq k$ which has non-zero probability (since b is picked uniformly at random). This implies, by the probabilistic method, that there exists b, ϕ such that for $T \leq k-1$, $M_T \geq 1$. Therefore, by definition of M_T, f_T is not be robustly correct on S for $T \leq k-1$. □

Proof of Theorem 4.4. Let $\mathcal{U}$ be an arbitrary adversary and $\mathsf{O}_{\mathcal{U}}$ its corresponding mistake oracle. Let $\mathcal{C} \subseteq \mathcal{Y}^{\mathcal{X}}$ be an arbitrary target class, and $\mathcal{A}$ an online learner for $\mathcal{C}$ with mistake bound $M_{\mathcal{A}} < \infty$. We assume w.l.o.g. that the online learner $\mathcal{A}$ is conservative, meaning that it does not update its state unless it makes a mistake. Algorithm 3 in essence is a standard conversion of a learner in the mistake bound model to a learner in the PAC model (see e.g. Balcan [2010]):

Algorithm 3: Robust Learner with a Mistake Oracle.

Input: $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, ε, δ, black-box access to a an online learner $\mathcal{A}$, black-box access to a mistake oracle $\mathsf{O}_{\mathcal{U}}$

1 Initialize $h_0 = \mathcal{A}(\emptyset)$.
2 **for** $i \leq m$ **do**
3 Certify the robustness of h on (x_i, y_i) by asking the mistake oracle $\mathsf{O}_{\mathcal{U}}$.
4 If h_t is not robust on (x_i, y_i), update h_t by running $\mathcal{A}$ on (z, y_i), where z is the perturbation returned by $\mathsf{O}_{\mathcal{U}}$.
5 Break when h_t is robustly correct on a consecutive sequence of length $\frac{1}{\varepsilon}\log\left(\frac{M_{\mathcal{A}}}{\delta}\right)$.

Output: h_t.

Analysis Let $\mathcal{D}$ be an arbitrary distribution over $\mathcal{X} \times \mathcal{Y}$ that is robustly realizable with some concept $c \in \mathcal{C}$,i.e., $\mathrm{R}_{\mathcal{U}}(c; \mathcal{D}) = 0$. Fix $\varepsilon, \delta \in (0,1)$ and a sample size $m = 2\frac{M_{\mathcal{A}}}{\varepsilon}\log\left(\frac{M_{\mathcal{A}}}{\delta}\right)$.

Since online learner $\mathcal{A}$ has a mistake bound of $M_{\mathcal{A}}$, Algorithm 3 will terminate in at most $\frac{M_{\mathcal{A}}}{\varepsilon}\log\left(\frac{M_{\mathcal{A}}}{\delta}\right)$ steps of certification, which of course is an upperbound on the number of calls to the mistake oracle $\mathsf{O}_{\mathcal{U}}$, and the number of calls to the online learner $\mathcal{A}$.

It remains to show that the output of Algorithm 3, the final predictor h, has low robust risk $\mathrm{R}_{\mathcal{U}}(h; \mathcal{D})$. Throughout the runtime of Algorithm 3, the online learner can generate a sequence of at most $M_{\mathcal{A}} + 1$ predictors. There's the initial predictor from Step 1, plus the $M_{\mathcal{A}}$ updated predictors corresponding to potential updates by online learner $\mathcal{A}$. Observe that the probability that the final h has robust risk more than ε

$$\Pr_{S \sim \mathcal{D}^m}[\mathrm{R}_{\mathcal{U}}(h; \mathcal{D}) > \varepsilon] \leq \Pr_{S \sim \mathcal{D}^m}[\exists j \in [M_{\mathcal{A}}+1] \text{ s.t. } \mathrm{R}_{\mathcal{U}}(h_j; \mathcal{D}) > \varepsilon] \leq (M_{\mathcal{A}}+1)(1-\varepsilon)^{\frac{1}{\varepsilon}\log\left(\frac{M_{\mathcal{A}}+1}{\delta}\right)} \leq \delta.$$

Therefore, with probability at least $1 - \delta$ over $S \sim \mathcal{D}^m$, Algorithm 3 outputs a predictor h with robust risk $\mathrm{R}_{\mathcal{U}}(h; \mathcal{D}) \leq \varepsilon$. Thus, Algorithm 3 robustly PAC learns $\mathcal{C}$ w.r.t. adversary $\mathcal{U}$. □