IMPROVED ROBUSTNESS TO ADVERSARIAL EXAMPLES USING LIPSCHITZ REGULARIZATION OF THE LOSS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial training is an effective method for improving robustness to adversarial attacks. We show that adversarial training using the Fast Signed Gradient Method can be interpreted as a form of regularization. We implemented a more effective form of adversarial training, which in turn can be interpreted as regularization of the loss in the 2-norm, $\|\nabla_x \ell(x)\|_2$. We obtained further improvements to adversarial robustness, as well as provable robustness guarantees, by augmenting adversarial training with Lipschitz regularization.

1 INTRODUCTION

1.1 CONTRIBUTIONS OF THIS WORK

Adversarial training is an effective method for improving robustness to adversarial attacks. We show that adversarial training using the Fast Signed Gradient Method (Goodfellow et al., 2014) can be interpreted as *regularization* by the average of the 1-norm of the gradient of the loss over the data,

$$J^1[w] = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} [\ell(x) + \varepsilon \|\nabla \ell(x)\|_1] \tag{J^1}$$

The choice of norm for the adversarial perturbation can lead to different interpretations: using the 2-norm for adversarial training corresponds to

$$J^2[w] = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} [\ell(x) + \varepsilon \|\nabla \ell(x)\|_2] \tag{J^2}$$

We present theoretical justification and empirical evidence that training with (J^2) is more adversarially robust than (J^1).

We consider Lipschitz regularization in §3. Write $L_{\ell\circ f}$ for the Lipschitz constant of loss of the model, $\ell(x) = \ell \circ f(x)$. We found existing methods of Lipschitz regularization based on norms of weight matrices (Bartlett, 1996; Szegedy et al., 2013) to be ineffective. As an alternative, we consider a tractable Lipschitz regularization of the loss of the model, by taking the *maximum of over the data* of the norm of the gradient of the loss of the model.

$$\max_{x\in\mathcal{D}} \|\nabla \ell(x)\|_2 \leq L_{\ell\circ f}.$$

Moreover, we show in Theorem 3.2 that $L_{\ell\circ f}$ controls the adversarial robustness of the model. Thus we interpret adversarial training (in the 2-norm) augmented with Lipschitz regularization as minimization of the objective function

$$J^{2-\mathrm{Lip}}[w] = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} [\ell(x) + \varepsilon \|\nabla \ell(x)\|_2] + \lambda \max_{(x,y)\in\mathcal{D}} \|\nabla_x \ell(x)\|_2.$$

which we refer to as $2-\mathrm{Lip}$ (tulip). In practice, $J^{2-\mathrm{Lip}}$ outperforms J^2 and J^1. For example on CIFAR-10, for a ResNeXt model, adversarial training alone reduced adversarial training error by 29% (measured at adversarial ℓ_2 distance[1] $\varepsilon = 0.1$) over an undefended model. In contrast, J^2 with Lipschitz regularization ($J^{2-\mathrm{Lip}}$) reduces adversarial error by 42% over baseline. See Table 3. We trained with hyperparameters $\lambda = 0.1$ and $\varepsilon = 0.01$. Other values of λ and ε may work better; we did not tune these hyperparameters. See §4 for empirical results.

[1] Apologies for overloading 'ℓ' for both the loss and for norms may be confusing: we hope the meaning is clear from context

1.2 BACKGROUND ON ADVERSARIAL EXAMPLES AND ADVERSARIAL TRAINING

Improving robustness to adversarial samples is a first step towards model verification (Szegedy et al., 2013; Goodfellow et al., 2018). However robustness guarantees to adversarial samples are difficult to obtain, since in practice it is only possible to generate suboptimal adversarial attacks.

Adversarial samples are unlikely to occur randomly. Rather, they are generated by an adversary. Adversarial attacks are classified according to the amount of information available to the attacker. White box attacks occur when the attacker has full access to the loss, model, and label. Typically white box attacks are generated using loss gradients: these attacks include L-BFGS (Szegedy et al. (2013)), Fast Signed Gradient (Goodfellow et al. (2014)), Jacobian Saliency (Papernot et al. (2016a)), and Projected Gradient Descent (Madry et al. (2017)). Black box attacks rely on less information, using model outputs rather than model gradients. Black box attacks require more effort (Papernot et al. (2017)) to implement, but their brute force approach may make them more effective evading adversarial defences (Brendel et al. (2018)).

The recent review Goodfellow et al. (2018) discusses defences against adversarial attacks and their limitations. The earliest and most successful defense is adversarial training (Szegedy et al. (2013); Goodfellow et al. (2014); Tramèr et al. (2018); Madry et al. (2017)). Top entries in a recent adversarial defence competition (Kurakin et al. (2017)) used Ensemble Adversarial Training (Tramèr et al. (2018)), where a model is adversarially trained with inputs generated by an ensemble of other models.

In adversarial training, the model, f, is trained to solve the minimax problem

$$\min_{w} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \left[\max_{\|\delta\|\leq\varepsilon} \ell(f(x+\delta;w),y) \right]. \tag{1}$$

However in practice this problem is not computationally feasible. Instead, (1) is approximated. A popular and effective approximation is the Fast Gradient Sign Method (FGSM) Goodfellow et al. (2014), which also defines an attack.

Other forms of defences against gradient based attacks besides adversarial training include (Papernot et al., 2017; 2016b) as well as adding stochastic noise to the model, using a non-differentiable classifier (Lu et al. (2017)), or defense distillation (Hinton et al. (2015); Papernot et al. (2016c)). Gradient based methods may be less successful against black box attacks (Brendel et al. (2018)).

Other possible defences discussed in Goodfellow et al. (2018) include input validation and preprocessing, which would potentially allow adversarial samples to be recognized before being input to the model, and architecture modifications designed to improve robustness to adversarial samples. For more information we refer to the review (Goodfellow et al. (2018)) and the discussion of attack methods in (Brendel et al. (2018)).

1.3 BACKGROUND ON LIPSCHITZ REGULARIZATION OF THE MODEL

A form of robustness guarantees for a network is provided by the global Lipschitz constant of the model. Weng et al. (2018) show that the Lipschitz constant of the model gives an certifiable minimum adversarial distance: a successful attack on image x will have adversarial distance at least

$$\delta \geq \min_{j\neq i^*} \frac{f_{i^*}(x) - f_j(x)}{2L_f} \tag{2}$$

where L_f is the Lipschitz constant of the model, f, and i^* is the correct label of x. Thus training models to have small Lipschitz constant could improve adversarial robustness (Hein & Andriushchenko (2017); Tsuzuku et al. (2018)). Oberman & Calder (2018) recently showed that Lipschitz regularization leads to a proof of generalization. The Lipschitz constant of a model may be estimated using only the product of the norms of model weight matrices (Bartlett (1996); Szegedy et al. (2013)), which is independent of the data. Models have been trained using this estimate as a regularization term in (Cissé et al., 2017; Gouk et al., 2018; Miyato et al., 2018a; Tsuzuku et al., 2018).

For deep neural networks, we argue that there is a large gap between the empirical Lipschitz constant of a model on the data and the estimate of the model Lipschitz constant provided using the model weights Bartlett (1996), see §4.

2 ADVERSARIAL TRAINING AND REGULARIZATION

Definition 2.1 (Adversarial attacks)**.** Write $c^*(x)$ for the correct label and $c(x) = \arg\max_i f(x)_i$ for the classifier. An adversarial attack $a = a(x)$, is a perturbation of the input x which leads to incorrect classification $c(x + a(x)) \neq c^*(x)$.

Adversarial attacks seek to find the minimum norm attack vector, which is an intractable problem (Athalye et al., 2018). An alternative which permits loss gradients to be used, is to consider the attack vector of a given norm which most increases the loss, ℓ.

$$\max_{\|a\| \leq \varepsilon} \ell(f(x + a), y) \tag{3}$$

2.1 DERIVATION OF ATTACK DIRECTIONS

The solution of (3) can be approximated using the dual norm (Boyd & Vandenberghe, 2004, A.1.6). If the ∞-norm is used, we recover the Signed Gradient (Goodfellow et al., 2014). However a different attack vector is obtained if we measure attacks in the 2-norm.

Theorem 2.2. *The optimal attack vector defined by (3) in a generic norm $\|\cdot\|$ can be approximated to $\mathcal{O}(\varepsilon^2)$ with the vector εa, where a is the solution of*

$$a \cdot v = \|v\|_*, \qquad \textit{with } v = \nabla_x \ell(f(x), y)$$

and $\|\cdot\|_*$ is the dual norm. In particular a is given by

$$\begin{cases} a_i^{SG} = \dfrac{\nabla \ell(x)_i}{|\nabla \ell(x)_i|} & \textit{for the } \infty\textit{-norm} \\ a^{\ell_2} = \dfrac{\nabla \ell(x)}{\|\nabla \ell(x)\|_2} & \textit{for the 2-norm} \end{cases} \tag{4}$$

Proof. Write $g(x) = \ell(f(x), y)$ and use the Taylor expansion of

$$g(x + a) = g(x) + a \cdot \nabla_x g(x) + \mathcal{O}(\|a\|^2)$$

Then we can approximate (3) by solving

$$\max_{\|a\| \leq \varepsilon} \nabla_x \ell(f(x), y) \cdot a \tag{5}$$

The value of the solution of (5) is given by the dual norm (Boyd & Vandenberghe, 2004, A.1.6) of the gradient, $\epsilon \|\nabla_x \ell(f(x), y)\|_*$, and the optimal vector is then given by the ε-scaled solution of

$$a \cdot v = \|v\|_*.$$

In the case of the ∞-norm, the dual norm is the 1-norm, and the solution is given by the Signed Gradient vector a^{SG}. In the case of the 2-norm the dual norm is itself the 2-norm and the solution of (5) is given by $a^{\ell_2} = v/\|v\|_2$. □

The 2-norm attack vector, a^{ℓ_2} points in the direction of the gradient of the loss, while the signed gradient attack vector points in the direction of the optimal dual vector.

2.2 INTERPRETATION OF ADVERSARIAL TRAINING

Adversarial training can be interpreted as minimizing

$$\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \left[\ell\left(x + a(x)\right)\right], \qquad \text{where } a \text{ is given by (4)} \tag{6}$$

Theorem 2.3. *Adversarial training using the attack vector (4) can be interpreted as augmenting the loss function with the regularization $\varepsilon R[\ell]$ where*

$$R[\ell] = \begin{cases} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \|\nabla \ell(x)\|_1, & \textit{FGSM} \\ \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \|\nabla \ell(x)\|_2, & \textit{2-norm} \end{cases} \tag{7}$$

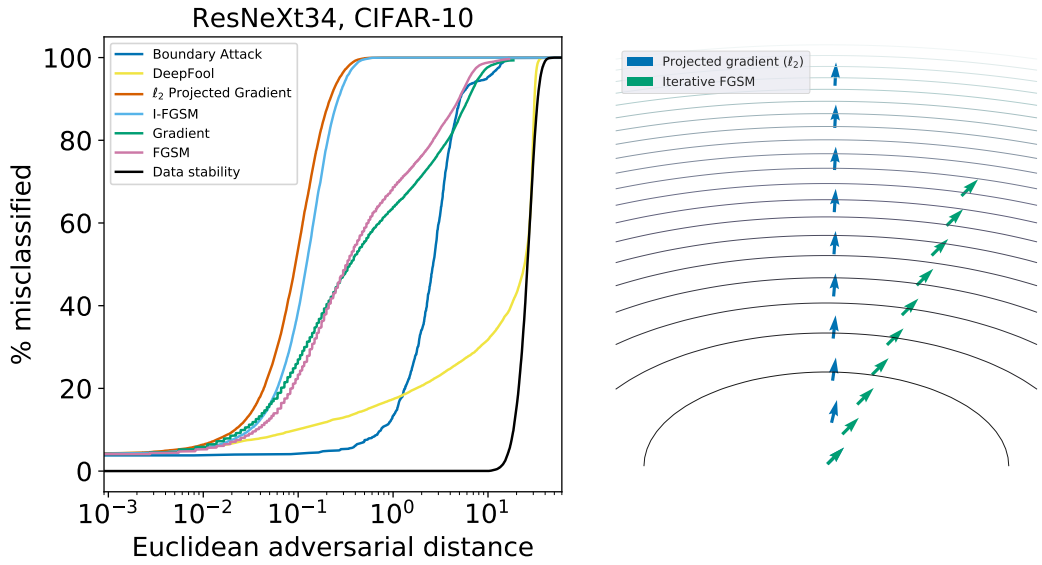


Figure 1: Left: Comparison of attack methods using error curves for undefended ResNeXt-34, on the CIFAR-10 test set. The data stability curve is black. A higher curve means more probability of error, so the ℓ_2 projected gradient method is the most effective attack. Right: Comparison of Iterative FGSM and ℓ_2 gradient descent on a quadratic function in two dimensions. The search direction chosen by FGSM is not the direction of steepest ascent.

Table 1: Adversarial statistics with ResNeXt-34. The columns $\|\nabla \ell\|$ and $\|\nabla f\|$ report the maximum observed norm on the test data.

Dataset	defense method	median ℓ_2 distance	% Err at $\varepsilon = 0.1$	$\|\nabla \ell\|$	$\|\nabla f\|$	$\|\prod W^k\|$
CIFAR-10	J^0 (baseline)	0.09	53.98	3.33	85.85	5.42e6
	J^1 (AT, FGSM)	0.18	24.63	7.04	48.29	2.62e6
	J^2 (AT, ℓ_2)	0.30	13.54	3.94	33.19	1.47e6
	$J^{2-\text{Lip}}$ & tanh	**0.56**	**12.12**	1.62	9.54	1.64e4
CIFAR-100	J^0 (*baseline*)	4.74e−2	74.18	3.54	43.28	6.53e6
	J^1 (AT, FGSM)	8.08e−2	56.34	7.50	56.62	1.95e6
	J^2 (AT, ℓ_2)	8.61e−2	53.77	7.78	44.9	5.08e6
	$J^{2-\text{Lip}}$ & tanh	**0.136**	**42.58**	3.75	21.03	1.61e4

Proof. The adversarial vector given by (4) combined with the Taylor expansion gives

$$\ell(x+a) = \ell(x) + \varepsilon\|\nabla\ell(x)\| + \mathcal{O}(\varepsilon^2)$$

Substitute the last equation into the adversarial training equation (6) to obtain

$$\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}[\ell(x) + \varepsilon\|\nabla\ell(x)\|_*] + \mathcal{O}(\varepsilon^2)$$

which, up to $\mathcal{O}(\varepsilon^2)$ give the regularization term (7). □

2.3 Iterative attacks based on gradient norms

Iterative attacks based on gradient ascent such as iterative FGSM (Madry et al., 2017) should be performed using the 2-norm direction $v = \nabla\ell(x)/\|\nabla\ell(x)\|_2$, since this follows the gradient ascent curve, see Figure **??**.

The angle between a^{SG} and a^{ℓ_2} is given by

$$\cos\theta = \frac{\|\nabla\ell\|_1}{\sqrt{n}\|\nabla\ell\|_2},$$

where n is the input dimension. Because $\|\cdot\|_1 \leq \sqrt{n}\|\cdot\|$, this ratio is always between zero and one. On the networks we studied, the ratio above could be as small as 0.32. To illustrate, Figure **??** shows the angle between iterative FGSM and the iterative gradient ascent on a toy loss (convex quadratic) in two dimensions. In practice we find iterative attacks using the steepest ascent direction are more effective than iterative FGSM based attacks, see Section 4.2.

3 LIPSCHITZ REGULARIZATION

3.1 EVALUATING THE LIPSCHITZ CONSTANT OF A MODEL

Definition 3.1. The Lipschitz constant of a function f is given by

$$\mathrm{Lip}_{2,\infty}(f) = \max_{x_1 \neq x_2} \frac{\|f(x_1) - f(x_2)\|_\infty}{\|x_1 - x_2\|_2}$$

When f is differentiable on a closed, bounded domain, X, then

$$\mathrm{Lip}(f) = \max_x \|\nabla f(x)\|_{2,\infty}. \tag{8}$$

Here for vector value functions, f, the induced matrix norm must be used, based on the norms for $f(x)$ and x (Horn et al., 1990, Chapter 5.6.4). The result is standard in analysis, it follows from the Mean Value Theorem and the definition of the derivative. Using (8), we can approximate the Lipschitz constant by testing on the data

$$\max_{x \in \mathcal{D}} \|\nabla f(x)\|_{2,\infty} \leq \mathrm{Lip}(f) \tag{9}$$

Because the loss is a scalar, Lipschitz regularization of the loss is implemented by taking $\lambda > 0$ and minimizing the regularized loss function

$$J^{\mathrm{Lip}}(w) = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} [\ell(f(x;w), y)] + \lambda \max_{x \in \mathcal{D}} \|\nabla_x \ell(f(x;w), y)\|_2. \tag{J^{Lip}}$$

The first term in (J^{Lip}) is the expected loss, and the second term is the approximation of the Lipschitz constant of the loss coming from (9). During training with Stochastic Gradient Descent, both terms are evaluated over mini-batches.

3.2 LIPSCHITZ CONSTANT OF DATA AND OPTIMAL EXTENSIONS

Define the Lipschitz constant of the data (in the $2, \infty$ norms) to be

$$\mathrm{Lip}_{2,\infty}(\mathcal{D}) = \max_{x_1, x_2 \in \mathcal{D}} \left\{ \frac{\|c^*(x_1) - c^*(x_2)\|_\infty}{\|x_1 - x_2\|_2} \;\middle|\; c^*(x_1) \neq c^*(x_2) \right\}$$

Table 2 lists the Lipschitz constant of the training data for common datasets, which are all small: all by one are below 1.

The Lipschitz extension theorem (Valentine, 1945) says that given function values $\{f(x)\}_{x \in \mathcal{D}}$, there exists an extension f_{ext} which perfectly fits the data, and has the same Lipschitz constant, provided the appropriate norm are used on the X and Y spaces. This can be done using, for example, the 2-norm for X and the ∞ norm on the label space. In other norms, we can also make an extension, but the Lipschitz constant may increase (Johnson & Lindenstrauss, 1984). Of course, such a function may not be consistent with a given architecture.

3.3 ROBUSTNESS GUARANTEES FROM THE LIPSCHITZ CONSTANT

The following Lemma shows that the Lipschitz constant of the loss function gives a robustness guarantee for the loss incurred by an adversarial perturbation of norm ε. An analogous formula gives the corresponding robustness result using the Lipschitz constant of the model (2).

Lemma 3.2 (Stability of network)**.** *Suppose the composed loss function $\ell(x) = \ell(f(x), y)$ is L-Lipschitz continuous. Let $a(x)$ be an adversarial perturbation of norm $\varepsilon = \|a(x)\|$. Then*

$$\ell(x + a(x)) \leq \ell(x) + L\varepsilon \tag{10}$$

Proof. By Lipschitz continuity of ℓ

$$|\ell(x + a(x)) - \ell(x)| \leq L\|a(x)\| = L\varepsilon$$

There are two cases for the left-hand side, depending on the sign. In both cases we obtain (10). □

3.4 Regularization of the model versus the loss of the model

If the goal is adversarial robustness, then regularization of the loss is just as effective (empirically) as regularizing the model, at a much lower cost. Since the loss is a scalar, regularizing by the Lipschitz constant of the loss is equivalent to corresponds to regularization of the model f in one direction. By the chain rule,

$$\nabla_x \ell(f(x), y) = \nabla_f \ell(f(x), y) \nabla_x f(x)$$

For example, when ℓ is the KL divergence, and when $f = \mathrm{softmax}(z(x))$ then

$$\nabla_x \ell(f(x), y) = (f(x) - y)) \nabla_x z(x)$$

Thus, in this case, regularizing $\ell(x)$ corresponds to regularization of $z(x)$ in the direction $f(x) - y$.

3.5 Upper bounds on the Lipschitz constant

The estimate (9) is a lower bound of the Lipschitz constant of the loss. It is well known that data independent upper bounds on the Lipschitz constant of the model are available (Bartlett, 1996) using the product of the norm of the weight matrices. See also Szegedy et al. (2013); Cissé et al. (2017); Gouk et al. (2018); Miyato et al. (2018b;a); Tsuzuku et al. (2018).

Let W^k be the weight matrix of the k-th layer of a network f comprised of N layers, and suppose all non linearities of a network are at most 1-Lipschitz. Then via the chain rule and properties of induced matrix norms, it can be shown

$$\mathrm{Lip}_{p,q}(f) \leq \prod_{k=1}^{N} \|W^k\|_{p_k, p_{k-1}} \tag{11}$$

with $p_0 = p$ and $p_n = q$. Certain conditions on the p_k's must be met. For a proof with $p, q = 2$ see Tsuzuku et al. (2018). A similar bound is available with $p = 2$ and $q = \infty$.

Weng et al. (2018) uses extreme value theory to estimate the local Lipschitz constant of a model in the context of adversarial robustness. However, for deep models, we found for the networks used that this bound was way to large: by a factor on the order of 10^{12} to 10^{23}.

For many networks, a tighter bound is available. Here we prove a bound on the Lipschitz constant (in the $2, \infty$-norm), as the norm of the product of weight matrices. Networks with other non-linearities, such as $\max$-pooling, are not captured by the following lemma, but we believe a generalization is possible.

Lemma 3.3. *Let z be the last layer of model before the* $\mathrm{softmax}$ *layer. Suppose the only non-linearities in the model up to the z layer are entry-wise activation functions which are 1-Lipschitz. Let $W^k \in \mathbb{R}^{m_k \times n_k}$ be the weight matrix of the k-th layer of a network u with N layers. Then*

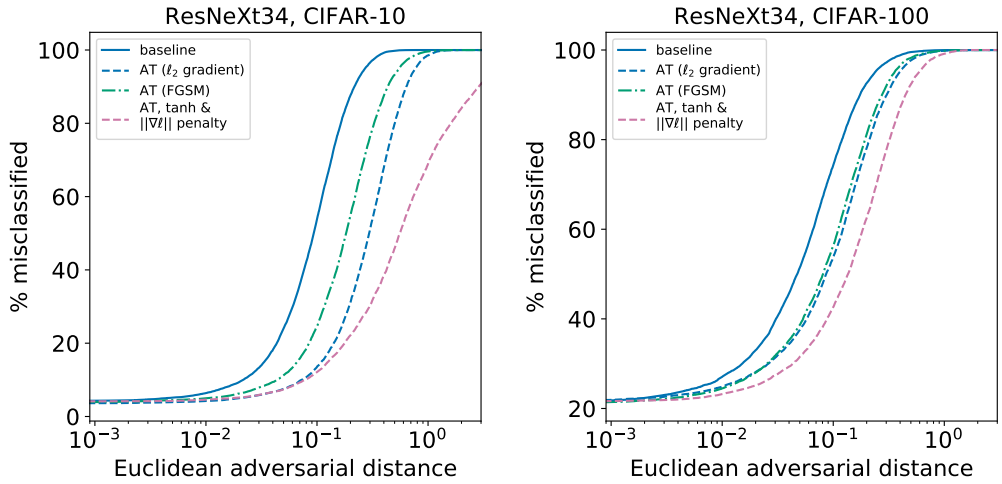
$$\mathrm{Lip}_{2,\infty}(f) \leq \left\| \prod_{k=0}^{N} W^k \right\|_{2,\infty} \tag{12}$$

The proof is in §C. This is a tighter bound, but we found empirically that it is still an over-estimate of the Lipschitz constant for deep networks.

In practice, computing $\prod W^k$ from a predefined network u is straightforward. The following is a simple method for calculating the product of weights. Let $\bar{u}$ be a linearized network defined using the weight matrices of u, but without activation functions or a final $\mathrm{softmax}$. Then the product is determined by evaluating the linearized network at the identity I, $\prod W^k = \bar{u}(I)$. If the layers also have biases, then the linearized bias is recovered by evaluating $\bar{u}$ at 0. In this case the product of weights is given by subtracting off the linearized bias.

Table 2: Lipschitz constants of common training sets. CIFAR-100 has several duplicated images with different labels, these were removed from the calculation.

Dataset	MNIST	FashionMNIST	CIFAR-10	CIFAR-100
$\mathrm{Lip}_{2,\infty}(\mathcal{D})$	0.417	0.626	0.364	1.245



(a) CIFAR-10

(b) CIFAR-100

Figure 2: Comparison of adversarial regularization types for ResNeXt networks on CIFAR 10 and 100.

4 Empirical Results

We considered two toy problems, using image classification on the CIFAR-10 and CIFAR-100 datasets (Krizhevsky & Hinton (2009)). We tested our methods on three networks, chosen to represent a broad range of architectures: AllCNN-C (Springenberg et al. (2014)), a 34 layer ResNet (He et al. (2016)), and a 34 layer ResNeXt (Xie et al. (2017)). Training and model details are provided in Appendix A.

4.1 Error curves for models and robustness metrics

We define the error curve of a model given an attack. The curve provides information about the robustness of a model to attacks of different norms.

Definition 4.1. The error curve $C_{\mathrm{err}}(\varepsilon)$ of the model f for the attack a is the probability over the test data $\mathcal{D}$ that an attack of size ε leads to a misclassification

$$C_{\mathrm{err}}(\varepsilon) = p_{\mathcal{D}}\{c(x + a(x)) \neq c^*(x) \mid \text{ for an attack } \|a(x)\|_X \leq \varepsilon\}$$

See Figure **??** for error curves for a given model over a range of attacks. We also plot the data stability curve, which is the probability that a perturbation can move one data point in the direction of another data point with a different label (which can be interpreted as a very weak attack).

We can compare how two different models perform against various attacks using the model error curve. In Table 3 we report $\varepsilon = 0$ and 0.1, using the 2-norm on X. These values corresponded to the test error, and noise which is slightly smaller than a human perceptible perturbation (see Figure 3). We also report the median ℓ_2 distance which corresponds to the x-intercept of 50% error on the curve.

4.2 Attack evaluation

We attacked each model on the test/validation set using six untargeted attack methods: gradient attack; projected gradient descent (constrained in ℓ_2); the Fast Gradient Sign Method (FGSM)

Table 3: Adversarial statistics with ResNeXt-34. The columns $\|\nabla \ell\|$ and $\|\nabla f\|$ report the maximum observed norm on the test data.

Dataset	defense method	median ℓ_2 distance	% Err at $\varepsilon = 0.1$	$\|\nabla \ell\|$	$\|\nabla f\|$	$\|\prod W^k\|$
CIFAR-10	J^0 (baseline)	0.09	53.98	3.33	85.85	5.42e6
	J^1 (AT, FGSM)	0.18	24.63	7.04	48.29	2.62e6
	J^2 (AT, ℓ_2)	0.30	13.54	3.94	33.19	1.47e6
	$J^{2-\text{Lip}}$ & $\tanh$	**0.56**	**12.12**	1.62	9.54	1.64e4
CIFAR-100	J^0 (*baseline*)	4.74e−2	74.18	3.54	43.28	6.53e6
	J^1 (AT, FGSM)	8.08e−2	56.34	7.50	56.62	1.95e6
	J^2 (AT, ℓ_2)	8.61e−2	53.77	7.78	44.9	5.08e6
	$J^{2-\text{Lip}}$ & $\tanh$	**0.136**	**42.58**	3.75	21.03	1.61e4

(Goodfellow et al. (2014)); Iterative FGSM (I-FGSM) (Kurakin et al. (2016)); DeepFool (Moosavi-Dezfooli et al. (2016)); and Boundary attack (Brendel et al., 2018). The first five methods are white-box attacks, while the last is a black-box attack. I-FGSM and the ℓ_2 projected gradient attack are iterative methods, whereas FGSM and the gradient attack are single step. All attacks were implemented with Foolbox 1.3.1 (Rauber et al. (2017)). Hyperparameters were set to Foolbox defaults, except for the Boundary attack[2]. For each image and attack method, each attack reports an adversarial distance (in ℓ_2).

On each model, dataset, and regularization method, we tested all six attack methods on the entire test/validation set. We compared attack methods using the attack error curve. For example see Figure **??**, where we plot attack error curves for each attack method on an undefended model. The attack error curve plots the percent of misclassified test images as a function of adversarial distance. We report against Euclidean distance, ε, because it is an often reported measure of adversarial robustness, although other choices (MSE, distance in ∞-norm) are equally valid. Common adversarial metrics are easily read off the attack error curve. For example, median Euclidean adversarial distance occurs at the ε-intercept of 50% error. The percent error given a maximum adversarial distance (for example $\varepsilon = 0.1$) is also readily available.

On all models and for all defence methods studied, projected gradient descent (constrained in ℓ_2) consistently outperformed the other attack methods: Projected gradient descent had the smallest mean adversarial distance and the highest attack error curve. See Figure **??** for an illustration. A close second was I-FGSM, the other iterative attack method tested. The next two best attack methods were gradient attack, followed by FGSM. The Boundary attack outperformed DeepFool. We observed the same ranking of attacks on all models and defences studied. For this reason in the following section, we only report model statistics using projected gradient descent, constrained in ℓ_2, which could also be regarded as the strongest attack of all the attacks listed.

4.3 Evaluation of defence methods

Each model was trained with combinations of up to three adversarial defences. The methods were: (i) J^0, the baseline undefended model; (ii) J^1, adversarial training with FGSM; (iii) J^2, adversarial training with 2-norm; each of J^0, J^1, J^2 can be augmented with Lipschitz regularization, which in the last case we call $J^{2-\text{Lip}}$; we also considered adding a final sigmoid layer to the network, prior to the softmax.

The choice of sigmoid we choose is $\tanh$, and is inspired by (but not equivalent to) $\tanh$-estimators used in classical statistics as a robust estimator (Hampel et al., 2011, Chapter 2). The intuition behind this choice is to normalize the logit scores of the model, which we believe should improve robustness to outliers. Outside of deep learning, $\tanh$-estimators have been successfully used to normalize scores and improve robustness, for example in machine learning biometrics (Jain et al. (2005)). See Appendix A.1 for layer details.

[2] The Boundary attack is a computationally demanding attack, and so due to resource constraints we ran the Boundary attack for only 500 iterations per test image. The boundary attack should have better performance with more iterations.

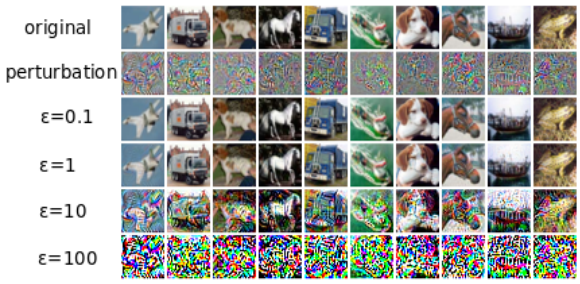


Figure 3: Adversarial perturbations of CIFAR-10 with increasing magnitudes ε of attack, measured in ℓ_2.

Model robustness is evaluated on the entire test/validation set using the median adversarial distance (in ℓ_2), and the percent misclassified at adversarial ℓ_2 distance $\varepsilon = 0.1$. We chose $\varepsilon = 0.1$ because at this magnitude attacks are still imperceptible to the human eye. We argue it is reasonable to ask that models classify images with imperceptible perturbations correctly. At $\varepsilon = 1$ attacks are perceptible, albeit only slightly. See Figure 3. We also plot the attack error curve for each model. These statistics were generated with the ℓ_2 projected gradient attack. Table 3 and Figure 2 present results for ResNeXt-34. The best statistics are in bold.

Here we summarize our results for ResNeXt-34, the model studied with the greatest capacity, and defer results for the other models to Appendix B. Without adversarial perturbations, all ResNeXt-34 models achieve roughly 4% test error on CIFAR-10. However, the undefended (baseline, J^0) model achieves 54% test error at adversarial ℓ_2 distance $\varepsilon = 0.1$. Adversarial training via FGSM (J^1) reduces test error to 24.6%, whereas ℓ_2 adversarial training (J^2) reduces test error to 13.5%. A combination of all defenses ($J^{2-\mathrm{Lip}}$ with $\tanh$) further reduces test error to 12.1%. The models are ranked in the same order when instead measured with median ℓ_2 adversarial distance. The model with all defenses has median adversarial distance six times that of the undefended model. FGSM (J^1) only doubles the median adversarial distance relative to the baseline undefended model. Figure 2a illustrates that this ranking of defenses holds over all distances of adversarial perturbations.

We observe a similar ranking on CIFAR-100. See for Figure 2b. Unperturbed, all models achieve between 21% and 22% test error. Without adversarial defenses, ResNeXt-34 (4x32d) has a test error of 74% at adversarial ℓ_2 distance $\varepsilon = 0.1$. Adversarial training alone brings the test error down to 56.3% and 53.7%, with respectively FGSM and ℓ_2 adversarial training. A combination of all defenses further reduces test error to 42.6%. Median ℓ_2 adversarial distance increase from 0.05 on the undefended model to 0.14 on the model with all defenses.

In Table 3 we also report statistics measuring the model's Lipschitz constant. The columns $\|\nabla l\|_2$ and $\|\nabla f\|_{2,\infty}$ give the maximum of these norms over the test/validation set. The norm of the product of weights is independent of test data, and is an upper bound on the global Lipschitz constant of the model. Employing all defenses dramatically decreases the norm of the model Jacobian on the test data, and hence improves model robustness. On CIFAR-10 the model with all defenses has Jacobian norm nearly 10 times smaller than the undefended model, whereas adversarial training only improves the Jacobian norm by a factor of three at most. On CIFAR-100, adversarial training alone does not appear to improve the norm of the Jacobian significantly. However a combination of all defenses decreases the norm of the model Jacobian by a factor of two.

In Appendix B we report results for all models and combinations of defense methods. Of the individual defenses by themselves, adversarial training (J^1 or J^2) improves model robustness the most. We find ℓ_2 adversarial training (J^2) to be more effective than FGSM (J^1). We observe the same ranking of defense methods for AllCNN and ResNet-34. Adversarial training improves model robustness. However model robustness is further improved by adding Lipschitz regularization, which empirically decreases the Jacobian norm of the model on the test data.

Both adversarial training and Lipschitz regularization increase training time by a factor of no more than four. In contrast, adding a final $\tanh$ layer to normalize the logits is nearly free, and consistently improves model robustness by itself.

Rather than using $\max_{i \in \mathcal{I}} \| \nabla_x \ell \|$ as the Lipschitz penalty, we also tried training models with direct estimates of the Lipschitz constant. We tried both the product of layer weight norms $\prod \|W^k\|$, and the tighter estimate $\| \prod W^k \|$. However, neither of these direct estimates were effective as regularizers. The gap between the empirical Lipschitz constant on the data (the modulus of continuity on the data), and the estimated Lipschitz constant is too large. See for example Table 3, where we report the maximum Jacobian norm and $\| \prod W^k \|$. These two statistics differ by at least four orders of magnitude. The estimate $\prod \|W^k\|$ is worse, and is numerically infeasible for models with more than a few layers. For example, on the two 34-layer networks we studied, this estimate was at least 10^{12}, and was as large as 10^{23}. Another estimate of the local Lipschitz constant is available using a statistic from Extreme value theory (Weng et al. (2018)). However this estimate requires at a minimum many tens of model evaluations for each image, and so is not tractable as a Lipschitz estimate during training.

REFERENCES

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 274–283, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/athalye18a.html.

Peter L. Bartlett. For valid generalization the size of the weights is more important than the size of the network. In *Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2-5, 1996*, pp. 134–140, 1996. URL http://papers.nips.cc/paper/1204-for-valid-generalization-the-size-of-the-weights-is-more-important-than-the-s

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University press, 2004.

Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SyZI0GWCZ.

Moustapha Cissé, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 854–863, 2017. URL http://proceedings.mlr.press/v70/cisse17a.html.

Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017. URL http://arxiv.org/abs/1708.04552.

Ian Goodfellow, Patrick McDaniel, and Nicolas Papernot. Making machine learning robust against adversarial inputs. *Communications of the ACM*, 61(7):56–66, June 2018. URL http://dl.acm.org/citation.cfm?doid=3234519.3134599.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014. URL http://arxiv.org/abs/1412.6572.

Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael Cree. Regularisation of neural networks by enforcing lipschitz continuity. *CoRR*, abs/1804.04368, 2018. URL http://arxiv.org/abs/1804.04368.

Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust statistics: the approach based on influence functions*, volume 196. John Wiley & Sons, 2011.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, pp. 630–645, 2016. URL https://doi.org/10.1007/978-3-319-46493-0_38.

Matthias Hein and Maksym Andriushchenko. Formal Guarantees on the Robustness of a Classifier against Adversarial Manipulation. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017,*

Long Beach, CA, USA, pp. 2263–2273, 2017. URL http://papers.nips.cc/paper/6821-formal-guarantees-on-the-robustness-of-a-classifier-against-adversarial-manip

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531.

Roger A Horn, Roger A Horn, and Charles R Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

Anil K. Jain, Karthik Nandakumar, and Arun Ross. Score normalization in multimodal biometric systems. *Pattern Recognition*, 38(12):2270–2285, 2005. URL https://doi.org/10.1016/j.patcog.2005.01.012.

William B Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26(189-206):1, 1984.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016. URL http://arxiv.org/abs/1607.02533.

Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Nips 2017: Defense against adversarial attack. https://www.kaggle.com/c/nips-2017-defense-against-adversarial-attack, 2017.

Jiajun Lu, Theerasit Issaranon, and David A. Forsyth. SafetyNet: Detecting and rejecting adversarial examples robustly. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 446–454, 2017. URL https://doi.org/10.1109/ICCV.2017.56.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *CoRR*, abs/1706.06083, 2017. URL http://arxiv.org/abs/1706.06083.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *CoRR*, abs/1802.05957, 2018a. URL http://arxiv.org/abs/1802.05957.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018b. URL https://openreview.net/forum?id=B1QRgziT-.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. DeepFool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 2574–2582, 2016. URL https://doi.org/10.1109/CVPR.2016.282.

Adam M. Oberman and Jeff Calder. Lipschitz regularized deep neural networks converge and generalize. *CoRR*, abs/1808.09540, 2018. URL http://arxiv.org/abs/1808.09540.

Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*, pp. 372–387, 2016a. URL https://doi.org/10.1109/EuroSP.2016.36.

Nicolas Papernot, Patrick D. McDaniel, Arunesh Sinha, and Michael P. Wellman. Towards the science of security and privacy in machine learning. *CoRR*, abs/1611.03814, 2016b. URL http://arxiv.org/abs/1611.03814.

Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, pp. 582–597, 2016c. URL https://doi.org/10.1109/SP.2016.41.

Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, pp. 506–519, 2017. URL http://doi.acm.org/10.1145/3052973.3053009.

Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox v0.8.0: A Python toolbox to benchmark the robustness of machine learning models. *CoRR*, abs/1707.04131, 2017. URL http://arxiv.org/abs/1707.04131.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. *CoRR*, abs/1412.6806, 2014. URL http://arxiv.org/abs/1412.6806.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013. URL http://arxiv.org/abs/1312.6199.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rkZvSe-RZ.

Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. *CoRR*, abs/1802.04034, 2018. URL http://arxiv.org/abs/1802.04034.

Frederick Albert Valentine. A Lipschitz condition preserving extension for a vector function. *American Journal of Mathematics*, 67(1):83–93, 1945.

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. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BkUHlMZ0b.

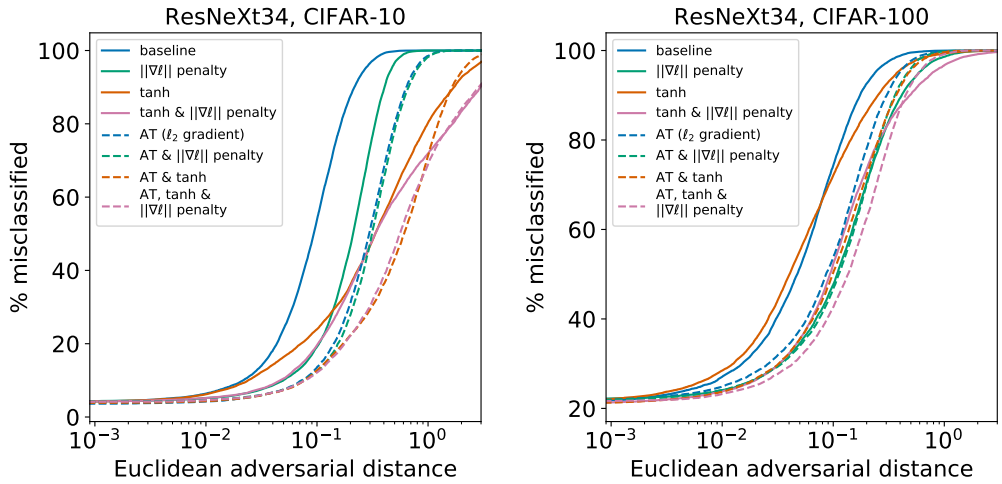
Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 5987–5995, 2017. URL https://doi.org/10.1109/CVPR.2017.634.

A Model and training details

We used standard data augmentation for the CIFAR dataset, comprising of horizontal flips, and random crops of padded images, four pixels per side. We used square cutout (Devries & Taylor (2017)) of width 16 on CIFAR-10, and width 8 on CIFAR-100, but no dropout. Batch normalization was used after every convolution layer. We used SGD with an initial learning rate of 0.1, momentum set to 0.9, and a batch size of 128. CIFAR-10 was trained for 200 epochs, dropping the learning rate by a factor of five after epochs 60, 120, and 180. On CIFAR-100, networks were trained for 300 epochs, and the learning rate was dropped by a factor of 10 after epochs 150 and 225. For CIFAR-10 weight decay (Tikhonov/ℓ_2 regularization) was set to $5\mathrm{e}{-4}$; on CIFAR-100 it was $1\mathrm{e}{-4}$.

For networks with Lipschitz regularization, the Lagrange multiplier λ of the excess Lipschitz term was set to $\lambda = 0.1$. Adversarially trained models were trained with images perturbed to an ℓ_2 distance of $\varepsilon = 0.01$. We did not tune either of these hyperparameters.

For CIFAR-10, the ResNeXt architecture we used had a depth of 34 layers, cardinality 2 and width 32, with a basic residual block rather than a bottleneck. The branches (convolution groups) of the blocks were aggregated via a mean, rather than using a fully connected layer. For CIFAR-100 the architecture was the same, but had cardinality 4.



(a) CIFAR-10

(b) CIFAR-100

Figure 4: Comparison of results for ResNeXt networks on CIFAR 10 and 100.

Table 4: CIFAR-10 adversarial statistics

Model	variant	Without adversarial training			Adversarial training (ℓ_2)		
		Median ℓ_2	% error at ε		Median ℓ_2	% error at ε	
			0	0.1		0	0.1
AllCNN	baseline	0.13	6.01	38.11	0.29	5.90	17.09
	$\|\|\nabla\ell\|\|$ penalty	0.17	6.26	29.27	0.29	5.84	16.86
	tanh	0.19	5.41	32.61	**0.38**	5.10	16.19
	tanh, $\|\|\nabla\ell\|\|$ penalty	0.21	5.45	25.04	0.35	5.27	**15.00**
ResNet34	baseline	0.09	6.00	56.00	0.25	5.57	18.19
	$\|\|\nabla\ell\|\|$ penalty	0.17	5.43	27.08	0.28	5.65	16.74
	tanh	0.20	5.54	34.44	**0.46**	5.52	17.45
	tanh, $\|\|\nabla\ell\|\|$ penalty	0.21	6.14	28.66	0.40	5.81	**15.84**
ResNeXt34 (2x32d)	baseline	0.09	4.07	53.98	0.30	3.58	13.54
	$\|\|\nabla\ell\|\|$ penalty	0.21	4.28	19.13	0.31	4.13	12.52
	tanh	0.34	4.05	23.97	**0.61**	3.80	12.71
	tanh, $\|\|\nabla\ell\|\|$ penalty	0.33	4.18	19.64	0.56	4.08	**12.12**

A.1 PRE-softmax SIGMOID LAYER

Prior to the final softmax layer, we found inserting a sigmoid activation function improved model robustness. In this case, the sigmoid layer comprised of first batch normalization (without learnable parameters), followed by the activation function $t \tanh(\frac{x}{t})$, where t is a single learnable parameter, common across all layer inputs.

B FURTHER EXPERIMENTAL RESULTS

Here we present complete results for all regularization types, on all models and datasets considered. Because ℓ_2 adversarial training outperforms FGSM, we only report results for the former.

[3] We believe this value is an error, but we report it regardless

Table 5: CIFAR-10 stability statistics

Model	variant	Without adversarial training			Adversarial training (ℓ_2)		
		$\|\nabla l\|$	$\|\nabla u\|$	$\|\prod W\|$	$\|\nabla l\|$	$\|\nabla u\|$	$\|\prod W^k\|$
AllCNN	baseline	0.31	16.75	6.93e1	1.03	6.38	3.58e1
	$\|\|\nabla\ell\|\|$ penalty	0.18	9.80	3.55e1	0.84	6.26	2.41e1
	tanh	0.40	13.64	1.10e3	1.44	9.72	9.73e1
	tanh, $\|\|\nabla\ell\|\|$ penalty	0.16	7.15	3.44e2	1.04	8.20	2.90e2
ResNet34	baseline	3.17	72.14	3.66e3	2.51	18.62	4.24e2
	$\|\|\nabla\ell\|\|$ penalty	0.81	20.43	5.73e1	1.88	12.40	3.47e1
	tanh	2.28	41.76	1.26e5	1.97	15.66	4.76e3
	tanh, $\|\|\nabla\ell\|\|$ penalty	0.38	8.45	6.97e1	1.33	9.23	2.74e1
ResNeXt34 (2x32d)	baseline	3.33	85.85	5.42e6	3.94	33.19	1.47e6
	$\|\|\nabla\ell\|\|$ penalty	0.79	20.91	3.88e2	2.30	13.65	1.14e3
	tanh	2.30	50.70	1.01e8	0.76	18.91	2.03e7
	tanh, $\|\|\nabla\ell\|\|$ penalty	0.55	8.23	2.69e2	1.62	9.54	1.64e4

Table 6: CIFAR-100 adversarial statistics

Model	variant	Without adversarial training			Adversarial training (ℓ_2)		
		Median ℓ_2	% error at ε		Median ℓ_2	% error at ε	
			0	0.1		0	0.1
AllCNN	baseline	6.24e−2	25.25	63.58	8.63e−2	25.64	53.85
	$\|\|\nabla\ell\|\|$ penalty	7.78e−2	25.89	56.45	9.71e−2	25.60	**50.71**
	tanh	6.14e−2	26.06	64.77	8.24e−2	26.27	54.81
	tanh, $\|\|\nabla\ell\|\|$ penalty	7.86e−2	26.23	56.26	8.73e−2	26.05	53.24
ResNet34	baseline	2.56e−2	27.42	90.41	5.55e−2	28.21	66.12
	$\|\|\nabla\ell\|\|$ penalty	4.78e−2	28.18	70.94	6.87e−2	28.21	66.12
	tanh	1.46e−2	40.72	81.19	5.59e−2	29.19	64.51
	tanh, $\|\|\nabla\ell\|\|$ penalty	3.58e−2	38.61	68.34	7.12e−2	28.01	**58.40**
ResNeXt34 (4x32d)	baseline	4.74e−2	21.24	74.18	8.61e−2	21.57	53.77
	$\|\|\nabla\ell\|\|$ penalty	1.08e−1	21.97	47.64	1.12e−1	21.73	46.79
	tanh	9.35e−2	21.05	52.28	9.88e−2	21.01	50.33
	tanh, $\|\|\nabla\ell\|\|$ penalty	9.35e−2	21.05	52.28	1.36e−1	21.47	**42.58**

Table 7: CIFAR-100 stability statistics

Model	variant	Without adversarial training			Adversarial training (ℓ_2)		
		$\|\nabla l\|$	$\|\nabla u\|$	$\|\prod W\|$	$\|\nabla l\|$	$\|\nabla u\|$	$\|\prod W^k\|$
AllCNN	baseline	4.18	27.57	2.40e2	3.21	20.92	1.64e4
	$\|\|\nabla\ell\|\|$ penalty	3.08	20.95	1.41e2	2.62	16.30	1.53e2
	tanh	3.41	10.69	2.42e3	3.65	23.32	1.74e3
	tanh, $\|\|\nabla\ell\|\|$ penalty	2.35	6.42	1.27e3	3.04	20.70	1.14e3
ResNet34	baseline	17.78	90.83	6.90e4	5.13	31.15	2.62e3
	$\|\|\nabla\ell\|\|$ penalty	5.41	27.72	1.60e1[3]	3.55	19.19	6.04e1
	tanh	11.36	31.66	1.35e6	4.48	27.99	8.99e3
	tanh, $\|\|\nabla\ell\|\|$ penalty	3.66	7.8	6.23e3	3.44	16.88	1.04e2
ResNeXt34 (4x32d)	baseline	3.54	43.28	6.53e6	7.78	44.9	5.08e6
	$\|\|\nabla\ell\|\|$ penalty	0.65	23.27	1.69e4	3.26	27.67	3.93e3
	tanh	14.03	53.43	1.45e10	7.90	41.73	1.22e8
	tanh, $\|\|\nabla\ell\|\|$ penalty	4.74	10.09	1.95e6	3.75	21.03	1.61e4

C PROOF 1

Proof of Lemma 3.3. Let $l^k(x) = \sigma_k(W^k l^{k-1}(x))$ be the k-th layer of a network, with activation σ_k. Then the gradient of the k-th layer is

$$\nabla l^k(x) = \operatorname{diag}\left(\sigma_k'\left(W^k l^{k-1}(x)\right)\right) W^k \nabla l^{k-1}(x)$$

Note that $\sigma_k'\left(W^k l^{k-1}(x)\right)$ is a vector, with $i_k = 1, \ldots, m_k$ components, where m_k is the number of rows of W^k. For brevity let $d^k_{i_k}(x)$ be the i_k-th component of this vector, and let $w^k_{i_k,j_k}$ be the entries of W^k.

The Jacobian ∇u is defined entry-wise. The entry of ∇u in the i_n-th row and i_0-th column is given by

$$\begin{aligned}
[\nabla u]_{i_n,i_0} &= \sum_{i_1,\ldots,i_{n-1}} \left(\prod_{k=1}^{N} d^k_{i_k}(x) w^k_{i_k,i_{k-1}}\right) \\
&= \sum_{i_1,\ldots,i_{n-1}} \left(\prod_{k=1}^{N} d^k_{i_k}(x)\right)\left(\prod_{k=1}^{N} w^k_{i_k,i_{k-1}}\right)
\end{aligned}$$

Because each activation function is at most 1-Lipschitz, $\left|\prod_{k=1}^{N} d^k_{i_k}(x)\right| \leq 1$. Pulling the maximum of this term out of the matrix multiplication bounds

$$\max_x \|\nabla u(x)\|_{2,\infty} \leq \left\|\prod_{k=1}^{N} W^k\right\|_{2,\infty}$$

as desired. □

The gap between (11) and (12) can be very large. We have observed that (11) is only tractable for very shallow networks, with fewer than half a dozen layers. For a modern deep network, with tens of layers, (11) is typically many orders of magnitude larger than (12). For this reason we do not recommend using (11).