On the Effectiveness of Adversarial Training Against Common Corruptions

Klim Kireev[*1] **Maksym Andriushchenko**[*1] **Nicolas Flammarion**[1]

[1]EPFL, Lausanne, Switzerland

Abstract

The literature on robustness towards common corruptions shows no consensus on whether adversarial training can improve the performance in this setting. First, we show that, when used with an appropriately selected perturbation radius, ℓ_p adversarial training can serve as a strong baseline against common corruptions improving both accuracy and calibration. Then we explain why adversarial training performs better than data augmentation with simple Gaussian noise which has been observed to be a meaningful baseline on common corruptions. Related to this, we identify the *σ-overfitting* phenomenon when Gaussian augmentation overfits to a particular standard deviation used for training which has a significant detrimental effect on common corruption accuracy. We discuss how to alleviate this problem and then how to further enhance ℓ_p adversarial training by introducing an *efficient relaxation* of adversarial training with *learned perceptual image patch similarity* as the distance metric. Through experiments on CIFAR-10 and ImageNet-100, we show that our approach does not only improve the ℓ_p adversarial training baseline but also has cumulative gains with data augmentation methods such as AugMix, DeepAugment, ANT, and SIN, leading to state-of-the-art performance on common corruptions. The code of our experiments is publicly available at `https://github.com/tml-epfl/adv-training-corruptions`.

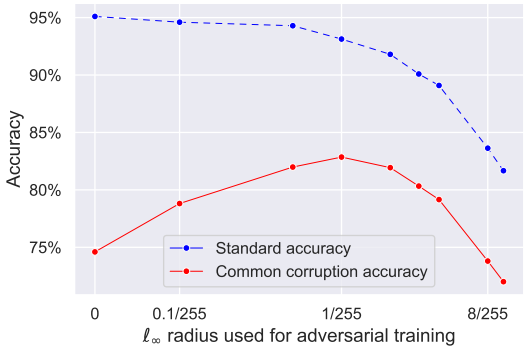


Figure 1: Accuracy on common corruptions from CIFAR-10-C for ResNet-18 models adversarially trained using different ℓ_∞ radii. We observe that the performance with $\varepsilon = 1/255$ is significantly higher than with the standardly used $\varepsilon = 8/255$.

1 INTRODUCTION

Despite achieving human-level performance on many computer vision tasks, deep neural networks are still not as robust as humans towards various distribution shifts [Szegedy et al., 2014, Taori et al., 2020] including common image corruptions [Hendrycks and Dietterich, 2019]. Attempts to understand the vulnerability towards such shifts include analysis of the network architecture [Azulay and Weiss, 2019], the features contained in the data [Ilyas et al., 2019], and frequency analysis of neural networks [Yin et al., 2019, Ortiz-Jimenez et al., 2020]. Many approaches have been suggested to improve their robustness to these shifts including approaches based on data augmentations [Cubuk et al., 2019, Hendrycks et al., 2019b], adversarial training [Madry et al., 2018, Laidlaw et al., 2021], and pretraining [Hendrycks et al., 2019a].

Although data augmentation methods tend to improve the performance under common synthetic corruptions [Hendrycks et al., 2019b], these augmentations are often ad hoc and may have substantial overlap with the corruptions evaluated at test time. At the same time, there is a large amount of literature on adversarial training with ℓ_p-bounded perturbations [Goodfellow et al., 2015, Madry et al., 2018]. Adversarial training emerged as a principled approach to

[*]Equal contribution.

Accepted for the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022).

improve the worst-case performance of the model against *small* ℓ_p perturbations. However, common image corruptions have a very high ℓ_p distance from clean samples, so the utility of using ℓ_p adversarial training for them is not obvious. This leads us to explore the following question:

How can we improve the performance on common image corruptions using adversarial training?

We make the following contributions in our paper:

- We show that ℓ_p adversarial training with an *appropriately selected* perturbation radius can serve as a strong baseline against common image corruptions improving both accuracy and calibration on corrupted images.
- We analyze the success of ℓ_p adversarial training via a comparison to other natural baselines such as Gaussian data augmentation. We observe that it can overfit to the perturbation size it has been trained which, however, does not happen for adversarial training.
- We introduce an efficient relaxation of adversarial training with *learned perceptual image patch similarity* (LPIPS) [Zhang et al., 2018b] based on layerwise adversarial perturbations. This new relaxation is at least as effective as previous approaches [Laidlaw et al., 2021] but significantly faster to train.
- We show that our relaxation approach has cumulative gains with existing data augmentation methods such as AugMix, DeepAugment, ANT, and SIN leading to state-of-the-art performance on common corruptions from CIFAR-10-C and ImageNet-100-C.

2 RELATED WORK

We provide here an overview of relevant works on common image corruptions, different data augmentation methods proposed to improve the performance on corruptions, and then we discuss papers on adversarial robustness with respect to both ℓ_p and non-ℓ_p perturbations.

Common image corruptions. Dodge and Karam [2017] first find that despite being on par with the human vision on standard images, deep networks perform suboptimally on common corruptions such as noise and blur. Geirhos et al. [2018] measure the performance of deep networks on 12 different image corruption types but find that data augmentation on one type of corruption does not tend to improve the performance on others. However, these findings are reconsidered in Rusak et al. [2020] where Gaussian data augmentation is shown to help for a wide range of image corruptions. In a standardization effort, Hendrycks and Dietterich [2019] introduce a few image classification datasets—in particular, CIFAR-10-C and ImageNet-C—with 15 different common corruptions from four categories: noise, blur, weather, and digital corruptions. Ovadia et al. [2019] show that not only acccuracy but also calibration deteriorates under these common corruptions. [Schneider et al., 2020, Nandy et al., 2021] show that robustness to common corruptions can be improved by using test-time adaptation, e.g., via recomputing the batch normalization statistics. Radford et al. [2021] show that contrastive pretraining on a very large set of image-caption pairs can substantially improve robustness on various distribution shifts including common corruptions.

Data augmentations. Data augmentation is a widely used technique to improve the generalization. Besides classical image transformations like random flipping or cropping, many other approaches have been proposed such as linearly interpolating between images and their labels [Zhang et al., 2018a], replacing a part of the image with either a black-colored patch [DeVries and Taylor, 2017] or a part of another image [Yun et al., 2019]. One of the best-performing methods in terms of accuracy and calibration on common corruptions is AugMix [Hendrycks et al., 2019b], which combines carefully selected augmentations with a regularization term based on the Jensen-Shannon divergence. Taori et al. [2020] observe that improvements on synthetic distribution shifts (such as common corruptions) do not necessarily transfer to real distribution shifts. However, Hendrycks et al. [2021] show an example when improving robustness against synthetic blurs also helps against naturally obtained blurred images.

ℓ_p adversarial robustness. Adversarial training in deep learning has been first considered in Goodfellow et al. [2015] and later framed as a robust optimization problem by Madry et al. [2018]. The view that adversarial training damages or at least does not improve the performance on *common corruptions* has been prevalent in the literature [Hendrycks et al., 2019b, Rusak et al., 2020, Hendrycks et al., 2021]. However, previous works directly use publicly available robust models without adjusting the perturbation radius used for adversarial training. For example, Rusak et al. [2020] show that adversarially trained ImageNet models from Xie et al. [2019], Shafahi et al. [2019], and Shafahi et al. [2020] do not help on ImageNet-C compared to standardly trained models. However, Ford et al. [2019] report that ℓ_∞ adversarially trained models on CIFAR-10 from Madry et al. [2018] do lead to an improvement on CIFAR-10-C compared to a standard model. The approach of Xie et al. [2020], AdvProp, relies on ℓ_∞ adversarial training to improve standard and corruption accuracy but they advocate the use of *auxiliary* batch normalization layers for standard and adversarial training examples. We find that similar performance can be achieved on common corruptions using vanilla adversarial training without a customized use of BatchNorm layers. Kang et al. [2019] study the robustness transfer between ℓ_p-robust models and *adversarially optimized* elastic and JPEG corruptions. They show that ℓ_p adversarial training can increase robustness against these

two types of adversarial perturbations, but robustness does not transfer in all the cases and sometimes may even hurt robustness against other perturbation types.

Non-ℓ_p adversarial robustness. Volpi et al. [2018] propose Lagrangian-style adversarial training in the input space and in the last layer of the network. Stutz et al. [2019] propose *on-manifold* adversarial training which is performed in the latent space of a VAE-GAN generative model. However, its success crucially depends on the quality of the generative model which could not be scaled beyond simple image recognition datasets. Wei and Ma [2020] derive generalization bounds that motivate adversarial training with respect to all network layers which they use to improve ℓ_p robustness. Recently, Laidlaw et al. [2021] provided algorithms for approximate *perceptual adversarial training* based on the LPIPS distance [Zhang et al., 2018b] which is defined via activations of a neural network. They aim at improving robustness against new types of adversarial perturbations that were unseen during training.

3 ℓ_p ADVERSARIAL TRAINING IMPROVES THE PERFORMANCE ON COMMON CORRUPTIONS

Here we formally introduce adversarial training and show that it can lead to non-trivial improvements in accuracy and calibration on common corruptions.

Background on adversarial training. Let $\ell(x, y; \theta)$ denote the loss of a classifier parametrized by $\theta \in \mathbb{R}^m$ on the sample $(x, y) \sim D$ where D is the data distribution. Previous works [Shaham et al., 2018, Madry et al., 2018] formalized the goal of training adversarially robust models as the following optimization problem:

$$\min_{\theta} \mathbb{E}_{(x,y)\sim D} \left[\max_{\delta \in \Delta} \ell(x + \delta, y; \theta) \right]. \tag{1}$$

In this section, we focus on the ℓ_p threat model, i.e. $\Delta = \{\delta \in \mathbb{R}^d : \|\delta\|_p \leq \varepsilon,\ x + \delta \in [0, 1]^d\}$, where the adversary can change each input x in an ε-ball around it while making sure that the input $x + \delta$ does not exceed its natural range. A common way to solve the inner maximization problem is the *projected gradient descent* method (PGD) defined by the following recursion initialized at $\delta^{(0)}$:

$$\delta^{(t+1)} \stackrel{\text{def}}{=} \Pi_{\Delta} \left[\delta^{(t)} + \alpha \nabla_{\delta^{(t)}} \ell(x + \delta^{(t)}, y; \theta) \right], \tag{2}$$

where Π is the projection operator on the set Δ, and α is the step size of PGD. Instead of the gradient, one often uses the gradient sign update for ℓ_∞ perturbations or the ℓ_2 normalized update for ℓ_2 perturbations. $\delta^{(0)}$ can be initialized as any point inside Δ, e.g. as zero, or randomly [Madry et al., 2018].

The one-iteration variant of PGD is known as the *fast gradient method* (FGM) when the normalized ℓ_2 update is used and as the *fast gradient sign method* (FGSM) when the ℓ_∞ sign update is used [Goodfellow et al., 2015]. Note that in both cases the step size is $\alpha = \varepsilon$ which leads to perturbations located on the boundary of the set Δ. These methods are fast but sometimes prone to *catastrophic overfitting* when the model overfits to FGM/FGSM but is not robust to iterative PGD attacks [Tramèr et al., 2018, Wong et al., 2020]. This problem can be alleviated by specific regularization methods like CURE [Moosavi-Dezfooli et al., 2019, Huang et al., 2020] or GradAlign [Andriushchenko and Flammarion, 2020]. However, for small enough ε, adversarial training with FGM/FGSM works as well as multi-step PGD [Andriushchenko and Flammarion, 2020].

Table 1: Accuracy and calibration of ResNet-18 models trained on CIFAR-10 and ImageNet-100. ℓ_∞ and ℓ_2 adversarial training substantially improves accuracy and calibration error (ECE) on corrupted samples.

Training	Standard accuracy	Corruption accuracy	Corruption calibration error
	CIFAR-10		
Standard	95.1%	74.6%	16.6%
ℓ_∞ adversarial	93.3%	82.7%	10.8%
ℓ_2 adversarial	93.6%	**83.4%**	**10.5%**
	ImageNet-100		
Standard	86.6%	47.5%	10.0%
ℓ_∞ adversarial	86.5%	47.7%	12.4%
ℓ_2 adversarial	86.3%	**48.4%**	**9.4%**

Experimental details. We do experiments on two common image classification datasets: CIFAR-10 [Krizhevsky and Hinton, 2009] which has 32×32 images, and ImageNet-100 [Russakovsky et al., 2015] with 224×224 images where we take each tenth class following Laidlaw et al. [2021]. We choose ImageNet-100 since we always perform a grid search over the main hyperparameters such as the perturbation radius for adversarial training which would be too expensive to do on the full ImageNet. Unless mentioned otherwise, we use PreAct ResNet-18 architecture [He et al., 2016]. We specify the exact hyperparameters in App. A. We evaluate the accuracy on common corruptions using CIFAR-10-C and ImageNet-C datasets from [Hendrycks and Dietterich, 2019] which contain 15 different synthetic corruptions in 4 categories: blur, noise, digital, weather corruptions. We report the accuracy by averaging over all 5 severity levels.

Adversarial training improves accuracy and calibration. We start by showing in Fig. 1 the common corruption accuracy of ℓ_∞ adversarially trained models as it is the most widely studied setting [Madry et al., 2018] and has been reported multiple times in common corruption literature [Hendrycks et al., 2019b, Ford et al., 2019, Rusak et al., 2020]. Since we are interested primarily in small-ε adversarial training, we rely throughout the paper on FGM/FGSM

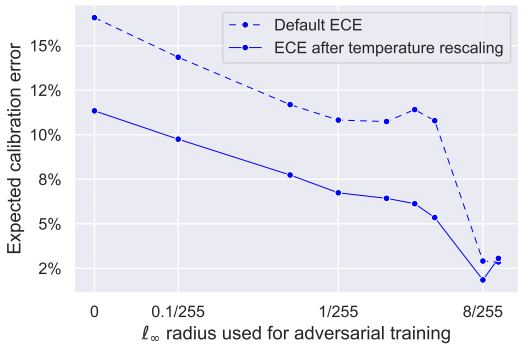


Figure 2: Expected calibration error on CIFAR-10-C for ℓ_∞ adversarially trained models.

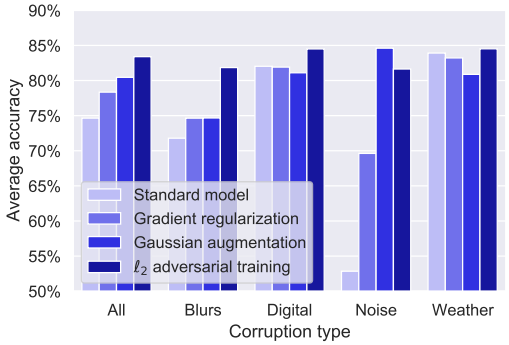


Figure 3: Accuracy for different corruption types on CIFAR-10-C. Unlike other methods, adversarial training improves the performance on each corruption.

for ℓ_2/ℓ_∞ norms respectively to solve the inner maximization problem (1) which only leads to a $2\times$ computational overhead. Note however that we exceptionally use PGD with 10 steps for $\varepsilon \in \{8/255, 10/255\}$ to prevent catastrophic overfitting and allow a direct comparison with previous works. We observe that *for the small-ε regime* around $\varepsilon = 1/255$, we get a significant improvement in corruption accuracy: 74.5% accuracy is achieved with standard training, 82.7% with adversarial training using $\varepsilon = 1/255$, and 73.8% using the standardly reported threshold $\varepsilon_\infty = 8/255$.[1] The reason is that the tradeoff between robustness and accuracy [Tsipras et al., 2019] has to be carefully balanced—if the standard accuracy drops for higher ε, the corruption accuracy also deteriorates. Thus, selecting the most robust ℓ_p-model does not lead to the optimal performance on common corruptions. Alternatively, one can also balance this tradeoff by mixing clean and adversarial samples, but it overall leads to similar results (see App. C for details), so we focus on adversarial training with 100% adversarial samples for the rest of the paper.

Additionally, we show that predicted probabilities of adversarially trained models are significantly better *calibrated on common corruptions*. We believe that calibration is another important aspect of the model's trustworthiness, which is particularly important in the presence of out-of-distribution data such as corrupted images. In Fig. 2, we plot the expected calibration error (ECE) [Guo et al., 2017] on CIFAR-10-C for models trained with different ℓ_∞-radii. We observe that the ECE—both with and without temperature rescaling (see App. B for details)—follows a decreasing trend over ℓ_∞-radii which is expected since a classifier that predicts uniform probabilities over classes is perfectly calibrated. In particular, the most accurate model trained with $\varepsilon_\infty = 1/255$ has a much lower ECE than the standard model: 10.8% instead of 16.6%, and with temperature rescaling 6.7% instead of 11.3%.

[1]The exact numbers differ from [Ford et al., 2019] since we use ResNet-18 instead of WRN-28-10 and different hyperparameters.

We further compare the performance in the ℓ_2 perturbation model. In Table 1, we report results of standard, ℓ_∞, and ℓ_2 adversarial training on CIFAR-10 and ImageNet-100 where we perform a detailed grid search for each model over the perturbation radius ε. To the best of our knowledge, we show for the first time that adversarial training improves calibration (see also App. B) while increasing the accuracy and that it helps on ImageNet-C, and not only on CIFAR-10-C. We generally observe that ℓ_2 adversarial training performs better than ℓ_∞, thus we focus on it in the next section.

4 UNDERSTANDING THE EFFECT OF ADVERSARIAL TRAINING ON IMAGE CORRUPTIONS

Here we compare ℓ_2 adversarial training to other natural baselines and discuss the main conceptual differences.

Comparing natural baselines across corruption types. We compare ℓ_2 adversarial training with a few simple baselines: standard training, gradient regularization [Drucker and LeCun, 1992], and standard Gaussian data augmentation. To ensure a fair comparison, we perform a grid search for each method over the perturbation radius ε, regularization parameter λ, and noise standard deviation σ respectively. We choose to compare to gradient regularization since it is an established regularization method that may have a similar effect to adversarial training with small perturbations [Simon-Gabriel et al., 2019]. We aggregate the corruptions over each type (blurs, digital, noise, weather) and plot the results in Fig. 3 and report results over each corruption in Fig. 12 in the Appendix.

First, we observe that adversarial training is the best performing method and that unlike other methods, ℓ_2 adversarial training helps for *each* corruption type. At the same time, Gaussian augmentation *degrades* the performance on digital

and weather corruptions while very significantly improving the performance for noise corruptions which is expected as the Gaussian noise used for training is also contained in the noise corruptions. Interestingly, for the fog and contrast corruptions, the performance degrades for *all* methods (see Table 10 in App. H), consistently with the observation made in Ford et al. [2019]. Our results also suggest that the impact of gradient regularization is limited and it cannot explain the accuracy gains of both adversarial training and Gaussian augmentation as one could expect from the fact that these methods are equivalent to gradient regularization when used with *sufficiently* small parameters σ and ε [Bishop, 1995].

Worst-case vs average-case behavior. Ford et al. [2019] show that the robustness to Gaussian noise and adversarial perturbations are closely related. More precisely, they show using concentration of measure arguments that a non-zero error rate under Gaussian perturbation implies the existence of small adversarial perturbations and consequently that improving adversarial robustness leads to an improvement in robustness against Gaussian perturbations. This finding is consistent with what we observe here. What remains to be understood is why adversarial training performs *better* than Gaussian augmentation on common corruptions. The main difference between both methods appears when analyzing the objectives that both methods minimize. For a single sample x, the loss function considered in Gaussian augmentation is:

$$\mathbb{E}_{d\sim N(0,I\sigma^2)}\left[\ell(\theta, x+d)\right] \;\sim\; \mathbb{E}_{\rho:||\rho||_2=\sigma\sqrt{d}}\left[\ell(\theta, x+\rho)\right],$$

since Gaussian vectors with variance $\sigma^2 I$ are highly concentrated on the sphere of radius $\sigma\sqrt{d}$ in high dimensions. Therefore Gaussian augmentation amounts to minimize an *averaged* objective where perturbations are averaged over the *sphere*. However, the objective behind adversarial training defined in Eq. (1) amounts to minimize a *worst-case* loss based on the worst-case perturbation in the *ball*. The key difference is that minimization of the expected value of the loss *does not guarantee* any behavior inside the sphere.

To investigate this behavior, we perform the following experiment in Fig. 4. For random 1000 test set images from CIFAR-10, we evaluate the loss with additive Gaussian noise of $\sigma \in [0, 0.1]$ and average the loss function over both images and perturbations for (1) a standard model, (2) a model trained with Gaussian augmentation with $\sigma = 0.05$ where all 100% training samples are augmented, (3) a model trained with Gaussian augmentation for $\sigma = 0.1$ where only 50% training samples are augmented, and (4) ℓ_2 adversarially trained model with $\varepsilon = 0.1$. We notice that the loss function for 100% Gaussian augmentation is minimal at σ which is only slightly less than $\sigma = 0.05$ used for its training. *Hence, the model has overfitted not only to the type of noise but also to its magnitude.* The loss function outside *and inside* of the sphere is bigger than on its surface. However, there is a simple fix if we train with 50%

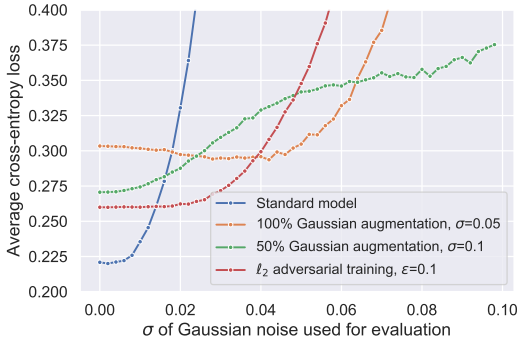


Figure 4: Average cross-entropy loss under Gaussian noise for different training methods.

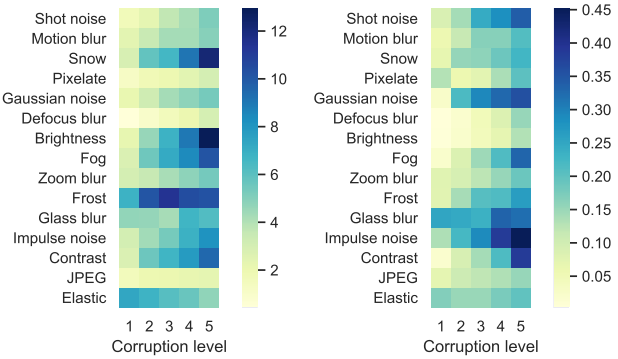


Figure 5: Average ℓ_2 and LPIPS distance for different common corruptions from CIFAR-10-C.

Gaussian noise in each batch, as suggested, e.g., in Rusak et al. [2020] in contrast to Ford et al. [2019]. This scheme allows to alleviate the σ-overfitting behavior and also achieve better accuracy on clean samples (93.2% instead of 92.5%) and, most importantly, *significantly* improve on common corruptions (85.0% instead of 80.5%). At the same time, ℓ_2 adversarial training does not suffer from this problem and both 100% and 50% schemes work nearly equally well (details can be found in App. C). We provide a further discussion on σ-overfitting in App. D together with additional experiments on ImageNet-100 where σ-overfitting has even more noticeable behavior.

Local vs global ℓ_p behavior. Interestingly, adversarial training with worst-case perturbations bounded within a *tiny* ℓ_2 ball leads to robustness significantly beyond this radius. Fig. 5 illustrates that common corruptions have an ℓ_2 norm an *order of magnitude larger* than $\varepsilon = 0.1$ used for ℓ_2 adversarial training. This is in contrast with adversarial robustness that does not significantly extend beyond the radius used for training [Madry et al., 2018]. Related to this, Ford et al. [2019] argue that *for Gaussian noise* improving the minimum distance to the decision boundary (e.g. via adversarial training) also leads to an improvement of the average distance. We have a similar mechanism at play for adversarial ℓ_2 perturbations and common corruptions which may

explain the generalization of adversarial training to large average-case perturbations. However, our setting is more complex compared to Ford et al. [2019] since at the training and test time we deal with *different* and *diverse* types of noise.

5 IMPROVING ADVERSARIAL TRAINING BY RELAXING A PERCEPTUAL DISTANCE

As shown above, ℓ_p adversarial training already leads to encouraging results on common corruptions. Moreover, the ℓ_2 distance appears to be more suitable for adversarial training than ℓ_∞ on both datasets as implied by Table 1. This observation suggests that using more advanced distances such as perceptual ones can further improve corruption robustness.

From ℓ_p distances to LPIPS. One of the main disadvantages of ℓ_p-norms is that they are very sensitive under simple transformations such as rotations or translations [Sharif et al., 2018]. One possible solution is to consider *perceptual distances*[2] which capture these invariances better such as the *learned perceptual image patch similarity* (LPIPS) distance introduced in Zhang et al. [2018b] and which is based on the activations of a convolutional network. The LPIPS distance is formally defined as

$$\mathrm{d}_{\text{LPIPS}}(x, x')^2 = \sum_{l=1}^{L} \alpha_l \|\phi_l(x) - \phi_l(x')\|_2^2, \qquad (3)$$

where L is the depth of the network, ϕ_l is its feature map up to the l-th layer, and $\{\alpha_l\}_{l=1}^L$ are some constants that weigh the contributions of the ℓ_2 distances between activations. There are two crucial elements in LPIPS: the learned network and learned coefficients $\{\alpha_l\}_{l=1}^L$. Zhang et al. [2018b] propose to take a network pre-trained on ImageNet and learn coefficients on their collected dataset of human judgemenets about which images are closer to each other. Both Zhang et al. [2018b] and Laidlaw et al. [2021] argue about better suitability of LPIPS to measure image similarity. In App. E we analyse the suitability of LPIPS over ℓ_2 specifically on the images from CIFAR-10-C with a detailed breakdown over corruption types. In particular, we show that the LPIPS distance is better correlated with the error rate of the network, and the increase over severity levels is more monotonic compared to ℓ_2 as can be also seen in Fig. 5.

LPIPS adversarial training. In view of the positive features of LPIPS, adversarial training using LPIPS appears to be a promising approach to improve the performance on common corruptions. The worst-case loss problem considered in (1) using the LPIPS distance can be formulated as:

$$\max_{\delta} \ell(x + \delta, y; \theta) \quad \text{s.t.} \quad \mathrm{d}_{\text{LPIPS}}(x, x + \delta) \leq \varepsilon. \qquad (4)$$

However, this optimization problem is challenging since $\mathrm{d}_{\text{LPIPS}}$ is itself defined by a neural network, and the projection onto the LPIPS-ball—as required when using PGD to solve (4)—does not admit a closed-form expression. This problem was considered in Laidlaw et al. [2021] who propose two approximate attacks: the Perceptual Projected Gradient Descent (PPGD) and the Lagrangian Perceptual Attack (LPA). We discuss their approach in more detail in App. F but emphasize that they either need to perform an approximate projection which is computationally expensive or come up with some scheme for tuning the Lagrange multiplier λ in the Lagrangian formulation. Furthermore, they suggest in both cases to use 10-step iterative attacks for approximate LPIPS adversarial training which limits the scalability of the method to large datasets such as ImageNet.

Relaxed LPIPS adversarial training. We propose here a relaxation of the LPIPS adversarial objective (4). For the simplicity of presentation, let us start by assuming that the LPIPS distance is defined using a *single* intermediate layer of the network, i.e. $\mathrm{d}_{\text{LPIPS}}(x, x') = \|\phi(x) - \phi(x')\|_2$. Then we can write a neural network f as the composition of the feature map ϕ and the remaining part of the network $f(x) = h(\phi(x))$. The LPIPS adversarial objective (4) in this notation becomes

$$\max_{\delta} \ell(h(\phi(x + \delta))) \quad \text{s.t.} \quad \|\phi(x + \delta) - \phi(x)\|_2 \leq \varepsilon.$$

We first introduce the slack variable $\tilde{\delta} = \phi(x + \delta) - \phi(x)$ which allows us to rewrite the objective as

$$\max_{\delta, \tilde{\delta}} \ell(h(\phi(x) + \tilde{\delta})) \;\; \text{s.t.} \;\; \|\tilde{\delta}\|_2 \leq \varepsilon, \;\; \tilde{\delta} = \phi(x + \delta) - \phi(x).$$

Then we perform the key step: we omit the constraint on the slack variable and obtain the following relaxation

$$\max_{\tilde{\delta}} \ell(h(\phi(x) + \tilde{\delta})) \quad \text{s.t.} \quad \|\tilde{\delta}\|_2 \leq \varepsilon, \qquad (5)$$

i.e. we lift the requirement that there should exist a δ in the *input* space that corresponds to the layerwise perturbation $\tilde{\delta}$.

A similar relaxation can be derived when the LPIPS distance is defined using multiple layers (see App. F):

$$\max_{\tilde{\delta}^{(1)}, \ldots, \tilde{\delta}^{(L)}} \quad \ell(g_L(\ldots g_1(x + \tilde{\delta}^{(1)}) \cdots + \tilde{\delta}^{(L)})) \qquad (6)$$

$$\text{s.t.} \quad \|\tilde{\delta}^{(l)}\|_2 \leq \varepsilon_l \;\; \forall l \in \mathcal{L}_{LPIPS}, \quad \tilde{\delta}^{(l)} = 0 \;\; \forall l \notin \mathcal{L}_{LPIPS},$$

where the network is written under its compositional form $f = g_L \circ \cdots \circ g_1$, $\mathcal{L}_{LPIPS}$ is the set of layer indices used in LPIPS and ε_l denotes the ℓ_2 bound imposed at the l-th layer. We denote this relaxation as *relaxed LPIPS adversarial training* (RLAT) and solve it efficiently using a single-iteration adversarial attack similar to FGM. We emphasize

[2] Not necessarily distances in a strict mathematical sense that assumes a certain set of axioms to hold.

that the projection of each $\tilde{\delta}^{(l)}$ onto the corresponding ℓ_2 balls is computationally cheap to perform, unlike the LPIPS projection.

Since we perform relaxation and *train* the network which is also used to compute LPIPS, the exact layerwise coefficients α_l from the original LPIPS Zhang et al. [2018b] are no longer applicable and cannot be used to set the layerwise bounds ε_l. Therefore, we set our own values of ε_l which we specify in App. F together with detailed derivations of RLAT, its precise algorithm and other implementation details. Finally, we remark that related layerwise adversarial training methods have been proposed before [Stutz et al., 2019, Volpi et al., 2018, Wei and Ma, 2020]. However, viewing layerwise adversarial training as an efficient relaxation of LPIPS adversarial training is novel, as well as applying these methods for general robustness such as common corruptions.

6 EMPIRICAL EVALUATION OF RLAT

Here we first show that RLAT indeed substantially improves the LPIPS robustness. Second, we compare RLAT to other established methods and show that it consistently leads to improved accuracy and calibration on common corruptions.

LPIPS robustness of RLAT. We use the Lagrangian Perceptual Attack attack developed in Laidlaw et al. [2021] to estimate the LPIPS adversarial accuracy under different LPIPS radii and plot results in Fig. 6 on CIFAR-10. We use standard, ℓ_2 adversarial training (AT), Fast PAT, and RLAT models with their main hyperparameters selected to perform best on common corruptions.[3] We observe that RLAT indeed substantially improves LPIPS robustness, even more than other approaches such as ℓ_2 AT and Fast PAT. This gives further evidence that both ℓ_2 and RLAT training *do not suffer from catastrophic overfitting*, even though trained with one-step perturbations similar to FGSM. We provide a similar evaluation for ℓ_2 robustness in App. F (Fig. 10).

Main experimental setup. We compare the results for RLAT with additional baselines: ℓ_2 and ℓ_∞ adversarial training (with 100% adversarial samples per batch), Gaussian augmentation (with both 50% and 100% augmentations per batch), AdvProp [Xie et al., 2020], Fast PAT [Laidlaw et al., 2021], and also four data augmentation approaches: DeepAugment [Hendrycks et al., 2021], AugMix [Hendrycks et al., 2019b], adversarial noise training (ANT) [Rusak et al., 2020], and Stylized ImageNet (SIN) [Geirhos et al., 2019]. We use AugMix method additionally with the Jensen-

[3]We note that Laidlaw et al. [2021] focus on robustness to unseen adversarial examples that involve a *worst-case* optimization process, while we focus on unseen *average-case* common corruptions. This is the reason why the optimal perturbation radii that we consider are noticeably smaller than in their paper.

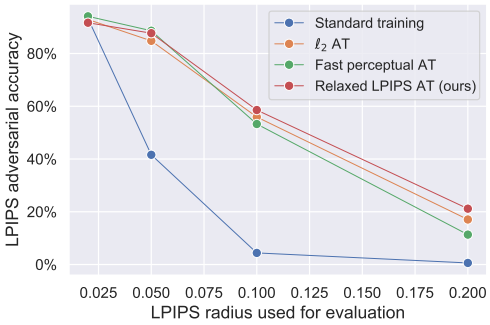


Figure 6: LPIPS adversarial robustness of different training schemes on CIFAR-10.

Shannon regularization term as proposed in Hendrycks et al. [2019b]. We train all methods from random initialization except ANT where we follow the scheme of Rusak et al. [2020]. All comparisons between methods are performed with a grid search over their main hyperparameters (reported in App. A) such as σ in Gaussian augmentation or ϵ in adversarial training which we perform on the main 15 corruptions from CIFAR-10-C / ImageNet-C. In App. H we further verify that selecting the main hyperparameters on validation corruptions leads to the same results. For Fast PAT on CIFAR-10, we do a grid search over their parameter ε, but on ImageNet-100 we report the results based on the models provided by the authors due to limited computational resources. To assess calibration, we report the expected calibration error (ECE) (see App. H for ECE with temperature rescaling Guo et al. [2017]). More details can be found in our repository `https://github.com/tml-epfl/adv-training-corruptions`.

Since the main goal of the common corruption benchmark [Hendrycks and Dietterich, 2019] is to show the model's behavior on *unseen* corruptions, we do not use overlapping augmentations in training (see App. A). The only exception is Gaussian augmentation which we mark in gray in Table 2 following [Rusak et al., 2020] since it belongs to common corruptions. We note that removing only Gaussian noise from evaluation is not sufficient, because other noises can be affected as well by training with Gaussian augmentation. Thus, the results of 100% and 50% Gaussian augmentation are shown only for illustrative purposes suggesting that adversarial training with no prior knowledge about the corruptions can obtain almost the same results as direct augmentation.

Main experimental results. We show the main experimental results on CIFAR-10-C and ImageNet-100-C in Table 2. First of all, we observe that ℓ_p adversarial training is a strong baseline on common corruptions on both datasets with a larger gain on CIFAR-10-C. Using our proposed relaxed LPIPS adversarial training further improves the corruption accuracy on both datasets: from 74.6% to 84.1% on CIFAR-

Table 2: Accuracy and calibration of ResNet-18 models trained on CIFAR-10 and ImageNet-100. Gray-colored numbers correspond to methods partially trained with the corruptions from CIFAR-10-C and ImageNet-100-C.

Training	Standard accuracy	Corruption accuracy	Corruption calibr. error
		CIFAR-10	
Standard	95.1%	74.6%	16.6%
100% Gaussian	92.5%	80.5%	13.2%
50% Gaussian	93.2%	85.0%	9.1%
Fast PAT	93.4%	80.6%	12.0%
AdvProp	94.7%	82.9%	10.1%
ℓ_∞ adversarial	93.3%	82.7%	10.8%
ℓ_2 adversarial	93.6%	83.4%	10.5%
RLAT	93.1%	**84.1%**	**9.9%**
DeepAugment	94.1%	85.3%	8.7%
DeepAugment + RLAT	93.6%	**87.8%**	**6.1%**
AugMix	95.0%	86.6%	6.9%
AugMix + RLAT	94.8%	**88.5%**	**4.5%**
AugMix + JSD	95.0%	88.6%	6.5%
AugMix + JSD + RLAT	94.8%	**89.6%**	**5.4%**
		ImageNet-100	
Standard	86.6%	47.5%	10.0%
100% Gaussian	86.4%	46.7%	11.7%
50% Gaussian	83.8%	55.2%	6.1%
Fast PAT	71.5%	45.2%	8.0%
ℓ_∞ adversarial	86.5%	47.7%	12.4%
ℓ_2 adversarial	86.3%	48.4%	9.4%
RLAT	86.5%	**48.8%**	**9.1%**
AugMix	86.7%	52.3%	7.5%
AugMix + RLAT	86.8%	**54.8%**	**4.7%**
AugMix + JSD	88.4%	59.3%	1.9%
AugMix + JSD + RLAT	87.1%	**61.1%**	**1.8%**
SIN	86.6%	53.7%	6.7%
SIN + RLAT	86.5%	**54.3%**	**6.0%**
$\text{ANT}^{3\text{x}3}$	85.9%	57.7%	5.1%
$\text{ANT}^{3\text{x}3}$ + RLAT	85.3%	**58.3%**	**4.4%**

10-C and from 47.5% to 48.8% compared to standard models. Moreover, RLAT also improves calibration compared to the standard model: from 16.6% to 9.9% ECE on CIFAR-10-C and from 10.0% to 9.1% ECE on ImageNet-100-C. We also observe that 100% Gaussian augmentation even deteriorates the performance on ImageNet-100-C while 50% Gaussian augmentation significantly improves the average accuracy which is consistent with Rusak et al. [2020].

We observe that RLAT can be successfully combined with existing data augmentations, leading to better accuracy and calibration. E.g., adding RLAT on top of DeepAugment helps to improve the CIFAR-10-C accuracy from 85.3% to 87.8%. Combining RLAT with the AugMix augmentation improves the corruption accuracy from 86.6% to 88.5% on CIFAR-10-C and on ImageNet-100-C from 52.3% to 54.8%. Combining SIN and $\text{ANT}^{3\text{x}3}$ improves the accuracy on ImageNet-100-C from 53.7% to 54.3% and from 57.7% to 58.3%, respectively. Moreover, we see that RLAT consistently improves ECE in all settings, and we refer to App. H for ECE with temperature rescaling which qualitatively shows the same behavior.

Table 3: Wall-clock time in hours for ResNet-18 trained with different methods on CIFAR-10 and ImageNet-100 using one Nvidia V100 GPU. * denotes the time reported by Laidlaw et al. [2021] for a larger model (ResNet-50) using different hardware (4 Nvidia RTX 2080 Ti GPUs).

	Dataset	
Training	CIFAR-10	ImageNet-100
Standard	0.8h	3.9h
ℓ_2/ℓ_∞ adversarial	1.3h	5.8h
RLAT	1.8h	6.2h
Fast PAT	9.4h	*120h

Additionally, we added our models to the RobustBench leaderboard[4] where our method has the best performance among the architectures of comparable sizes (i.e., ResNet-18). The models which perform better have larger architectures and some of them additionally rely on ensembles.

Runtime of RLAT. We report a full runtime comparison between standard training, ℓ_2 / ℓ_∞ adversarial training, RLAT, and Fast PAT in Table 3. The main observation is that RLAT is significantly faster than Fast PAT (e.g., 1.8 hours vs. 9.4 hours on CIFAR-10) and leads only to a slight overhead compared to ℓ_2 / ℓ_∞ adversarial training (1.8 hours vs 1.3 hours on CIFAR-10). These runtimes show further the advantage of the single-step adversarial training procedure of RLAT compared to the multi-step approach of Fast PAT. It would be interesting in future work to develop a single-step version of Fast-LPA which is, however, not straightforward because of their Lagrangian formulation and the need to tune the parameter λ over the iterations of Fast-LPA.

Additional experiments. We refer to the Appendix for further experimental results. In App. G, we evaluate the performance of the models from Table 2 on ImageNet-A, ImageNet-R, and Stylized ImageNet to better understand how well the improvements on common corruptions transfer to other distribution shifts. In App. H, we provide more detailed results such as those presented in Table 2 but with breakdowns over different corruptions and severities. We also present results for larger network architectures and for AugMix combined with ℓ_p adversarial training in App. H, as well as results of RLAT over multiple random seeds.

[4] https://robustbench.github.io/

7 CONCLUSIONS AND FUTURE WORK

Our findings suggest that adversarial training can be successfully used to improve accuracy and calibration on common image corruptions. Even simple ℓ_p adversarial training can serve as a strong baseline if the optimal perturbation radius is chosen for the given problem. More advanced adversarial training schemes involve perceptual distances, such as LPIPS, and we provide a relaxation of LPIPS adversarial training with an efficient single-step procedure. We observe that the developed relaxation (RLAT) substantially improves the LPIPS robustness and can be successfully combined with existing data augmentations. We hope that RLAT would be of interest also for other domains such as natural language processing where robustness to commonly occurring corruptions (e.g., typos) is an important task.

REFERENCES

Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. In *NeurIPS*, 2020.

Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *JMLR*, 20(184):1–25, 2019.

Chris M. Bishop. Training with noise is equivalent to Tikhonov regularization. *Neural Computation*, 7(1):108–116, January 1995.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. In *CVPR*, 2019.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Samuel Dodge and Lina Karam. A study and comparison of human and deep learning recognition performance under visual distortions. In *ICCCN*, 2017.

Harris Drucker and Yann LeCun. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 1992.

Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. URL `https://github.com/MadryLab/robustness`.

Nic Ford, Justin Gilmer, Nicolas Carlini, and Dogus Cubuk. Adversarial examples are a natural consequence of test error in noise. In *ICML*, 2019.

Robert Geirhos, Carlos R Medina Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In *NeurIPS*, 2018.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. *ICLR*, 2019.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *ICLR*, 2015.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *ECCV*, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.

Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *ICML*, 2019a.

Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *ICLR*, 2019b.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019c.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 2021.

Tianjin Huang, Vlado Menkovski, Yulong Pei, and Mykola Pechenizkiy. Bridging the performance gap between fgsm and pgd adversarial training. *arXiv preprint arXiv:2011.05157*, 2020.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *NeurIPS*, 2019.

Daniel Kang, Yi Sun, Tom Brown, Dan Hendrycks, and Jacob Steinhardt. Transfer of adversarial robustness between perturbation types. *arXiv preprint arXiv:1905.01034*, 2019.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

Cassidy Laidlaw, Sahil Singla, and Soheil Feizi. Perceptual adversarial robustness: Generalizable defenses against unforeseen threat models. In *ICLR*, 2021.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Eric Mintun, Alexander Kirillov, and Saining Xie. On interaction between augmentations and corruptions in natural corruption robustness. *arXiv preprint arXiv:2102.11273*, 2021.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *CVPR*, 2019.

Jay Nandy, Sudipan Saha, Wynne Hsu, Mong Li Lee, and Xiao Xiang Zhu. Adversarially trained models with test-time covariate shift adaptation. *arXiv*, 2021.

Guillermo Ortiz-Jimenez, Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Hold me tight! influence of discriminative features on deep network boundaries. In *NeurIPS*, 2020.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *arXiv preprint arXiv:1906.02530*, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. 2021.

Evgenia Rusak, Lukas Schott, Roland S Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. In *ECCV*, 2020.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. In *NeurIPS*, 2020.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *NeurIPS*, 2019.

Ali Shafahi, Mahyar Najibi, Zheng Xu, John Dickerson, Larry S Davis, and Tom Goldstein. Universal adversarial training. In *AAAI*, 2020.

Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 2018.

Mahmood Sharif, Lujo Bauer, and Michael K Reiter. On the suitability of lp-norms for creating and preventing adversarial examples. In *CVPR Workshops*, 2018.

Carl-Johann Simon-Gabriel, Yann Ollivier, Leon Bottou, Bernhard Schölkopf, and David Lopez-Paz. First-order adversarial vulnerability of neural networks and input dimension. *ICML*, 2019.

David Stutz, Matthias Hein, and Bernt Schiele. Disentangling adversarial robustness and generalization. In *CVPR*, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. *NeurIPS*, 2020.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *ICLR*, 2018.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *ICLR*, 2019.

Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *NeurIPS*, 2018.

Colin Wei and Tengyu Ma. Improved sample complexities for deep neural networks and robust classification via an all-layer margin. In *ICLR*, 2020.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020.

Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *CVPR*, 2019.

Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *CVPR*, 2020.

Dong Yin, Raphael Gontijo Lopes, Jonathon Shlens, Ekin D Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. In *NeurIPS*, 2019.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018a.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018b.

Appendix

ORGANIZATION OF THE APPENDIX

The appendix contains additional implementation details for our methods and the baselines we compare to, as well as more detailed derivations and experimental results. The appendix is organized as follows:

- Sec. A: further details on our experimental setup, hyperparameter choice, and runtime of different methods.
- Sec. B: further details on the calibration experiments and results for ℓ_2 adversarial training.
- Sec. C: an ablation study for AdvProp comparing it to adversarial training with some fraction of clean images.
- Sec. D: further experiments related to σ-overfitting on ImageNet-100 and more detailed discussion.
- Sec. E: discussion on why LPIPS distance is particularly suitable for the corrupted images from CIFAR-10-C.
- Sec. F: full derivations for our Relaxed LPIPS adversarial training method, further implementation details, and evaluation of ℓ_2 robustness.
- Sec. G: evaluation of the performance of various models on different distribution shifts such as ImageNet-A, ImageNet-R, and Stylized ImageNet.
- Sec. H: more detailed experimental results related to Sections 4 and 6 such as breakdowns over different corruptions and severity levels.

A EXPERIMENTAL DETAILS

In this section, we provide more details regarding our experimental settings, hyperparameters, evaluation metrics, and runtime of our method.

Dataset details. We perform experiments on two common image classification datasets: CIFAR-10 [Krizhevsky and Hinton, 2009] which has 32×32 images, and ImageNet-100 [Russakovsky et al., 2015] with 224×224 images where we take each tenth class according to the WordNet ID order following Laidlaw et al. [2021]. We choose ImageNet-100 instead of the full ImageNet since we have limited computation resources for performing grid searches on large-scale datasets over the main hyperparameters such as the perturbation radius for adversarial training or the standard deviation of the Gaussian noise.

The CIFAR-10-C and ImageNet-C datasets that were introduced in Hendrycks and Dietterich [2019] contain 15 main synthetic corruptions: Gaussian noise, shot noise, impulse noise, defocus blur, glass blur, motion blur, zoom blur, snow, frost, fog, brightness, contrast, elastic, pixelation, and JPEG. Both datasets contain also 4 additional corruptions (speckle noise, Gaussian blur, spatter, saturation) that are not commonly used. Each corruption has 5 levels of severity. We use the CIFAR-10-C and ImageNet-C images provided by Hendrycks and Dietterich [2019], although one could alternatively apply the corruptions in-memory (as done, e.g., in [Ford et al., 2019]).

In addition, we make use of three more variants of ImageNet: Stylized ImageNet, ImageNet-A and ImageNet-R. Stylized ImageNet (SIN) is a variant of ImageNet which is obtained using style transfer. It has been first introduced to induce a shape bias in convolutional networks [Geirhos et al., 2019]. ImageNet-A [Hendrycks et al., 2019c] is a test set of $7\,500$ natural but adversarially collected images with, e.g., unusual backgrounds or occlusions for 200 ImageNet classes. ImageNet-R [Hendrycks et al., 2021] is a test set of $30\,000$ image renditions (e.g., paintings, sculptures, embroidery) for another set of 200 ImageNet classes. When evaluating on these datasets, we only use the classes intersecting with ImageNet-100.

Evaluation details. There are 15 corruptions and 5 severity levels in CIFAR-10-C and ImageNet-C. Thus there are multiple ways of reporting the performance of a model on these two datasets. For a model f, let $E^f_{s,c}$ denote the top-1 error rate on the corruption c with severity level s averaged over the whole test set, then three popular metrics are often reported:

- *Average accuracy*: the accuracy is averaged over all severity levels and corruptions:

$$\text{Accuracy}_f = 1 - \frac{1}{15 \cdot 5} \sum_{c=1}^{15} \sum_{s=1}^{5} E^f_{s,c}.$$

- *Mean corruption error* (mCE, proposed in Hendrycks and Dietterich [2019]): the error rate on each corruption is normalized by the error rate, $E^{\text{AlexNet}}_{s,c}$, of the standard deep learning model, AlexNet [Krizhevsky et al., 2012]:

$$\text{mCE}_f = \frac{1}{15} \sum_{c=1}^{15} \frac{\sum_{s=1}^{5} E^f_{s,c}}{\sum_{s=1}^{5} E^{\text{AlexNet}}_{s,c}}.$$

  The motivation is to make the error rates on different corruptions more comparable. Indeed they do not all have the same inherent level of difficulty.

- *Relative mean corruption error* (relative mCE, proposed in Hendrycks and Dietterich [2019]): instead of measuring the error rate, one can also consider the degradation of the error rate compared to the standard error rate E^f_{standard} of the model f taken relative to the degradation of the error rate $E^{\text{AlexNet}}_{\text{standard}}$ of AlexNet:

$$\text{Relative mCE}_f = \frac{1}{15} \sum_{c=1}^{15} \frac{\sum_{s=1}^{5} E^f_{s,c} - E^f_{\text{standard}}}{\sum_{s=1}^{5} E^{\text{AlexNet}}_{s,c} - E^{\text{AlexNet}}_{\text{standard}}}.$$

Table 4: The main hyperparameters used for CIFAR-10 and ImageNet experiments.

	Dataset	
Hyperparameter	CIFAR-10	ImageNet-100
Architecture	PreAct ResNet-18	PreAct ResNet-18
Number of epochs	150	100
Learning rate of SGD	0.1	0.1
Epochs for learning rate decay (by $10\times$ factor)	{50, 100}	{33, 66}
Momentum	0.9	0.9
Batch size	128	128
Weight decay	0.0005	0.0005

Table 5: Grid searches performed for CIFAR-10 experiments. For each grid, we boldface the hyperparameter that leads to the model with the best common corruption accuracy.

Method	**Grid values**
100% Gaussian augmentation	$\sigma \in \{0.02, \mathbf{0.05}, 0.08, 0.1, 0.2\}$
50% Gaussian augmentation	$\sigma \in \{0.02, 0.05, 0.08, \mathbf{0.1}, 0.2\}$
ℓ_∞ adversarial training	$\varepsilon \in \{0.1, 0.5, \mathbf{1.0}, 2.0, 4.0, 8.0, 16.0\}/255$
ℓ_2 adversarial training	$\varepsilon \in \{0.01, 0.05, 0.08, \mathbf{0.1}, 0.15, 0.2, 0.5, 1.0\}$
Fast PAT	$\varepsilon \in \{0.005, 0.01, \mathbf{0.02}, 0.05, 0.08\}$
AdvProp	$\varepsilon \in \{0.5, 1.0, \mathbf{2.0}, 4.0, 8.0\}$
RLAT	$\varepsilon \in \{0.05, \mathbf{0.08}, 0.1, 0.15, 0.2, 0.25\}$
RLAT + DeepAugment	$\varepsilon \in \{0.005, \mathbf{0.02}, 0.05\}$
RLAT + AugMix	$\varepsilon \in \{0.01, \mathbf{0.02}, 0.05, 0.1, 0.15, 0.2\}$
RLAT + AugMix + JSD	$\varepsilon \in \{\mathbf{0.01}, 0.02, 0.05, 0.1, 0.15, 0.2\}$

However, this metric has to be carefully interpreted since it does not take absolute accuracy into account, e.g., a *constant* model achieves the perfect score of 0 for this metric.

Since there is no standard AlexNet model on CIFAR-10, the mean corruption error and the relative mean corruption error are not well defined on this dataset. Therefore we focus on reporting average accuracy on both CIFAR-10-C and ImageNet-100-C to be consistent throughout the paper.

For the LPIPS robustness evaluation shown in Fig. 6, we use the following settings: Fast Lagrangian Attack from Laidlaw et al. [2021] using their suggested hyperparameters and AlexNet as the network to compute the LPIPS distance. For the ℓ_2 robustness evaluation, we use the APGD-CE attack [Croce and Hein, 2020] with 100 iterations and 5 random restarts.

Training details. In all our experiments, we use SGD with momentum to train a PreAct ResNet-18 network (both on CIFAR-10 and ImageNet-100). The momentum coefficient is set to the value 0.9. The learning rate is initially set to the value 0.1 and then is decayed by a factor 10 according to a predefined schedule. We train for 150 epochs on CIFAR-10 and 100 epochs on ImageNet-100 and always report the results of the *last* model, i.e. we do not perform any early stopping. We specify all the main training hyperparameters in Table 4.

A recent work of Mintun et al. [2021] suggests that the most effective data augmentations are those which are perceptually similar to the target corruptions from CIFAR-10-C and ImageNet-C. Along the same lines, Rusak et al. [2020] mention that for AugMix, there is a visual similarity, e.g., between the posterize operation and the JPEG corruption. Thus, to prevent training on augmentations which resemble the ones from CIFAR-10-C and ImageNet-C, we use only random horizontal flip and random crops unless mentioned otherwise. Moreover, when we train Fast PAT on CIFAR-10, we make sure to remove the overlapping augmentations used in the robustness library [Engstrom et al., 2019] such as random brightness and contrast change. For the experiments whose results are reported in Table 2, we use additional augmentations like AugMix, SIN, ANT$^{3\text{x}3}$ whenever it is explicitly mentioned.

For every method that we reported, we performed a grid search over the main hyperparameters such as the standard deviation σ for Gaussian data augmentation or the perturbation radius ε for adversarial training. We report all the used grids in Table 5 and Table 6. We note that the grids are not of the same size for all methods since in case an initial grid of values came out to be suboptimal, we expanded it further until the optimal value (according to common corruption accuracy) was attained not at the boundary of the grid. The only exception is for 100% Gaussian augmentation on ImageNet-100 in Table 6 where the best performance is attained for the smallest $\sigma = 0.001$ out of the final grid. We

Table 6: Grid searches performed for ImageNet-100 experiments. For each grid, we boldface the hyperparameter that leads to the model with the best common corruption accuracy.

Method	Grid values
100% Gaussian augmentation	$\sigma \in \{\mathbf{0.001}, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.4, 0.5, 0.6\}$
50% Gaussian augmentation	$\sigma \in \{0.01, 0.02, 0.05, 0.1, 0.2, \mathbf{0.5}, 0.8\}$
ℓ_∞ adversarial training	$\varepsilon \in \{0.01, \mathbf{0.05}, 0.1, 0.5, 1.0, 2.0, 3.0, 4.0\}/255$
ℓ_2 adversarial training	$\varepsilon \in \{0.02, \mathbf{0.05}, 0.1, 0.2\}$
RLAT	$\varepsilon \in \{0.01, \mathbf{0.02}, 0.05, 0.1, 0.2\}$
RLAT + AugMix	$\varepsilon \in \{0.01, 0.02, \mathbf{0.05}, 0.1, 0.2\}$
RLAT + AugMix + JSD	$\varepsilon \in \{0.01, 0.02, \mathbf{0.05}, 0.1, 0.2\}$
RLAT + SIN	$\varepsilon \in \{0.005, \mathbf{0.01}, 0.02, 0.05\}$
RLAT + ANT^{3x3}	$\varepsilon \in \{0.005, 0.01, \mathbf{0.02}, 0.05\}$

found out that even such a small σ still harms the overall performance due to σ-overfitting and elaborate further on this phenomenon on ImageNet-100 in Sec. D.

Data augmentation experiments. For the experiments with DeepAugment [Hendrycks et al., 2021], we generate distorted images once before training using the CAE model from their public repository. For the experiments that involve training on Stylized ImageNet, we use in each batch 28 stylized images and 100 standard ImageNet images following Rusak et al. [2020]. ANT [Rusak et al., 2020] is the only exception where instead of training from a random initialization, we follow the scheme of the authors and fine-tune a standardly pretrained ImageNet-100 model. Additionally, we note that the noise generator of Rusak et al. [2020] uses skip connections with Gaussian noise and experience replay of previous noise generators. This means that there is a certain Gaussian noise component in the final noise which implies that it partially overlaps with the common corruptions from CIFAR-10-C and ImageNet-100-C.

Licenses for the used and released assets. We release all our models under the MIT license. Throughout the paper, we used ImageNet [Russakovsky et al., 2015], CIFAR-10 [Krizhevsky and Hinton, 2009], ImageNet-C [Hendrycks and Dietterich, 2019], and CIFAR-10-C [Hendrycks and Dietterich, 2019] datasets and the Fast PAT [Laidlaw et al., 2021] model trained on ImageNet. Their licenses can be found in their repositories or webpages. Importantly, all their licenses are compatible for the purposes of academic research.

B ADDITIONAL DETAILS AND RESULTS ON CALIBRATION

In this section, we discuss the details on how we compute the expected calibration error and perform temperature rescaling to improve calibration. We also show calibration results for ℓ_2 adversarially trained models.

Calibration details. To compute the expected calibration error (ECE) we follow the code of Guo et al. [2017] with their default settings using 15 equally-sized bins to compute the calibration error. However, we change the implementation of the temperature rescaling. Since optimization of ECE over the softmax temperature is a simple one-dimensional optimization problem, it can be solved efficiently using a grid search. Moreover, we can optimize directly the metric of interest, i.e. ECE, instead of the cross-entropy loss as in Guo et al. [2017] who relied on a differentiable loss since they used gradient descent to optimize the temperature. As the grid, we use the interval $t \in [0.001, 1.0]$ with a grid step 0.001 and we test both t and $1/t$ temperatures. Moreover, we make sure that for all methods the optimal t is located not at the boundary of the grid. We optimize the temperature only on the *in-distribution* samples from the test sets of CIFAR-10 and ImageNet-100 to make sure that the out-distribution samples from CIFAR-10-C and ImageNet-100-C stay unseen.

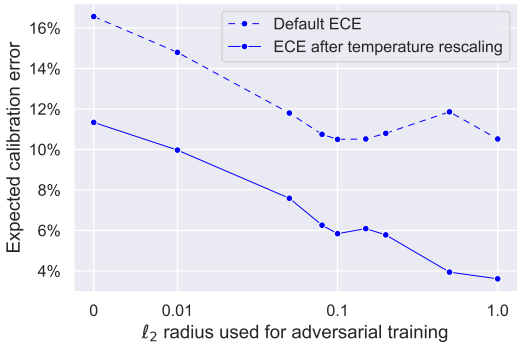


Figure 7: Expected calibration error on CIFAR-10-C for ℓ_2 adversarially trained models.

Calibration of ℓ_2 adversarially trained models. In Fig. 7, we additionally present the expected calibration error for models adversarially trained with different ℓ_2 perturbation radii. We observe a decreasing trend over the perturbation radius similarly to the ℓ_∞-trained models shown in Fig. 2. We see that the most accurate ℓ_2 model on corruptions is

significantly better calibrated: 10.5% ECE vs 16.6% ECE of the standard model. Moreover, the same amount of improvement can be observed even after temperature rescaling: 5.8% ECE vs 11.3% ECE of the standard model. Thus, we conclude that both ℓ_∞ and ℓ_2 adversarial training substantially improve calibration both before and after temperature rescaling. Moreover, ℓ_2 adversarial training leads to better calibration than ℓ_∞: 10.8% vs 10.5% ECE by default and 6.7% vs 5.8% ECE after temperature rescaling.

C ABLATION STUDY FOR ADVPROP AND ADVERSARIAL TRAINING

In this section, we provide experimental details for the AdvProp baseline and compare it with the standard ℓ_∞ adversarial training.

AdvProp [Xie et al., 2020] is a method based on adversarial training where the objective consists of a mixture of clean and adversarial examples for which separate BatchNorm layers are used, and only the clean BatchNorm layers are used at test time. This method was shown to improve the accuracy on clean images compared to standard adversarial training and to help to generalize under distribution shifts such as common corruptions, so we consider it here in more detail. As shown in Table 2, AdvProp achieves 94.7% standard accuracy and 82.9% common corruption accuracy. Thus AdvProp performs comparably to 100% ℓ_∞ adversarial training (82.9% vs 82.7% accuracy on CIFAR-10-C), however, AdvProp still performs worse than 100% ℓ_2 adversarial training (83.4%) and 100% RLAT (84.1%).

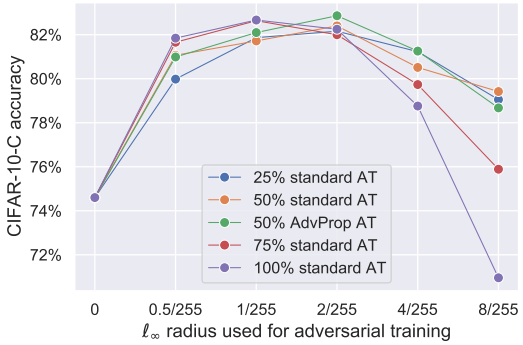


Figure 8: Accuracy on common corruptions from CIFAR-10-C for ResNet-18 models adversarially trained using different ℓ_∞ radii and different proportions of adversarial and clean examples together with the AdvProp scheme.

In Fig. 8, we show the results of an ablation study for Pre-Act ResNet-18 models adversarially trained using different ℓ_∞ radii and different proportions of adversarial and clean examples (25%, 50%, 75%, 100%) together with the AdvProp scheme. We can see that AdvProp outperforms 100% standard AT but only by a small margin (+0.2%). Moreover, 100% standard AT performs comparably to 75% standard AT and better than 50% and 25% standard AT. Thus, we observe no benefit in mixing clean and adversarial samples for standard AT unlike for Gaussian data augmentation. We emphasize here that the advantage of the standard ℓ_p adversarial training is that it is a conceptually simpler method as it does not require using separate BatchNorms during training, and balancing clean and adversarial samples. Besides, AdvProp requires up to 50% more training time if the same number of adversarial examples is used as for 100% standard AT.

D DETAILS ON σ-OVERFITTING

In this section, we provide experiments related to the σ-overfitting phenomenon on ImageNet-100 and provide a further discussion on it.

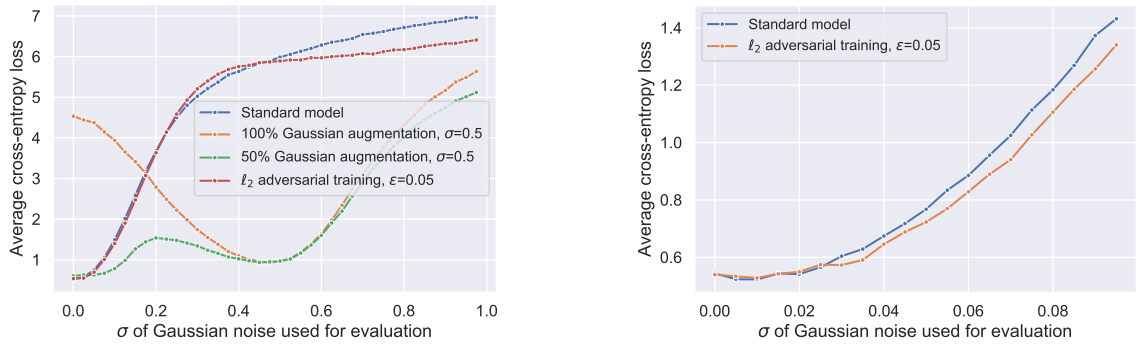
σ-overfitting on ImageNet. On ImageNet-100-C, the gap between 50% Gaussian augmentation and 100% Gaussian augmentation is even larger than on CIFAR-10-C (see Table. 2). To study the σ-overfitting phenomenon in more detail, we repeat the experiment behind Fig. 4 on ImageNet-100 and show the results in Fig. 9.

We first focus on large Gaussian perturbations (up to $\sigma = 1$ for image pixels in $[0, 1]$) in Fig. 9 (a). We observe that 100% Gaussian augmentation *severely* overfits to the noise magnitude used for training while the 50% Gaussian augmentation scheme mitigates this problem. However, 50% Gaussian augmentation does not completely solve the σ-overfitting problem since we observe that the loss still has a local maximum around $\sigma = 0.2$ (with a $\approx 1.6\times$ increase compared to the loss at $\sigma = 0.45$). Therefore, 50% Gaussian augmentation may be a suboptimal method against σ-overfitting. We believe that future research is needed to better understand improved mitigation strategies.

We additionally plot the performance of standard and ℓ_2 adversarially trained models for a smaller range of Gaussian perturbations with $\sigma \in [0, 0.1]$ in Fig. 9 (b). We can observe that there is no σ-overfitting trend for standard and ℓ_2 adversarially trained models. Moreover, the latter has a slightly smaller loss for small values of σ.

Ford et al. [2019] in the context of σ-overfitting. As a Gaussian data augmentation baseline, Ford et al. [2019] use a scheme that is actually different from both 50% and 100% Gaussian augmentation schemes. They perform Gaussian augmentation on *each* sample, however they sample the standard deviation uniformly at random from the range $[0, \sigma]$[5]. This strategy can be seen as an interpolation between the

[5]We confirmed this implementation detail via private communication with the authors. Note that this also corresponds to what [Rusak et al., 2020] report in Appendix H.



(a) A larger range of the standard deviation $\sigma \in [0, 1]$

(b) A smaller range of the standard deviation $\sigma \in [0, 0.1]$

Figure 9: The cross-entropy loss under Gaussian noise of different training methods for ImageNet-100. We see that 100% Gaussian augmentation *severely* overfits to the noise magnitude used for training while the 50% Gaussian augmentation scheme mitigates this problem, but not completely, since it has a noticeable local maximum $\sigma = 0.2$, and ℓ_2 adversarial training does not suffer from this problem.

50% and 100% Gaussian augmentation schemes. Moreover, another difference in the results from Ford et al. [2019] compared to our paper is that they computed corruptions for ImageNet-C in-memory which leads to some discrepancy compared to static images (e.g., see Fig. 5 in Ford et al. [2019] for an illustration).

σ-overfitting and adversarial training. As we can see from Fig. 4 and Fig. 9, adversarial training does not suffer from the σ-overfitting problem which may explain why applying 50% adversarial training (as in Sec. C) does not lead to better results compared to 100% adversarial training. We believe that the difference between Gaussian augmentation and adversarial training can occur due to a much larger norm of the perturbation used for data augmentation. On CIFAR-10-C, the optimal ℓ_2 radius for adversarial training is 0.1 while the norm of Gaussian vectors is approximately $\sigma_{100\%} \cdot \sqrt{d} = 0.05 \cdot \sqrt{32 \cdot 32 \cdot 3} \approx 2.77$ for 100% Gaussian augmentation and $\sigma_{50\%} \cdot \sqrt{d} = 0.1 \cdot \sqrt{32 \cdot 32 \cdot 3} \approx 5.54$ for 50% Gaussian augmentation. It is therefore likely that due to a much larger norm of the perturbation, the model trained with 100% Gaussian noise fails to generalize to perturbations of smaller norms, while this failure can be alleviated to some extent by using the 50% scheme. On a related note, some variations of adversarial training are known to suffer from a phenomenon called *catastrophic overfitting* [Tramèr et al., 2018, Wong et al., 2020]. Catastrophic overfitting shares some similarities with σ-overfitting, in the sense that the model overfits to a particular perturbation type used during training. Clarifying the relation of catastrophic overfitting and σ-overfitting is an interesting direction for future work.

E SUITABILITY OF LPIPS FOR COMMON CORRUPTIONS

We first show that the LPIPS distance is more suitable to common image corruptions than the ℓ_2 distance. We note that previous works [Zhang et al., 2018b, Laidlaw et al., 2021] have discussed its advantages over other commonly used distances, however not in the context of corruptions from CIFAR-10-C. Another difference to Zhang et al. [2018b] is that here we compute the LPIPS distance using *full images* instead of smaller image patches as we are eventually interested in performing adversarial training using full inputs.

We compute the average ℓ_2 and LPIPS distances based on a standardly trained VGG network using the code from Zhang et al. [2018b] for different corruptions from CIFAR-10-C. The results are shown in Fig. 5 (see also Table 8 in the Appendix), where we observe that for certain corruptions, LPIPS clearly demonstrates a more preferable behavior than the ℓ_2 distance. For example, the ℓ_2 distances of elastic transformations are monotonically *decreasing* over the corruption severity which is the opposite of what we would expect from a suitable distance between images. At the same time, the LPIPS distance is first slightly decreasing and then increasing which is a better behavior compared to ℓ_2. Similarly, the ℓ_2 distances for JPEG corruption appear to be roughly constant while they are noticeably increasing for LPIPS. The LPIPS distances for the frost corruption start from a low value and then monotonically increase whereas the ℓ_2 distances start from a very high value and then fail to be monotonic. At the same time, LPIPS behavior is more unexpected on noise corruption where it shows a very fast increase for impulse, Gaussian, and shot noises.

Therefore, the LPIPS distance appears to better capture the

Table 7: Correlation between distances and error rates of a standard model taken across different corruption severities and averaged over multiple corruptions. LPIPS distance is better correlated with the error rates on different corruption severity levels.

	Corruption type				
Metric	All	Noise	Blur	Weather	Digital
ℓ_2	0.807	**0.998**	0.986	0.835	0.561
LPIPS	**0.963**	0.993	**0.994**	**0.968**	**0.921**

severity of these corruptions. To investigate further this qualitative assessment, we compute the correlation between the l_2 and the LPIPS distances and the error rates of a standardly trained model (shown in Fig. 11) in Table 7. More precisely, each corruption has five severity levels for which both the average distance value and the average error rate can be calculated. Therefore, for a given corruption, we compute the correlation between the vector composed of the distances corresponding to each severity level and the error-rate vector defined similarly. We take then the average correlation over each corruption type. We observe that the LPIPS distance is *more correlated* with the error rates for all corruption types except the noise one. The main difference comes from the digital corruptions where LPIPS leverages the monotonic behavior of the frost and elastic transform corruptions. Therefore, we conclude that LPIPS is quantifiably more suitable to capture the distance between common image corruptions.

Average perturbation distance for different corruptions. In Table 8 we report the average distances between clean and corrupted images for the LPIPS and l_2 norms which is the same data as in Fig. 5. We report exact numbers to further illustrate the point that adversarial training with worst-case perturbations in a *small* ℓ_2 ball (such as $\varepsilon = 0.1$) leads to robustness against corruptions of a much larger magnitude. We can observe from Table 8 that for some corruptions, the perturbation norm does not grow monotonically (particularly, glass blur and elastic transform) which we highlight in **red**. We observe that for the LPIPS distance such behavior occurs less often. Another observation is that the l_2 distance itself does not always accurately reflects the strength of the performance degradation. For example, fog and brightness have similar magnitude (and the largest among the other corruptions), but very different behavior in terms of accuracy: degradation under fog is much higher than under brightness.

F DETAILS ON THE RELAXED LPIPS ADVERSARIAL TRAINING

In this section, we first discuss in more detail the approach of Laidlaw et al. [2021], then present complete derivations for RLAT where LPIPS is defined using multiple layers, discuss implementation details of RLAT, and evaluate the ℓ_2 robustness of RLAT and a few other baselines.

Details on the perceptual attacks of Laidlaw et al. [2021]. At each step of the Perceptual Projected Gradient Descent (PPGD) proposed in Laidlaw et al. [2021], both the loss ℓ and the neural network f used to define $\mathrm{d}_{\text{LPIPS}}$ are linearized, and the constrained problem (4) is approximated with a large linear system which is solved approximately with K iterations of the conjugate gradient method. To satisfy the constraint in (4), the solution δ is then approximately projected onto the LPIPS-ball, i.e. onto the set $\{\delta : \mathrm{d}_{\text{LPIPS}}(x, x+\delta) \leq \varepsilon\}$, for which n iterations of the bisection method are used. For T iterations of PPGD, the algorithm in total requires $T(K+n+4)$ forward passes and $T(K+n+3)$ backward passes of the network which makes it significantly more expensive than standard PGD which requires T forward and backward passes.

The Lagrangian Perceptual Attack (LPA) uses the following Lagrangian relaxation of the objective:

$$\max_{\delta} \ell(x+\delta, y; \theta) - \lambda \max\{\mathrm{d}_{\text{LPIPS}}(x, x+\delta) - \varepsilon, 0\},$$

which is solved by gradient descent for several values of the Lagrange multiplier λ (usually $S = 5$ in their experiments) and whose solution is then projected back onto the LPIPS-ball. In total, the attack requires $2ST+n+2$ forward passes and $ST+n+2$ backward passes of the network which also makes it expensive due to the outer loop over S different values of λ.

These two attacks are too computationally expensive to be efficiently used during adversarial training. To speed up the method, they additionally propose Fast-LPA where λ is not searched over but is instead increased during the training according to a fixed schedule and no projection steps are included. Then Fast-LPA requires $2T+1$ forward and $T+1$ backward passes that represent a small overhead compared to PGD but a large one when compared to single-step methods such as FGSM. We note that the possibility of using Fast-LPA with a few iterations is worth investigating in future work, although it appears to be not straightforward because of the Lagrangian formulation and the need to tune the parameter λ over iterations of Fast-LPA.

Relaxation for multi-layer LPIPS. We derive here the relaxation of LPIPS adversarial training for a general *multi-layer* version of the LPIPS distance. We recall from Eq. (3) that the LPIPS distance can be written as

$$\mathrm{d}_{\text{LPIPS}}(x, x+\delta)^2 = \sum_{l=1}^{L} \alpha_l \|\phi_l(x) - \phi_l(x+\delta)\|_2^2.$$

We use the convention that $\alpha_l = 0$ if a layer l is not in the set of the layers used in the LPIPS distance, i.e. if $l \notin \mathcal{L}_{LPIPS}$. We consider the network f written in its compositional form,

Table 8: Average ℓ_2 and LPIPS distance for different corruptions and severity levels from CIFAR-10-C. Note that the distances do not always monotonically increase with the corruption level. We mark such cases in **red** and observe that they occur less often for LPIPS than for the ℓ_2 distance.

	ℓ_2 distance at different severity levels					LPIPS distance at different severity levels				
Corruption	1	2	3	4	5	1	2	3	4	5
Shot noise	1.67	2.35	3.68	4.22	5.13	0.089	0.143	0.245	0.284	0.341
Motion blur	2.40	3.51	4.33	4.32	4.97	0.059	0.114	0.166	0.166	0.211
Snow	2.92	5.83	6.60	9.10	12.17	0.068	0.155	0.160	0.188	0.227
Pixelate	1.25	1.77	1.97	2.48	3.04	0.130	0.060	0.074	0.138	0.202
Gaussian noise	2.19	3.26	4.31	4.83	5.34	0.027	0.217	0.294	0.328	0.359
Defocus blur	0.43	1.06	1.61	2.09	2.98	0.003	0.019	0.048	0.085	0.153
Brightness	2.26	4.59	6.85	9.01	12.95	0.006	0.022	0.045	0.071	0.132
Fog	2.81	5.54	7.18	8.45	10.20	0.022	0.086	0.147	0.215	0.332
Zoom blur	3.04	3.56	4.20	4.83	5.40	0.079	0.094	0.124	0.153	0.189
Frost	6.86	10.14	11.39	10.45	10.26	0.077	0.141	0.208	0.214	0.266
Glass blur	4.70	4.64	4.27	6.73	6.28	0.253	0.248	0.235	0.333	0.321
Impulse noise	3.08	4.37	5.35	6.92	8.20	0.136	0.223	0.289	0.387	0.452
Contrast	2.80	5.60	6.71	7.83	9.51	0.020	0.092	0.143	0.216	0.386
JPEG compression	1.59	1.95	2.07	2.20	2.38	0.073	0.108	0.121	0.134	0.153
Elastic transform	7.37	6.79	6.11	5.67	4.76	0.168	0.152	0.151	0.175	0.198

i.e., $f(x) = g_L \circ \cdots \circ g_1(x)$. The LPIPS adversarial problem defined in Eq. 4 is then equivalent to the problem

$$\max_{\delta} \quad \ell(g_L(\dots g_1(x+\delta)\dots))$$
$$\text{s.t.} \quad \sum_{l=1}^{L} \alpha_l \left\|\phi_l(x) - \phi_l(x+\delta)\right\|_2^2 \leq \varepsilon^2 .$$

We introduce the slack variables $\tilde{\delta}^{(l)}$ for $l = 1, \dots, L$, defined as $\tilde{\delta}^{(l)} = g_l(g_{l-1}(\dots g_1(x + \tilde{\delta}^{(1)})\dots) + \tilde{\delta}^{(l-1)})g_l(\dots g_1(x)\dots)$ when $l \in \mathcal{L}_{LPIPS}$ and $\tilde{\delta}^{(l)} = 0$ otherwise. The previous problem can be written as:

$$\max_{\tilde{\delta}^{(1)},\dots,\tilde{\delta}^{(L)}} \quad \ell(g_L(\dots g_1(x+\tilde{\delta}^{(1)})\cdots + \tilde{\delta}^{(L)}))$$
$$\text{s.t.} \quad \sum_{l=1}^{L} \alpha_l \left\|\tilde{\delta}^{(l)}\right\|_2^2 \leq \varepsilon^2,$$
$$\tilde{\delta}^{(l)} = g_l(g_{l-1}(\dots g_1(x+\tilde{\delta}^{(1)})\dots) + \tilde{\delta}^{(l-1)}) - g_l(\dots g_1(x)\dots) \quad \forall l \in \mathcal{L}_{LPIPS},$$
$$\tilde{\delta}^{(l)} = 0 \quad \forall l \notin \mathcal{L}_{LPIPS}.$$

When relaxing the equality constraints on the slack variables $\tilde{\delta}^{(l)}$ for $l \in \mathcal{L}_{LPIPS}$ we obtain the following relaxation

$$\max_{\tilde{\delta}^{(1)},\dots,\tilde{\delta}^{(L)}} \quad \ell(g_L(\dots g_1(x+\tilde{\delta}^{(1)})\cdots + \tilde{\delta}^{(L)}))$$
$$\text{s.t.} \quad \sum_{l=1}^{L} \alpha_l \left\|\tilde{\delta}^{(l)}\right\|_2^2 \leq \varepsilon^2,$$
$$\tilde{\delta}^{(l)} = 0 \quad \forall l \notin \mathcal{L}_{LPIPS}.$$

We further relax the inequality constraint on $\sum_{l=1}^{L} \alpha_l \left\|\tilde{\delta}^{(l)}\right\|_2^2$ as individual constraints on each $\left\|\tilde{\delta}^{(l)}\right\|_2$ in the following way

$$\max_{\tilde{\delta}^{(1)},\dots,\tilde{\delta}^{(L)}} \quad \ell(g_L(\dots g_1(x+\tilde{\delta}^{(1)})\cdots + \tilde{\delta}^{(L)}))$$
$$\text{s.t.} \quad \left\|\tilde{\delta}^{(l)}\right\|_2 \leq \frac{\varepsilon}{\sqrt{\alpha_l}} \quad \forall l \in \mathcal{L}_{LPIPS},$$
$$\tilde{\delta}^{(l)} = 0 \quad \forall l \notin \mathcal{L}_{LPIPS}.$$

Denoting by $\varepsilon_l = \frac{\varepsilon}{\sqrt{\alpha_l}}$, we finally obtain the RLAT relaxation from Eq. 6

$$\max_{\tilde{\delta}^{(1)},\dots,\tilde{\delta}^{(L)}} \quad \ell(g_L(\dots g_1(x+\tilde{\delta}^{(1)})\cdots + \tilde{\delta}^{(L)})) \tag{7}$$
$$\text{s.t.} \quad \|\tilde{\delta}^{(l)}\|_2 \leq \varepsilon_l \quad \forall l \in \mathcal{L}_{LPIPS},$$
$$\tilde{\delta}^{(l)} = 0 \quad \forall l \notin \mathcal{L}_{LPIPS}.$$

We provide the algorithm for a single iteration of weight updates for RLAT in Algorithm 1. We show the weight update for standard SGD but any other optimizer can be used as well. We emphasize that one of the important advantages of RLAT is its computational efficiency since it leads to only $2\times$ overhead since we can successfully use a single-step adversarial training for it (see Fig. 6 for LPIPS robustness evaluation with an iterative attack). We refer to Table 3 for exact timings and comparison to Fast PAT and other methods.

Layer selection. We choose to use the following layers for LPIPS used for RLAT: input, conv1, conv2_x, conv3_x, conv4_x, and conv5_x. We tried various combinations of layers including all layers in the network, all BatchNorm layers, and all convolution layers, and the best results were

Algorithm 1: Single iteration of relaxed LPIPS adversarial training (RLAT)

input : network weights θ, batch of training samples $(x_i, y_i)_{i=1}^b$

output : updated network weights θ_{new}

for i ***in*** $\{1, \dots, b\}$ **do**
  $\forall l \in \{1, \dots, L\}$: $\tilde{\delta}_i^{(l)} := 0$
  for l ***in*** $\mathcal{L}_{LPIPS}$ **do**
    $\nabla_i^{(l)} := \nabla_{\tilde{\delta}_i^{(l)}} \ell(g_L(\dots g_1(x_i + \tilde{\delta}_i^{(1)}) \cdots + \tilde{\delta}_i^{(L)}), y_i)$
  for l ***in*** $\mathcal{L}_{LPIPS}$ **do**
    $\tilde{\delta}_i^{(l)} := \varepsilon_l \nabla_i^{(l)} / \left\| \nabla_i^{(l)} \right\|_2$
$\theta_{new} := \theta - \eta \nabla_\theta \frac{1}{b} \sum_{i=1}^b \ell(g_L(\dots g_1(\Pi_{[0,1]^d}[x_i + \tilde{\delta}_i^{(1)}]) \cdots + \tilde{\delta}_i^{(L)}), y_i)$
return θ_{new}

obtained when perturbations are added to the outputs of residual blocks and the first convolution layer.

Magnitude of layerwise perturbations. Similarly to $\{\alpha_l\}_{l=1}^L$ in the definition of the LPIPS distance in Eq. (3), the constraints $\{\varepsilon_l\}_{l=1}^L$ for different layers in Eq. (7) also need to be carefully selected so that the perturbation magnitudes on different layers are balanced. Our final approach sets the layerwise bounds ε_l proportionally to the layer's dimensionality and depth:

$$\varepsilon_l = \frac{1}{l} \frac{d_l}{d_{in}} \varepsilon,$$

where d_l is the dimension of the l-th feature maps and d_{in} is the input dimension. We choose this scaling since it is simple enough and empirically more effective than other simple scaling methods that we have tried such as the constant or inverse proportional strategies, or even more involved methods such as dynamic adjusting of the scale to the average of the layer's BatchNorm.

ℓ_2 robustness of RLAT. To complement the LPIPS robustness evaluation in Fig. 6, we also report the ℓ_2 robustness of the same set of models in Fig. 10: standard, ℓ_2 AT, Fast perceptual AT, and RLAT models with their main hyperparameters selected to perform best on common corruptions. We evaluate ℓ_2 robustness using the APGD-CE attack with 100 iterations and 5 random restarts [Croce and Hein, 2020] for different ℓ_2 radii $\varepsilon \in \{0.05, 0.1, 0.25, 0.5\}$. We observe that *all* three adversarial training methods improve the ℓ_2 robustness substantially compared to the standard model.

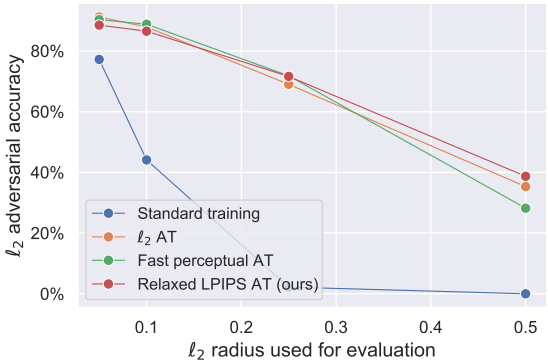


Figure 10: ℓ_2 adversarial robustness of different training schemes on CIFAR-10. All three adversarial training methods improve the ℓ_2 robustness substantially compared to the standard model.

G PERFORMANCE UNDER VARIOUS DISTRIBUTION SHIFTS

In this section, we provide additional experiments on distribution shifts that are different from the common corruptions that we studied throughout this paper.

For this, we use three distribution shifts to evaluate our models: ImageNet-A, ImageNet-R, and Stylized ImageNet (SIN). We report the results in Table 9 where we compute the accuracy on Stylized ImageNet on all 100 classes, and the accuracy on ImageNet-A and ImageNet-R on all classes that overlap with the classes of ImageNet-100. We use the same models for this evaluation as the ones reported in Table 2, i.e. these models have been selected after a grid search to maximize the performance on ImageNet-100-C. We observe that for evaluations on ImageNet-100-A and Stylized ImageNet-100, RLAT moderately improves the accuracy ($+0.4\%$ and $+1.1\%$ respectively) but does not yield improvements on ImageNet-100-R (-0.1%). As expected, training on SIN gives very significant improvements for an evaluation on SIN since the same distribution was used during training and testing. We also note that the performance of all methods could be improved if the model selection was performed on the target datasets, and not on ImageNet-100-C. Finally, we observe that for some data augmentation methods like AugMix and SIN, using RLAT leads to further improvements, e.g. from 35.1% to 37.0% for SIN + RLAT on ImageNet-100-R. Overall, we conclude that there is no method that performs best on all distribution shifts, and the obtained improvements are relatively small unless one uses a target distribution shift for training.

Table 9: Accuracy of various methods on different distribution shifts: ImageNet-100-A, ImageNet-100-R, and Stylized ImageNet-100. Gray-colored numbers correspond to models trained and evaluated using Stylized ImageNet.

Method	**Standard**	**IN-100-A**	**IN-100-R**	**IN-100-Stylized**
Standard training	**86.6%**	5.9%	33.2%	16.6%
100% Gaussian augmentation	86.4%	5.8%	31.2%	17.1%
50% Gaussian augmentation	83.8%	5.7%	32.6%	**18.9%**
Fast PAT	71.5%	5.4%	**34.6%**	17.7%
ℓ_∞ AT	86.5%	5.0%	33.2%	18.1%
ℓ_2 AT	86.3%	5.5%	33.2%	17.7%
RLAT	86.5%	**6.3%**	33.1%	17.7%
AugMix	86.7%	**5.5%**	31.5%	20.1%
AugMix + RLAT	**86.8%**	5.1%	**33.2%**	**20.6%**
Stylized ImageNet	**86.6%**	**6.5%**	35.1%	62.5%
Stylized ImageNet + RLAT	86.5%	**6.5%**	**37.0%**	**63.4%**
ANT$^{3\text{x}3}$	**85.9%**	**5.6%**	**33.3%**	**20.8%**
ANT$^{3\text{x}3}$ + RLAT	85.3%	5.3%	32.8%	20.7%

H SUPPLEMENTARY FIGURES AND TABLES

In this section, we present additional experimental results related to Sections 4 and 6.

Stability analysis. To check the stability of the results reported in Table 2, we repeated RLAT training on CIFAR-10 over 3 different random seeds. The corruption accuracy has the average 84.0% with the standard deviation of 0.2% which suggests that the method is quite stable with respect to random seeds.

Performance of different methods across individual corruptions. First, we show the performance of the simple baselines considered in Sec. 4 over each of the 15 corruptions of CIFAR-10-C. We show in Fig. 11 the error rates of the standard and the ℓ_2 adversarially trained models and in Fig. 12 a breakdown of the accuracy of all simple baselines (i.e., standard and ℓ_2 adversarially trained models, together with the models trained with gradient regularization and Gaussian data augmentation). We first note that ℓ_2 adversarial training leads to better average performance compared to other baselines, and improves on each corruption type (blurs, digital, noises, weather) and particularly on JPEG compression, elastic transform, pixelate, and zoom blur. However, compared to the standard model, adversarial training worsens the performance on contrast and fog, and slightly on brightness as has been observed in previous work for ℓ_∞ adversarial training with a large $\varepsilon = 8/255$ [Ford et al., 2019]. Moreover, Fig. 11 complements Fig. 5 and helps to motivate why the LPIPS distance can be more suitable than the ℓ_2 distance for the problem of being robust to common corruptions (see the discussion in Sec. E).

We additionally compare these baselines together with 50% Gaussian augmentation, AdvProp and RLAT in Table 10.

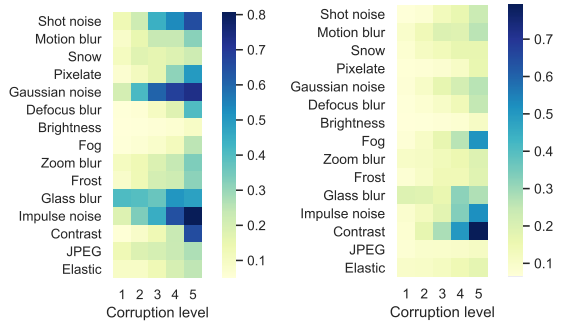


Figure 11: Error rates of a standard and ℓ_2 adversarially trained models on different common corruptions from CIFAR-10-C.

We observe that RLAT leads to better average performance than other baselines (not taking into account 50% Gaussian augmentation since it is partially trained with a corruption from CIFAR-10-C) and consistently improves upon ℓ_2 adversarial training. AdvProp helps for the snow and elastic transform corruptions, has a better behavior on corruptions on which ℓ_2/RLAT adversarial training and Gaussian augmentation perform poorly (such as brightness, fog, or contrast) but performs suboptimally on noise. Finally, 50% Gaussian augmentation obtains high accuracy, consistently improving upon 100% Gaussian augmentation (in particular on blur corruptions) due to its usage of clean samples which mitigates σ-overfitting.

Detailed performance of ℓ_2 adversarial training trained with different ε. We study further the influence of the radius ε on the performance of ℓ_2 adversarial training on CIFAR-10-C. We present in Table 11 the accuracy of ℓ_2 adversarially trained model, trained with different values of ε. Interestingly, Table 11 suggests that the same degradation

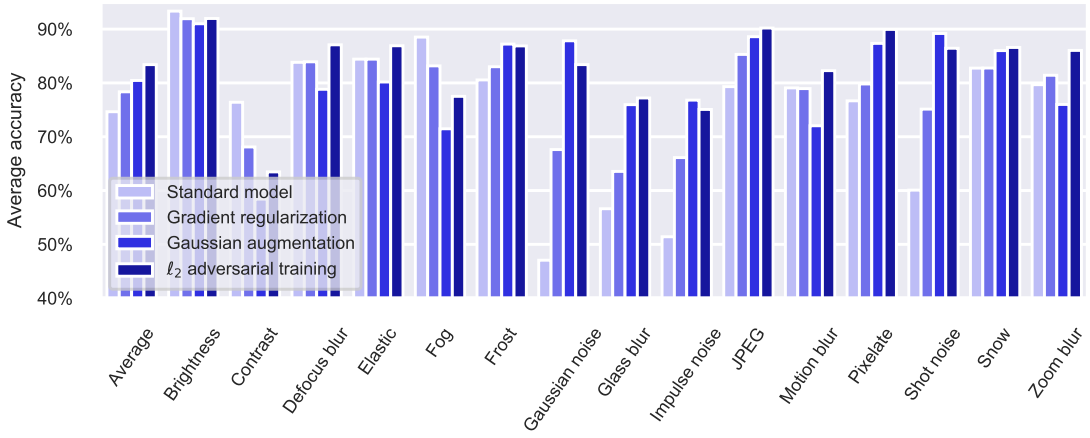


Figure 12: Accuracy for different individual corruptions on CIFAR-10-C. ℓ_2 adversarial training leads to better average performance compared to other baselines.

for fog and contrast occurs even for the *smallest* ε used for training. We also observe that different corruptions require different ε for optimal performance. In particular, the optimal ε for the frost corruptions is 0.05 while for impulse noise the optimal ε is an order of magnitude larger, i.e. 0.5. Despite being widely used in the literature on ℓ_2 robustness, $\varepsilon = 0.5$ is suboptimal for many other corruptions, and the optimal average performance over *all* corruptions for a single model is obtained at $\varepsilon = 0.1$.

Detailed model comparison for different corruption severity. In Tables 12 and 13 we report the accuracy of selected models for each corruption level separately. We first note that for all models, as one could expect, the accuracy is gradually decreasing with the corruption severity. When comparing all the baselines, RLAT shows the best accuracy on the majority of corruption levels (three on Imagenet-100 and four on CIFAR-10). It is also worth mentioning that AdvProp on CIFAR-10 works better for smaller corruption levels which can be explained by its higher standard accuracy. On ImageNet-100-C, we can observe that the Fast PAT model has the best accuracy at the highest severity level despite having clearly suboptimal standard accuracy (71.5% compared to 86.6% of the standard model) and average corruption accuracy (45.2% compared to 47.5% of the standard model). We also observe that for the severity level 4, ℓ_2 adversarial training is slightly better than RLAT (37.0% vs 36.8%). Moreover, we notice that when RLAT is combined with different data augmentation schemes, the improvement is often achieved at multiple severity levels simultaneously. For example, when RLAT is combined with AugMix, the accuracy is improved on all severity levels.

Results of AugMix combined with ℓ_2 and ℓ_∞ adversarial training. We report these results on CIFAR-10 in Table 14. We can observe that RLAT still outperforms ℓ_2 and ℓ_∞ adversarial training when they are combined with AugMix, both with and without the JSD consistency term from Hendrycks et al. [2019b]. Moreover, our RLAT model achieves 89.4% accuracy with the JSD term outperforming the best model from the AugMix paper Hendrycks et al. [2019b] despite being much smaller: we use ResNet-18 while the best model of Hendrycks et al. [2019b] uses ResNeXt-29.

Model selection based on validation corruptions. Many previous works use the same model selection scheme as we used, i.e. selecting the optimal hyperparameters on the test corruptions Hendrycks et al. [2019b], Xie et al. [2020], Rusak et al. [2020]. However, a proper validation-test split would lead to a more rigorous evaluation which we report in Table 15. We do a grid search based on the accuracy on the four validation corruptions from CIFAR-10-C. We see that for all adversarially trained models, *the optimal ε is the same* as we reported in Table 2 except for the ℓ_∞ model on ImageNet-100 for which, however, the final test accuracy is unchanged. Overall, this is perhaps not too surprising as the validation corruptions are from the same category of corruptions and we optimize over a small one-dimensional grid so we are unlikely to overfit to the test corruptions in this way.

Results on larger architectures. Since we performed all experiments for Table 15 on ResNet-18, we also test here another architecture to make sure that our findings generalize to other models. For this, we take the WRN-28-10 architecture on CIFAR-10 and similarly to other experiments, perform a grid search over ε for adversarial training methods (ℓ_∞, ℓ_2, RLAT). We report the results on the best ε for each method in Table 16 and observe that RLAT also outperforms other adversarial training methods for this architecture.

Table 10: Accuracy of clean training, gradient regularization, 100% and 50% Gaussian data augmentation, AdvProp, ℓ_2 adversarial training, and RLAT on CIFAR-10-C using ResNet-18. Gray-colored numbers correspond to methods that were at least partially trained with the corruptions from CIFAR-10-C.

Corruption	**Clean**	**GradReg**	**100% Gauss**	**50% Gauss**	**AdvProp**	**ℓ_2 AT**	**RLAT**
Shot noise	60.1%	75.1%	89.2%	**91.2%**	82.0%	86.4%	88.8%
Motion blur	79.1%	78.9%	72.0%	82.2%	82.2%	**82.3%**	**82.3%**
Snow	82.8%	82.8%	86.0%	85.9%	**86.9%**	86.6%	86.2%
Pixelate	76.7%	79.8%	87.4%	87.7%	87.7%	89.9%	**90.2%**
Gaussian noise	47.1%	67.6%	87.8%	90.8%	77.0%	83.4%	**86.0%**
Defocus blur	83.8%	83.9%	78.8%	86.5%	87.1%	87.1%	**87.2%**
Brightness	**93.3%**	91.9%	91.0%	91.2%	92.8%	92.0%	91.5%
Fog	**88.5%**	83.1%	71.5%	78.7%	88.2%	77.5%	76.7%
Zoom blur	79.7%	81.4%	76.0%	85.2%	85.8%	86.0%	**86.1%**
Frost	80.6%	83.0%	**87.2%**	**87.2%**	86.4%	86.9%	87.0%
Glass blur	56.6%	63.5%	76.0%	79.3%	74.4%	77.2%	**80.4%**
Impulse noise	51.4%	66.1%	76.8%	**87.3%**	71.8%	75.1%	79.6%
Contrast	**76.4%**	68.1%	58.4%	66.3%	68.6%	63.5%	62.6%
JPEG	79.3%	85.3%	88.6%	89.8%	89.8%	90.2%	**90.5%**
Elastic	84.4%	84.4%	80.2%	86.3%	**87.9%**	86.9%	87.2%
Average	74.6%	78.3%	80.5%	85.0%	82.9%	83.4%	**84.1%**

Table 11: Accuracy of ℓ_2 adversarial training for different ε on CIFAR-10-C. Similarly to Fig. 1 which was done for ℓ_∞ norm, we observe here that the widely used ℓ_2 radius $\varepsilon = 0.5$ leads to suboptimal corruption accuracy.

	Radius ε used for ℓ_2 adversarial training								
Corruption	$\varepsilon = 0$	$\varepsilon = 0.01$	$\varepsilon = 0.05$	$\varepsilon = 0.08$	$\varepsilon = 0.1$	$\varepsilon = 0.15$	$\varepsilon = 0.2$	$\varepsilon = 0.5$	$\varepsilon = 1$
Shot noise	60.1%	70.6%	81.6%	84.9%	86.4%	**87.0%**	86.9%	85.1%	79.6%
Motion blur	79.1%	79.4%	81.4%	81.8%	**82.3%**	82.1%	82.0%	80.0%	75.3%
Snow	84.7%	82.8%	86.6%	**86.9%**	86.6%	86.0%	85.5%	81.3%	75.0%
Pixelate	76.7%	82.0%	88.8%	89.7%	89.9%	**90.2%**	89.7%	86.5%	81.2%
Gaussian noise	47.1%	59.6%	76.0%	81.3%	83.4%	84.6%	**85.1%**	83.8%	78.4%
Defocus blur	83.8%	84.5%	86.2%	86.7%	**87.1%**	86.8%	86.5%	83.7%	78.6%
Brightness	**93.3%**	93.2%	93.0%	92.5%	92.0%	91.3%	90.2%	85.4%	79.0%
Fog	**88.5%**	86.7%	80.8%	78.5%	77.5%	74.4%	71.5%	61.5%	54.1%
Zoom blur	79.7%	81.9%	84.7%	85.6%	**86.0%**	85.7%	85.4%	82.7%	77.8%
Frost	80.6%	84.4%	**87.5%**	**87.5%**	86.9%	86.4%	85.7%	79.7%	71.5%
Glass blur	56.6%	62.7%	72.9%	76.1%	77.2%	80.7%	**81.6%**	81.5%	76.7%
Impulse noise	51.5%	58.7%	66.4%	73.2%	75.1%	76.5%	78.1%	**79.7%**	75.0%
Contrast	**76.4%**	72.3%	65.9%	63.7%	63.5%	61.1%	59.3%	50.5%	44.0%
JPEG	79.3%	86.0%	89.9%	**90.4%**	90.2%	90.3%	89.9%	86.7%	81.5%
Elastic	84.4%	85.7%	**87.2%**	**87.2%**	86.9%	86.6%	85.9%	82.4%	77.1%
Average	74.6%	78.2%	81.9%	83.1%	**83.4%**	83.3%	82.9%	79.4%	73.7%

Calibration before and after temperature rescaling. In Table 17 we report ECE before and after calibration via temperature rescaling [Guo et al., 2017] using in-distribution data for the models reported in Table 2. We observe that calibration is substantially improved after temperature rescaling, however it does not affect the ranking between different methods. In particular, all adversarial training methods significantly improve the calibration error, and RLAT leads to the best calibration reaching 1.3% ECE on CIFAR-10-C when combined with AugMix. Moreover, we note that all the CIFAR-10 models are *overconfident* since their optimal calibration temperature is greater than 1 (mostly in the range from 1.2 to 1.5). On ImageNet-100, the picture is similar and RLAT also improves calibration. For example, ANT$^{3\text{x}3}$ leads to 4.5% ECE after calibration while ANT$^{3\text{x}3}$ combined with RLAT achieves 2.8% ECE.

Comparison of in-distribution vs. out-distribution calibration. Throughout the paper, we focused only on calibration and accuracy on *common image corruptions*. Regarding calibration on in-distribution images, it is known that ℓ_p adversarial training can degrade the calibration quality (e.g., see Croce et al. [2020]). However, with the small ε that we use for adversarial training, this degradation is minimal. To

Table 12: Accuracy of different methods on CIFAR-10-C. Gray-colored numbers correspond to methods that were at least partially trained with the corruptions from CIFAR-10-C.

		Accuracy at different severity levels					
Method	Standard	1	2	3	4	5	Average
Standard training	95.1%	87.6%	82.3%	76.2%	69.2%	57.9%	74.6%
Gradient regularization	93.4%	87.9%	84.9%	81.0%	74.8%	65.5%	78.8%
100% Gaussian augmentation	92.5%	89.2%	86.3%	82.3%	76.4%	68.1%	80.5%
50% Gaussian augmentation	93.2%	91.0%	89.1%	86.8%	82.6%	75.8%	85.0%
Fast PAT	93.4%	89.5%	87.2%	83.7%	79.0%	72.6%	82.4%
ℓ_∞ AT	93.3%	90.8%	88.3%	84.5%	78.5%	71.2%	82.7%
AdvProp	94.7%	**91.5%**	88.9%	85.2%	79.2%	69.5%	82.9%
ℓ_2 AT	93.6%	91.1%	88.8%	85.5%	79.8%	71.8%	83.4%
RLAT	93.1%	91.1%	**89.0%**	**85.9%**	**80.9%**	**73.5%**	**84.1%**
DeepAugment	94.1%	91.0%	89.0%	86.6%	82.7%	77.3%	85.3%
DeepAugment + RLAT	93.6%	**91.7%**	**90.6%**	**89.1%**	**86.1%**	**81.6%**	**87.8%**
AugMix	95.0%	92.2%	90.5%	88.5%	84.7%	78.8%	86.9%
AugMix + RLAT	94.8%	**93.1%**	**91.8%**	**90.3%**	**87.0%**	**80.6%**	**88.5%**
AugMix + JSD	95.0%	92.9%	91.5%	89.9%	86.8%	82.1%	88.6%
AugMix + JSD + RLAT	94.8%	**93.3%**	**92.3%**	**90.9%**	**88.3%**	**83.3%**	**89.6%**

Table 13: Accuracy of different methods on ImageNet-100-C. Gray-colored numbers correspond to methods that were at least partially trained with the corruptions from ImageNet-100-C.

		Accuracy at different severity levels					
Method	Standard	1	2	3	4	5	Average
Standard training	86.6%	70.9%	58.7%	47.3%	35.2%	25.4%	47.5%
100% Gaussian augmentation	86.4%	70.1%	57.2%	46.2%	34.9%	25.3%	46.7%
50% Gaussian augmentation	83.8%	73.8%	65.0%	56.9%	45.7%	34.8%	55.2%
Fast PAT	71.5%	61.7%	52.7%	45.7%	36.6%	**29.1%**	45.2%
ℓ_∞ AT	86.5%	70.6%	57.9%	46.5%	36.2%	27.4%	47.7%
ℓ_2 AT	86.3%	70.1%	58.3%	47.8%	**37.0%**	27.9%	48.4%
RLAT	86.5%	**71.6%**	**59.6%**	**48.8%**	36.8%	27.1%	**48.8%**
AugMix	86.7%	73.9%	63.3%	53.9%	41.1%	29.5%	52.3%
AugMix + RLAT	86.8%	**75.4%**	**65.1%**	**56.2%**	**44.7%**	**32.8%**	**54.8%**
Stylized ImageNet	86.6%	73.0%	63.5%	55.1%	43.5%	**33.2%**	53.7%
Stylized ImageNet + RLAT	86.5%	**74.1%**	**64.4%**	**56.0%**	**44.1%**	33.0%	**54.3%**
$\text{ANT}^{3\text{x}3}$	85.9%	74.2%	**66.5%**	**58.9%**	49.8%	39.3%	57.7%
$\text{ANT}^{3\text{x}3}$ + RLAT	85.3%	**74.5%**	66.4%	58.8%	**50.5%**	**41.2%**	**58.3%**

illustrate this, we report below the calibration results on clean and corrupted CIFAR-10 data in Table 18. We can see that, in line with the literature, ℓ_p adversarial training degrades the calibration on clean data but only slightly (from 2.9% to 4.0% for RLAT). At the same time, all adversarial training methods significantly improve the calibration on corrupted data (from 16.6% to 9.9% for RLAT).

Table 14: CIFAR-10-C accuracy of AugMix without and with the JSD term combined with methods based on adversarial training (ℓ_∞, ℓ_2, RLAT).

Method	**Accuracy**
AugMix + ℓ_∞ adversarial training	87.8%
AugMix + ℓ_2 adversarial training	88.3%
AugMix + RLAT	**88.5%**
AugMix + JSD + ℓ_∞ adversarial training	89.0%
AugMix + JSD + ℓ_2 adversarial training	89.0%
AugMix + JSD + RLAT	**89.6%**

Table 15: Accuracy on the 15 test and 4 extra validation corruptions from *CIFAR-10-C* / *ImageNet-100-C* for the best models out of a grid of ε values for each method.

Method	Test accuracy	Validation accuracy	Same ε?
ℓ_∞ AT	82.7% / 47.7%	86.6% / 53.2%	**Yes** / No
ℓ_2 AT	83.4% / 48.4%	86.9% / 54.0%	**Yes / Yes**
RLAT	**84.1% / 48.8%**	**87.1% / 54.7%**	**Yes / Yes**

Table 16: CIFAR-10-C accuracy of adversarial training methods (ℓ_∞, ℓ_2, RLAT) using WRN-28-10 architecture.

Method	Accuracy
Standard training	75.6%
ℓ_∞ adversarial training	84.8%
ℓ_2 adversarial training	85.5%
RLAT	**85.9%**

Table 17: Calibration of ResNet-18 models trained on CIFAR-10 and ImageNet-100 before and after calibration via temperature rescaling. Gray-colored numbers correspond to methods partially trained with the corruptions from CIFAR-10-C and ImageNet-100-C.

Training	ECE before calibration	ECE after calibration
	CIFAR-10-C	
Standard	16.6%	11.3%
100% Gaussian	13.2%	7.8%
50% Gaussian	9.1%	4.6%
Fast PAT	12.0%	6.6%
AdvProp	10.1%	6.4%
ℓ_∞ adversarial	10.8%	6.5%
ℓ_2 adversarial	10.5%	5.8%
RLAT	**9.9%**	**5.1%**
DeepAugment	8.7%	4.4%
DeepAugment + RLAT	**6.1%**	**2.3%**
AugMix	6.9%	3.2%
AugMix + RLAT	**4.5%**	**1.3%**
AugMix + JSD	6.5%	4.2%
AugMix + JSD + RLAT	**5.4%**	**3.3%**
	ImageNet-100-C	
Standard	10.0%	6.4%
100% Gaussian	11.7%	8.9%
50% Gaussian	6.1%	4.5%
Fast PAT	8.0%	12.7%
ℓ_∞ adversarial	12.4%	10.9%
ℓ_2 adversarial	9.4%	6.5%
RLAT	**9.1%**	**5.4%**
AugMix	7.5%	5.8%
AugMix + RLAT	**4.7%**	**5.3%**
AugMix + JSD	1.9%	4.0%
AugMix + JSD + RLAT	**1.8%**	**2.1%**
SIN	6.7%	5.8%
SIN + RLAT	**6.0%**	**5.1%**
$\text{ANT}^{3\text{x}3}$	5.1%	4.5%
$\text{ANT}^{3\text{x}3}$ + RLAT	**4.4%**	**2.8%**

Table 18: A comparison of the expected calibration error (ECE) on in-distribution (CIFAR-10) vs. out-distribution (CIFAR-10-C) data.

Method	ECE on CIFAR-10	ECE on CIFAR-10-C
Standard training	**2.9%**	16.6%
ℓ_∞ adversarial training	3.9%	10.8%
ℓ_2 adversarial training	3.7%	10.5%
RLAT	4.0%	**9.9%**