Constructing a provably adversarially-robust classifier from a high accuracy one

Grzegorz Głuch
EPFL

Rüdiger Urbanke
EPFL

Abstract

Modern machine learning models with very high accuracy have been shown to be vulnerable to small, adversarially chosen perturbations of the input. Given black-box access to a high-accuracy classifier f, we show how to construct a new classifier g that has high accuracy and is also robust to adversarial L2-bounded perturbations. Our algorithm builds upon the framework of randomized smoothing that has been recently shown to outperform all previous defenses against L2-bounded adversaries. Using techniques like random partitions and doubling dimension, we are able to bound the adversarial error of g in terms of the optimum error. In this paper we focus on our conceptual contribution, but we do present two examples to illustrate our framework. We will argue that, under some assumptions, our bounds are optimal for these cases.

1 INTRODUCTION

Modern neural networks achieve high accuracy on tasks such as image classification (Krizhevsky et al. (2012)) or speech recognition (Collobert and Weston (2008)) but have been shown to be susceptible to small, adversarially-chosen perturbations of the inputs (Szegedy et al. (2014), Nguyen et al. (2015), Biggio et al. (2013)): given an input x, which is correctly classified by a neural network, one is often able to find a small perturbation δ such that $x + \delta$ is misclassified by the network, whereas x and $x + \delta$ are virtually indistinguishable to the human eye.

Many empirical approaches have been proposed for building "robust" classifiers. One of the most successful ones is the framework of *adversarial training* (Goodfellow et al. (2015), Kurakin et al. (2017), Madry et al. (2017)). Unfortunately these techniques usually protect only against restricted types of adversaries. Moreover, many of the heuristic defenses were shown to break in the presence of suitably powerful adversaries (Carlini and Wagner (2017), Athalye et al. (2018), Uesato et al. (2018)).

Certifiable robust classifiers, on the other hand, are classifiers whose predictions are verifiably constant within a neighborhood of a query point. The first such classifiers were introduced by Raghunathan et al. (2018) and Wong and Kolter (2018). *Randomized smoothing* was considered in Lécuyer et al. (2019), Li et al. (2018), Cohen et al. (2019) and Salman et al. (2019). It works as follows.

Let f be any classifier which maps $\mathbb{R}^d$ to classes $\mathcal{Y}$. The smoothed classifier g classifies an input x as that class c that is most likely to be returned by f on input $x + \delta$, where $\delta \sim \mathcal{N}(0, \sigma^2 I)$.

It was shown in Lécuyer et al. (2019) that this approach scales well and one can use it to train certifiably robust classifier for ImageNet. In Cohen et al. (2019) it is shown that for ℓ_2 perturbations *randomized smoothing* outperforms other certifiably defenses previously proposed. Moreover the authors show how to derive a robustness radius guarantee for an input x. To derive the bound one defines for a class $c \in \mathcal{Y}$ the probability $p_c := \mathbb{P}_\delta(f(x + \delta) = c)$, where the perturbation δ is chosen according to $\delta \sim \mathcal{N}(0, \sigma^2 I)$. Then one argues that if there exists a c such that $p_c \gg \max_{c' \neq c} p_{c'}$ then the robustness radius at x is big. Unfortunately, even if the base classifier f has very high accuracy we don't know much about the structure of $\{p_c\}_{c \in \mathcal{Y}}$. Thus it's hard to reason about the robustness radii. These short-

Proceedings of the 23rdInternational Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

comings point to the following question:

Having a black-box access to a high accuracy classifier f is it possible to construct a new classifier g that is guaranteed to be both robust and achieve high accuracy?

Note that robustness without an accuracy constraint is trivially achieved by a constant classifier, and high accuracy without a robustness constraints has also been shown to be achievable in many settings of interest. The real question of interest therefore only appears if we require both types of constraints.

Our contributions: We show a framework for transforming **any** high accuracy classifier f into a provably robust and high accuracy classifier g. Moreover we show what the optimal classifier for a given learning task is and then relate the performance of g to this optimum. We present two instances of this framework. To keep the exposition simple, we limit our setting to ℓ_2-robustness. The ideas apply more generally, but the details differ.

In the first instance we show that if f satisfies a suitable property (similar to a property implicitly assumed in Cohen et al. (2019) and Salman et al. (2019)) then g can be evaluated with only black-box access to f.

In the second instance we prove that, without any assumptions on f, a robust classifier g can be evaluated if we also have access to an oracle $\mathcal{O}$ that provides *unlabeled* i.i.d. samples from the underlying distribution. Notice that this model is not very restrictive. A similar setting occurs in semi-supervised learning where the learner has access to a dataset of labeled data D_l and also to (an often much larger) dataset D_u of unlabeled samples (see Chapelle et al. (2010)). In this scenario D_u serves as the oracle $\mathcal{O}$.

Even though our main contribution is a conceptual one, we also present two implementations of these methods that achieve different runtime/robustness tradeoffs. In the end we give examples of binary classification tasks (e.g. adversarial spheres from Gilmer et al. (2018)) and compare the performance of our methods on these tasks to the optimum.

2 OUR TECHNIQUES

Let us present an overview of our approach.

2.1 Randomized smoothing

Our techniques build upon *randomized smoothing* from Lécuyer et al. (2019), Li et al. (2018), Cohen et al. (2019) and Salman et al. (2019). Consider a classifier f that maps $\mathbb{R}^d$ to classes $\mathcal{Y}$. *Randomized smoothing* is a method that produces a new, *smoothed* classifier g. The smoothed classifier g assigns to a query point x the class that is most likely to be returned by f under random Gaussian noise:

$$g(x) := \arg\max_{c\in\mathcal{Y}} \mathbb{P}[f(x+\delta)=c], \delta\sim\mathcal{N}(0,\sigma^2 I). \quad (1)$$

Note that g can also be expressed as:

$$g(x) = \arg\max_{c\in\mathcal{Y}} \int_{\mathbb{R}^d} \mathbb{1}_{\{f(x)=c\}}\gamma(x-z)dz, \quad (2)$$

where γ is the density function of $\mathcal{N}(0,\sigma^2 I)$.

Unfortunately it is easy to design a learning task and a classifier f with low standard error such that g, computed according to (1), has high error. For instance, imagine the following binary classification task in $\mathbb{R}^2$. We generate $x\in\mathbb{R}^2$ uniformly at random from a union of two discs B_-, B_+ of radius 1 centered at $(-2,0)$ and $(2,0)$, respectively. We assign the label $y=-1$ if x belongs to B_- and the label $y=+1$ otherwise. Let $f(x)=-1$ if $x\in B_-$ and $f(x)=+1$ otherwise (i.e., for all points $x\notin B_-$). Observe that f has error equal 0. If we now compute g according to (1), then $g(x)=+1$ for all x if $\sigma \geq 1/(\sqrt{2}\ \mathrm{InvErfc}(\frac{1}{2}))\sim 1.4826$. This means that g has an error of $\frac{1}{2}$.

The reason that we were able to construct such an example is that in (1) the smoothing is performed independent of the data. A natural idea to fix this is to perform the smoothing "conditioned" on the data distribution. For instance, in the example above we would like to not take points outside $B_-\cup B_+$ into account during smoothing. The formal definition of this approach is as follows:

$$g(x) := \arg\max_{c\in\mathcal{Y}} \int_{\mathbb{R}^d} \mathbb{1}_{\{f(x)=c\}}\gamma(x-z)p_X(z)dz, \quad (3)$$

where p_X is the density function of the data distribution. Notice the difference between (3) and (2). Unfortunately we can construct "counter examples" even for this modification (see next section).

2.2 Hard distribution for randomized smoothing described in (3)

It is possible to create a separable, binary classification task on $\mathbb{R}^d$ and a classifier f such that the standard error of f is $e^{-\Theta(d)}$ but the error of the smoothed classifier g is $\Theta(1)$ (see Appendix A for details). That is, the standard error grows by a factor $e^{\Theta(d)}$ when we perform smoothing!

This example shows that when we use *randomized smoothing* then already the *standard* error can grow by a factor exponential in the dimension of the ambient space. As we aim for creating g that is robust **and** has small error we must try something different.

2.3 Partitions

The intuitive reason why we were able to construct the example in the previous section is that in *randomized smoothing* it might happen that 1 misclassified point of f contributes to $e^{\Theta(d)}$ misclassified points of g. To prevent that we use space partitions.

Assume that in a binary classification task the distance between the two classes is at least ϵ. Assume further that we partition $\mathbb{R}^d$ into sets $S_1, S_2, \ldots$, each of diameter at most ϵ. Now for $x \in \mathbb{R}^d$ we define $g(x)$ as the class that is most likely returned by f **on points sampled from the data distribution conditioned on being in set S_i to which x belongs**. As the classes are at least ϵ away from each other and the diameters of the sets in the partition are at most ϵ, each misclassified point of f contributes to at most 2 misclassified points of g (this will be proven in Lemma 5). This means that the error of g is bounded in terms of the error of f.

But we also want g to be robust. Intuitively we want a big fraction of points to be far from the boundaries of sets $S_1, S_2, \ldots$. To do that we use *padded random partitions*, which previously found applications in low distortion embeddings (Gupta et al. (2003)), locality sensitive hashing (Andoni and Indyk (2008)) and even spectral algorithms (Lee et al. (2014)). Definitions of random and padded partitions are presented in Section 5.

2.4 Doubling dimension

Some random partitions suffer from the big dimension of the space $\mathbb{R}^d$. To improve the guarantees for binary classification tasks that have data lying on lower dimensional manifolds we resort to the notion of *doubling dimension* (Section 4). This definition captures the intuition that it should be easier to "describe" a manifold that is lower dimensional.

2.5 Examples

In Section 9 we give two examples to analyze the tightness of the bounds obtained in Section 7. The first one is a data distribution from Gilmer et al. (2018). For this example we show that our approach is competitive against a certain class of classifiers (see Section 9.1 for an in-depth discussion). The second example is a data distribution supported on two low dimensional manifolds embedded in high-dimensional space for which we show optimality of our method up to constant factors.

3 PRELIMINARIES

For a distribution $\mathcal{D}$ over $\mathbb{R}^d$ and for a set $A \subseteq \mathbb{R}^d$ let $\mu(A) := \mathbb{P}_X(X \in A)$. For us, $\mathcal{D}$ will denote the distribution of the data. For simplicity in this section and the rest of the paper we consider only *separable* binary classification tasks. Such tasks are fully specified by $\mathcal{D}$ as well as a *ground truth* $h : \mathbb{R}^d \to \{-1, 1\}$. We note however that one can generalize the results to any binary classification task (see Appendix B for a generalization of the definitions from this section). For $x \in \mathbb{R}^d$ and $\epsilon > 0$ we write $B_\epsilon(x)$ to denote the *open ball* with center x and radius ϵ. Most of the proofs are deferred to the Appendix D.

Definition 1. (**Risk**) Consider a binary classification task for separable classes with a ground truth $h : \mathbb{R}^d \to \{-1, 1\}$. For a classifier $f : \mathbb{R}^d \to \{-1, 1\}$ we define the **R**isk as

$$R(f) := \mathbb{P}_X(f(X) \neq h(X)).$$

Definition 2. (**Adversarial Risk**) Consider a binary classification task for separable classes with a ground truth $h : \mathbb{R}^d \to \{-1, 1\}$. For a classifier $f : \mathbb{R}^d \to \{-1, 1\}$ and $\epsilon \in \mathbb{R}_{\geq 0}$ we define the **A**dversarial **R**isk as

$$AR(f, \epsilon) := \mathbb{P}_X(\exists\ \eta \in B_\epsilon\ f(X + \eta) \neq h(X)).$$

We also introduce the notation:

$$AR(\epsilon) := \inf_f AR(f, \epsilon).$$

to denote the smallest achievable adversarial risk for that classification task with a given ϵ.

Fact 1. ***(R versus AR)***

- $AR(f, 0) = R(f)$,
- *$AR(f, \epsilon)$ and $AR(\epsilon)$ are nondecreasing functions of ϵ; combined with the previous point this implies that for $\epsilon \in \mathbb{R}_{\geq 0}$, $AR(f, \epsilon) \geq R(f)$,*
- $AR(f, \epsilon) \leq R(f) + \mathbb{P}_{X \sim \mathcal{D}}[f \text{ } \neg \text{ const. on } B_\epsilon(X)]$.

Definition 3 (**Separation function**)**.** For a binary classification task for separable classes with a ground truth $h : \mathbb{R}^d \to \{-1, 1\}$ we define a separation function as follows:

$$S(\epsilon) := \inf_{E \subseteq \mathbb{R}^d, d(M_- \setminus E, M_+ \setminus E) \geq \epsilon} [\mathbb{P}_X(X \in E)]\,.$$

Here $M_- = h^{-1}(\{-1\}), M_+ = h^{-1}(\{1\})$. For a given $\epsilon > 0$ this function returns the probability mass that needs to be removed so that the classes are separated by an ϵ-margin.

Lemma 1. *For all separable binary classification tasks and all $\epsilon \in \mathbb{R}_{\geq 0}$ we have that:*

$$AR(\epsilon) = S(2\epsilon).$$

4 DOUBLING DIMENSION

Definition 4. (ϵ**-Net**) Let (M, d) be a metric space. For $N \subseteq M$ we say that N is an ϵ-net of M if it satisfies:

- For every $u, w \in N$ if $u \neq w$ then $d(u, w) \geq \epsilon$,
- $M \subseteq \bigcup_{u \in N} B_\epsilon(u)$.

Definition 5 (ϵ**-Doubling dimension**)**.** For a metric space (M, d), let λ be the smallest value such that every ball of radius at most ϵ in M can be covered by λ balls of radius $\epsilon/2$. We define the ϵ**-doubling dimension** of M as $dd((M, d), \epsilon) := \log_2 \lambda$. Often we will omit the metric and write $dd(M, \epsilon)$ when the metric is clear from the context.

Definition 6 (**Doubling dimension**)**.** For a metric space (M, d) it's doubling dimension is:

$$dd((M, d)) := \sup_{\epsilon > 0} dd((M, d), \epsilon)$$

Fact 2. $dd((\mathbb{R}^d, \ell_2)) \leq 3d$

Next fact was implicitly proven in Dasgupta (2007).

Fact 3. *Let $M \subseteq \mathbb{R}^d$ be a d' dimensional manifold such that the second fundamental form is uniformly bounded by κ. Pick $\epsilon \leq 1/2\kappa$. If for all $e' \leq \epsilon$, for all $x \in M$ we have that $B_\epsilon(x) \cap M$ has at most $2^{O(d')}$ connected components then*

$$dd(M, \epsilon) = O(d'),$$

where the metric on M is the inherited ℓ_2 metric from $\mathbb{R}^d$.

Lemma 2. *Let (M, d) be a metric space with ϵ-doubling dimension dd. If all pairwise distances in $N \subseteq M$ are at least r then for any point $x \in M$ and radius $r \leq t \leq \epsilon$ we have $|B_t(x) \cap N| \leq 2^{dd\lceil \log \frac{2t}{r} \rceil}$.*

Remark 1. *In the remainder of the paper we will only consider subsets of $\mathbb{R}^d$ and the metric we use is always the inherited ℓ_2 metric from the whole space.*

5 RANDOM PARTITIONS

We now discuss *random partitions*, the main technical tool of the paper. For a metric space (M, d) a *partition* π of M is as a function $\pi : M \to 2^M$, mapping a point $x \in M$ to the unique set $\pi(x)$ in π that contains x.

Although in this section we formulate all statements with respect to a generic M, in the sequel it will be important that M equals the support of the data distribution, i.e., $M = \text{supp}(\mathcal{D})$. In particular this will come into play when the data lies on a manifold of small dimension embedded in the ambient space. **To simplify our notation we will not repeat this assertion in each subsequent statement.**

For $\epsilon > 0$ we say that π is ϵ-*bounded* if $\text{diam}(\pi(x)) \leq \epsilon$ for all $x \in M$. The main object of interest will be *random* partitions. We denote a random partition by Π and assume that it has distribution $\mathcal{P}$. We say that Π is ϵ-bounded if Π, drawn according to $\mathcal{P}$, is ϵ-bounded with probability 1.

Definition 7 (**Padded partitions**)**.** For a metric space (M, d) we say that a random partition $\Pi \sim \mathcal{P}$ is $(\epsilon, \beta, \delta)$-*padded* if it is ϵ-bounded and for every $x \in M$:

$$\mathbb{P}_{\Pi \sim \mathcal{P}}[B_{\epsilon/\beta}(x) \not\subseteq \Pi(x)] \leq \delta.$$

Corollary 1. *Let $\Pi \sim \mathcal{P}$ be an $(\epsilon, \beta, \delta)$-padded random partition of a metric space (M, d). Then for every distribution $\mathcal{D}$ we have that:*

$$\mathbb{E}_{\Pi \sim \mathcal{P}}[\mathbb{P}_{X \sim \mathcal{D}}[B_{\epsilon/\beta}(X) \not\subseteq \Pi(X)]] \leq \delta.$$

Now let's consider two random partitions:

Definition 8 (**Cube partition**)**.** For the space $(\mathbb{R}^d, \ell_2)$ and parameter ϵ we define a Cube partition as a partition of $\mathbb{R}^d$ into cubes of width $\epsilon/\sqrt{d}$ corresponding to the shifted lattice $v + \frac{\epsilon}{\sqrt{d}} \cdot \mathbb{Z}^d$. Here the shift $v \sim U([0, \frac{\epsilon}{\sqrt{d}}]^d)$, i.e., v is drawn uniformly at random from a fundamental region of the lattice $\frac{\epsilon}{\sqrt{d}} \cdot \mathbb{Z}^d$. A point x which lies in the intersection of two or more cubes is assigned to the one that is crossed first by a ray $x + \alpha(1, 1, \ldots, 1), \alpha \in \mathbb{R}_{\geq 0}$.

Definition 9 (**Ball carving partition**)**.** For a bounded $M \subseteq \mathbb{R}^d$ and $\epsilon > 0$ we define a ball carving partition as follows. Let N be an $\epsilon/4$-net of M. Pick R uniformly at random from the interval $(\epsilon/4, \epsilon/2]$. Let σ be a random permutation of N. Then for each $u \in N$ define

$$\hat{\Pi}(u) := B_R(u) \setminus \bigcup_{w : \sigma(w) < \sigma(u)} B_R(w).$$

Since the radius R can be strictly larger than some pairwise distances it can happen that for some $u \in N$, $\hat{\Pi}(u)$ does not contain u itself, leading to a potential inconsistency in our notation for the points of the net N. Hence, for all $x \in M$ (and in particular the points of the net N itself) let us define $\Pi(x)$ to be the unique $\hat{\Pi}(w)$, $w \in N$, that contains x.

Lemma 3. *Let Π be a Cube partition with parameter ϵ. Then for every $\beta > 2\sqrt{d}$ it is $\left(\epsilon, \beta, \frac{O(d^{1.5})}{\beta}\right)$-padded.*

The proof of the following Lemma is a slight modification of a proof presented in (Gupta et al. (2003)).

Lemma 4. *Let Π be a Ball carving partition of a bounded $M \subseteq \mathbb{R}^d$ with parameter ϵ. Then for every $\beta > 1$ it is $(\epsilon, \beta, \frac{O(dd(M,\epsilon))}{\beta})$-padded.*

Proof. Recall that the net N underlying the ball carving partition is an $\epsilon/4$-net of M. Fix a point $x \in M$ and some $t \in [0, \epsilon/4]$. Let $W = B_{\epsilon/2+t}(x) \cap N$, and note that by Lemma 2 we have that $m = |W| \leq 6^{dd(M,\epsilon)}$. Arrange the points $w_1, \ldots, w_n \in W$ in order of increasing distance from x, and let I_k be the interval $[d(x, w_k) - t, d(x, w_k) + t]$. Let us say that $B_t(x)$ is *cut* by a cluster $\hat{\Pi}(w_k)$ if $\hat{\Pi}(w_k) \cap B_t(x) \neq \emptyset$ and $B_t(x) \not\subseteq \hat{\Pi}(w_k)$. Finally, write $\mathcal{E}_k$ for the event that w_k is the minimal element in W (according to σ) for which $\hat{\Pi}(w_k)$ cuts $B_t(x)$. Then,

$$\begin{aligned}\mathbb{P}[B_t(x) \text{ is cut}] &\leq \sum_{k=1}^{m} \mathbb{P}[\mathcal{E}_k] \\ &= \sum_{k=1}^{m} \mathbb{P}[R \in I_k] \cdot \mathbb{P}[\mathcal{E}_k | R \in I_k] \\ &\leq \sum_{k=1}^{m} \frac{4t}{\epsilon} \cdot \frac{1}{k} \leq \frac{4t}{\epsilon}(1 + \ln m).\end{aligned}$$

Using the fact that $m = |W| \leq 6^{dd(M,\epsilon)}$ we get that:

$$\mathbb{P}[B_t(x) \text{ is cut}] \leq \frac{t \cdot (8 \cdot dd(M, \epsilon) + 4)}{\epsilon}.$$

□

Corollary 2. *If Π is a Ball carving partition of a bounded $M \subseteq \mathbb{R}^d$ with parameter ϵ then for every $\beta > 1$ it is $\left(\epsilon, \beta, \frac{O(d)}{\beta}\right)$-padded.*

Proof. Follows from Fact 2 and Lemma 4. □

6 FROM A PARTITION TO A CLASSIFIER

To create a robust classifier g from a low-risk classifier f we will use the following framework:

Algorithm 1 SMOOTH$(f, \mathcal{P})$

1: Partition "the space" using $\Pi \sim \mathcal{P}$
2: **return** $g(x) := \text{sgn}(\mathbb{E}_{Z \sim \mathcal{D}}[f(Z) | Z \in \Pi(x)])$

First we want to argue that if a partition π is ϵ-bounded then g defined in Algorithm 1 will have small Risk.

Lemma 5. *Let π be an ϵ-bounded partition. For a given f let $g(x) = sgn(\mathbb{E}_{Z \sim \mathcal{D}}[f(Z) | Z \in \pi(x)])$. Then*

$$R(g) \leq 2S(\epsilon) + 2R(f).$$

The following lemma collects the results from previous sections to obtain a bound on the Adversarial Risk of the classifier g in terms of the optimum.

Lemma 6. *For all $\epsilon > 0$ and any binary classification task with underlying distribution $\mathcal{D}$ if there exists an $(\epsilon\beta, \beta, \delta)$-padded random partition Π of supp($\mathcal{D}$) then the following conditions hold. There exists a randomized algorithm ALG that given black-box access to classifier f produces a classifier g such that in expectation over the random choices of ALG:*

$$AR(g, \epsilon) \leq 2S(\epsilon\beta) + 2R(f) + \delta$$

and if $AR(\epsilon) > 0$ then:

$$AR(g, \epsilon) \leq \frac{2S(\epsilon\beta)}{S(2\epsilon)} AR(\epsilon) + 2R(f) + \delta.$$

7 MAIN RESULTS

In this section we use the partitions defined in Section 6 to derive explicit bounds for the Adversarial Risk of the created classifier.

Theorem 1. *Assume that Algorithm 1 uses Cube partitions (see Definition 8). Let $\alpha > 0$ and $\epsilon > 0$. Then, in expectation over the randomness of the algorithm,*

$$AR(g, \epsilon) \leq 2S\left(\frac{d^{\frac{3}{2}} \cdot \epsilon}{\alpha}\right) + 2R(f) + O(\alpha)$$

and if $AR(\epsilon) > 0$ then

$$AR(g, \epsilon) \leq \frac{2S\left(\frac{d^{\frac{3}{2}} \cdot \epsilon}{\alpha}\right)}{S(2\epsilon)} AR(\epsilon) + 2R(f) + O(\alpha).$$

Proof. Follows from Lemma 6 and 3. □

To understand the interplay of the parameters it's instructive to consider the following case. If $S\left(\frac{O\left(d^{\frac{3}{2}}\right)}{\alpha}\epsilon\right)$ and $S(2\epsilon)$ are comparable, say their ratio is upper-bounded by a constant C, and α is some small constant then the theorem says that the classifier produced by the algorithm satisfies: $AR(g,\epsilon) \leq 2C \cdot AR(\epsilon) + 2 \cdot R(f) + O(1)$. That is, the produced classifier is at most $2C$ times (plus additive error) worse than the optimal one.

Next we present an algorithm with a better bound that uses Ball carving partitions.

Theorem 2. *Assume that Algorithm 1 uses Ball carving partitions (see Definition 9). Let $\alpha > 0$ and $\epsilon > 0$. Then, in expectation over the randomness of the algorithm,*

$$AR(g,\epsilon) \leq 2S\left(\frac{d \cdot \epsilon}{\alpha}\right) + 2R(f) + O(\alpha)$$

and if $AR(\epsilon) > 0$ then:

$$AR(g,\epsilon) \leq \frac{2S\left(\frac{d\cdot\epsilon}{\alpha}\right)}{S(2\epsilon)} AR(\epsilon) + 2R(f) + O(\alpha).$$

Proof. Follows from Lemma 6 and Corollary 2. □

Remark 2. *The bound obtained in Theorem 2 degrades with d. We show however that in some regimes it might be inevitable (see Section 9.1).*

Finally we generalize Theorem 2 to the case when the support of the underlying distribution is a low-dimensional manifold.

Theorem 3. *Assume that Algorithm 1 uses Ball carving partitions (see Definition 9) and that $supp(\mathcal{D}) \subseteq \mathbb{R}^d$ is a $\boldsymbol{d'} \leq d$ dimensional manifold such that the second fundamental form is uniformly bounded by κ. Assume further that for all $x \in M$, and for all $r \leq 1/2\kappa$ the intersection $B_r(x) \cap M$ has at most $2^{O(d')}$ connected components. Let $\alpha > 0$ and $\epsilon \leq \frac{\alpha}{O(d')\kappa}$. Then, in expectation over the randomness of the algorithm,*

$$AR(g,\epsilon) \leq 2S\left(\frac{\boldsymbol{d'}\cdot\epsilon}{\alpha}\right) + 2R(f) + O(\alpha)$$

and if $AR(\epsilon) > 0$ then:

$$AR(g,\epsilon) \leq \frac{2S\left(\frac{\boldsymbol{d'}\cdot\epsilon}{\alpha}\right)}{S(2\epsilon)} AR(\epsilon) + 2R(f) + O(\alpha).$$

Proof. Follows from Lemma 6, 4 and Fact 3. □

Note that all theorems in this section give bounds in the expectation over the randomness of the algorithms. By applying Markov inequality, we can convert these bounds to bounds that are worse by a factor $\gamma > 1$ but hold with probability $1 - 1/\gamma$.

8 COMPUTING $\mathbf{sgn}(\mathbb{E}[f(Z)|Z\in\pi(x)])$

Recall that $g(x) := \mathrm{sgn}(\mathbb{E}_{Z\sim\mathcal{D}}[f(Z)|Z \in \pi(x)])$. As we do not know the distribution $\mathcal{D}$ we cannot compute this expectation directly.

8.1 Scheme A: Approximation with oracle

One approach is to approximate the expectation by a sample mean

$$\hat{g}(x) := \mathrm{sgn}\left(\frac{1}{s}\sum_{i=1}^{s} f(Z_i)\right), \tag{4}$$

where the Z_i's are i.i.d. samples from the distribution $\mathcal{D}$ conditioned on being inside $\pi(x)$. To compute this sum we need samples from $\mathcal{D}$. Note that *unlabeled* samples suffice.

We will bound the number of samples needed to estimate $\hat{g}$ so that $\hat{g}$ has small adversarial risk. Let $x \in \mathbb{R}^d$, and assume that $|\mathbb{E}_{Z\sim\mathcal{D}}[f(Z)|Z\in\pi(x)] - \frac{1}{2}| \geq 0.1$. If we use s samples to estimate $\hat{g}(x)$ according to (4), then, using standard tail bounds,

$$\mathbb{P}[g(x) \neq \hat{g}(x)] \leq e^{-\Theta(s)}. \tag{5}$$

Now assume that π has Q sets $S_1, S_2, \ldots, S_Q$. For $i \in \{1,\ldots,Q\}$ let $p_i := \mathbb{P}_{X\sim\mathcal{D}}[X \in S_i]$. We only need to worry about sets S_i whose probability p_i is not too small. Hence, let $H \subseteq \{i \in \{1,\ldots,Q\} : p_i \geq \frac{R(f)}{Q}\}$. We can argue now, as in the coupon collector's problem, that if we draw

$$O\left(\frac{Q}{R(f)}\log\left(\frac{Q}{R(f)}\right) + \frac{Q\log(Q)}{R(f)}\log\log\left(\frac{Q}{R(f)}\right)\right) \tag{6}$$

samples from $\mathcal{D}$ then with constant probability, for every $i \in H$ at least $\Theta(\log(Q))$ samples will be in S_i. Note that sets outside H cover negligible mass:

$$\sum_{i\in\{1,\ldots,Q\}\setminus H} p_i \leq R(f). \tag{7}$$

Now let $F := \{i \in \{1,\ldots,Q\} : |\mathbb{E}_{Z\sim\mathcal{D}}[f(Z)|Z \in S_i] - \frac{1}{2}| \leq 0.1\}$ and notice that sets from F also cover negligible mass of $\mathcal{D}$:

$$\sum_{i\in F} p_i \leq O(R(f)), \tag{8}$$

because if $i \in F$ then at least a 0.4 fraction of points from S_i is misclassified. Combining: by (5) and the union bound over Q sets, if we sample (6) points from $\mathcal{D}$ then with constant probability, for every $i \in \{1, \dots, Q\} \setminus (F \cup H)$ $\hat{g}$ is equal to g on S_i, which by using (7), (8) and Lemma 5 implies that $R(\hat{g}) \leq O(R(f) + S(\epsilon))$. As a consequence, all theorems from Sections 7 remain true in this setting up to some changes in the constant factors. For instance a variant of Theorem 2 would state:

Theorem 4. *Assume that we sample*

$$O\left(\frac{Q}{R(f)} \log\left(\frac{Q}{R(f)}\right) + \frac{Q \log(Q)}{R(f)} \log\log\left(\frac{Q}{R(f)}\right)\right)$$

points from $\mathcal{D}$ to estimate $\hat{g}$. Assume further that Algorithm 1 uses Ball carving partitions (see Definition 9). Let $\alpha > 0$ and $\epsilon > 0$. Then, with constant probability over the randomness of the algorithm,

$$AR(\hat{g}, \epsilon) \leq O\left(S\left(\frac{d \cdot \epsilon}{\alpha}\right) + R(f) + \alpha\right)$$

and if $AR(\epsilon) > 0$ then:

$$AR(\hat{g}, \epsilon) \leq O\left(\frac{S\left(\frac{d \cdot \epsilon}{\alpha}\right)}{S(2\epsilon)} AR(\epsilon) + R(f) + \alpha\right).$$

8.2 Scheme B: Approximation by uniform sampling

If Q is large then an alternative approach to estimating g might be preferable. One might hope that

$$g(x) \approx \operatorname{sgn}(\mathbb{E}_{Z \sim U(\pi(x))}[f(Z)]). \tag{9}$$

In words, the expectation of f over the whole set $\pi(x)$ is a good proxy to the expectation of f with respect to $\mathcal{D}$ conditioned on being in set $\pi(x)$. If that is the case then instead of performing the smoothing with respect to the data distribution $\mathcal{D}$ we smooth with respect to the uniform distribution on a set of the partition. There are experimental results that indicate that assumption (9) is reasonable. In particular, the approach to approximate $g(x)$ according to (9) is similar to the smoothing used in Cohen et al. (2019) and Salman et al. (2019) – in these works the smoothing is performed by adding a random Gaussian noise to the input. So in this case also the smoothing does not depend on $\mathcal{D}$. Authors of these papers show that their methods outperform all previous defenses against ℓ_2 restricted adversaries. This suggests that assumption (9) holds.

A disadvantage of that approach is that it's hard to prove any theoretical guarantees for this algorithm because, as we discussed before, classifiers with small risk can still behave widely outside of $\operatorname{supp}(\mathcal{D})$. The main advantage of this approach is that we don't require any additional data, apart from access to f, to compute $\hat{g}$. So if (9) holds then the theorems from Section 7 give a direct, affirmative answer to the question posed in the introduction. For running times discussion see Appendix C.

9 THOUGHT EXPERIMENTS

In this section we will present two data distributions and we will show how the implied guarantees from Section 7 compare to the optimum.

9.1 Concentric spheres

First let's analyze the concentric spheres dataset considered in Gilmer et al. (2018). The data distribution consists of two concentric spheres in d dimensions: we generate $x \in \mathbb{R}^d$ where $||x||_2$ is either 1.0 or 1.3, with equal probability assigned to each norm. We associate with each x a label y such that $y = -1$ if $||x||_2 = 1.0$ and $y = +1$ otherwise.

First observe that the data is perfectly separable and that the optimal classifier

$$g_{opt}(x) = \begin{cases} -1, & \text{if } ||x||_2 \leq 1.15 \\ +1, & \text{otherwise} \end{cases}$$

obtains $AR(g_{opt}, 0.15) = 0$, which is the information-theoretic optimum. Assume that we have access to a classifier f such that $R(f) = \delta$. Now we want to analyze the performance of our algorithm. More precisely, we compare our algorithm to the set of classifiers

$$\mathcal{H} := \{g : \mathbb{R}^d \to \{-1, 1\} \mid R(g) \geq \delta\},$$

and not g_{opt}. The constraint $R(g) \geq \delta$ is natural as it means that we want to be competitive against classifiers that are no better than input classifier f.

Now assume that we want to produce a classifier $ALG(f)$ such that $AR(ALG(f), \epsilon) \leq \eta$, for some $\eta \in \mathbb{R}_+$. We should compare the following quantities:

$$\epsilon_{alg} := \arg\max_{\epsilon \in \mathbb{R}_+} [AR(ALG(f), \epsilon) \leq \eta], \tag{10}$$

$$\epsilon_{opt} := \arg\max_{\epsilon \in \mathbb{R}_+} \left[\min_{g \in \mathcal{H}} AR(g, \epsilon) \leq \eta\right]. \tag{11}$$

Observe that the S function for this dataset is:[1]

$$S(\epsilon) = \begin{cases} 0, & \text{if } \epsilon < 0.3, \\ 1/2, & \text{otherwise.} \end{cases}$$

Then Theorem 2 guarantees that we can produce $ALG(f)$ so that: $AR(ALG(f), \epsilon) \leq 2\delta + O(\epsilon \cdot d)$. Using (10) this gives us that $\epsilon_{alg} \geq \Theta(\frac{\eta - 2\delta}{d})$.

Now let $g \in \mathcal{H}$. Recall that by definition $R(g) \geq \delta$. Let S_{in} and S_{out} denote the inner and outer sphere, respectively. Assume that E and E' are the sets of misclassified points on the inner and outer sphere respectively. Without loss of generality we may assume that $\mu(E) \geq \delta/2$ (μ is the measure corresponding to $\mathcal{D}$). Notice that for all ϵ we have $AR(g, \epsilon) \geq \mu(E + B_\epsilon)$. Moreover, the isoperimetric inequality for spheres states that among all sets of measure $\delta/2$ the one that minimizes $\mu(E + B_\epsilon)$ is a spherical cap of this volume, see Gilmer et al. (2018). Let's call this cap C. Now observe that $\mu(C + B_\epsilon) \approx \frac{\delta}{2}(1+\epsilon)^d$. This means that $\epsilon_{opt} \leq O\left(\frac{\log(\eta/\delta)}{d}\right)$.

Combining lower and upper bounds we get that:

$$\frac{\epsilon_{opt}}{\epsilon_{alg}} \leq O\left(\frac{\log(\eta/\delta)}{\eta - 2\delta}\right). \tag{12}$$

That is, our method achieves the target adversarial risk but for perturbations that are $O\left(\frac{\log(\eta/\delta)}{\eta - 2\delta}\right)$ smaller than the optimum. For example in a regime where $\log(\eta/\delta)$ remains smaller than a constant we get a Markov-style tradeoff between the target adversarial risk η and the optimality of ϵ.

It was shown in Gilmer et al. (2018) that neural networks trained on concentric spheres dataset achieve very small risk. When one of the trained networks was evaluated on 20 million samples no errors were observed. This means that $R(f)$ might be really small for this dataset. If for the target adversarial risk we have $\eta >> R(f)$ then the bound (12) might not be satisfactory. It is an interesting research direction to analyze the regime where $\eta >> R(f)$.

9.2 Intersecting circles

Let u_1, u_2 be a pair of orthonormal vectors in $\mathbb{R}^d$. Let $C_{-1}, C_{+1} \subseteq \mathbb{R}^d$ be two circles in the 2-dimensional subspace spanned by u_1, u_2 of radius 1 centered at 0 and u_1 respectively. The distribution is defined as follows: we generate $x \sim U(C_{-1} \cup C_{+1})$ and we associate with each x a label y such that $y = -1$ if $x \in C_{-1}$ and $y = +1$ otherwise.

Note that for $\epsilon \leq 1/10$, $S(\epsilon) = \Theta(\epsilon)$. This is true since in order to ϵ-separate the classes we need to remove the points close to the two intersection points. Note that $\text{supp}(\mathcal{D})$ is a union of two 1-dimensional manifolds whose second fundamental form is bounded by $\Theta(1)$ (Theorem 3 also works in this case). Hence, using Theorem 3 for all $\epsilon < 1/10$ and $\alpha > 0$:

$$AR(g, \alpha\epsilon) \leq O(AR(\epsilon) + R(f) + \alpha).$$

That is, if α is a small constant and $R(f)$ is small then g is only a constant times (plus an additive error) worse than the optimal classifier for adversarial perturbations which are α times smaller. Note that the guarantee does not depend on the dimension of the ambient space but only on the dimension of the manifolds themselves, which in this case is 1.

10 OPEN PROBLEMS & RESEARCH DIRECTIONS

One important open problem is to consider improvements of Theorem 2. In this theorem the guaranteed robustness radius degrades with the dimensionality d of the space. One might hope to get a better dependence on d. In some regimes however it might be hard to achieve an improvement as discussed in Subsection 9.1 (see competitive guarantee (12)).

It is also interesting to analyze different threat models. Imagine that we want the classifier to be robust against an **oblivious** adversary, that is an adversary that has access to f and the algorithm's code but does not know the randomness used by the algorithm. In Appendix E we show that in this model it's possible to achieve the bound

$$AR(g, \epsilon) \leq 2S\left(\frac{\sqrt{\mathbf{d}} \cdot \epsilon}{\alpha}\right) + 2R(f) + O(\alpha).$$

Note that the difference compared to Theorem 2 is that we have the factor $\sqrt{\mathbf{d}}$ instead of $\mathbf{d}$. Intuitively this means that we are be able to get the same adversarial risk for perturbations that are $\sqrt{d}$ bigger.

Another direction is to improve running times of presented algorithms, especially the ones using Ball carving partition. These methods suffer from the high dimension of the ambient space $\mathbb{R}^d$, but as discussed in Subsection C.0.2 there is hope to improve the runtime per query to $2^{O(dd(\text{supp}(\mathcal{D}), \epsilon))}$. This would be a significant improvement for low-dimensional distributions.

[1] The separating function $S(\epsilon)$ does not reach 1 for large values of ϵ since one can always completely remove one class in order to guarantee a separation of ∞.

References

Andoni, A. and Indyk, P. (2008). Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122.

Athalye, A., Carlini, N., and Wagner, D. A. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 274–283.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. (2013). Evasion attacks against machine learning at test time. In Blockeel, H., Kersting, K., Nijssen, S., and Železný, F., editors, *Machine Learning and Knowledge Discovery in Databases*, pages 387–402, Berlin, Heidelberg. Springer Berlin Heidelberg.

Carlini, N. and Wagner, D. A. (2017). Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2017, Dallas, TX, USA, November 3, 2017*, pages 3–14.

Chapelle, O., Schlkopf, B., and Zien, A. (2010). *Semi-Supervised Learning*. The MIT Press, 1st edition.

Cohen, J., Rosenfeld, E., and Kolter, Z. (2019). Certified adversarial robustness via randomized smoothing. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1310–1320, Long Beach, California, USA. PMLR.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA. ACM.

Dasgupta, S. (2007). Random projection trees and low dimensional manifolds. Technical report.

Gilmer, J., Metz, L., Faghri, F., Schoenholz, S. S., Raghu, M., Wattenberg, M., and Goodfellow, I. J. (2018). Adversarial spheres. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*.

Goodfellow, I., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

Gupta, A., Krauthgamer, R., and Lee, J. R. (2003). Bounded geometries, fractals, and low-distortion embeddings. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 534–543.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.

Kurakin, A., Goodfellow, I. J., and Bengio, S. (2017). Adversarial machine learning at scale.

Lécuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., and Jana, S. (2019). Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pages 656–672.

Lee, J. R., Gharan, S. O., and Trevisan, L. (2014). Multiway spectral partitioning and higher-order cheeger inequalities. *Journal of the ACM (JACM)*, 61(6):37.

Li, B., Chen, C., Wang, W., and Carin, L. (2018). Second-order adversarial attack and certifiable robustness. *CoRR*, abs/1809.03113.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *ArXiv*, abs/1706.06083.

Nguyen, A. M., Yosinski, J., and Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, pages 427–436. IEEE Computer Society.

Raghunathan, A., Steinhardt, J., and Liang, P. (2018). Certified defenses against adversarial examples. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Salman, H., Yang, G., Li, J., Zhang, P., Zhang, H., Razenshteyn, I. P., and Bubeck, S. (2019). Provably robust deep learning via adversarially trained smoothed classifiers. *ArXiv*, abs/1906.04584.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014).

Intriguing properties of neural networks. In *International Conference on Learning Representations*.

Uesato, J., O'Donoghue, B., Kohli, P., and van den Oord, A. (2018). Adversarial risk and the dangers of evaluating against weak attacks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 5032–5041.

Wong, E. and Kolter, J. Z. (2018). Provable defenses against adversarial examples via the convex outer adversarial polytope. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 5283–5292.