A Closer Look at Accuracy vs. Robustness

Yao-Yuan Yang[*1] **Cyrus Rashtchian**[*1] **Hongyang Zhang**[2]

Ruslan Salakhutdinov[3] **Kamalika Chaudhuri**[1]

[1]University of California, San Diego
[2]Toyota Technological Institute at Chicago
[3]Carnegie Mellon University
{yay005, crashtchian}@eng.ucsd.edu hongyanz@ttic.edu
rsalakhu@cs.cmu.edu kamalika@cs.ucsd.edu

Abstract

Current methods for training robust networks lead to a drop in test accuracy, which has led prior works to posit that a robustness-accuracy tradeoff may be inevitable in deep learning. We take a closer look at this phenomenon and first show that real image datasets are actually separated. With this property in mind, we then prove that robustness and accuracy should both be achievable for benchmark datasets through locally Lipschitz functions, and hence, there should be no inherent tradeoff between robustness and accuracy. Through extensive experiments with robustness methods, we argue that the gap between theory and practice arises from two limitations of current methods: either they fail to impose local Lipschitzness or they are insufficiently generalized. We explore combining dropout with robust training methods and obtain better generalization. We conclude that achieving robustness and accuracy in practice may require using methods that impose local Lipschitzness and augmenting them with deep learning generalization techniques.[1]

1 Introduction

A growing body of research shows that neural networks are vulnerable to *adversarial examples*, test inputs that have been modified slightly yet strategically to cause misclassification [56, 17]. While a number of defenses have been proposed [10, 32, 46, 65], they are known to hurt test accuracy on many datasets [41, 32, 68]. This observation has led prior works to claim that a tradeoff between robustness and accuracy may be *inevitable* for many classification tasks [57, 65].

We take a closer look at the tradeoff between robustness and accuracy, aiming to identify properties of data and training methods that enable neural networks to achieve *both*. A plausible reason why robustness may lead to lower accuracy is that different classes are very close together or they may even overlap (which underlies the argument for an inevitable tradeoff [57]). We begin by testing if this is the case in real data through an empirical study of four image datasets. Perhaps surprisingly, we find that these datasets actually satisfy a natural separation property that we call r-separation: examples from different classes are at least distance $2r$ apart in pixel space. This r-separation holds for values of r that are higher than the perturbation radii used in adversarial example experiments.

We next consider separation as a guiding principle for better understanding the robustness-accuracy tradeoff. Neural network classifiers are typically obtained by rounding an underlying continuous

[*]Equal contribution
[1]Code available at https://github.com/yangarbiter/robust-local-lipschitz.

function $f : \mathcal{X} \to \mathbb{R}^C$ with C classes. We take inspiration from prior work, which shows that Lipschitzness of f is closely related to its robustness [10, 21, 46, 60, 64]. However, one drawback of the existing arguments is that they do not provide a compelling and realistic assumption on the data that guarantees robustness and accuracy. We show theoretically that any r-separated data distribution has a classifier that is both robust up to perturbations of size r, and accurate, and it can be obtained by rounding a function that is locally Lipschitz around the data. This suggests that there should exist a robust and highly accurate classifier for real image data. Unfortunately, the current state of robust classification falls short of this prediction, and the discrepancy remains poorly understood.

To better understand the theory-practice gap, we empirically investigate several existing methods on a few image datasets with a special focus on their local Lipschitzness and generalization gaps. We find that of the methods investigated, adversarial training (AT) [32], robust self-training (RST) [42] and TRADES [64] impose the highest degree of local smoothness, and are the most robust. We also find that the three robust methods have large gaps between training and test accuracies as well as adversarial training and test accuracies. This suggests that the disparity between theory and practice may be due to the limitations of existing training procedures, particularly in regards to generalization. We then experiment with dropout, a standard generalization technique, on top of robust training methods on two image datasets where there is a significant generalization gap. We see that dropout in particular narrows the generalization gaps of TRADES and RST, and improves test accuracy, test adversarial accuracy as well as test Lipschitzness. In summary, our contributions are as follows.

- Through empirical measurements, we show that several image datasets are separated.
- We prove that this separation implies the existence of a robust and perfectly accurate classifier that can be obtained by rounding a locally Lipschitz function. In contrast to prior conjectures [12, 16, 57], robustness and accuracy can be achieved together in principle.
- We investigate smoothness and generalization properties of classifiers produced by current training methods. We observe that the training methods AT, TRADES, and RST, which produce robust classifiers, also suffer from large generalization gaps. We combine these robust training methods with dropout [54], and show that this narrows the generalization gaps and sometimes makes the classifiers smoother.

What do our results imply about the robustness-accuracy tradeoff in deep learning? They suggest that this tradeoff is not inherent. Rather, it is a consequence of current robustness methods. The past few years of research in robust machine learning has led to a number of new loss functions, yet the rest of the training process – network topologies, optimization methods, generalization tools – remain highly tailored to promoting accuracy. We believe that in order to achieve both robustness and accuracy, future work may need to redesign other aspects of the training process such as better network architectures using neural architecture search [13, 19, 47, 69]. Combining this with improved optimization methods and robust losses may be able to reduce the generalization gap in practice.

2 Preliminaries

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be an instance space equipped with a metric $\text{dist} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$; this is the metric in which robustness is measured. Let $[C] = \{1, 2, \ldots, C\}$ denote the set of possible labels with $C \geq 2$. For a function $f : \mathcal{X} \to \mathbb{R}^C$, let $f(\boldsymbol{x})_i$ denote the value of the ith coordinate.

Robustness and Astuteness. Let $\mathbb{B}(\boldsymbol{x}, \varepsilon)$ denote a ball of radius $\varepsilon > 0$ around $\boldsymbol{x}$ in a metric space. We use $\mathbb{B}_\infty$ to denote the ℓ_∞ ball. A classifier g is *robust* at $\boldsymbol{x}$ with radius $\varepsilon > 0$ if for all $\boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \varepsilon)$, we have $g(\boldsymbol{x}') = g(\boldsymbol{x})$. Also, g is *astute* at $(\boldsymbol{x}, y)$ if $g(\boldsymbol{x}') = y$ for all $\boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \varepsilon)$. The *astuteness* of g at radius $\varepsilon > 0$ under a distribution μ is

$$\Pr_{(\boldsymbol{x},y)\sim\mu}[g(\boldsymbol{x}') = y \text{ for all } \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \varepsilon)].$$

The goal of robust classification is to find a g with the highest astuteness [59]. We sometimes use *clean accuracy* to refer to standard test accuracy (no adversarial perturbation), in order to differentiate it from *robust accuracy* a.k.a. astuteness (with adversarial perturbation).

Local Lipschitzness. Here we define local Lipschitzness theoretically; Section 5 later provides an empirical way to estimate this quantity.

Definition 1. *Let $(\mathcal{X}, \mathsf{dist})$ be a metric space. A function $f : \mathcal{X} \to \mathbb{R}^C$ is L-locally Lipschitz at radius r if for each $i \in [C]$, we have $|f(\boldsymbol{x})_i - f(\boldsymbol{x}')_i| \leq L \cdot \mathsf{dist}(\boldsymbol{x}, \boldsymbol{x}')$ for all $\boldsymbol{x}'$ with $\mathsf{dist}(\boldsymbol{x}, \boldsymbol{x}') \leq r$.*

Separation. We formally define separated data distributions as follows. Let $\mathcal{X}$ contain C disjoint classes $\mathcal{X}^{(1)}, \ldots, \mathcal{X}^{(C)}$, where all points in $\mathcal{X}^{(i)}$ have label i for $i \in [C]$.

Definition 2 (r-separation)**.** *We say that a data distribution over $\bigcup_{i \in [C]} \mathcal{X}^{(i)}$ is r-separated if $\mathsf{dist}(\mathcal{X}^{(i)}, \mathcal{X}^{(j)}) \geq 2r$ for all $i \neq j$, where $\mathsf{dist}(\mathcal{X}^{(i)}, \mathcal{X}^{(j)}) = \min_{\boldsymbol{x} \in \mathcal{X}^{(i)}, \boldsymbol{x}' \in \mathcal{X}^{(j)}} \mathsf{dist}(\boldsymbol{x}, \boldsymbol{x}')$.*

In other words, the distance between any two examples from different classes is at least $2r$. One of our motivating observations is that many real classification tasks comprise of separated classes; for example, if dist is the ℓ_∞ norm, then images with different categories (e.g., dog, cat, panda, etc) will be r-separated for a value $r > 0$ depending on the image space (see Figure 1). In the next section, we empirically verify that this property actually holds for a number of standard image datasets.

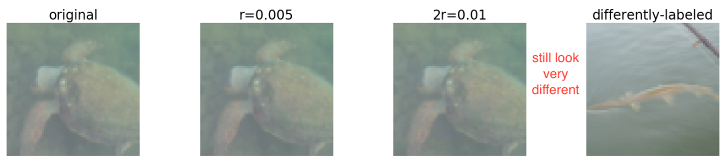


Figure 1: Intuitive example of r-separated images from Restricted ImageNet [57]. Even with a $2r$-perturbation, classes do not overlap. Moving $2r$ from turtle to fish still looks more like a turtle.

3 Real Image Datasets are r-Separated

We begin by addressing the question: Are image datasets r-separated for $\varepsilon \ll r$ and attack radii ε in standard robustness experiments? While we cannot determine the underlying data distribution, we can empirically measure whether current training and test sets are r-separated. These measurements can potentially throw light on what can be achieved in terms of test robustness in real data.

We consider four datasets: MNIST, CIFAR-10, SVHN and Restricted ImageNet (ResImageNet), where ResImageNet contains images from a subset of ImageNet classes [32, 45, 57]. We present two statistics in Table 1 The *Train-Train Separation* is the ℓ_∞ distance between each training example and its closest neighbor with a different class label in the training set, while the *Test-Train Separation* is the ℓ_∞ distance between each test example and its closest example with a different class in the training set. See Figure 3 for histograms. We use exact nearest neighbor search to calculate distances. Table 1 also shows the typical adversarial attack radius ε for the datasets; more details are in Appendix D.

	adversarial perturbation ε	minimum Train-Train separation	minimum Test-Train separation
MNIST	0.1	0.737	0.812
CIFAR-10	0.031	0.212	0.220
SVHN	0.031	0.094	0.110
ResImageNet	0.005	0.180	0.224

Table 1: Separation of real data is $3\times$ to $7\times$ typical perturbation radii.

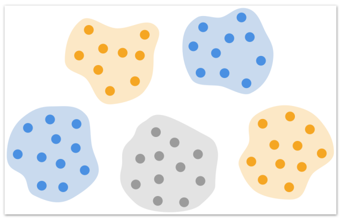
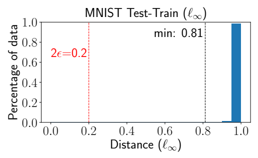
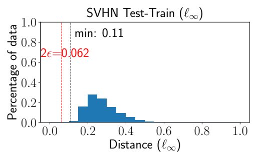


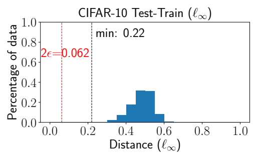
Figure 2: Robust classifers exist if the perturbation is less than the separation.



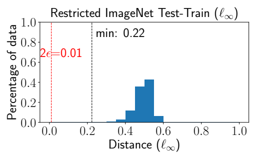
(a) MNIST



(b) SVHN



(c) CIFAR-10



(d) Restricted ImageNet

Figure 3: Train-Test separation histograms: MNIST, SVHN, CIFAR-10 and Restricted ImageNet.

Both the Train-Train and Test-Train separations are higher than 2ε for all four datasets. We note that SVHN contains a single duplicate example with multiple labels, and one highly noisy example; removing these three examples out of $73,257$ gives us a minimum Train-Train Separation of 0.094, which is more than enough for attack radius $\varepsilon = 0.031 \approx 8/255$. Restricted ImageNet is similar with three pairs of duplicate examples, and two other highly noisy training examples (see Figures 7 and 8 in Appendix D). Barring a handful of highly noisy examples, real image datasets are indeed r-separated when r is equal to the attack radii commonly used in adversarial robustness experiments.

These results imply that in real image data, the test images are far apart from training images from a different class. There perhaps are images of dogs which look like cats, but standard image datasets are quite clean, and such images mostly do not occur in either their test nor the training sets. In the next section, we explore consequences of this separation.

4 Robustness and Accuracy for r-Separated Data

We have just shown that four image datasets are indeed r-separated, for $\varepsilon \ll r$ where ε is the typical adversarial perturbation used in experiments. We now show theoretically that if a data distribution is r-separated, then there exists a robust and accurate classifier that can be obtained by rounding a locally Lipschitz function. Additionally, we supplement these results in Appendix C by a constructive "existence proof" that demonstrates proof-of-concept neural networks with both high accuracy and robustness on some of these datasets; this illustrates that at least on these image datasets, these classifiers can potentially be achieved by neural networks.

4.1 r-Separation implies Robustness and Accuracy through Local Lipschitzness

We show that it is theoretically possible to achieve both robustness and accuracy for r-separated data. In particular, we exhibit a classifier based on a locally Lipschitz function, which has astuteness 1 with radius r. Working directly in the multiclass case, our proof uses classifiers of the following form. If there are C classes, we start with a vector-valued function $f : \mathcal{X} \to \mathbb{R}^C$ so that $f(\boldsymbol{x})$ is a C-dimensional real vector. We let $\mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(i)}) = \min_{\mathbf{z}\in\mathcal{X}^{(i)}} \mathsf{dist}(\boldsymbol{x}, \mathbf{z})$. We analyze the following function

$$f(\boldsymbol{x}) = \frac{1}{r} \cdot \left(\mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(1)}), \ldots, \mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(C)})\right). \tag{1}$$

In other words, we set $f(\boldsymbol{x})_i = \frac{1}{r} \cdot \mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(i)})$. Then, we define a classifier $g : \mathcal{X} \to [C]$ as

$$g(\boldsymbol{x}) = \operatorname*{argmin}_{i\in[C]} f(\boldsymbol{x})_i. \tag{2}$$

We show that accuracy and local Lipschitzness together imply astuteness (proofs in Appendix A).

Lemma 4.1. *Let $f : \mathcal{X} \to \mathbb{R}^C$ be a function, and consider $\boldsymbol{x} \in \mathcal{X}$ with true label $y \in [C]$. If*

- *f is $\frac{1}{r}$-Locally Lipschitz in a radius r around $\boldsymbol{x}$, and*
- *$f(\boldsymbol{x})_j - f(\boldsymbol{x})_y \geq 2$ for all $j \neq y$,*

then $g(\boldsymbol{x}) = \operatorname{argmin}_i f(\boldsymbol{x})_i$ is astute at $\boldsymbol{x}$ with radius r.

Finally, we show that there exists an astute classifier when the distribution is r-separated.

Theorem 4.2. *Suppose the data distribution $\mathcal{X}$ is r-separated, denoting C classes $\mathcal{X}^{(1)}, \ldots, \mathcal{X}^{(C)}$. There exists a function $f : \mathcal{X} \to \mathbb{R}^C$ such that*

(a) f is $\frac{1}{r}$-locally-Lipschitz in a ball of radius r around each $\boldsymbol{x} \in \bigcup_{i\in[C]} \mathcal{X}^{(i)}$, and

(b) the classifier $g(\boldsymbol{x}) = \operatorname{argmin}_i f(\boldsymbol{x})_i$ has astuteness 1 with radius r.

While the classifier g used in the proof of Theorem 4.2 resembles the 1-nearest-neighbor classifier, it is actually different on any finite sample, and the classifiers only coincide in the *infinite sample limit* or when the class supports are known.

Binary case. We also state results for the special case of binary classification. Let $\mathcal{X} = \mathcal{X}^+ \cup \mathcal{X}^-$ be the instance space with disjoint class supports $\mathcal{X}^+ \cap \mathcal{X}^- = \emptyset$. Then, we define $f : \mathcal{X} \to \mathbb{R}$ as

$$f(\boldsymbol{x}) = \frac{\mathsf{dist}(\boldsymbol{x}, \mathcal{X}^-) - \mathsf{dist}(\boldsymbol{x}, \mathcal{X}^+)}{2r}.$$

It is immediate that if $\mathcal{X}$ is r-separated, then f is locally Lipschitz with constant $1/r$, and also the classifier $g = \mathsf{sign}(f)$ achieves the guarantees in the following theorem using the next lemma.

Lemma 4.3. *Let $f : \mathcal{X} \to \mathbb{R}$, and let $\boldsymbol{x} \in \mathcal{X}$ have label y. If (a) f is $\frac{1}{r}$-Locally Lipschitz in a ball of radius $r > 0$ around $\boldsymbol{x}$, (b) $|f(\boldsymbol{x})| \geq 1$, and (c) $g(\boldsymbol{x})$ has the same sign as y, then the classifier $g = \mathsf{sign}(f)$ is astute at $\boldsymbol{x}$ with radius r.*

Theorem 4.4. *Suppose the data distribution $\mathcal{X} = \mathcal{X}^+ \cup \mathcal{X}^-$ is r-separated. Then, there exists a function f such that (a) f is $\frac{1}{r}$-locally Lipschitz in a ball of radius r around all $\boldsymbol{x} \in \mathcal{X}$ and (b) the classifier $g = \mathsf{sign}(f)$ has astuteness 1 with radius r.*

A visualization of the function (and resulting classifier) from Theorem 4.4 for a binary classification dataset appears in Figure 4. Dark colors indicate high confidence (far from decision boundary) and lighter colors indicate the gradual change from one label to the next. The classifier $g = \mathsf{sign}(f)$ guaranteed by this theorem will predict the label based on which decision region (positive or negative) is closer to the input example. Figure 5 shows a pictorial example of why using a locally Lipschitz function can be just as expressive while also being robust.

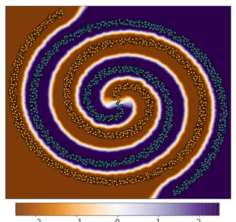


Figure 4: Plot of $f(\boldsymbol{x})$ from Theorem 4.4 for the spiral dataset. The classifier $g = \mathsf{sign}(f)$ has high accuracy and astuteness because it gradually changes near the decision boundary.

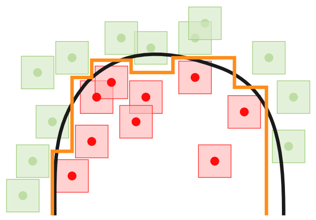


Figure 5: The classifier corresponding to the orange boundary has small local Lipschitzness because it does not change in the ℓ_∞ balls around data points. The black curve, however, is vulnerable to adversarial examples even though it has high clean accuracy.

4.2 Implications

We consider the consequences of Table 1 and Theorem 4.2 taken together. Then, in Section 5, we empirically explore the limitations of current robustness techniques.

Significance for real data. Theorem 4.2 refers to supports of distributions, while our measurements in Table 1 are on actual data. Hence, the results do not imply perfect *distributional* accuracy and robustness. However, our test set measurements suggest that even if the distributional supports may be close in the infinite sample limit, the close images are rare enough that we do not see them in the test sets. Thus, we still expect high accuracy and robustness on these test sets. Additionally, if we are willing to assume that the data supports are representative of the support of the distribution, then we can conclude the existence of a distributionally robust and accurate classifier. Combined with proof-of-concept results in Appendix C, we deduce that these classifiers can be implemented by neural networks. The remaining question is how such networks can be *trained* with existing methods.

Optimally astute classifier and non-parametrics (comparison to Yang et al. [62]). Prior work proposes adversarial pruning, a method that removes training examples until different classes are r-separated. They exhibit connections to maximally astute classifier, which they call the r-Optimal classifier for size r perturbations [62]. Follow-up work proved that training various non-parametric classifiers after pruning leads them to converge to maximally astute classifiers under certain conditions [5]. Our result in Theorem 4.2 complements these efforts, showing that the r-Optimal classifier can be obtained by the classifier in Theorem 4.2. Moreover, we provide additional justification for adversarial pruning by presenting a new perspective on the role of data separation in robustness.

Lower bounds on test error. Our results also corroborate some recent works that use optimal transport to estimate a lower bound on the robust test accuracy of any classifier on standard datasets. They find that it is actually zero for typical perturbation sizes [4, 38]. In other words, we have further evidence that well-curated benchmark datasets are insufficient to demonstrate a tradeoff between robustness and accuracy, in contrast to predictions of an inevitable tradeoff [11, 12, 16, 57].

Robustness is *not* inherently at odds with accuracy (comparison to Tsipras et al. [57]). Prior work provides a theoretical example of a data distribution where any classifier with high test accuracy must also have small adversarial accuracy under ℓ_∞ perturbations. Their theorem led the authors to posit that (i) accuracy and robustness may be unachievable together due their inherently opposing goals, and (ii) the training algorithm may not be at fault [57]. We provide an alternative view.

Their distribution is defined using the following sampling process: the first feature is the binary class label (flipped with a small probability), and the other $d-1$ features are sampled from one of two $(d-1)$-dimensional Gaussian distributions either with mean r or $-r$ depending on the true example label. While the means are separated with distance $2r$, their distribution is *not* r-separated due to the noise in the first feature combined with the infinite support of the Gaussians. Their lower bound is tight and only holds for ℓ_∞ perturbations ε satisfying $\varepsilon \geq 2r$. Our experiments in Section 3 have already shown that r-separation is a realistic assumption, and typical perturbations ε satisfy $\varepsilon \ll r$. Taken together with Theorem 4.2, we conclude that the robustness-accuracy tradeoff in neural networks and image classification tasks is *not intrinsic*.

5 A Closer Look at Existing Training Methods

So far we have shown that robustness and accuracy should both be achievable in principle, but practical networks continue to trade robustness off for accuracy. We next empirically investigate why this tradeoff might arise. One plausible reason might be that existing training methods do not impose local Lipschitzness properly; another may be that they do not generalize enough. We next explore these hypotheses in more detail, considering the following questions:

- How locally Lipschitz are the classifiers produced by existing training methods?
- How well do classifiers produced by existing training methods generalize?

These questions are considered in the context of one synthetic and four real datasets, as well as several plausible training methods for improving adversarial robustness. We do not aim to achieve best performance for any method, but rather to understand smoothness and generalization.

5.1 Experimental Methodology

We evaluate train/test accuracy, adversarial accuracy and local lipschitzness of neural networks trained using different methods. We also measure generalization gaps: the difference between train and test clean accuracy (or between train and test adversarial accuracy).

Training Methods. We consider neural networks trained via Natural training (Natural), Gradient Regularization (GR) [14], Locally Linear Regularization (LLR) [40], Adversarial Training (AT) [32], and TRADES [65]. Additionally, we use Robust Self Training (RST) [42], a recently introduced method that minimizes a linear combination of clean and robust accuracy in an attempt to improve robustness-accuracy tradeoffs. For fair comparison between methods, we use a version of RST that only uses labeled data. Both RST and TRADES have a parameter; for RST higher λ means higher weight is given to the robust accuracy, while for TRADES higher β means higher weight given to enforcing local Lipschitzness. Details are provided in Appendix B.

Adversarial Attacks. We evaluate robustness with two attacks. In this section, we use Projected gradient descent (PGD) [25] for adversarial accuracy with step size $\varepsilon/5$ and a total of 10 steps. The Multi-Targeted Attack (MT) [18] leads to similar conclusions; results in Appendix E.1.

Measuring Local Lipschitzness. For each classifier, we empirically measure the local Lipschitzness of the underlying function by the *empirical Lipschitz constant* defined as the following quantity

$$\frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{x}_i' \in \mathbb{B}_\infty(\boldsymbol{x}_i,\varepsilon)} \frac{\|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_i')\|_1}{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_\infty}. \tag{3}$$

A lower value of the empirical Lipschitz constant implies a smoother classifier. We estimate this through a PGD-like procedure, where we iteratively take a step towards the gradient direction $(\nabla_{\boldsymbol{x}_i'} \frac{\|f(\boldsymbol{x}_i)-f(\boldsymbol{x}_i')\|_1}{\|\boldsymbol{x}_i-\boldsymbol{x}_i'\|_\infty})$ where ε is the perturbation radius. We use step size $\varepsilon/5$ and a total of 10 steps.

architecture	CNN1						CNN2					
	train acc.	test acc.	adv test acc.	test lipschitz	gap	adv gap	train acc.	test acc.	adv test acc.	test lipschitz	gap	adv gap
Natural	100.00	99.20	59.83	67.25	0.80	0.45	100.00	99.51	86.01	23.06	0.49	-0.28
GR	99.99	99.29	91.03	26.05	0.70	3.49	99.99	99.55	93.71	20.26	0.44	2.55
LLR	100.00	99.43	92.14	30.44	0.57	4.42	100.00	99.57	95.13	9.75	0.43	2.28
AT	99.98	99.31	97.21	8.84	0.67	2.67	99.98	99.48	98.03	6.09	0.50	1.92
RST(λ=.5)	100.00	99.34	96.53	11.09	0.66	3.16	100.00	99.53	97.72	8.27	0.47	2.27
RST(λ=1)	100.00	99.31	96.96	11.31	0.69	2.95	100.00	99.55	98.27	6.26	0.45	1.73
RST(λ=2)	100.00	99.31	97.09	12.39	0.69	2.87	100.00	99.56	98.48	4.55	0.44	1.52
TRADES(β=1)	99.81	99.26	96.60	9.69	0.55	2.10	99.96	99.58	98.10	4.74	0.38	1.70
TRADES(β=3)	99.21	98.96	96.66	7.83	0.25	1.33	99.80	99.57	98.54	2.14	0.23	1.18
TRADES(β=6)	97.50	97.54	93.68	2.87	-0.04	0.37	99.61	99.59	98.73	1.36	0.02	0.80

Table 2: MNIST (perturbation 0.1). We compare two networks: CNN1 (smaller) and CNN2 (larger). We evaluate adversarial accuracy with the PGD-10 attack and compute Lipschitzness with Eq. (3). We also report the standard and adversarial generalization gaps.

.

	CIFAR-10						Restricted ImageNet					
	train acc.	test acc.	adv test acc.	test lipschitz	gap	adv gap	train acc.	test acc.	adv test acc.	test lipschitz	gap	adv gap
Natural	100.00	93.81	0.00	425.71	6.19	0.00	97.72	93.47	7.89	32228.51	4.25	-0.46
GR	94.90	80.74	21.32	28.53	14.16	3.94	91.12	88.51	62.14	886.75	2.61	0.19
LLR	100.00	91.44	22.05	94.68	8.56	4.50	98.76	93.44	52.62	4795.66	5.32	0.22
RST(λ=.5)	99.90	85.11	39.58	20.67	14.79	36.26	96.08	92.02	79.24	451.57	4.06	4.57
RST(λ=1)	99.86	84.61	40.89	23.15	15.25	41.31	95.66	92.06	79.69	355.43	3.61	4.67
RST(λ=2)	99.73	83.87	41.75	23.80	15.86	43.54	96.02	91.14	81.41	394.40	4.87	6.19
AT	99.84	83.51	43.51	26.23	16.33	49.94	96.22	90.33	82.25	287.97	5.90	8.23
TRADES(β=1)	99.76	84.96	43.66	28.01	14.80	44.60	97.39	92.27	79.90	2144.66	5.13	6.66
TRADES(β=3)	99.78	85.55	46.63	22.42	14.23	47.67	95.74	90.75	82.28	396.67	5.00	6.41
TRADES(β=6)	98.93	84.46	48.58	13.05	14.47	42.65	93.34	88.92	82.13	200.90	4.42	5.31

Table 3: CIFAR-10 (perturbation 0.031) and Restricted ImageNet (perturbation 0.005). We evaluate adversarial accuracy with the PGD-10 attack and compute Lipschitzness with Eq. (3).

Datasets. We evaluate the various algorithms on one synthetic dataset: Staircase [41] and four real datasets: MNIST [26], SVHN [33], CIFAR-10 [24] and Restricted ImageNet [57]. We consider adversarial ℓ_∞ perturbations for all datasets. More details are in Appendix B.

5.2 Observations

Our experimental results, presented in Tables 2 and 3, provide a number of insights into the smoothness and generalization properties of classifiers trained by existing methods.

How well do existing methods impose local Lipschitzness? There is a large gap in the degree of local Lipschitzness in classifiers trained by AT, RST and TRADES and those trained by natural training, GR and LLR. Classifiers in the former group are considerably smoother than the latter. Classifiers produced by TRADES are the most locally Lipschitz overall, with smoothness improving with increasing β. AT and RST also produce classifiers of comparable smoothness – but less smooth than TRADES. Overall, local Lipschitzness appears mostly correlated with adversarial accuracy; the more robust methods are also the ones that impose the highest degree of local Lipschitzness. But there are diminishing returns in the correlation between robustness *and* accuracy and local Lipschitzness; for example, the local smoothness of TRADES improves with higher β; but increasing β sometimes leads to drops in test accuracy even though the Lipschitz constant continues to decrease.

How well do existing methods generalize? We observe that for the methods that produce locally Lipschitz classifiers – namely, AT, TRADES and RST – also have large generalization gaps while natural training, GR and LLR generalize much better. In particular, there is a large gap between training and test accuracies of AT, RST and TRADES, and an even larger one between training and test adversarial accuracies. Although RST has better test accuracy than AT, it continues to have a large generalization gap with only labeled data. An interesting fact is that this generalization behaviour is quite unlike linear classification, where imposing local Lipschitzness leads to higher margin and better generalization [61] – imposing local Lipschitzness in neural networks, at least through these methods, appears to hurt generalization instead of helping. This suggests that these robust training methods may not be generalizing properly.

		SVHN					CIFAR-10				
	dropout	test acc.	adv test acc.	test lipschitz	gap	adv gap	test acc.	adv test acc.	test lipschitz	gap	adv gap
Natural	False	95.85	2.66	149.82	4.15	0.87	93.81	0.00	425.71	6.19	0.00
Natural	True	96.66	1.52	152.38	3.34	1.22	93.87	0.00	384.48	6.13	0.00
AT	False	91.68	54.17	16.51	5.11	25.74	83.51	43.51	26.23	16.33	49.94
AT	True	93.05	57.90	11.68	-0.14	6.48	85.20	43.07	31.59	14.51	44.05
RST(λ=2)	False	92.39	51.39	23.17	6.86	36.02	83.87	41.75	23.80	15.86	43.54
RST(λ=2)	True	95.19	55.22	17.59	1.90	11.30	85.49	40.24	34.45	14.00	33.07
TRADES(β=3)	False	91.85	54.37	10.15	7.48	33.33	85.55	46.63	22.42	14.23	47.67
TRADES(β=3)	True	94.00	62.41	4.99	0.48	7.91	86.43	49.01	14.69	12.59	35.03
TRADES(β=6)	False	91.83	58.12	5.20	5.35	23.88	84.46	48.58	13.05	14.47	42.65
TRADES(β=6)	True	93.46	63.24	3.30	0.45	5.97	84.69	52.32	8.13	11.91	26.49

Table 4: Dropout and generalization. SVHN (perturbation 0.031, dropout rate 0.5) and CIFAR-10 (perturbation 0.031, dropout rate 0.2). We evaluate adversarial accuracy with the PGD-10 attack and compute Lipschitzness with Eq. (3).

5.3 A Closer Look at Generalization

A natural follow-up question is whether the generalization gap of existing methods can be reduced by existing generalization-promoting methods in deep learning. In particular, we ask: *Can we improve the generalization gap of AT, RST and TRADES through generalization tools?*

To better understand this question, we consider two medium-sized datasets, SVHN and CIFAR-10, which nevertheless have a reasonably high gap between the test accuracy of the model produced by natural training and the best robust model. We then experiment with dropout [54], a standard and highly effective generalization method. For SVHN, we use a dropout rate of 0.5 and for CIFAR-10 a rate of 0.2. More experimental details are provided in the Appendix B.

Table 4 shows the results, contrasted with standard training. We observe that dropout narrows the generalization gap between training and test accuracy, as well as adversarial training and test accuracy significantly for all methods. For SVHN, after incorporating dropout, the best test accuracy is achieved by RST (95.19%) along with an adversarial test accuracy of 55.22%; the best adversarial test accuracy (62.41%) is with TRADES ($\beta = 3$) along with a test accuracy of (94.10%). Both accuracies are much closer to the accuracy of natural training (96.66%), and the test adversarial accuracies are also significantly higher. A similar narrowing of the generalization gap is visible for CIFAR-10 as well. Dropout also appears to make the networks smoother as test Lipschitzness also appears to improve for all algorithms for SVHN, and for TRADES for CIFAR-10.

Dropout Improvements. Our results show that the generalization gap of AT, RST and TRADES can be reduced by adding dropout; this reduction is particularly effective for RST and TRADES. Dropout additionally decreases the test local Lipschitzness of all methods – and hence promotes generalization all round – in accuracy, adversarial accuracy, and also local Lipschitzness. This suggests that combining dropout with the robust methods may be a good strategy for overall generalization.

5.4 Implications

Our experimental results lead to three major observations. We see that the training methods that produce the smoothest and most robust classifiers are AT, RST and TRADES. However, these robust methods also do not generalize well, and the generalization gap narrows when we add dropout.

Comparison with Rice et al. [43]. An important implication of our results is that generalization is a particular challenge for existing robust methods. The fact that AT may sometimes overfit has been previously observed by [41, 43, 55]; in particular, Rice et al. [43] also experiments with a few generalization methods (but *not* dropout) and observes that only early stopping helps overcome overfitting to a certain degree. We expand the scope of these results to show that RST and TRADES also suffer from large generalization gaps, and that dropout can help narrow the gap in these two methods. Furthermore, we demonstrate that dropout often decreases the local Lipschitz constant.

6 Related Work

There is a large body of literature on developing adversarial attacks as well as defenses that apply to neural networks [7, 29, 56, 30, 32, 36, 35, 53, 58, 67]. While some of this work has noted that an increase in robustness is sometimes accompanied by a loss in accuracy, the phenomenon remains ill-understood. Raghunathan et al. [41] provides a synthetic problem where adversarial training overfits, which we take a closer look at in Appendix E. Raghunathan et al. [42] proposes the robust self training method that aims to improve the robustness and accuracy tradeoff; however, our experiments show that they do not completely close the gap particularly when using only labeled data.

Outside of neural networks, prior works suggest that lack of data may be responsible for low robustness [1, 5, 8, 9, 28, 48, 49, 59, 66]. For example, Schmidt et al. [48] provides an example of a linear classification problem where robustness in ℓ_∞ norm requires more samples than plain accuracy, and Wang et al. [59] shows that nearest neighbors would be more robust with more data. Some prior works also suggest that the robustness accuracy tradeoff may arise due to limitations of existing algorithms. Bubeck et al. [6] provides an example where finding a robust and accurate classifier is significantly more computationally challenging than finding one that is simply accurate, and Bhattacharjee and Chaudhuri [5] shows that certain non-parametric methods do not lead to robust classifiers in the large sample limit. It is also known that the Bayes optimal classifier is not robust in general [4, 44, 38, 57, 59], and it differs from the maximally astute classifier [5, 62].

Prior work has also shown a connection between adversarial robustness and local or global Lipschitzness. For linear classifiers, it is known that imposing Lipschitzness reduces to bounding the norm of the classifier, which in turn implies large margin [31, 61]. Thus, for linear classification of data that is linearly separable with a large margin, imposing Lipschitzness *helps* generalization.

For neural networks, Anil et al. [2], Qian and Wegman [39], Huster et al. [22] provide methods for imposing global Lipschitzness constraints; however, the state-of-the-art methods for training such networks do not lead to highly expressible functions. For local Lipschitzness, Hein and Andriushchenko [21] first showed a relationship between adversarial robustness of a classifier and local Lipschitzness of its underlying function. Following this, Weng et al. [60] provides an efficient way of calculating a lower bound on the local Lipschitzness coefficient. Many works consider a randomized notion of local smoothness, and they prove that enforcing it can lead to certifiably robust classifiers [27, 10, 37, 46].

7 Conclusion

Motivated by understanding when it is possible to achieve both accuracy and robustness, we take a closer look at robustness in image classification and make several observations. We show that many image datasets follow a natural separation property and that this implies the existence of a robust and perfectly accurate classifier that can be obtained by rounding a locally Lipschitz function. Thus in principle robustness and accuracy should both be achievable together on real world datasets.

We investigate the gap between theory and practice by examining the smoothness and generalization properties of existing training methods. Our results show that generalization may be a particular challenge in robust learning since all robust methods that produce locally smooth classifiers also suffer from fairly large generalization gaps. We then experiment with combining robust classification methods with dropout and see that this leads to a narrowing of the generalization gaps.

Our results suggest that the robustness-accuracy tradeoff in deep learning is not inherent, but it is rather a consequence of current methods for training robust networks. Future progress that ensures both robustness and accuracy may come from redesigning other aspects of the training process, such as customized optimization methods [3, 15, 25, 34, 43, 50, 52, 51] or better network architectures [13, 47, 69] in combination with loss functions that encourage adversarial robustness, generalization, and local Lipschitzness. Some recent evidence for improved network architectures has been obtained by Guo et. al. [19], who search for newer architectures with higher robustness from increased model capacity and feature denoising. A promising direction is to combine such efforts across the deep learning stack to reduce standard and adversarial generalization gaps.

Broader Impact

In this paper we have investigated when it is possible to achieve both high accuracy and adversarial robustness on standard image classification datasets. Our motivation is partially to offer an alternative perspective to previous work that speculates on the inevitability of an accuracy-robustness tradeoff. In practice, if there were indeed a tradeoff, then robust machine learning technology is unlikely to be very useful. The vast majority of instances encountered by practical systems will be natural examples, whereas adversaries are few and far between. A self-driving car will mostly come across regular street signs and rarely come across adversarial ones. If increased robustness necessitates a loss in performance on natural examples, then the system's designer might be tempted to use a highly accurate classifier that is obtained through regular training and forgo robustness altogether. For adversarially robust machine learning to be widely adopted, accuracy needs to be achieved in conjunction with robustness.

While we have backed up our theoretical results by empirically verifying dataset separation, we are also ready to point out the many limitations of current robustness studies. The focus on curated benchmarks may lead to a false sense of security. Real life scenarios will likely involve much more complicated classification tasks. For example, the identification of criminal activity or the maneuvering of self-driving cars depend on a much broader notion of robustness than has been studied so far. Perturbations in ℓ_p distance cover only a small portion of the space of possible attacks.

Looking toward inherent biases, we observe that test accuracy is typically aggregated over all classes, and hence, it does not account for underrepresentation. For example, if a certain class makes up a negligible fraction of the dataset, then misclassifying these instances may be unnoticeable when we expect a drop in overall test accuracy. A more stringent objective would be to retain accuracy on each separate class, as well as being robust to targeted perturbations that may exploit dataset imbalance.

On a more positive note, we feel confident that developing a theoretically grounded discussion of robustness will encourage machine learning engineers to question the efficacy of various methods. As one of our contributions, we have shown that dataset separation guarantees the existence of an accurate and robust classifier. We believe that future work will develop new methods that achieve robustness by imposing both Lipschitzness and effective generalization. Overall, it is paramount to close the theory-practice gap by working on both sides, and we stand by our suggestion to further investigate the various deep learning components (architecture, loss function, training method, etc) that may compound the perceived gains in robustness and accuracy.

Acknowledgments and Disclosure of Funding

Kamalika Chaudhuri and Yao-Yuan Yang thank NSF under CIF 1719133, CNS 1804829 and IIS 1617157 for support. Hongyang Zhang was supported in part by the Defense Advanced Research Projects Agency under cooperative agreement HR00112020003. The views expressed in this work do not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred. Approved for public release; distribution is unlimited. This work was also supported in part by NSF IIS1763562 and ONR Grant N000141812861.

References

[1] Jean-Baptiste Alayrac, Jonathan Uesato, Po-Sen Huang, Alhussein Fawzi, Robert Stanforth, and Pushmeet Kohli. Are labels required for improving adversarial robustness? In *Advances in Neural Information Processing Systems*, pages 12192–12202, 2019.

[2] Cem Anil, James Lucas, and Roger Grosse. Sorting out lipschitz function approximation. In *International Conference on Machine Learning*, 2019.

[3] Pranjal Awasthi, Abhratanu Dutta, and Aravindan Vijayaraghavan. On robustness to adversarial examples and polynomial optimization. In *Advances in Neural Information Processing Systems*, pages 13737–13747, 2019.

[4] Arjun Nitin Bhagoji, Daniel Cullina, and Prateek Mittal. Lower bounds on adversarial robustness from optimal transport. In *Advances in Neural Information Processing Systems*, pages 7496–7508, 2019.

[5] Robi Bhattacharjee and Kamalika Chaudhuri. When are non-parametric methods robust? *arXiv preprint arXiv:2003.06121*, 2020.

[6] Sébastien Bubeck, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. *arXiv preprint arXiv:1805.10204*, 2018.

[7] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. *IEEE Symposium on Security and Privacy*, 2017.

[8] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems*, pages 11190–11201, 2019.

[9] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. *arXiv preprint arXiv:2003.12862*, 2020.

[10] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320, 2019.

[11] Elvis Dohmatob. Generalized no free lunch theorem for adversarial robustness. In *International Conference on Machine Learning*, pages 1646–1654, 2019.

[12] Alhussein Fawzi, Hamza Fawzi, and Omar Fawzi. Adversarial vulnerability for any classifier. In *Advances in Neural Information Processing Systems*, pages 1186–1195, 2018.

[13] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

[14] Chris Finlay and Adam M Oberman. Scaleable input gradient regularization for adversarial robustness. *arXiv preprint arXiv:1905.11468*, 2019.

[15] Shivam Garg, Vatsal Sharan, Brian Zhang, and Gregory Valiant. A spectral view of adversarially robust features. In *Advances in Neural Information Processing Systems*, pages 10138–10148, 2018.

[16] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. Adversarial spheres. *arXiv preprint arXiv:1801.02774*, 2018.

[17] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[18] Sven Gowal, Jonathan Uesato, Chongli Qin, Po-Sen Huang, Timothy Mann, and Pushmeet Kohli. An alternative surrogate loss for PGD-based adversarial testing. *arXiv preprint arXiv:1910.09338*, 2019.

[19] Minghao Guo, Yuzhe Yang, Rui Xu, and Ziwei Liu. When nas meets robustness: In search of robust architectures against adversarial attacks. *arXiv preprint arXiv:1911.10695*, 2019.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, pages 2266–2276, 2017.

[22] Todd Huster, Cho-Yu Jason Chiang, and Ritu Chadha. Limitations of the lipschitz constant as a defense against adversarial examples. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 16–29, 2018.

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[24] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[25] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations*, 2017.

[26] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

[27] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified adversarial robustness with additive gaussian noise. *arXiv preprint arXiv:1809.03113*, 2018.

[28] Yiming Li, Baoyuan Wu, Yan Feng, Yanbo Fan, Yong Jiang, Zhifeng Li, and Shutao Xia. Toward adversarial robustness via semi-supervised robust training. *arXiv preprint arXiv:2003.06974*, 2020.

[29] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *ICLR*, 2017.

[30] Daniel Lowd and Christopher Meek. Adversarial learning. In *SIGKDD*, pages 641–647, 2005.

[31] Ulrike von Luxburg and Olivier Bousquet. Distance-based classification with Lipschitz functions. *Journal of Machine Learning Research*, 5:669–695, 2004.

[32] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[33] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[34] Alexander G Ororbia II, Daniel Kifer, and C Lee Giles. Unifying adversarial training algorithms with data gradient regularization. *Neural computation*, 29(4):867–887, 2017.

[35] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. *arXiv preprint arXiv:1511.04508*, 2015.

[36] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Berkay Celik, and Ananthram Swami. Practical black-box attacks against deep learning systems using adversarial examples. In *ASIACCS*, 2017.

[37] Rafael Pinot, Laurent Meunier, Alexandre Araujo, Hisashi Kashima, Florian Yger, Cédric Gouy-Pailler, and Jamal Atif. Theoretical evidence for adversarial robustness through randomization: the case of the exponential family. *arXiv preprint arXiv:1902.01148*, 2019.

[38] Muni Sreenivas Pydi and Varun Jog. Adversarial risk via optimal transport and optimal couplings. *arXiv preprint arXiv:1912.02794*, 2019.

[39] Haifeng Qian and Mark N. Wegman. L2-nonexpansive neural networks. *ArXiv*, abs/1802.07896, 2018.

[40] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. In *Advances in Neural Information Processing Systems*, pages 13824–13833, 2019.

[41] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C Duchi, and Percy Liang. Adversarial training can hurt generalization. *arXiv preprint arXiv:1906.06032*, 2019.

[42] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang. Understanding and mitigating the tradeoff between robustness and accuracy. *arXiv preprint arXiv:2002.10716*, 2020.

[43] Leslie Rice, Eric Wong, and J Zico Kolter. Overfitting in adversarially robust deep learning. *arXiv preprint arXiv:2002.11569*, 2020.

[44] Eitan Richardson and Yair Weiss. A bayes-optimal view on adversarial examples. *arXiv preprint arXiv:2002.08859*, 2020.

[45] Andrew Slavin Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. *arXiv preprint arXiv:1711.09404*, 2017.

[46] Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *Advances in Neural Information Processing Systems*, pages 11289–11300, 2019.

[47] Shreyas Saxena and Jakob Verbeek. Convolutional neural fabrics. In *Advances in Neural Information Processing Systems*, pages 4053–4061, 2016.

[48] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Mądry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems 31*, pages 5019–5031, 2018.

[49] Vikash Sehwag, Arjun Nitin Bhagoji, Liwei Song, Chawin Sitawarin, Daniel Cullina, Mung Chiang, and Prateek Mittal. Analyzing the robustness of open-world machine learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, pages 105–116, 2019.

[50] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

[51] Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 307:195–204, 2018.

[52] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In *Advances in Neural Information Processing Systems*, pages 10802–10813, 2018.

[53] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifiable Distributional Robustness with Principled Adversarial Training. In *ICLR*, 2018. URL `https://openreview.net/forum?id=Hk6kPgZA-`.

[54] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[55] Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao. Is robustness the cost of accuracy?–a comprehensive study on the robustness of 18 deep image classification models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 631–648, 2018.

[56] 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.

[57] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

[58] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*, 2020.

[59] Yizhen Wang, Somesh Jha, and Kamalika Chaudhuri. Analyzing the robustness of nearest neighbors to adversarial examples. In *International Conference on Machine Learning*, pages 5133–5142, 2018.

[60] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. Evaluating the robustness of neural networks: An extreme value theory approach. *arXiv preprint arXiv:1801.10578*, 2018.

[61] Huan Xu, Constantine Caramanis, and Shie Mannor. Robustness and regularization of support vector machines. *Journal of machine learning research*, 10(Jul):1485–1510, 2009.

[62] Yao-Yuan Yang, Cyrus Rashtchian, Yizhen Wang, and Kamalika Chaudhuri. Robustness for Non-Parametric Classification: A Generic Attack and Defense. In *AISTATS*, 2020.

[63] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference*, 2016.

[64] Runtian Zhai, Tianle Cai, Di He, Chen Dan, Kun He, John Hopcroft, and Liwei Wang. Adversarially robust generalization just requires more unlabeled data. *arXiv preprint arXiv:1906.00555*, 2019.

[65] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 2019.

[66] Jingfeng Zhang, Bo Han, Gang Niu, Tongliang Liu, and Masashi Sugiyama. Where is the bottleneck of adversarial learning with unlabeled data? *arXiv preprint arXiv:1911.08696*, 2019.

[67] Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. *arXiv preprint arXiv:2002.11242*, 2020.

[68] Xiao Zhang, Jinghui Chen, Quanquan Gu, and David Evans. Understanding the intrinsic robustness of image distributions using conditional generative models. *arXiv preprint arXiv:2003.00378*, 2020.

[69] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

A Proofs for Astute Classifier Existence Results

Proof of Lemma 4.1. Suppose $\boldsymbol{x}' \in \mathcal{X}$ satisfies $\mathsf{dist}(\boldsymbol{x}, \boldsymbol{x}') \leq r$. By the assumptions that f is $\frac{1}{r}$-locally Lipschitz and $f(\boldsymbol{x})_j - f(\boldsymbol{x})_y \geq 2$, we have that

$$f(\boldsymbol{x}')_j \geq f(\boldsymbol{x})_j - 1 \geq f(\boldsymbol{x})_y + 1 \geq f(\boldsymbol{x}')_y,$$

where the first and third inequalities use Lipschitzness, and the middle inequality uses that

$$f(\boldsymbol{x})_j - f(\boldsymbol{x})_y \geq 2.$$

As this holds for all $j \neq y$, we have that

$$\operatorname*{argmin}_i f(\boldsymbol{x}')_i = \operatorname*{argmin}_i f(\boldsymbol{x})_i = y.$$

Thus, we see that $g(\boldsymbol{x}) = \operatorname{argmin}_i f(\boldsymbol{x})_i$ correctly classifies $\boldsymbol{x}$ while being astute with radius r. $\square$

Proof of Theorem 4.2. We first show that if the support of the distribution is r-separated, then there exists a function $f : \mathcal{X} \to \mathbb{R}^C$ satisfying:

1. If $\boldsymbol{x} \in \bigcup_{i \in [C]} \mathcal{X}^{(i)}$, then, $f(\boldsymbol{x})$ is $\frac{1}{r}$-locally-Lipschitz in a ball of radius r around $\boldsymbol{x}$.
2. If $\boldsymbol{x} \in \mathcal{X}^{(y)}$, then $f(\boldsymbol{x})_j - f(\boldsymbol{x})_y \geq 2$ for all $j \neq y$.

Define the function

$$f(\boldsymbol{x}) = \frac{1}{r} \cdot \left(\mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(1)}), \ldots, \mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(C)})\right).$$

In other words, we set $f(\boldsymbol{x})_i = \frac{1}{r} \cdot \mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(i)})$. Then, for any $\boldsymbol{x}$, we have:

$$f(\boldsymbol{x})_i - f(\boldsymbol{x}')_i = \frac{\mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(i)}) - \mathsf{dist}(\boldsymbol{x}', \mathcal{X}^{(i)})}{r} \leq \frac{\mathsf{dist}(\boldsymbol{x}, \boldsymbol{x}')}{r}$$

where we used the triangle inequality. This establishes (a). To establish (b), suppose without loss of generality that $\boldsymbol{x} \in \mathcal{X}^{(y)}$, which in particular implies that $f(\boldsymbol{x})_y = \mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(y)}) = 0$. Then,

$$f(\boldsymbol{x})_j - f(\boldsymbol{x})_y = \frac{\mathsf{dist}(\boldsymbol{x}, \mathcal{X}^{(j)})}{r} \geq \frac{\mathsf{dist}(\mathcal{X}^{(y)}, \mathcal{X}^{(j)})}{r} \geq 2$$

because every pair of classes has distance at least $2r$ from the r-separated property.

Now observe that by construction, f satisfies the two conditions in Lemma 4.1 at all $\boldsymbol{x} \in \bigcup_{i \in [C]} \mathcal{X}^{(i)}$. Thus, applying Lemma 4.1, we get that $g(\boldsymbol{x}) = \operatorname{argmin}_i f(\boldsymbol{x})_i$ has astuteness 1 with radius r over any distribution over points in $\bigcup_{i \in [C]} \mathcal{X}^{(i)}$. $\square$

B Experimental Setup: More Details

Experiments run with NVIDIA GeForce RTX 2080 Ti GPUs. We report the number with a single run of experiment. The code for the experiments is available at `https://github.com/yangarbiter/robust-local-lipschitz`.

Synthetic Staircase setup. As a toy example, we first consider a synthetic regression dataset, which is known to show that adversarial training can seriously overfit when the sample size is too small [41]. We use the code provided by the authors to reproduce the result for natural training and AT, and we add results for GR, LLR, and TRADES. The model for this dataset is linear regression in a kernel space using cubic B-splines as the basis. Let $\mathcal{F}$ be the hypothesis set and the regularization term $\|f\|^2$ is the RKHS norm of the weight vector in the kernel space. The regularization term is set to $\lambda = 0.1$ and the result is evaluated using the mean squared error (MSE). For GR, we set $\beta = 10^{-4}$ and for LLR, we only use the local linearity γ for regularization and the regularization strength is 10^{-2}. The perturbation set $P(x, \varepsilon) = \{x - \varepsilon, x, x - \varepsilon\}$ considers only the point-wise perturbation.

MNIST setup. We use two different convolutional neural networks (CNN) with different capacity. The first CNN (CNN1) has two convolutional layers followed by two fully connected layer[2] and the second larger CNN (CNN2) has four convolutional layers followed by three fully connected layers[3]. We set the perturbation radius to 0.1. The network is optimized with SGD with momentum 0.9.

SVHN setup. We use the wide residual network WRN-40-10 [63] and set the perturbation radius to 0.031. The initial learning rate is set to 0.01 except LLR, AT and RST. We set the initial learning rate 0.001 for them. The network is optimized with SGD without momentum (the default setting in `pytorch`).

CIFAR10 setup. Following [32, 65], we use the wide residual network WRN-40-10 [63] and set the perturbation radius to 0.031. The initial learning rate is set to 0.01 except RST. We set the initial learning rate 0.001 for them. Data augmentation is performed. When performing data augmentation, we randomly crop the image to 32×32 with 4 pixels of padding then perform random horizontal flips. The network is optimized with SGD without momentum (the default setting in `pytorch`).

Restricted ImageNet setup. Following [57], we set the perturbation radius $\varepsilon = 0.005$, use the residual network (ResNet50) [20] and use Adam [23] to optimize. Data augmentation is performed: During training, we resize an image to 72×72 and randomly crop to 64×64 with 8 pixels padding. When evaluating, we resize the image to 72×72 and crop in the center resulting in a 64×64 image.

dataset	MNIST	SVHN	CIFAR10	Restricted ImageNet
network structure	CNN1 / CNN2	WRN-40-10	WRN-40-10	ResNet50
optimizer	SGD	SGD	SGD	Adam
batch size	64	64	64	128
perturbation radius	0.1	0.031	0.031	0.005
perturbation step size	0.02	0.0062	0.0062	0.001
# train examples	60000	73257	50000	257748
# test examples	10000	26032	10000	10150
# classes	10	10	10	9

Table 5: Experimental setup and parameters for the four real datasets that we test on in this paper. No weight decay is applied to the model.

Details on the network structure.

- CNN1 is retrieved from pytorch repository.[4]
- CNN2 is retrieved from TRADES [65] github repository.[5]
- WRN-40-10 represents the wide residual network [63] with depth equals to forty and widen factor equals to ten.
- ResNet50 represents the residual network with 50 layers [20].

Learning rate schedulers for each dataset

- MNIST: We run 160 epochs on the training dataset, where we decay the learning rate by a factor 0.1 in the 40th, 80th 120th and 140th epochs.
- SVHN: We run 60 epochs on the training dataset, where we decay the learning rate by a factor 0.1 in the 30th and 50th epochs.
- CIFAR10: We run 120 epochs on the training dataset, where we decay the learning rate by a factor 0.1 in the 40th, 80th and 100th epochs.
- Restricted ImageNet: We run 70 epochs on the training dataset, where we decay the learning rate by a factor 0.1 in the 40th and 60th epochs.

[2]CNN1 is retrieved from pytorch repository `https://github.com/pytorch/examples/blob/master/mnist/main.py`

[3]CNN2 is retrieved from TRADES [65] github repository `https://github.com/yaodongyu/TRADES/blob/master/models/small_cnn.py`

[4]`https://github.com/pytorch/examples/blob/master/mnist/main.py`

[5]`https://github.com/yaodongyu/TRADES/blob/master/models/small_cnn.py`

B.0.1 Spiral Dataset

Here we provide the details for generating the spiral dataset in Figure 4.

We take x as a uniform sample $[0, 4.33\pi]$, the noise level is set to 0.75 (uniform $[0, 0.75]$).

We construct the negative examples using the transform:

$$(-x\cos x + uniform(noise), x\sin x + uniform(noise))$$

We construct the positive examples using the transform:

$$(-x\cos x + uniform(noise), -x\sin x + uniform(noise))$$

B.0.2 Details on the baseline algorithms

Gradient Regularization (GR). The Gradient Regularization (GR) is in the form of soft regularization. We use the latest work by Finlay and Oberman [14] for our experiments. In general, GR models can be formulated as adding a regularization term on the norm of gradient of the loss function:

$$\min_f \mathbb{E}\Big\{\mathcal{L}(f(\mathbf{X}), Y) + \beta\|\nabla_{\mathbf{X}}\mathcal{L}(f(\mathbf{X}), Y)\|_2^2\Big\}.$$

Finlay and Oberman [14] compute the gradient term through a finite difference approximation. Let $d = \frac{\nabla f(\mathbf{X})}{\|\nabla f(\mathbf{X})\|_2}$ and h be the step size. Then,

$$\|\nabla f(\mathbf{X})\|_2^2 \approx \left(\frac{\mathcal{L}(f(\mathbf{X}+hd), Y) - \mathcal{L}(f(\mathbf{X}), Y)}{h}\right)$$

We use the publicly available implementation.[6]

Locally-Linear Regularization model (LLR). Qin et al. [40] propose to regularize the local linearity through the motivation that AT with PGD increases the model's local linearity. The authors first formulate the function g to evaluate the local linearity of a model.

$$g(f, \delta, \mathbf{X}) = |\mathcal{L}(f(\mathbf{X}+\delta), Y) - \mathcal{L}(f(\mathbf{X}), Y) - \delta^T\nabla_{\mathbf{X}}\mathcal{L}(f(\mathbf{X}), Y)|$$

Define $\gamma(\varepsilon, \mathbf{X}) = \mathbb{E}\Big\{\max_{\delta\in B(\mathbf{X},\varepsilon)} g(f, \delta, \mathbf{X})\Big\}$ and also $\delta_{LLR} = \mathbb{E}\Big\{\operatorname{argmax}_{\delta\in B(\mathbf{X},\varepsilon)} g(f, \delta, \mathbf{X})\}$. The loss function for Locally-Linear Regularization (LLR) model is

$$\mathbb{E}\Big\{\mathcal{L}(f(\mathbf{X}), Y) + \lambda\gamma(\varepsilon, \mathbf{X}) + \mu\|\delta_{LLR}^T\nabla_{\mathbf{X}}\mathcal{L}(f(\mathbf{X}), Y)\|\Big\}$$

We use our own implementation of LLR.

Adversarial training (AT). Adversarial training is a successful defense by Madry et al. [32] that trains based on adversarial examples:

$$\min_f \mathbb{E}\Big\{\max_{\mathbf{X}'\in\mathbb{B}(\mathbf{X},\varepsilon)} \mathcal{L}(f(\mathbf{X}'), Y)\Big\}. \tag{4}$$

Robust self-training (RST). Robust self-training is a defense proposed by Raghunathan et al. [42] to improve the tradeoff between standard and robust error that occurs with AT. The RST in the original paper includes the use unlabeled data. However, to be fair in the experiments, we considers only the supervised learning part. RST uses the following optimization problem for the loss function:

$$\min_f \mathbb{E}\Big\{\mathcal{L}(f(\mathbf{X}), Y) + \beta \max_{\mathbf{X}'\in\mathbb{B}(\mathbf{X},\varepsilon)} \mathcal{L}(f(\mathbf{X}'), Y)\Big\}.$$

Locally-Lipschitz models (TRADES). One of the best methods for robustness via smoothness is TRADES [65], which has been shown to obtain state-of-the-art adversarially accuracy in many cases. TRADES uses the following optimization problem for the loss function:

$$\min_f \mathbb{E}\Big\{\mathcal{L}(f(\mathbf{X}), Y) + \beta \max_{\mathbf{X}'\in\mathbb{B}(\mathbf{X},\varepsilon)} \mathcal{L}(f(\mathbf{X}), f(\mathbf{X}'))\Big\},$$

where the second term encourages local Lipschitzness. We use the publicly available implementation.[7]

[6] https://github.com/cfinlay/tulip

[7] https://github.com/yaodongyu/TRADES

C Proof-of-Concept Classifier

Our theoretical result in Theorem 4.2 leaves two concerns about practical solutions: (i) we need to verify that existing networks can achieve high robustness and accuracy, and (ii) we need training methods to converge to such solutions. For the first concern, we present proof-of-concept networks that use standard architectures, but they have an unfair advantage: they can use the test data. While this may seem unreasonable, we argue that the results in Table 1 provide multiple insights. This process is sufficient to establish the *existence* of a robust and accurate classifier. With access to the test data, current training methods can plausibly train a nearly-optimal robust classifier (we use robust self-training with λ=2). Finally, the robust accuracies can actually get close to 100% on MNIST, SVHN, and CIFAR-10. These insights reinforce our claim that robustness and accuracy are both achievable by neural networks on image classification tasks.

	perturbation ε	accuracy	adversarial accuracy
MNIST	0.1	99.99	99.98
SVHN	0.031	100.00	99.90
CIFAR-10	0.031	100.00	99.99

Table 6: Proof-of-concept: demonstrating a robust network trained with access to the test set.

	MNIST	SVHN	CIFAR10
network structure	CNN2	WRN-40-10	WRN-40-10
optimizer	SGD	Adam	Adam
batch size	128	64	64
epochs	40	60	60
perturbation radius	0.1	0.031	0.031
perturbation step size	0.02	0.0062	0.0062
initial learning rate	0.0001	0.001	0.01

Table 7: Setup for the Proof-of-Concept classifiers.

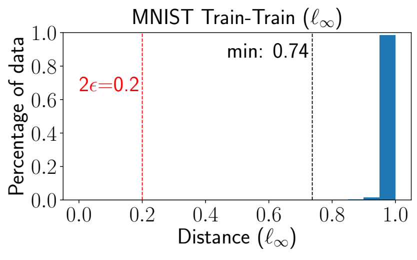
D Separation Experiment Results

Figures 6 shows the distribution of the train-train and test train separation for each dataset.

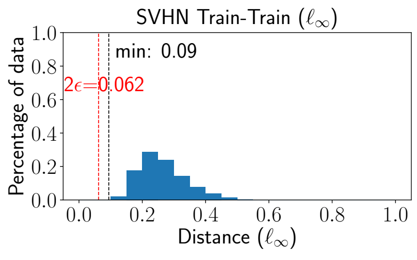
Random label. In addition, we conducted a random label experiment to show that in regular datasets, the distance between same-class examples are smaller than differently-labeled examples. Table 8 shows the minimum and average separation for randomly labeled and natural dataset. Figure 9 plots the minimum separation for these two dataset and clearly shows that natural dataset is more well-separated than random-labeled dataset.

	ε	randomly labeled				original labels			
		train-train		test-train		train-train		test-train	
		min	mean	min	mean	min	mean	min	mean
MNIST	0.100	0.231	0.902	0.290	0.904	0.737	0.990	0.812	0.990
CIFAR-10	0.031	0.125	0.476	0.098	0.475	0.212	0.479	0.220	0.479
SVHN	0.031	0.012	0.259	0.102	0.271	0.094	0.264	0.110	0.274
ResImageNet	0.005	0.000	0.485	0.000	0.483	0.180	0.492	0.224	0.492

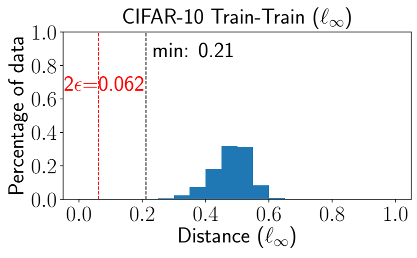
Table 8: Separation results on real datasets for both original labels and randomly assigned labels.



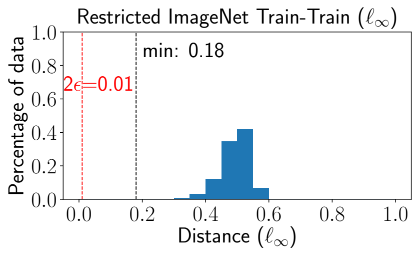
(a) MNIST train



(b) SVHN train

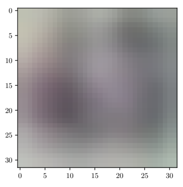


(c) CIFAR-10 train

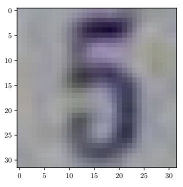


(d) Restricted ImageNet train

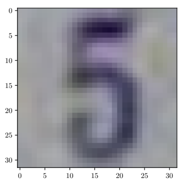
Figure 6: Train-Train separation histograms: MNIST, SVHN, CIFAR-10 and Restricted ImageNet.



(a)

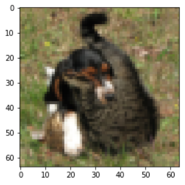


(b)

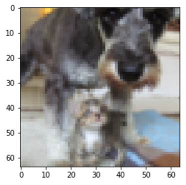


(c)

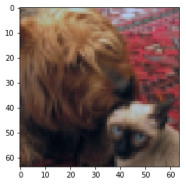
Figure 7: Images ignored in SVHN. The left most image appeared twice in the training set and one labeled as a one and the other one labeled as a five. The other two images are the closest images to each other in ℓ_∞ distance. The middle one is labeled as a five and the right most one is labeled as a one (which is clearly miss labeled).



(a)



(b)



(c)

Figure 8: Images ignored in Restricted ImageNet. These three images appeared twice in the training set and labeled differently (one labeled as a cat and the other one labeled as a dog).

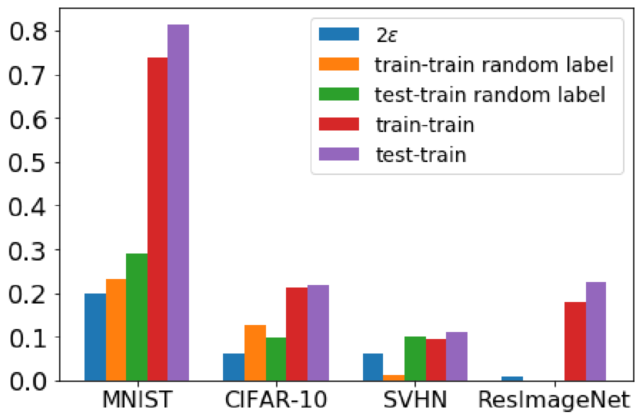


Figure 9: Separation results for four image datasets. We measure the separation for the original labels, and we also perform the experiment where we randomly label the test and train examples. We see that in MNIST, CIFAR-10, and ResImageNet, the separation diminishes quite a bit when using random labels. Indeed for ResImageNet, there are a number of duplicated examples that appear multiple times in the dataset. Overall, we conclude that the separation is much larger between *different* classes, while this is not the case *within* the same class.

E Further Experimental Results

E.1 Multi-targeted Attack Results

Certain prior works have suggested that the multi-targeted (MT) attack [18] is stronger than PGD. For example, the MT attack is highlighted as a selling point for LLR [40]. For completeness, we complement our empirical results from earlier by running all of the experiments using the MT attack. We run MT attack with 20 iterations for each target. Tables 9 to 14 provide the results.

We verify that our discussion about accuracy, robustness, and Lipschitzness remains valid using this attack. Comparing with the results using the PGD attack (Tables 2 to 4 above), the results with the MT attack gives a slightly lower adversarial test accuracy for all methods. The drop in accuracy is usually around 1–5%. This is within our expectation as this attack is regarded as a stronger attack than PGD.

The MT results still justify the previous discussion from Section 5 in general. Training methods leading to models with higher adversarial test accuracy are more locally smooth (smaller local Lipschitz constant during testing). Overall, we believe that seeing consistent results between PGD and MT only strengthens our argument that robustness requires some local Lipschitzness, and moreover, that the accuracy-robustness trade-off may not be necessary for separated data.

	train accuracy	test accuracy	adv test accuracy	test lipschitz	gap	adv gap
Natural	100.00	99.20	47.30	67.25	0.80	-0.53
GR	99.99	99.29	89.99	26.05	0.70	3.30
LLR	100.00	99.43	90.49	30.44	0.57	4.06
AT	99.98	99.31	97.23	8.84	0.67	2.65
RST(λ=.5)	100.00	99.34	96.46	11.09	0.66	3.22
RST(λ=1)	100.00	99.31	96.93	11.22	0.69	2.97
RST(λ=2)	100.00	99.31	97.00	12.39	0.69	2.95
TRADES(β=1)	99.81	99.26	96.53	9.69	0.55	2.12
TRADES(β=3)	99.21	98.96	96.60	7.83	0.25	1.34
TRADES(β=6)	97.50	97.54	93.54	2.86	-0.04	0.39

Table 9: MNIST on CNN1, multi-targeted attack

	train accuracy	test accuracy	adv test accuracy	test lipschitz	gap	adv gap
Natural	100.00	99.51	81.35	23.06	0.49	-0.87
GR	99.99	99.55	92.93	20.26	0.44	2.39
LLR	100.00	99.57	93.76	9.75	0.43	1.70
AT	99.98	99.48	98.01	6.09	0.50	1.94
RST(λ=.5)	100.00	99.53	97.69	8.27	0.47	2.30
RST(λ=1)	100.00	99.55	98.25	6.26	0.45	1.74
RST(λ=2)	100.00	99.56	98.46	4.56	0.44	1.53
TRADES(β=1)	99.96	99.58	98.06	4.74	0.38	1.73
TRADES(β=3)	99.80	99.57	98.54	2.14	0.23	1.18
TRADES(β=6)	99.61	99.59	98.73	1.36	0.02	0.81

Table 10: MNIST on CNN2, multi-targeted attack

	train accuracy	test accuracy	adv test accuracy	test lipschitz	gap	adv gap
Natural	100.00	95.85	1.06	149.82	4.15	0.43
GR	96.73	87.80	14.59	40.83	8.94	2.41
LLR	100.00	95.48	20.95	61.64	4.51	3.47
AT	95.20	92.45	49.47	13.03	2.75	14.96
RST(λ=.5)	99.99	93.09	45.98	19.56	6.90	27.26
RST(λ=1)	99.91	93.01	47.06	23.19	6.90	29.19
RST(λ=2)	99.25	92.39	47.58	23.18	6.86	29.99
TRADES(β=1)	98.96	92.45	46.40	18.75	6.51	29.22
TRADES(β=3)	99.33	91.85	49.41	10.15	7.48	32.70
TRADES(β=6)	97.19	91.83	52.82	5.20	5.35	24.28

Table 11: SVHN, multi-targeted attack

	train accuracy	test accuracy	adv test accuracy	test lipschitz	gap	adv gap
Natural	100.00	93.81	0.00	425.71	6.19	0.00
GR	94.90	80.74	19.15	28.53	14.16	2.88
LLR	100.00	91.44	14.58	94.68	8.56	1.32
AT	99.84	83.51	42.11	26.23	16.33	48.99
RST(λ=.5)	99.90	85.11	38.17	20.61	14.79	32.92
RST(λ=1)	99.86	84.61	39.29	22.92	15.25	38.84
RST(λ=2)	99.73	83.87	40.06	23.95	15.86	41.97
TRADES(β=1)	99.76	84.96	42.22	28.01	14.80	43.69
TRADES(β=3)	99.78	85.55	44.53	22.42	14.23	47.91
TRADES(β=6)	98.93	84.46	46.05	13.05	14.47	43.40

Table 12: CIFAR-10, multi-targeted attack

	train accuracy	test accuracy	adv test accuracy	test lipschitz	gap	adv gap
Natural	97.72	93.47	4.21	32228.51	4.25	-0.24
GR	91.12	88.51	60.61	886.75	2.61	-0.16
LLR	98.76	93.44	50.21	4795.66	5.32	-0.31
AT	96.22	90.33	81.91	287.97	5.90	8.27
RST($\lambda = .5$)	96.78	92.13	78.53	439.77	4.65	4.91
RST($\lambda = 1$)	95.61	92.06	79.33	366.21	3.55	4.68
RST($\lambda = 2$)	96.00	91.14	81.12	390.61	4.86	6.15
TRADES($\beta = 1$)	97.39	92.27	79.46	2144.66	5.13	6.61
TRADES($\beta = 3$)	95.74	90.75	82.00	396.67	5.00	6.35
TRADES($\beta = 6$)	93.34	88.92	81.90	200.90	4.42	5.28

Table 13: Restricted ImageNet, multi-targeted attack

		SVHN					CIFAR-10				
	dropout	test acc.	adv test acc.	test lipschitz	gap	adv gap	test acc.	adv test acc.	test lipschitz	gap	adv gap
Natural	False	95.85	1.06	149.82	4.15	0.87	93.81	0.00	425.71	6.19	0.00
Natural	True	96.66	1.52	152.38	3.34	1.22	93.87	0.00	384.48	6.13	0.00
AT	False	91.68	49.22	16.51	5.11	25.74	83.51	42.11	26.23	16.33	49.94
AT	True	93.05	52.44	11.68	-0.14	6.48	85.20	41.31	31.59	14.51	44.05
RST(λ=2)	False	92.39	47.58	23.17	6.86	36.02	83.87	40.06	23.80	15.86	43.54
RST(λ=2)	True	95.19	50.37	17.59	1.90	11.30	85.49	38.66	34.45	14.00	33.07
TRADES(β=3)	False	91.85	49.41	10.15	7.48	33.33	85.55	44.53	22.42	14.23	47.67
TRADES(β=3)	True	94.00	57.11	4.99	0.48	7.91	86.43	46.38	14.69	12.59	35.03
TRADES(β=6)	False	91.83	52.82	5.20	5.35	23.88	84.46	46.05	13.05	14.47	42.65
TRADES(β=6)	True	93.46	58.53	3.30	0.45	5.97	84.69	49.72	8.13	11.91	26.49

Table 14: Dropout and generalization. SVHN (perturbation 0.031, dropout rate 0.5) and CIFAR-10 (perturbation 0.031, dropout rate 0.2). We evaluate adversarial accuracy with the multi-targeted attack and compute Lipschitzness with Eq. (3).