Intriguing Properties of Robust Classification

Bernd Prach, Christoph H. Lampert
Institute of Science and Technology Austria (ISTA)
Klosterneuburg, Austria
bprach@ist.ac.at, chl@ist.ac.at

Abstract

Despite extensive research since the community learned about adversarial examples *10 years ago, we still do not know how to train high-accuracy classifiers that are guaranteed to be robust to small perturbations of their inputs. Previous works often argued that this might be because no classifier exists that is robust and accurate at the same time. However, in computer vision this assumption does not match reality where humans are usually accurate and robust on most tasks of interest. We offer an alternative explanation and show that in certain settings robust generalization is only possible with unrealistically large amounts of data. Specifically, we find a setting where a robust classifier exists, it is easy to learn an accurate classifier, yet it requires an exponential amount of data to learn a robust classifier. Based on this theoretical result, we evaluate the influence of the amount of training data on datasets such as CIFAR-10. Our findings indicate that the the amount of training data is the main factor determining the robust performance. Furthermore we show that that there are low magnitude directions in the data which are useful for non-robust generalization but are not available for robust classifiers. This implies that robust classification is a strictly harder tasks than normal classification, thereby providing an explanation why robust classification requires more data.*

1. Introduction

Deep learning has proven useful in numerous computer vision tasks, however, there are still shortcomings that come with these large end-to-end trained models. In particular, most state-of-the-art models suffer from adversarial examples [36], which are tiny perturbations of the input that can result in large changes to the output of such models. This phenomenon can negatively affect the trust of users in the models, it might constitute a security issue, and –because it contradicts human experience– it makes it impossible to create faithfully interpretable models.

Research on mitigation strategies has concentrated on three pillars: In *adversarial training* [15, 36] adversarial examples are created during training and the models are trained to classify those examples correctly. This procedure makes it harder to find adversarial examples, however, it cannot guarantee that no adversarial examples exist. In contrast, *randomized smoothing* [10] acts at prediction time. It mitigates the effect of adversarial inputs by repeatedly evaluating the network, each time with different noise added to the input. It then constructs a final prediction by combining the predictions, by a majority vote. This construction provides probabilistic robustness guarantees, however, usually several thousand predictions need to be performed for each input, which results in an undesirable slowdown. Finally, *Lipschitz networks* [9] prevent adversarial examples by constraining the network architecture such that only models with a small Lipschitz constant (typically equal to 1) can be learned. As a consequence, any input perturbation cannot cause a change in the network's output of larger magnitude than the perturbation itself, which yields deterministic and overhead-free guarantees on the presence of adversarial examples for any given input. This makes Lipschitz networks currently the most practical method for robust learning with guarantees.

Unfortunately, despite many years of research, robust networks still achieve results far worse than what one might hope for. Even on fairly simple datasets and for fairly small perturbations the robust accuracy is much worse than what we believe is possible. Furthermore, a recent large study [31] indicated that even architectures and training techniques that differ strongly in terms of their memory and computational demands, ultimately achieve quite similar robust accuracy values. This suggests the presence of a more fundamental barrier for the development of robust networks. Several explanations of this phenomenon have been put forward. For example, it has been suggested that there might be a natural trade-off between robustness and accuracy [37]: this would imply that high robust accuracy is just impossible to achieve. Alternatively, the hypothesis has been put forward that robust networks are not expressive enough [14, 26] or that the computational overhead of

training robust network is the limiting factor [7, 12]. At the same time, there exist also recent works that do report that higher robust accuracy is achievable if *additional training data* is exploited [1, 16, 18, 19, 40]. This would suggest that the problem is fundamentally one of *generalization*.

Overall, however, we still lack a solid understanding of what makes the task of robust classification difficult. Our main contributions in this work are three *insights* that we hope will clear up some misconceptions and hopefully guide future research on training robust networks in new directions.

Insight 1: There are settings in which learning robust accurate classifiers requires much more data than learning just accurate classifiers. Specifically, we present a learning problem in which any learning algorithm requires an amount of training data exponential in the data dimension, otherwise it cannot learn a robust classifier that is better than chance level. Our construction is based on the fact that non-robust classifiers are able to exploit low-magnitude features in the data, while robust classifiers have to rely on high-magnitude features. The exponential gap between robust and non-robust learning opens up when the former generalize well but the latter ones do not.

Insight 2: Also on real data, the amount of data is a major determinant of performance. We provide evidence that the problem of Insight 1 is not just theoretical, but happens in (less drastic) form also for real datasets. Specifically, for MNIST, CIFAR-10 and CIFAR-100 we demonstrate that increasing the size of the training data reliably increases robust performance. We further show that linear subspaces of the input space exist that only contain a tiny amount of the variance of the data, so those directions cannot be used for robust classification. However, when projecting our data into those subspaces, we can still obtain great (non-robust) accuracy. This implies that enforcing robustness makes classification a strictly hard task on CIFAR-10, providing an explanation why robust classification requires more data.

Insight 3: Robust architectures can fit and generalize non-robustly. Training robust models requires certain architectural choices that are different from standard networks. We show that this is not the reason for the lack of performance on test data. Architectures built for robust classification are expressive enough to robustly overfit the training data, and we can also learn classifiers that generalize well, we just struggle to learn robust classifiers that generalize well.

We believe these insights show how important the amount of training data is for robust classification. In the remainder of the paper, we state our insights more formally and report in detail on our theoretical and empirical findings.

2. Background & notation

2.1. Accuracy and Robust Accuracy

Traditionally, in machine learning, our goal is to maximize the *accuracy* of a classifier f. For a training or test set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ it is given by

$$\mathrm{acc}(f) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\Big[f(x_i) = y_i\Big]. \tag{1}$$

In contrast, in this paper we consider the accuracy on adversarial altered inputs. We want our classifiers to predict the correct class, even when an adversary is allowed to change the input by a small amount. In order to measure performance in this setting, we define the *robust accuracy* of margin ϵ as

$$\mathrm{RA}(f) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\Big[f(\tilde{x}) = y_i \quad \forall \tilde{x} : \|\tilde{x} - x_i\|_2 \leqslant \epsilon\Big] \tag{2}$$

Generally, computing a network's robust accuracy is NP-hard [39], and even approximations are hard to obtain [22]. Therefore, we usually put certain constraints on the classifier f. In this work, we will usually choose f to be a 1-Lipschitz classifier.

We define a *1-Lipschitz classifier* to be a function of the form

$$f(x) = \underset{i=1,\ldots,k}{\mathrm{argmax}}[g(x)]_i, \tag{3}$$

where g is a 1-Lipschitz function with k-dimensional outputs and $[\cdot]_i$ denotes the i-th component of a vector. Here, a function, layer or network f is 1-*Lipschitz* if

$$\|f(x) - f(y)\|_2 \leqslant \|x - y\|_2 \tag{4}$$

for all x, y, where $\|\cdot\|_2$ denotes the Euclidean norm.

Tsuzuku et al. [38] proved that with 1-Lipschitz classifier, we can easily compute a robustness guarantee. For a 1-Lipschitz function g, and the classifier f as defined in Equation (3), we have that the robust accuracy (Equation (2)) is bounded below by the *certified robust accuracy*, which we define as $\mathrm{CRA}(f) =$

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\Big[[g(x_i)]_{y_i} > \max_{c \neq y_i}[g(x_i)]_c + \sqrt{2}\epsilon\Big]. \tag{5}$$

Therefore, in this work we will use CRA as an efficient, yet conservative, proxy for a network's actual robust accuracy. Unless specified otherwise, we will use a perturbation radius $\epsilon = 36/255$, as it is common in the literature.

2.2. 1-Lipschitz networks

In order to train a 1-Lipschitz classifiers, we require a way of parameterizing the 1-Lipschitz function g. We do this by parameterizing g as a neural network, where every layer has the 1-Lipschitz property. There are many ways of creating 1-Lipschitz linear layers, in this work we will use two rescaling-based layers: AOL [29] and CPL [24]. For details about those methods, see *e.g.* [31].

3. Robust classification needs more data

We start our discussion by two observations. First, deep learning has been so successful for many classical computer vision tasks that it has become a routine task to train classifiers on a training dataset in a way so that the classifier also performs well on unseen test data afterwards. Second, for many such tasks, classifiers of high *robust* accuracy are provably possible. Namely, the human visual system provides proof for this, as human perception is typically not just highly accurate (we use it to generate the "ground truth" of our datasets), but also robust, in the sense that it is unaffected by small perturbations of its input.

It is tempting to assume that those two observations (classifiers learned from data generalize well, and high-accuracy robust classifiers do exist) imply that it is also possible to *learn* a high-accuracy robust classifier. However, in the following we show that this conclusion does not hold. Informally, we show that for any dataset size there exists a family of data distributions such that robust classification is possible, learning an accurate classifiers is easy, but learning an accurate robust classifier is impossible.

Our result is formalized in the following Theorem.

Theorem 1 (No Free Robustness)**.** *For any dataset size n there exists a family, $\mathcal{F}$, of binary classification problems such that the following 3 properties hold:*

1. *For any $\mathcal{D} \in \mathcal{F}$, there exists a classifier with 100% robust accuracy.*
2. *There is a learning algorithm that for any $\mathcal{D} \in \mathcal{F}$ and $S \sim \mathcal{D}$ finds a (linear) classifier with 100% test accuracy.*
3. *For any learning algorithm, $\mathcal{L}$, on average over $\mathcal{D} \in \mathcal{F}$ and $S \overset{i.i.d.}{\sim} \mathcal{D}$, the learned classifier $\mathcal{L}(S)$ achieves robust accuracy less than 51% on $\mathcal{D}$.*

Proof. This proof consist of an explicit construction of a family of data distributions that fulfills the three conditions.

Defining $\mathcal{F}$. We first need to define our family of classification problems, $\mathcal{F}$. For that, we first set the data dimension to $d = \lceil \log_2 n \rceil + 7$. We denote the set of all binary functions on the $(d-1)$-dimensional hypercube as Φ,

$$\Phi = \left\{\phi : \{\pm 1\}^{d-1} \to \{\pm 1\}\right\}. \tag{6}$$

Note that this is a very large set with size $|\Phi| = 2^{2^{d-1}}$. For every $\phi \in \Phi$ we will define a data distribution $\mathcal{D}_\phi$, then our family of distribution is given as

$$\mathcal{F} = \{\mathcal{D}_\phi : \phi \in \Phi\}. \tag{7}$$

In order to sample a pair $(\boldsymbol{x}, y)$ from $\mathcal{D}_\phi$, we will sample x_i uniformly from $\{+1, -1\}$ for $i = 1, \ldots, (d-1)$. Then we will set $x_d = \delta\phi(x_1, \ldots, x_{d-1})$ for some small scalar δ, and $y = \operatorname{sign}(x_d)$. Here, $x_1, \ldots, x_{d-1}$ are *robust* (large magnitude) features. Their relation to the ground truth label y is deterministic ($y = \phi(x_1, \ldots, x_{d-1})$), but it is hard to learn because of the size of Φ. In contrast, x_d is a useful, non-robust feature: It is perfectly correlated with the label, but because of its small magnitude it can be easily perturbed.

Proof of statement 1. For any $\mathcal{D}_\phi \in \mathcal{F}$ with associated mapping ϕ, consider the classifier $f(x_1, \ldots, x_d) = \phi\big(\operatorname{sign}(x_1), \ldots, \operatorname{sign}(x_{d-1})\big)$. f is robust against any perturbations of size $\epsilon < 1$, because any such perturbation of the robust features is undone by the sign function, and it has perfect accuracy, because it coincides with the labeling function ϕ.

Proof of statement 2. Consider a learning algorithm that always outputs the classifier $f(x_1, \ldots, x_d) = \operatorname{sign}(x_d)$. Then, because $y = \operatorname{sign}(x_d)$ holds for all data distributions, it follows that f has perfect accuracy on future data.

Proof of statement 3. The third property requires a slightly longer proof, resembling the *No Free Lunch* theorem, *e.g.* [35, Theorem 5.1]. Intuitively, it is based on the fact that a robust classifier cannot rely on the value of feature x_d, because that can be set to 0 by a perturbation of size δ. Furthermore, the functional relation between $(x_1, \ldots, x_{d-1})$ and y, is hard to learn because of the size of Φ.

In order to prove the statement, first note that the average adversarial test error for perturbation of size $\leqslant \delta$ is given as

$$\mathop{\mathbb{E}}_{\mathcal{D}_\phi \in \mathcal{F}} \mathop{\mathbb{E}}_{S \overset{iid}{\sim} \mathcal{D}_\phi} \mathop{\mathbb{P}}_{(\boldsymbol{x},y)\sim\mathcal{D}_\phi} \left[\exists_{\boldsymbol{x'} \in \mathcal{N}_\delta(\boldsymbol{x})} \text{ s.t. } \mathcal{L}(S)(\boldsymbol{x'}) \neq y \right], \tag{8}$$

for $\mathcal{N}_\delta(\boldsymbol{x}) = \{\boldsymbol{x'} : \|\boldsymbol{x} - \boldsymbol{x'}\|_2 \leqslant \delta\}$. We can get a lower bound to this quantity by considering only a single attack that sets the "non-robust" feature x_d to 0. We will write $\tilde{\boldsymbol{x}}$ for the result of applying this attack to an input $\boldsymbol{x}$. With this we can lower bound the average adversarial test error by

$$\mathop{\mathbb{E}}_{\mathcal{D}_\phi \in \mathcal{F}} \mathop{\mathbb{E}}_{S \overset{iid}{\sim} \mathcal{D}_\phi} \mathop{\mathbb{P}}_{(\boldsymbol{x},y)\sim\mathcal{D}_\phi} \mathcal{L}(S)(\tilde{\boldsymbol{x}}) \neq y. \tag{9}$$

Next we will rewrite sampling $\mathcal{D}_\phi \in \mathcal{F}$ and $S \overset{iid}{\sim} \mathcal{D}_\phi$ as sampling $\phi \in \Phi$, and once we know ϕ, sampling from $\mathcal{D}_\phi$

is equal to sampling the robust features from the hypercube $\{\pm 1\}^{d-1}$, as the non-robust feature x_d and the label depend deterministically on the robust features. We can furthermore change the order of sampling the robust features and sampling $\phi \in \Phi$. We will write X^{r} and $\boldsymbol{x}^{\mathrm{r}}$ for these robust features. Using this, the lower bound on the average adversarial test error becomes:

$$\mathop{\mathbb{E}}_{X^{\mathrm{r}}} \mathop{\mathbb{E}}_{\boldsymbol{x}^{\mathrm{r}}} \mathop{\mathbb{E}}_{\phi \in \Phi} \mathbb{1}\Big[\mathcal{L}\big(X^{\mathrm{r}},\, \phi(X^{\mathrm{r}})\big)(\tilde{\boldsymbol{x}}^{\mathrm{r}}) \neq \phi(\boldsymbol{x}^{\mathrm{r}})\Big] \tag{10}$$

Now note that when $\boldsymbol{x}^{\mathrm{r}} \notin X^{\mathrm{r}}$, then $\phi(\boldsymbol{x}^{\mathrm{r}})$ becomes independent of $\phi(X^{\mathrm{r}})$. Therefore, since we assumed a uniform distribution on Φ, any learner will be correct exactly $\frac{1}{2}$ of the time. When $\boldsymbol{x}^{\mathrm{r}} \in X^{\mathrm{r}}$, any reasonable learner should be able to predict correctly, we will bound the error in this case by 0. Using this, the lower bound on the average adversarial test error becomes

$$\mathop{\mathbb{E}}_{X^{\mathrm{r}}} \mathop{\mathbb{E}}_{\boldsymbol{x}^{\mathrm{r}}} \mathbb{1}\Big[\boldsymbol{x}^{\mathrm{r}} \notin X^{\mathrm{r}}\Big] \frac{1}{2}. \tag{11}$$

Now the probability that $\boldsymbol{x}^{\mathrm{r}} \in X^{\mathrm{r}}$ is at most $\frac{n}{2^{d-1}}$, so we know that the average adversarial test error is at least $\frac{1}{2} - \frac{n}{2^d}$, and therefore at least 49% by our choice of $d = \lceil \log_2 n \rceil + 7$. □

Theorem 1 establishes a lower bound on the worst case behavior, by showing that in certain settings we do require exponentially many data points (exponentially in the dataset dimension). Next, we provide a matching upper bound: As long as the input domain is bounded and some robust classifier exists, exponentially many data points suffice to learn an accurate and robust classifier. For this we do need to assume that there exists a robust classifier that is robust to perturbations with bounded L_∞ norm. Note that here is the only part of the paper where we use L_∞ norm, everywhere else we assume L_2 distances. More precisely:

Theorem 2. *Assume that there exists a L_∞ robust classifier (margin δ) on data distribution $\mathcal{D}$, where the data points are in $[0,1]^d$. Then as long as we have $n \geqslant 37 \left\lceil \frac{1}{\delta} \right\rceil^d$ training points independently sampled from $\mathcal{D}$, the 1-nearest neighbor classifier achieves average L_∞ robust test accuracy of $\geqslant 99\%$ (average over sampling training sets).*

Proof Sketch 1. *We first show that for any test point, the nearest point of a different class is at least 2δ away. Therefore, if there is a training point within distance δ of a test point, no perturbation of size at most $\delta/2$ applied to the test point can change the prediction of the 1-nearest neighbor algorithm. Finally, we show that the probability (over sampling training set and test point) of having a training example within δ is $\geqslant 99\%$.*

For the full proof see Section 8. Note that we can adapt the proof to work for any margin $\delta' < \delta$, and not just for $\delta/2$. Furthermore, if we only assume L_2 robustness we might need many more data points, namely $\mathcal{O}(c^d d^{d/2})$ for some constant c.

4. Experimental setup

In order to gather evidence towards quantifying the behavior of robust classifiers on datasets such as MNIST and CIFAR-10, we train some robust and standard (non-robust) models. We provide some background and describe architectures and training setup below.

SimpleConvNet. In order to obtain simple models that achieve good accuracy we rely on the *SimpleConvNet* [28, 41]. It consists of 8 convolutional layers and one linear layer. Each convolutional layer uses *BatchNorm* [21] and ReLU as the activation function. The model uses *MaxPooling* in order to reduce the resolution in the forward pass and also as a global pooling before the linear layer. In order to calculate the loss, we first apply the *Softmax* function with temperature $\frac{1}{8}$ to the predicted class scores, and then use *CrossEntropy*.

1-Lipschitz models. For the robust 1-Lipschitz models, we either use an 8-layer MLP or the ConvNet architecture from [31]. It constraints every single layer to be 1-Lipschitz, therefore the whole network is 1-Lipschitz as well. The architecture first concatenates channels with value 0 to the input, so that the total number of channels becomes 64. Then it applies 5 blocks with 3 convolutional layers each followed by a 1-Lipschitz linear layer. As 1-Lipschitz linear layers we use AOL [29] or CPL [24]. As common in 1-Lipschitz networks, the architecture uses MaxMin [2] as the activation function. As down sampling it uses *PixelUnshuffle*. Unless mentioned otherwise, the loss function we use is *OffsetCrossEntropy* [29], with offset and temperature both set to $\frac{1}{4}$. The only hyperparameter we tune is the peak learning rate, we train models with different learning rates for 100 epochs each, and pick the learning rate of the model with the highest certified robust accuracy on a validation set. For evaluation we usually train for 3000 epochs.

Randomized Smoothing We also estimate the robust performance of a *Randomized Smoothing* classifier [10]. Based on a classifier $f : \mathbb{R}^d \to [C]$, for C the number of classes, the *smoothed classifier* h is defined as

$$h(\boldsymbol{x}) = \mathop{\arg\max}_{c \in [C]} \mathbb{P}(f(\boldsymbol{x} + \boldsymbol{\epsilon}) = c), \tag{12}$$

where $\boldsymbol{\epsilon}$ follows a certain multivariate Gaussian distribution: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$ for some fixed standard deviation σ. Suppose $f(\boldsymbol{x} + \boldsymbol{\epsilon})$ returns the two most likely classes with probabilities p_1 and p_2. Then, its smoothed classifier is robust to perturbations of size $\frac{\sigma}{2}\left(F_{\mathrm{G}}^{-1}(p_1) - F_{\mathrm{G}}^{-1}(p_2)\right)$, for F_{G}^{-1} the inverse of a standard Gaussian cumulative distribution function.

Unfortunately, we cannot evaluate this classifier.[1] However, we can approximate it by sampling ϵ. Furthermore, we can also estimate the robust performance of this (theoretical) classifier. Since in this chapter we are purely interested in estimating robust performance, we directly report this approximation in our results.

We use a SimpleConvNet for the base classifier f and during training we add Gaussian noise to the images in addition to the data augmentation described in Section 4. We found that setting the standard deviation σ to $\frac{1}{8}$ and training for 100 epochs gave us good results on a validation set, therefore we used this values for our evaluation runs. We approximate the class probabilities by sampling ϵ 1000 times.

Optimization. We train all our models using SGD with Nesterov momentum of 0.9 and batch size of 256 with *OneCycleLR* as a learning rate scheduler. As data preprocessing we subtract the training data mean from every channel, we do not rescale the data. We use the same data augmentation as [28, 41]. It consists of random crops, random flips and setting a random patch of the image to zero.

5. Experimental results

5.1. Robust scaling behavior

Recent work on robust image classification has shown that additional data can greatly increase robust accuracy on CIFAR-10. Also, in Section 3 we have shown that the amount of training data can be an important limiting factor for robust classification. Therefore, in this section, we want to explore how the size of the training data influences the performance of a robust classifier trained on real data.

We generated datasets of different sizes by sub-sampling the training partition of existing datasets such as CIFAR-10. We evaluated the performance of models trained on those smaller datasets. In order to keep the amount of compute the same for all settings, when we divide the dataset size by some value k we also multiply the number of epochs by k. The results can be found in Figure 1. We found that increasing the size of the dataset size does indeed make a big difference for the (test) performance of these models, and doubling the size of the dataset seems to reliably increase the certified robust accuracy by about 5%.

We repeated the experiments on MNIST and CIFAR-100, see Figure 2 for the results. Considering all results together, we can nicely see that the certified robust accuracy follows a sigmoid curve. No matter how little data, we can always robustly classifier at chance level. This is visible on CIFAR-100, as the curve starts out almost flat for little training data. With more data, the certified robust accuracy

[1]When f is a piecewise-affine classifier, we could in theory evaluate the probability. This is usually not possible in practice though.

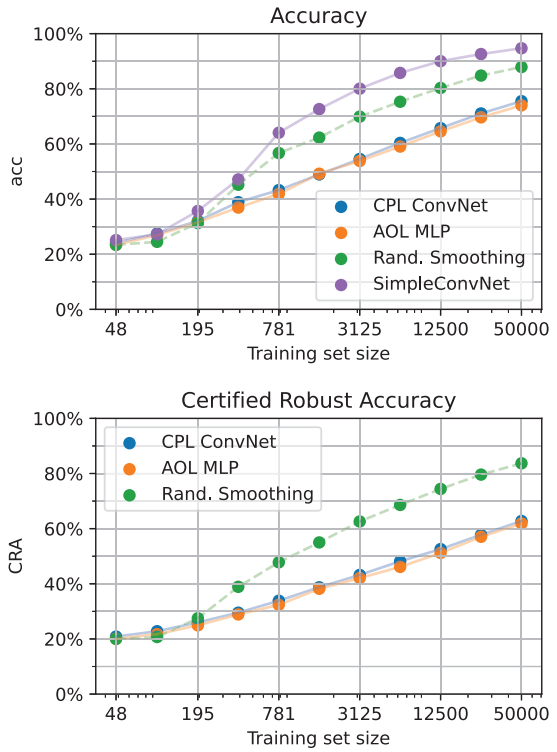

Figure 1. Accuracy (top) and certified robust accuracy for $\epsilon = 36/255$ (bottom). Training on subsets of CIFAR-10.

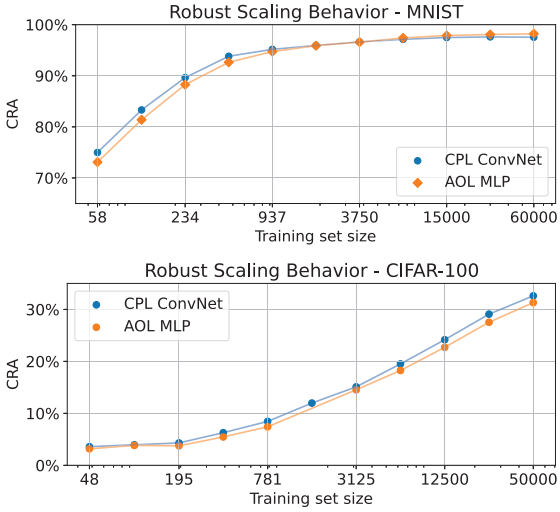

Figure 2. Scaling behavior on MNIST (top) and CIFAR-100 (bottom).

increases about linearly with the logarithm of the amount of training data. We can never get above 100% certified robust accuracy, so for larger dataset sizes the curve flattens again, which can be seen in the MNIST experiment.

Interestingly, we also observe that across dataset, when

training long enough, the convolutional architecture and the MLP have a very similar performance. It does seem that for robust classification, the inductive biases from the convolutional architecture are not very helpful, which is a big difference to non-robust classification.

We did a similar set of experiments to evaluate the influence of the amount of compute. The results are in Figure 5. Whilst increasing only the compute does also improve the performance, it has much less of an effect, and the curves flatten out with increasing compute. Note that of course with more data, additional compute will be more useful. So we do expect that ideally we scale up both, dataset size as well as compute.

Recently, different pieces of work [1, 16, 18, 19, 40] have used additional data in the style of CIFAR-10 generated by a diffusion model in order to improve the performance of robust classifiers. We also conduct scaling experiments on additional data, and find that the scaling nicely extends to larger sizes. See Section 9 for results.

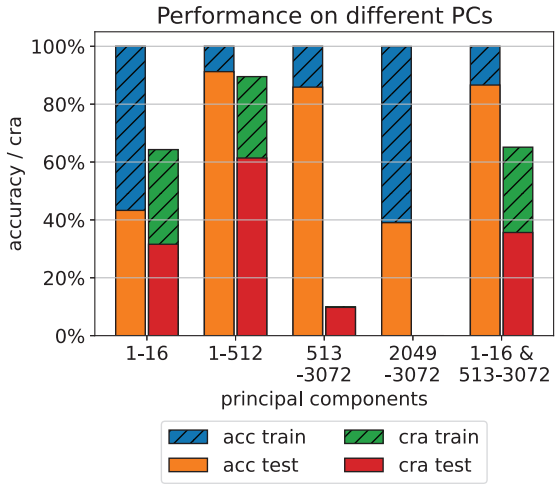

Figure 3. Performance on different subsets of the principal components.

5.2. Robust and non-robust features

In our theoretical section we have established that robust classification can be much harder than non-robust classification, which is also what we observe in experiments. In our theoretical example this hardness comes from a feature of small magnitude and high predictive power. In this section we aim to evaluate whether CIFAR-10 has similar properties: We want to know whether there is a subspace of the input space, such that the data has very low variance when projected onto that subspace, yet the projection is useful for classification. It turns out that it is the case. For example, we can find such a subspace by considering the principal components [27] of the dataset. The principal components of a data set are orthogonal basis vectors, such that principal component i maximizes the variance of the data that lies in the subspace spanned by the first i principal components. For visualizations of the first principal components of CIFAR-10 as well as the variance explained by subsets of principal components see Section 10.

For our experiments we flatten the training images in order to evaluate the principal components. Then in different experiments we project the flattened (train and test) images onto different subsets of principal components. After the projection, we transform the vectors back into the image space, so that we can train a standard and a 1-Lipschitz convolutional CPL network on the data without modifications to the setup.

Find our results in Table 1 and Figure 3 as well as some more in Figure 10. It turns out that when considering the subspace spanned by all but the first 512 components, only about 2% of the data variance are in this subspace. However, when training a standard network on this data, we obtain about 85% test accuracy. Similarly, when projecting on the last 1024 (out of 3072) components, we get only 0.02% of the variance, not enough to allow any robust classification, not even on the training data. However, we trained a standard network to achieve about 39% accuracy on the test set, so there is still some weak signal in the last principal component, a signal that we cannot use for robust classification due to its low magnitude.

From this we conclude that there are in fact low magnitude directions on CIFAR-10 that are useful for classification, yet because of the small magnitude they are not useful for robust classification. This is a property that CIFAR-10 shares with our example from Section 3.

We are also interested in the principal components of high variance. It turns out that the first principal components are not very useful for generalizing. When using only the first 16 principal components for example, we do get about 72% of the total variance, and we can fit standard networks to get $\geqslant 99\%$ training accuracy. Furthermore we can also train 1-Lipschitz ConvNets to get good certified robust training accuracy (64% with augmentation, 97% without) on this low-dimensional subspace. However, the standard convolution network only achieves about 43% accuracy on the test set, suggesting that there is only a weak signal in those features despite high variance.

Based on this observation, we created another dataset which contains the PCA components 1–16 together with the components 513–3072, that is, high magnitude and low magnitude features but not the intermediates. This data suffices for non-robust learning (86% test accuracy), but robust learning fails (35% robust test accuracy). We take this result as an indication that CIFAR-10 as a real dataset shares some characteristics with the hypercube example in Theorem 1: it contains high-magnitude features, which do not allow gen-

eralizing, and features of tiny magnitude, which generalize, but which no robust classifier is not able to exploit.

5.3. Robust overfitting

Previous works have suspected that the lack of robust performance might be due to underfitting: Our models might not have enough capacity to fit the data robustly. In our next experiment we will show that this is not the case, we can train a 1-Lipschitz networks to perfectly fit CIFAR-10, and do so in a robust way. We train an CPL ConvNet and an AOL MLP. In order to overfit robustly, we set the offset in our loss function to $\sqrt{2}$, and we train without augmentation for 3000 epochs.

We show the results in Figure 4. First note that we can clearly fit the training data robustly. For the MLP, even for perturbations of size 1, we get almost perfect certified robust accuracy on the training set. However, it is also visible that the classifiers do not generalize well. The performance on the test set is much worse for any perturbation size. Notably, robust overfitting does seem to require training for a lot of epochs.

Importantly, this result does not imply that our models have enough capacity to fit the data distribution robustly, only that they have enough capacity for the amount of training data we currently have. We believe that when scaling up to amount of training data by a few magnitudes, of course we will benefit from larger models. However, the results in Figure 4 shows that the capacity is not the current bottleneck.

5.4. Robust architectures can generalize

In this final experimental section we want to explore whether it is the model architecture that prevents robust models from generalizing. In order to prevent vanishing gradients and other problems when training 1-Lipschitz networks, there are a few important adaptation from standard convolutional networks. Apart from the different linear layers, in 1-Lipschitz model we use different activation layers, no MaxPooling, no BatchNorm and different initialization strategies, see also Table 2.

In order to evaluate whether the difference in architecture (*e.g.* the lack of global pooling in 1-Lipschitz networks) is the reason for the worse generalization, we carefully constructed a single architecture that can be trained to either competitive accuracy (93.2%) or competitive certified robust accuracy (61.7%), depending on the choice of loss function. This shows that 1-Lipschitz architectures can achieve compareable accuracy to traditional ConvNets, so we do not require layers such as MaxPooling for generalization. We report details and results in Section 11.

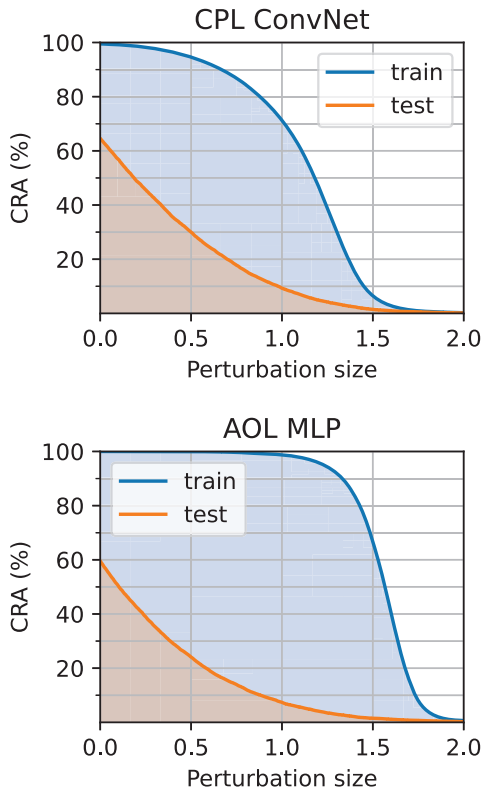

Figure 4. We can robustly overfit the CIFAR-10 training set with a CPL ConvNet (top) and an AOL MLP (bottom).

6. Related work

We will first describe related work on **robust generalization**. One closely related piece of work is [34]. In their work, the authors show that even in a very simple example on Gaussian distributions, robust classification can require many more training examples than non-robust classification. In their example, we can construct an accurate classifier from just a single training example, but for ϵ-robust classification with L_∞-norm we need about $\Omega(\epsilon^2\sqrt{d}/\log(d))$ examples, for d the data dimension.[2] This is an very interesting results, that could explain part of the gap in performance between standard classification and robust classification. Our paper differs in that we are mainly interested in L_2 robustness, and in our example we show that the gap in samples required could potentially be much larger, and we might require $\Omega(2^d)$ training examples.

The work of [34] has been extended by [5] and [11], where the authors also consider different L_p norms, and prove some bounds about the excess risk. We are more interested in distribution where the optimal robust classifier

[2]Here, Ω is the Bachmann–Landau notation: for functions f and g, we write $f(d) = \Omega(g(d))$ if for some M and d_0 it holds that $|f(d)| \geqslant M|g(d)|$ for all $d \geqslant d_0$.

actually achieves 0 risk, and furthermore we think that the convergence rate to this optimal risk is not that important, but the sample complexity of getting within (*e.g.*) 1% of the optimal risk is a more useful quantity to study.

Other works have also produced results about robust generalization. For example, in [32] the authors introduce a data distribution where adversarial training can hurt the test performance of a (regression) model. The authors also argue that we do not actually need labeled data in order to improve performance. As long as we have unlabeled data, we can use a standard network to produce pseudo-labels, and (as long as the data is not adversarial) those labels should be fairly accurate. In [25] the authors show that for Gaussian data the test loss of a linear classifier might actually get worse with additional data, or might show some double descent behavior. In [6] the authors consider data distribution where an perfectly accurate robust classifier exists. They show that in this scenario, learning a robust classifier with maximal possible margin can need d times more samples. We show in our paper that robust classification can be hard not just when aiming for maximum possible margin, but also when the goal is robustness to smaller perturbations.

A different attempt of explaining the lack of performance of current robust models is by blaming it on the **robustness-accuracy trade-off** [37]. It has been observed theoretically as well as empirically that on certain data distributions a classifier can be either very accurate or reasonably robust, but not both [4, 13, 33, 37, 43]. Whilst this trade-off offers interesting insights in general, we think it is not the most promising way to study the lack of performance in image classification tasks, where a classifier that is both accurate and robust at the same time does exist.

Other authors argue that robust classification might require much more complex models, where complexity can refer to the hypothesis class [14, 26, 30], the amount of compute required [7, 12] or the size of the model required [8, 23]. The distributions used to prove results are often similar to our example distribution, there is a map that is hard to learn (*e.g.* from the hypercube to the label), but the data comes with an additional feature of small magnitude that allows us read off the label. These results are further related to our work as in order to use exponentially (in d) many examples, one definitely also requires compute exponentially in d, at either training (*e.g.* for a neural network) or at inference time (*e.g.* for a 1-nearest neighbor classifier). So our result also implies the result that on certain distributions we do require exponential amount of compute.

Another related concept are **robust and non-robust features** [20]. The authors introduce the idea that adversarial examples might not directly be artifacts of the way we train the models, but exist because of the data distribution. Furthermore, they exist because of features that are useful and generalize well but are not robust. In [20] the authors use a very general definition of features, and consider any map from the input space to the real numbers a feature. We believe this definition is too general to give us insights about the datasets. We show that even linear subspaces of the input space exist with tiny variance, and yet projecting into these subspaces still allows achieving great (non-robust) performance.

There is also a recent piece of work [3] that studied **robust scaling laws**, however they consider perturbations with bounded L_∞ norm. In their setting they concluded that (with current techniques) it will require unreasonable amounts of compute (much more than to train recent LLMs) to match human performance on CIFAR-10.

7. Conclusion

Even 10 years after adversarial examples have entered the community's attention, robust classification is far from solved. Furthermore it is also not clear what makes the problem of robust classification so hard, and we still struggle on very simple datasets with robustness to fairly small perturbations. In our paper we have aimed to collect theoretical facts and empirical evidence about robust classification, in particular about robust generalization, in order to give the field a better understanding of the phenomena.

We first showed that there are data distributions on which it is not possible to train a robust classifier, unless the amount of data is unreasonably large. Moreover, this can be the case even for distribution where we can easily learn a good (non-robust) classifier, and where a perfect robust classifier exists.

Based on this insight, we evaluated whether similar results also hold on real data. We showed that on popular datasets including CIFAR-10, the performance of current models seems to be mainly determined by the amount of training data. Furthermore, we showed that as in our theoretical example, CIFAR-10 does have low-magnitude directions that cannot be used for robust classification, yet they are useful for training a standard classifier.

Finally, we showed that the lack of performance of 1-Lipschitz classifiers is not a consequence of the architecture. In particular, 1-Lipschitz models are expressive enough to fit the training data very robustly. Furthermore, 1-Lipschitz architecture are also able to do (non-robust) generalization very well and the same architecture can be trained either to good test accuracy, or to good certified robust accuracy based on the choice of loss function. We just currently fail do both (robust fitting and generalizing) at the same.

Overall we highlighted how important the amount of data is for training a robust classifier. We hope that future research will further explore scaling up datasets and classifiers, and that awareness of the intriguing properties of robust classification we presented will allow the community to create better robust image classifiers in the future.

References

[1] Thomas Altstidl, David Dobre, Björn Eskofier, Gauthier Gidel, and Leo Schwinn. Raising the bar for certified adversarial robustness with diffusion models. *arXiv preprint arXiv:2305.10388*, 2023. 2, 6

[2] Cem Anil, James Lucas, and Roger Grosse. Sorting out Lipschitz function approximation. In *International Conference on Machine Learning (ICML)*, 2019. 4

[3] Brian R. Bartoldson, James Diffenderfer, Konstantinos Parasyris, and Bhavya Kailkhura. Adversarial robustness limits via scaling-law and human-alignment studies. In *2nd Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ ICML 2024)*, 2024. 8

[4] Louis Béthune, Thibaut Boissin, Mathieu Serrurier, Franck Mamalet, Corentin Friedrich, and Alberto Gonzalez Sanz. Pay attention to your loss: understanding misconceptions about Lipschitz neural networks. *Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 8

[5] Arjun Nitin Bhagoji, Daniel Cullina, and Prateek Mittal. Lower bounds on adversarial robustness from optimal transport. *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 7

[6] Robi Bhattacharjee, Somesh Jha, and Kamalika Chaudhuri. Sample complexity of robust linear classification on separated data. In *International Conference on Machine Learning (ICML)*, 2021. 8

[7] Sébastien Bubeck, Yin Tat Lee, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. In *International Conference on Machine Learning (ICML)*, 2019. 2, 8

[8] Sébastien Bubeck, Yuanzhi Li, and Dheeraj M Nagaraj. A law of robustness for two-layers neural networks. In *Conference on Learning Theorey (COLT)*, 2021. 8

[9] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2017. 1

[10] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning (ICML)*, 2019. 1, 4

[11] Chen Dan, Yuting Wei, and Pradeep Ravikumar. Sharp statistical guaratees for adversarially robust gaussian classification. In *International Conference on Machine Learning (ICML)*, 2020. 7

[12] Akshay Degwekar, Preetum Nakkiran, and Vinod Vaikuntanathan. Computational limitations in robust classification and win-win results. In *Conference on Learning Theorey (COLT)*, 2019. 2, 8

[13] Elvis Dohmatob. Limitations of adversarial robustness: strong no free lunch theorem. *arXiv preprint arXiv:1810.04065*, 2018. 8

[14] Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Analysis of classifiers' robustness to adversarial perturbations. *Machine Learning*, 2018. 1, 8

[15] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015. 1

[16] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. Improving robustness using generated data. *Conference on Neural Information Processing Systems (NeurIPS)*, 2021. 2, 6

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4

[18] Kai Hu, Andy Zou, Zifan Wang, Klas Leino, and Matt Fredrikson. Unlocking deterministic robustness certification on Imagenet. *Conference on Neural Information Processing Systems (NeurIPS)*, 2023. 2, 6

[19] Kai Hu, Klas Leino, Zifan Wang, and Matt Fredrikson. A recipe for improved certifiable robustness. In *International Conference on Learning Representations (ICLR)*, 2024. 2, 6

[20] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 8

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015. 4

[22] Matt Jordan and Alexandros G. Dimakis. Exactly computing the local Lipschitz constant of ReLu networks. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 2

[23] Binghui Li, Jikai Jin, Han Zhong, John Hopcroft, and Liwei Wang. Why robust generalization in deep learning is difficult: Perspective of expressive power. *Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 8

[24] Laurent Meunier, Blaise J Delattre, Alexandre Araujo, and Alexandre Allauzen. A dynamical system perspective for Lipschitz neural networks. In *International Conference on Machine Learning (ICML)*, 2022. 3, 4

[25] Yifei Min, Lin Chen, and Amin Karbasi. The curious case of adversarially robust models: More data can help, double descend, or hurt generalization. In *Uncertainty in Artificial Intelligence (UAI)*, 2021. 8

[26] Preetum Nakkiran. Adversarial robustness may be at odds with simplicity. *arXiv preprint arXiv:1901.00532*, 2019. 1, 8

[27] Karl Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 1901. 6

[28] Bernd Prach. SimpleConvNet. `https://github.com/berndprach/SimpleConvNet`, 2024. 4, 5

[29] Bernd Prach and Christoph H. Lampert. Almost-orthogonal layers for efficient general-purpose Lipschitz networks. In *European Conference on Computer Vision (ECCV)*, 2022. 3, 4

[30] Bernd Prach and Christoph H. Lampert. 1-Lipschitz neural networks are more expressive with N-activations. *arXiv preprint arXiv:2311.06103*, 2023. 8

[31] Bernd Prach, Fabio Brau, Giorgio Buttazzo, and Christoph H. Lampert. 1-Lipschitz layers compared: Memory speed and certifiable robustness. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 3, 4

[32] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang. Adversarial training can hurt generalization. In *ICML Workshop on Identifying and Understanding Deep Learning Phenomena*, 2019. 8

[33] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang. Understanding and mitigating the tradeoff between robustness and accuracy. In *International Conference on Machine Learning (ICML)*, 2020. 8

[34] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. *Conference on Neural Information Processing Systems (NeurIPS)*, 2018. 7

[35] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014. 3

[36] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014. 1

[37] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations (ICLR)*, 2019. 1, 8, 4

[38] Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. *Conference on Neural Information Processing Systems (NeurIPS)*, 2018. 2

[39] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2018. 2

[40] Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. Better diffusion models further improve adversarial training. In *International Conference on Machine Learning (ICML)*, 2023. 2, 6

[41] Johan Sokrates Wind. 94% on CIFAR-10 in 94 lines and 94 seconds. `https://johanwind.github.io/2022/12/28/cifar_94.html`, 2022. Accessed: 2024-11-12. 4, 5

[42] Xiaojun Xu, Linyi Li, and Bo Li. LOT: Layer-wise orthogonal training on improving ℓ_2 certified robustness. *Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 4

[43] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning (ICML)*, 2019. 8