Analyzing the Robustness of Nearest Neighbors to Adversarial Examples

Yizhen Wang [1] **Somesh Jha** [2] **Kamalika Chaudhuri** [1]

Abstract

Motivated by safety-critical applications, test-time attacks on classifiers via adversarial examples has recently received a great deal of attention. However, there is a general lack of understanding on why adversarial examples arise; whether they originate due to inherent properties of data or due to lack of training samples remains ill-understood. In this work, we introduce a theoretical framework analogous to bias-variance theory for understanding these effects. We use our framework to analyze the robustness of a canonical non-parametric classifier – the k-nearest neighbors. Our analysis shows that its robustness properties depend critically on the value of k – the classifier may be inherently non-robust for small k, but its robustness approaches that of the Bayes Optimal classifier for fast-growing k. We propose a novel modified 1-nearest neighbor classifier, and guarantee its robustness in the large sample limit. Our experiments [1] suggest that this classifier may have good robustness properties even for reasonable data set sizes.

1. Introduction

Machine learning is increasingly applied in security-critical domains such as automotive systems, healthcare, finance and robotics. To ensure safe deployment in these applications, there is an increasing need to design machine-learning algorithms that are robust in the presence of adversarial attacks.

A realistic attack paradigm that has received a lot of recent attention (Goodfellow *et al.*, 2014; Papernot *et al.*, 2016a; Szegedy *et al.*, 2013; Papernot *et al.*, 2017b) is test-time attacks via *adversarial examples*. Here, an adversary has the ability to provide modified test inputs to an already-trained classifier, but cannot modify the training process in any way. Their goal is to perturb legitimate test inputs by a "small amount" in order to force the classifier to report an incorrect label. An example is an adversary that replaces a stop sign by a slightly defaced version in order to force an autonomous vehicle to recognize it as an yield sign. This attack is undetectable to the human eye if the perturbation is small enough.

Prior work has considered adversarial examples in the context of linear classifiers (Lowd and Meek, 2005), kernel SVMs (Biggio *et al.*, 2013) and neural networks (Szegedy *et al.*, 2013; Goodfellow *et al.*, 2014; Papernot *et al.*, 2017b; 2016a; Moosavi-Dezfooli *et al.*, 2016). However, most of this work has either been empirical, or has focussed on developing theoretically motivated attacks and defenses. Consequently, there is a general lack of understanding on why adversarial examples arise; whether they originate due to inherent properties of data or due to lack of training samples remains ill-understood.

This work develops a theoretical framework for robust learning in order to understand the effects of distributional properties and finite samples on robustness. Building on traditional bias-variance theory (Friedman *et al.*, 2000), we posit that a classification algorithm may be robust to adversarial examples due to three reasons. First, it may be *distributionally robust*, in the sense that the output classifier is robust as the number of training samples grow to infinity. Second, even the output of a distributionally robust classification algorithm may be vulnerable due to too few training samples – this is characterized by finite sample robustness. Finally, different training algorithms might result in classifiers with different degrees of robustness, which we call *algorithmic robustness*. These quantities are analogous to bias, variance and algorithmic effects respectively.

Next, we analyze a simple non-parametric classification algorithm: k*-nearest neighbors* in our framework. Our analysis demonstrates that large sample robustness properties of this algorithm depend very much on k.

Specifically, we identify two distinct regimes for k with vastly different robustness properties. When k is constant, we show that k-nearest neighbors has zero robustness in the

[*]Equal contribution [1]University of California, San Diego [2]University of Wisconsin-Madison. Correspondence to: Kamalika Chaudhuri <kamalika@cs.ucsd.edu>.

Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

[1]Code available at: `https://github.com/EricYizhenWang/robust_nn_icml`

large sample limit in regions where $p(y=1|x)$ lies in $(0,1)$. This is in contrast with accuracy, which may be quite high in these regions. For $k=\Omega(\sqrt{dn\log n})$, where d is the data dimension and n is the sample size, we show that the robustness region of k-nearest neighbors approaches that of the Bayes Optimal classifier in the large sample limit. This is again in contrast with accuracy, where convergence to the Bayes Optimal accuracy is known for a much slower growing k (Devroye *et al.*, 1994; Chaudhuri and Dasgupta, 2014).

Since $k=\Omega(\sqrt{dn\log n})$ is too high to use in practice with nearest neighbors, we next propose a novel robust version of the 1-nearest neighbor classifier that operates on a modified training set. We provably show that in the large sample limit, this algorithm has superior robustness to standard 1-nearest neighbors for data distributions with certain properties.

Finally, we validate our theoretical results by empirically evaluating our algorithm on three datasets against several popular attacks. Our experiments demonstrate that our algorithm performs better than or about as well as both standard 1-nearest neighbors and nearest neighbors with adversarial training – a popular and effective defense mechanism. This suggests that although our performance guarantees hold in the large sample limit, our algorithm may have good robustness properties even for realistic training data sizes.

1.1. Related Work

Adversarial examples have recently received a great deal of attention (Goodfellow *et al.*, 2014; Biggio *et al.*, 2013; Papernot *et al.*, 2016a; Szegedy *et al.*, 2013; Papernot *et al.*, 2017b). Most of the work, however, has been empirical, and has focussed on developing increasingly sophisticated attacks and defenses.

1.1.1. Related Work on Adversarial Examples

Prior theoretical work on adversarial examples falls into two categories – analysis and theory-inspired defenses. Work on analysis includes (Fawzi *et al.*, 2016), which analyzes the robustness of linear and quadratic classifiers under random and semi-random perturbations. (Hein and Andriushchenko, 2017) provides robustness guarantees on linear and kernel classifiers trained on a given data set. (Gilmer *et al.*, 2018) shows that linear classifiers for high dimensional datasets may have inherent robustness-accuracy trade-offs.

Work on theory-inspired defenses include (Mądry *et al.*, 2017; Kolter and Wong, 2017; Aman Sinha, 2018), who provide defense mechanisms for adversarial examples in neural networks that are relaxations of certain principled optimization objectives. (Katz *et al.*, 2017) shows how to use program verification to certify robustness of neural networks around given inputs for small neural networks.

Our work differs from these in two important ways. First, unlike most prior work which looks at a given training dataset, we consider effects of the data distribution and number of samples, and analyze robustness properties in the large sample limit. Second, unlike prior work which largely focuses on parametric methods such as neural networks, our focus is on a canonical non-parametric method – the nearest neighbors classifier.

1.1.2. Related Work on Nearest Neighbors

There has been a body of work on the convergence and consistency of nearest-neighbor classifiers and their many variants (Cover and Hart, 1967; Stone, 1977; Kulkarni and Posner, 1995; Devroye and Wagner, 1977; Chaudhuri and Dasgupta, 2014; Kontorovich and Weiss, 2015); all these works however consider accuracy and not robustness.

In the asymptotic regime, (Cover and Hart, 1967) shows that the accuracy of 1-nearest neighbors converges in the large sample limit to $1-2R^*(1-R^*)$ where R^* is the expected error rate of the Bayes Optimal classifier. This implies that even 1-nearest neighbor may achieve relatively high accuracy even when $p(y=1|x)$ is not 0 or 1. In contrast, we show that 1-nearest neighbor is *inherently non-robust* when $p(y=1|x)\in(0,1)$ under some continuity conditions.

For larger k, the accuracy of k-nearest neighbors is known to converge to that of the Bayes Optimal classifier if $k_n\to\infty$ and $k_n/n\to 0$ as the sample size $n\to\infty$. We show that the robustness also converges to that of the Bayes Optimal classifier when k_n grows at a much higher rate – fast enough to ensure uniform convergence. Whether this high rate is necessary remains an intriguing open question.

Finite sample rates on the accuracy of nearest neighbors are known to depend heavily on properties of the data distribution, and there is no distribution free rate as in parametric methods (Devroye and Wagner, 1977). (Chaudhuri and Dasgupta, 2014) provides a clean characterization of the finite sample rates of nearest neighbors as a function of natural *interiors* of the classes. Here we build on their results by defining a stricter, more robust version of interiors and providing bounds as functions of these new robust quantities.

1.1.3. Other Related Work

(Amsaleg *et al.*, 2016) provides a method for generating adversarial examples for nearest neighbors, and shows that the effectiveness of attacks grow with intrinsic dimensionality. Finally, (Papernot *et al.*, 2016b; 2017b) provides *black-box* attacks on substitute classifiers; their experiments show that attacks from other types of substitute classifiers are not successful on nearest neighbors; our experiments corroborate these results.

2. The Setting and Definitions

2.1. The Basic Setup

We consider test-time attacks in a white box setting, where the adversary has full knowledge of the training process – namely, the type of classifier used, the training data and any parameters – but cannot modify training in any way.

Given an input x, the adversary's goal is to perturb it so as to force the trained classifier f to report a different label than $f(x)$. The amount of perturbation is measured by an application-specific metric d, and is constrained to be within a radius r. Our analysis can be extended to any metric, but for this paper we assume that d is the Euclidean distance for mathematical simplicity; we also focus on binary classification, and leave extensions to multiclass for future work.

Finally, we assume that unlabeled instances are drawn from an instance space $\mathcal{X}$, and their labels are drawn from the label space $\{0, 1\}$. There is an underlying data distribution D that generates labeled examples; the marginal over $\mathcal{X}$ of D is μ and the conditional distribution of labels given x is denoted by η.

2.2. Robustness and astuteness

We begin by defining robustness, which for a classifier f at input x is measured by the robustness radius.

Definition 2.1 (Robustness Radius)**.** *The robustness radius of a classifier f at an instance $x \in \mathcal{X}$, denoted by $\rho(f, x)$, is the shortest distance between x and an input x' to which f assigns a label different from $f(x)$:*

$$\rho(f, x) = \inf_r \{\exists x' \in \mathcal{X} \cap B(x, r) \text{ s.t } f(x) \neq f(x')\}$$

Observe that the robustness radius measures a classifier's local robustness. A classifier f with robustness radius r at x guarantees that no adversarial example of x with norm of perturbation less than r can be created using any attack method. A plausible way to extend this into a global notion is to require a lower bound on the robustness radius everywhere; however, only the constant classifier will satisfy this condition. Instead, we consider robustness around *meaningful instances*, that we model as examples drawn from the underlying data distribution.

Definition 2.2 (Robustness with respect to a Distribution)**.** *The robustness of a classifier f at radius r with respect to a distribution μ over the instance space $\mathcal{X}$, denoted by $R(f, r, \mu)$, is the fraction of instances drawn from μ for which the robustness radius is greater than or equal to r.*

$$R(f, r, \mu) = \Pr_{x \sim \mu}(\rho(f, x) \geq r)$$

Finally, observe that we are interested in classifiers that are *both* robust and accurate. This leads to the notion of astuteness, which measures the fraction of instances on which a classifier is both accurate and robust.

Definition 2.3 (astuteness)**.** *The astuteness of a classifier f with respect to a data distribution D and a radius r is the fraction of examples on which it is accurate and has robustness radius at least r; formally,*

$$Ast_D(f, r) = \Pr_{(x,y) \sim D}(\rho(f, x) \geq r, f(x) = y),$$

Observe that astuteness is analogous to classification accuracy, and we argue that it is a more appropriate metric if we are concerned with both robustness and accuracy. Unlike accuracy, astuteness cannot be directly empirically measured unless we have a way to certify a lower bound on the robustness radius. In this work, we will prove bounds on the astuteness of classifiers, and in our experiments, we will approximate it by measuring resistance to standard attacks.

2.3. Sources of Robustness

There are three plausible reasons why classifiers lack robustness – *distributional, finite sample* and *algorithmic*. These sources are analogous to bias, variance, and algorithmic effects respectively in standard bias-variance theory.

Distributional robustness measures the effect of the data distribution on robustness when an infinitely large number of samples are used to train the classifier. Formally, if S_n is a training sample of size n drawn from D and $A(S_n, \cdot)$ is a classifier obtained by applying the training procedure A on S_n, then the distributional robustness at radius r is $\lim_{n\to\infty} \mathbb{E}_{S_n \sim D}[R(A(S_n, \cdot), r, \mu)]$.

In contrast, for finite sample robustness, we characterize the behaviour of $R(A(S_n, \cdot), r, \mu)$ for finite n – usually by putting high probability bounds over the training set. Thus, finite sample robustness depends on the training set size n, and quantifies how it changes with sample size. Finally, robustness also depends on the training algorithm itself; for example, some variants of nearest neighbors may have higher robustness than nearest neighbors itself.

2.4. Nearest Neighbor and Bayes Optimal Classifiers

Given a training set $S_n = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$ and a test example x, we use the notation $X^{(i)}(x)$ to denote the i-th nearest neighbor of x in S_n, and $Y^{(i)}(x)$ to denote the label of $X^{(i)}(x)$.

Given a test example x, the k-nearest neighbor classifier $A_k(S_n, x)$ outputs:

$$\begin{aligned} &= 1, \quad \text{if } Y^{(1)}(x) + \ldots + Y^{(k)}(x) \geq k/2 \\ &= 0, \quad \text{otherwise.} \end{aligned}$$

The Bayes optimal classifier g over a data distribution D

has the following classification rule:

$$g(x) = \begin{cases} 1 & \text{if } \eta(x) = \Pr(y = 1|x) \geq 1/2; \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

3. Robustness of Nearest Neighbors

How robust is the k-nearest neighbor classifier? We show that it depends on the value of k. Specifically, we identify two distinct regimes – constant k and $k = \Omega(\sqrt{dn \log n})$ where d is the data dimension – and show that nearest neighbors has different robustness properties in the two.

3.1. Low k Regime

In this region, k is a constant that does not depend on the training set size n. Provided certain regularity conditions hold, we show that k-nearest neighbors is inherently non-robust in this regime unless $\eta(x) \in \{0, 1\}$ – in the sense that the distributional robustness becomes 0 in the large sample limit.

Theorem 3.1. *Let $x \in \mathcal{X} \cap supp(\mu)$ such that (a) μ is absolutely continuous with respect to the Lebesgue measure (b) $\eta(x) \in (0, 1)$ (c) η is continuous with respect to the Euclidean metric in a neighborhood of x. Then, for fixed k, $\rho(A_k(S_n, \cdot), x)$ converges in probability to* 0.

Remarks. Observe that Theorem 3.1 implies that the distributional robustness (and hence astuteness) in a region where $\eta(x) \in (0, 1)$ is 0. This is in contrast with accuracy; for 1-NN, the accuracy converges to $1 - 2R^*(1 - R^*)$ as $n \to \infty$, where R^* is the error rate of the Bayes Optimal classifier, and thus may be quite high.

The proof of Theorem 3.1 in the Appendix shows that the absolute continuity of μ with respect to the Lebesgue measure is not strictly necessary; absolute continuity with respect to an embedded manifold will give the same result, but will result in a more complex proof.

In the Appendix A (Theorem A.2), we show that k-nearest neighbor is astute in the interior of the region where $\eta \in \{0, 1\}$, and provide finite sample rates for this case.

3.2. High k Regime

Prior work has shown that in the large sample limit, the accuracy of the nearest neighbor classifiers converge to the Bayes Optimal, provided k is set properly. We next show that if k is $\Omega(\sqrt{dn \log n})$, the regions of robustness and the astuteness of the k nearest neighbor classifiers also approach the corresponding quantities for the Bayes Optimal classifier as $n \to \infty$. Thus, if the Bayes Optimal classifier is robust, then so is k-nearest neighbors in the large sample limit.

The main intuition is that $k = \Omega(\sqrt{dn \log n})$ is large enough for *uniform convergence* – where, with high probability, *all* Euclidean balls with k examples have the property that the empirical averages of their labels are close to their expectations. This guarantees that for any x, the k-nearest neighbor reports the same label as the Bayes Optimal classifier for *all* x' close to x. Thus, if the Bayes Optimal classifier is robust, so is nearest neighbors.

3.2.1. Definitions

We begin with some definitions that we can use to characterize the robustness of the Bayes Optimal classifier. Following (Chaudhuri and Dasgupta, 2014), we use the notation $B^o(x, r)$ to denote an open ball and $B(x, r)$ to denote a closed ball of radius r around x. We define the probability radius of a ball around x as:

$$r_p(x) = \inf\{r \mid \mu(B(x, r)) \geq p\}$$

We next define the r-robust (p, Δ)-strict interiors as follows:

$$\begin{aligned} \mathcal{X}^+_{r,\Delta,p} &= \{x \in supp(\mu) \mid \forall x' \in B^o(x, r), \\ &\quad \forall x'' \in B(x', r_p(x')), \eta(x'') > 1/2 + \Delta\} \\ \mathcal{X}^-_{r,\Delta,p} &= \{x \in supp(\mu) \mid \forall x' \in B^o(x, r), \\ &\quad \forall x'' \in B(x', r_p(x')), \eta(x'') < 1/2 - \Delta\} \end{aligned}$$

What is the significance of these interiors? Let x' be an instance such that all $x'' \in B(x', r_p(x'))$ have $\eta(x'') > 1/2 + \Delta$. If $p \approx \frac{k}{n}$, then the k points x'' closest to x' have $\eta(x'') > 1/2 + \Delta$. Provided the average of the labels of these points is close to expectation, which happens when k is large relative to $1/\Delta$, k-nearest neighbor outputs label 1 on x'. When x is in the r-robust (p, Δ)-strict interior region $\mathcal{X}^+_{r,\Delta,p}$, this is true for all x' within distance r of x, which means that k-nearest neighbors will be robust at x. Thus, the r-robust (p, Δ)-strict interior is the region where we natually expect k-nearest neighbor to have robustness radius r, when k is large relative to $\frac{1}{\Delta}$ and $p \approx \frac{k}{n}$.

Readers familiar with (Chaudhuri and Dasgupta, 2014) will observe that the set of all x' for which $\forall x'' \in B(x', r_p(x')), \eta(x'') > 1/2 + \Delta$ forms a stricter version of the (p, Δ)-interiors of the 1 region that was defined in this work; these x' also represent the region where k-nearest neighbors are accurate when $k \approx \max(np, 1/\Delta^2)$. The r-robust (p, Δ)-strict interior is thus a somewhat stricter and more robust version of this definition.

3.2.2. Main Results

We begin by characterizing where the Bayes Optimal classifier is robust.

Theorem 3.2. *The Bayes Optimal classifier has robustness radius r at $x \in \mathcal{X}^+_{r,0,0} \cup \mathcal{X}^-_{r,0,0}$. Moreover, its astuteness is* $\mathbb{E}[\eta(x)\mathbf{1}(x \in \mathcal{X}^+_{r,0,0})] + \mathbb{E}[(1 - \eta(x))\mathbf{1}(x \in \mathcal{X}^-_{r,0,0})]$.

The proof is in the Appendix, along with analogous results for astuteness. The following theorem, along with a similar result for astuteness, proved in the Appendix, characterizes robustness in the large k regime.

Theorem 3.3. *For any n, pick a δ and a $\Delta_n \to 0$. There exist constant C_1 and C_2 such that if $k_n \geq C_1 \frac{\sqrt{dn\log n + n\log(1/\delta_n)}}{\Delta_n}$, and $p_n \geq \frac{k_n}{n}(1 + C_2\sqrt{\frac{d\log n + \log(1/\delta)}{k_n}})$, then, with probability $\geq 1 - 3\delta$, k_n-NN has robustness radius r in $x \in \mathcal{X}^+_{r,\Delta_n,p_n} \cup \mathcal{X}^-_{r,\Delta_n,p_n}$.*

Remarks. Some remarks are in order. First, observe that as $n \to \infty$, Δ_n and p_n tend to 0; thus, provided certain continuity conditions hold, $\mathcal{X}^+_{r,\Delta_n,p_n} \cup \mathcal{X}^-_{r,\Delta_n,p_n}$ approaches $\mathcal{X}^+_{r,0,0} \cup \mathcal{X}^-_{r,0,0}$, the robustness region of the Bayes Optimal classifier.

Second, observe that as r-robust strict interiors extend the definition of interiors in (Chaudhuri and Dasgupta, 2014), Theorem 3.3 is a robustness analogue of Theorem 5 in this work. Unlike the latter, Theorem 3.3 has a more stringent requirement on k. Whether this is necessary is left as an open question for future work.

4. A Robust 1-NN Algorithm

Section 3 shows that nearest neighbors is robust for k as large as $\Omega(\sqrt{dn\log n})$. However, this k is too high to use in practice – high values of k require even higher sample sizes (Chaudhuri and Dasgupta, 2014), and lead to higher running times. Thus a natural question is whether we can find a more robust version of the algorithm for smaller k. In this section, we provide a more robust version of 1-nearest neighbors, and analytically demonstrate its robustness.

Our algorithm is motivated by the observation that 1-nearest neighbor is robust when oppositely labeled points are far apart, and when test points lie close to training data. Most training datasets however contain nearby points that are oppositely labeled; thus, we propose to remove a subset of training points to enforce this property.

Which points should we remove? A plausible approach is to keep the largest subset where oppositely labeled points are far apart; however, this subset has poor stability properties even for large n. Therefore, we propose to keep all points x such that: (a) we are highly confident about the label of x and its nearby points and (b) all points close to x have the same label. Given that all such x are kept, we remove as few points as possible, and execute nearest neighbors on the remaining dataset.

The following definition characterizes data where oppositely labeled points are far apart.

Definition 4.1 (r-separated set). *A set $A = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ of labeled examples is said to be r-separated if for all pairs $(x_i, y_i), (x_j, y_j) \in A$, $\|x_i - x_j\| \leq r$ implies $y_i = y_j$.*

The full algorithm is described in Algorithm 1 and Algorithm 2. Given confidence parameters Δ and δ, Algorithm 2 returns a $0/1$ label when this label agrees with the average of k_n points closest to x; otherwise, it returns $\perp$. k_n is chosen such that with probability $\geq 1-\delta$, the empirical majority of k_n labels agrees with the majority in expectation, provided the latter is at least Δ away from $\frac{1}{2}$.

Algorithm 2 is used to determine whether an x_i should be kept. Let $f(x_i)$ be the output of Algorithm 2 on x_i. If $y_i = f(x_i)$ and if for all $x_j \in B(x_i, r)$, $f(x_i) = f(x_j) = y_i$, then we mark x_i as red. Finally, we compute the largest r-separated subset of the training data that includes all the red points; this reduces to a constrained matching problem as in (Kontorovich and Weiss, 2015). The resulting set, returned by Algorithm 1, is our new training set. We observe that this set is r-separated from Lemma B.2 in the Appendix, and thus oppositely labeled points are far apart. Moreover, we keep all (x_i, y_i) when we are confident about the label of x_i and its nearby points. Observe that our final procedure is a 1-NN algorithm, even though k_n neighbors are used to determine if a point should be retained in the training set.

4.1. Performance Guarantees

The following theorem establishes performance guarantees for Algorithm 1.

Theorem 4.2. *Pick a Δ_n and δ, and set $k_n = 3\log(2n/\delta)/\Delta_n^2$. Pick a* margin parameter τ. *Then, there exist constants C and C_0 such that the following hold. If we set $p_n = \frac{k_n}{n}(1 + C\sqrt{\frac{d\log n + \log(1/\delta)}{k_n}})$, and define the set:*

$$X_R = \left\{ x \middle| x \in \mathcal{X}^+_{r+\tau,\Delta_n,p_n} \cup \mathcal{X}^-_{r+\tau,\Delta_n,p_n}, \right. \\ \left. \mu(B(x,\tau)) \geq \frac{2C_0}{n}(d\log n + \log(1/\delta)) \right\}$$

Then, with probability $\geq 1 - 2\delta$ over the training set, Algorithm 1 run with parameters r, Δ_n and δ has robustness radius at least $r - 2\tau$ on X_R.

Remarks. The proof is in the Appendix, along with an analogous result for astuteness. Observe that X_R is roughly the *high density subset* of the $r + \tau$-robust strict interior $\mathcal{X}^+_{r+\tau,\Delta_n,p_n} \cup \mathcal{X}^-_{r+\tau,\Delta_n,p_n}$. Since $\eta(x)$ is constrained to be greater than $\frac{1}{2} + \Delta_n$ or less than $\frac{1}{2} - \Delta_n$ in this region, as opposed to 0 or 1, this is an improvement over standard nearest neighbors when the data distribution has a large high density region that intersects with the interiors.

A second observation is that as τ is an arbitrary constant, we can set to it be quite small and still satisfy the condition on $\mu(B(x, \tau))$ for a large fraction of x's when n is very large. This means that in the large sample limit, $r - 2\tau$ may be close to r and X_R may be close to the high density subset of $\mathcal{X}^{+}_{r,\Delta_n,p_n} \cup \mathcal{X}^{-}_{r,\Delta_n,p_n}$ for a lot of smooth distributions.

Algorithm 1 Robust_1NN(S_n, r, Δ, δ, x)

```
for $(x_i, y_i) \in S_n$ do
    $f(x_i) =$ Confident-Label$(S_n, \Delta, \delta, x_i)$
end for
$S_{RED} = \emptyset$
for $(x_i, y_i) \in S_n$ do
    if $f(x_i) = y_i$ and $f(x_i) = f(x_j)$ for all $x_j$ such that $\|x_i - x_j\| \leq r$ and $(x_j, y_j) \in S_n$ then
        $S_{RED} = S_{RED} \bigcup \{(x_i, y_i)\}$
    end if
end for
Let $S'$ be the largest $r$-separated subset of $S_n$ that contains all points in $S_{RED}$.
return new training set $S'$
```

Algorithm 2 Confident-Label(S_n, Δ, δ, x)

```
$k_n = 3\log(2n/\delta)/\Delta^2$
$\bar{y} = (1/k_n)\sum_{i=1}^{k_n} Y^{(i)}(x)$
if $\bar{y} \in [\frac{1}{2} - \Delta, \frac{1}{2} + \Delta]$ then
    return $\perp$
else
    return $\frac{1}{2}sgn(\bar{y} - \frac{1}{2}) + \frac{1}{2}$
end if
```

5. Experiments

The results in Section 4 assume large sample limits. Thus, a natural question is how well Algorithm 1 performs with more reasonable amounts of training data. We now empirically investigate this question.

Since there are no general methods that certify robustness at an input, we assess robustness by measuring how our algorithm performs against a suite of standard attack methods. Specifically, we consider the following questions:

1. How does our algorithm perform against popular white box and black box attacks compared with standard baselines?

2. How is performance affected when we change the training set size relative to the data dimension?

These questions are considered in the context of three datasets with varying training set sizes relative to the dimension, as well as two standard white box attacks and black box attacks with two kinds of substitute classifiers.

5.1. Methodology

Data. We use three datasets – Halfmoon, MNIST 1v7 and Abalone – with differing data sizes relative to dimension. Halfmoon is a popular 2-dimensional synthetic data set for non-linear classification. We use a training set of size 2000 and a test set of size 1000 generated with standard deviation $\sigma = 0.2$. The MNIST 1v7 data set is a subset of the 784-dimensional MNIST data. For training, we use 1000 images each of Digit 1 and 7, and for test, 500 images of each digit. Finally, for the Abalone dataset (Lichman, 2013), our classification task is to distinguish whether an abalone is older than 12.5 years based on 7 physical measurements. For training, we use 500 and for test, 100 samples. In addition, a validation set with the same size as the test set is generated for each experiment for parameter tuning.

Baselines. We compare Algorithm 1, denoted by RobustNN, against three baselines. The first is the standard 1-nearest neighbor algorithm, denoted by StandardNN. We use two forms of adversarially-trained nearest neighbors - ATNN and ATNN-all. Let S be the training set used by standard nearest neighbors. In ATNN, we augment S by creating, for each $(x, y) \in S$, an adversarial example x_{adv} using the attack method in the experiment, and adding (x_{adv}, y). The ATNN classifier is 1-nearest neighbor on this augmented data. In ATNN-all, for each $(x, y) \in S$, we create adversarial examples using *all* the attack methods in the experiment, and add them all to S. ATNN-all is the nearest neighbor classifier on this augmented data. For example, for white box Direct Attacks in Section 5.2, ATNN includes adversarial examples generated by the Direct Attack, and ATNN-all includes adversarial examples generated by both Direct and Kernel Substitute Attacks.

Observe that all algorithms except StandardNN have parameters to tune. RobustNN has three input parameters – Δ, δ and a defense radius r which is an approximation to the robustness radius. For simplicity, we set $\Delta = 0.45$, $\delta = 0.1$ and tune r on the validation set; this can be viewed as tuning the parameter τ in Theorem 4.2. For ATNN and ATNN-all, the methods that generate the augmenting adversarial examples need a perturbation magnitude r; we call this the *defense radius*. To be fair to all algorithms, we tune the defense radius for each. We consider the adversary with the highest attack perturbation magnitude in the experiment, and select the defense radius that yields the highest validation accuracy against this adversary.

5.2. White-box Attacks and Results

To evaluate the robustness of Algorithm 1, we use two standard classes of attacks – white box and black box. For white-box attacks, the adversary knows all details about the classifier under attack, including its training data, the

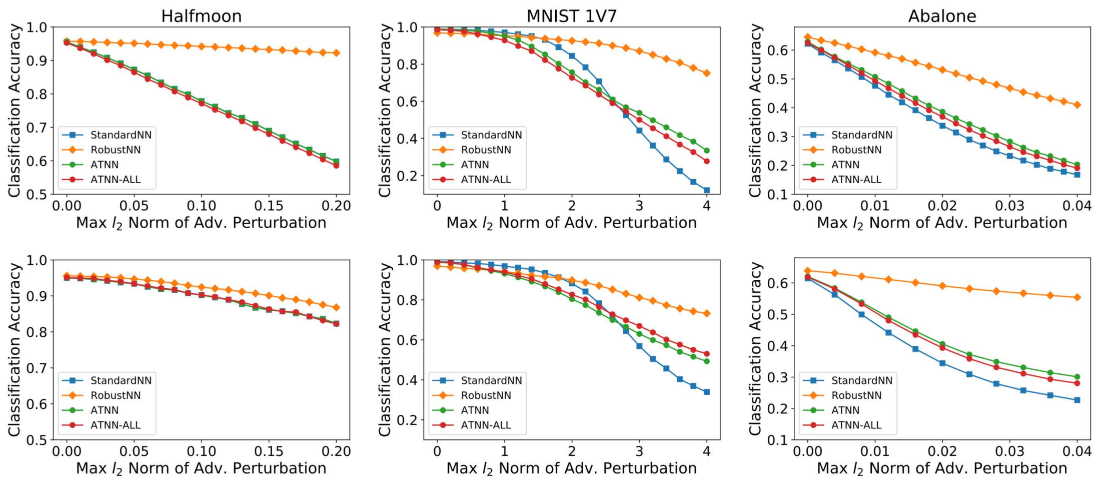

Figure 1. **White Box Attacks:** Plot of classification accuracy on adversarial examples v.s. attack radius. *Top row:* Direct Attack. *Bottom row:* Kernel Substitute Attack. *Left to right:* 1) Halfmoon, 2) MNIST 1v 7 and 3) Abalone.

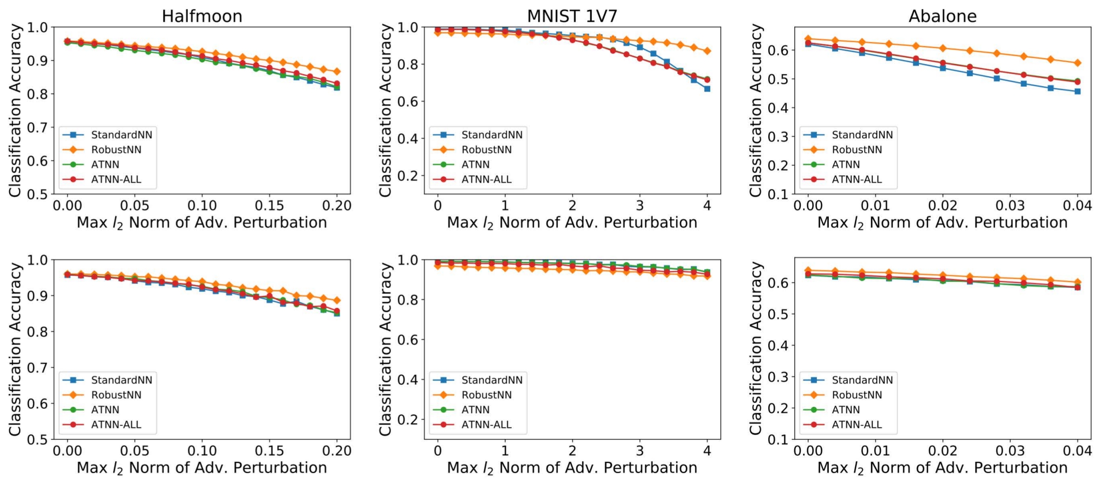

Figure 2. **Black Box Attacks:** Plot of classification accuracy on adversarial examples v.s. attack radius. *Top to Bottom:* 1) kernel substitute, 2) neural net substitute. *Left to right:* 1) Halfmoon, 2) MNIST 1 v.s. 7 and 3) Abalone.

training algorithm and any hyperparameters.

5.2.1. Attack Methods

We consider two white-box attacks – direct attack (Amsaleg *et al.*, 2016) and Kernel Substitute Attack (Papernot *et al.*, 2016b).

Direct Attack. This attack takes as input a test example x, an attack radius r, and a training dataset S (which may be an augmented or reduced dataset). It finds an $x' \in S$ that is closest to x but has a different label, and returns the adversarial example $x_{adv} = x + r\frac{x-x'}{||x-x'||_2}$.

Kernel Substitute Attack. This method attacks a substitute kernel classifier trained on the *same training set*. For a test input $\vec{x}$, a set of training points Z with one-hot labels Y, a kernel classifier f predicts the class probability as:

$$f : \vec{x} \rightarrow \frac{\left[e^{-||\vec{x}-\vec{z}||_2^2/c}\right]_{\vec{z}\in X}}{\sum_{\vec{z}\in X} e^{-||\vec{x}-\vec{z}||_2^2/c}} \cdot Y$$

The adversary trains a kernel classifier on the training set of the corresponding nearest neighbors, and then generates adversarial examples against this kernel classifier. The advantage is that the prediction of the kernel classifier is differentiable, which allows the use of standard gradient-based attack methods. For our experiments, we use the popular fast-gradient-sign method (FSGM). The parameter c is tuned to yield the most effective attack, and is set to 0.1 for Halfmoon and MNIST, and 0.01 for Abalone.

5.2.2. Results

Figure 1 shows the results. We see that RobustNN outperforms all baselines for Halfmoon and Abalone for all attack radii. For MNIST, for low attack radii, RobustNN's classification accuracy is slightly lower than the others, while it outperforms the others for large attack radii. Additionally, as is to be expected, the Direct Attack results in lower general accuracy than the Kernel Substitute Attack.

These results suggest that our algorithm mostly outperforms the baselines StandardNN, ATNN and ATNN-all. As predicted by theory, the performance gain is higher when the training set size is large relative to the dimension – which is the setting where nearest neighbors work well in general. It has superior performance for Halfmoon and Abalone, where the training set size is large to medium relative to dimension. In contrast, in the sparse dataset MNIST, our algorithm has slightly lower classification accuracy for small attack radii, and higher otherwise.

5.3. Black-box Attacks and Results

(Papernot *et al.*, 2017b) has observed that some defense methods that work by masking gradients remain highly amenable to *black box attacks*. In this attack, the adversary is unaware of the target classifier's nature, parameters or training data, but has access to a seed dataset drawn from the same distribution which they use to train and attack a substitute classifier. To establish robustness properties of Algorithm 1, we therefore validate it against black box attacks based on two types of substitute classifiers.

5.3.1. Attack Methods

We use two types of substitute classifiers – kernel classifiers and neural networks. The adversary trains the substitute classifier using the method of (Papernot *et al.*, 2017b) and uses the adversarial examples against the substitute to attack the target classifier.

Kernel Classifier. The kernel classifier substitute is the same as the one in Section 5.2, but trained using the seed data and the method of (Papernot *et al.*, 2017b).

Neural Networks. The neural network for MNIST is the ConvNet in (Papernot *et al.*, 2017a)'s tutorial. For Halfmoon and Abalone, the network is a multi-layer perceptron with 2 hidden layers.

Procedure. To train the substitute classifier, the adversary uses the method of (Papernot *et al.*, 2016b) to augment the seed data for two rounds; labels are obtained by querying the target classifier. Adversarial examples against the substitutes are created by FGSM, following (Papernot *et al.*, 2016b). As a sanity check, we verify the performance of the substitute classifiers on the original training and test sets. Details are in Table 1 in the Appendix. Sanity checks on the performance of the substitute classifiers are presented in Table 1 in the Appendix.

5.3.2. Results

Figure 2 shows the results. For all algorithms, black box attacks are less effective than white box, which corroborates the results of (Papernot *et al.*, 2016b), who observed that black-box attacks are less successful against nearest neighbors. We also find that the kernel substitute attack is more effective than the neural network substitute, which is expected as kernel classifiers have similar structure to nearest neighbors. Finally, for Halfmoon and Abalone, our algorithm outperforms the baselines for both attacks; however, for MNIST neural network substitute, our algorithm does not perform as well for small attack radii. This again confirms the theoretical predictions that our algorithm's performance is better when the training set is large relative to the data dimension – the setting in which nearest neighbors work well in general.

5.4. Discussion

The results show that our algorithm performs either better than or about the same as standard baselines against popular white box and black box attacks. As expected from our theoretical results, it performs better for denser datasets which have high or medium amounts of training data relative to the dimension, and either slightly worse or better for sparser datasets, depending on the attack radius. Since non-parametric methods such as nearest neighbors are mostly used for dense data, this suggests that our algorithm has good robustness properties even with reasonable amounts of training data.

6. Conclusion

We introduce a novel theoretical framework for learning robust to adversarial examples, and introduce notions of distributional and finite-sample robustness. We use these notions to analyze a non-parametric classifier, k-nearest neighbors, and introduce a novel modified 1-nearest neighbor algorithm that has good robustness properties in the large sample limit. Experiments show that these properties are still retained for reasonable data sizes.

Many open questions remain. The first is to close the gap in analysis of k-nearest neighbors for k in between our two regimes. The second is to develop nearest neighbor algorithms with better robustness guarantees. Finally, we believe that our work is a first step towards a comprehensive analysis of how the size of training data affects robustness; we believe that an important line of future work is to carry out similar analyses for other supervised learning methods.

Acknowledgement

We thank NSF under IIS 1253942 for support. This work was also partially supported by ARO under contract number W911NF-1-0405. We thank all anonymous reviewers for their constructive comments.

References

Aman Sinha, Hongseok Namkoong, J. D. (2018). Certifiable distributional robustness with principled adversarial training. *International Conference on Learning Representations*.

Amsaleg, L., Bailey, J., Erfani, S., Furon, T., Houle, M. E., Radovanović, M., and Vinh, N. X. (2016). The vulnerability of learning to adversarial perturbation increases with intrinsic dimensionality.

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 *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 387–402. Springer.

Chaudhuri, K. and Dasgupta, S. (2010). Rates of convergence for the cluster tree. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 343–351. Curran Associates, Inc.

Chaudhuri, K. and Dasgupta, S. (2014). Rates of convergence for nearest neighbor classification. In *Advances in Neural Information Processing Systems*, pages 3437–3445.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, **13**, 21–27.

Devroye, L., Gyorfi, L., Krzyzak, A., and Lugosi, G. (1994). On the strong universal consistency of nearest neighbor regression function estimates. *The Annals of Statistics*, pages 1371–1385.

Devroye, L. P. and Wagner, T. J. (1977). The strong uniform consistency of nearest neighbor density estimates. *The Annals of Statistics*, pages 536–540.

Fawzi, A., Moosavi-Dezfooli, S.-M., and Frossard, P. (2016). Robustness of classifiers: from adversarial to random noise. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1632–1640. Curran Associates, Inc.

Friedman, J., Hastie, T., Tibshirani, R., *et al.* (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, **28**(2), 337–407.

Gilmer, J., Metz, L., Faghri, F., Schoenholz, S. S., Raghu, M., Wattenberg, M., and Goodfellow, I. (2018). Adversarial spheres. *arXiv preprint arXiv:1801.02774*.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Hein, M. and Andriushchenko, M. (2017). Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, pages 2263–2273.

Katz, G., Barrett, C., Dill, D. L., Julian, K., and Kochenderfer, M. J. (2017). Towards proving the adversarial robustness of deep neural networks. *arXiv preprint arXiv:1709.02802*.

Kolter, J. Z. and Wong, E. (2017). Provable defenses against adversarial examples via the convex outer adversarial polytope. *arXiv preprint arXiv:1711.00851*.

Kontorovich, A. and Weiss, R. (2015). A bayes consistent 1-nn classifier. In *Artificial Intelligence and Statistics Conference*.

Kulkarni, S. and Posner, S. (1995). Rates of convergence of nearest neighbor estimation under arbitrary sampling. *IEEE Transactions on Information Theory*, **41**(4), 1028–1039.

Lichman, M. (2013). UCI machine learning repository.

Lowd, D. and Meek, C. (2005). Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 641–647. ACM.

Mądry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *stat*, **1050**, 9.

Mitzenmacher, M. and Upfal, E. (2005). *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge university press.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). Deepfool: a simple and accurate method to fool deep neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. (2016a). The limitations of deep learning in adversarial settings. In *Proceedings of the 1st IEEE European Symposium on Security and Privacy*. arXiv preprint arXiv:1511.07528.

Papernot, N., McDaniel, P., and Goodfellow, I. (2016b). Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*.

Papernot, N., Carlini, N., Goodfellow, I., Feinman, R., Faghri, F., Matyasko, A., Hambardzumyan, K., Juang, Y.-L., Kurakin, A., Sheatsley, R., Garg, A., and Lin, Y.-C. (2017a). cleverhans v2.0.0: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, B., and Swami, A. (2017b). Practical black-box attacks against deep learning systems using adversarial examples. In *Proceedings of the 2017 ACM Asia Conference on Computer and Communications Security*.

Stone, C. (1977). Consistent nonparametric regression. *Annals of Statistics*, **5**, 595–645.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.