Characterizing Model Robustness via Natural Input Gradients

Adrián Rodríguez-Muñoz
MIT CSAIL
adrianrm@mit.edu

Tongzhou Wang
MIT CSAIL
tongzhou@mit.edu

Antonio Torralba
MIT CSAIL
torralba@mit.edu

Abstract

Adversarially robust models are locally smooth around each data sample so that small perturbations cannot drastically change model outputs. In modern systems, such smoothness is usually obtained via *Adversarial Training*, which explicitly enforces models to perform well on perturbed examples. In this work, we show the surprising effectiveness of instead regularizing the gradient w.r.t. model inputs *on natural examples only*. Penalizing input *Gradient Norm* is commonly believed to be a much inferior approach. Our analyses identify that the performance of *Gradient Norm* regularization critically depends on the smoothness of activation functions, and are in fact extremely effective on modern vision transformers that adopt smooth activations over piecewise linear ones (*e.g.*, ReLU), contrary to prior belief. On ImageNet-1k, *Gradient Norm* training achieves $> 90\%$ the performance of state-of-the-art PGD-3 *Adversarial Training* (52% vs. 56%), while using only 60% computation cost of the state-of-the-art without complex adversarial optimization. Our analyses also highlight the relationship between model robustness and properties of natural input gradients, such as asymmetric sample and channel statistics. Surprisingly, we find model robustness can be significantly improved by simply regularizing its gradients to concentrate on image edges without explicit conditioning on the gradient norm.

1 Introduction

Deep neural networks have become the gold standard for computer vision tasks due to their strong empirical performances. However, they are also extremely brittle. Adversarial examples [53] are small manipulations to input images that can cause highly-performant models to fail catastrophically. For example, the accuracy of an ImageNet deep classifier drops from 84% to 0% under such attacks, even though the perturbed images look identical to humans.

To safely deploy these models in critical tasks such as medicine or autonomous vehicles, extensive research has been devoted to making robust models that are invariant to these small perturbations. The current foremost paradigm for obtaining robust models in practice has been *Adversarial Training* [39, 2], which trains models in a minimax fashion by optimizing classification losses over the attack-perturbed images. While this approach is effective (yielding 60% robust accuracy under attack), it is also extremely computationally expensive, taking $3.92\times$ wallclock time per training iteration compared to normal training (see Table 3). Therefore, it is important to seek properties of robust models that can be optimized much more efficiently.

In this work, we aim to characterize model robustness via *loss-input gradients* [48, 44]:

$$\nabla_x \mathcal{L} := \underbrace{\nabla_x \mathcal{L}_{\text{CE}}(f_\theta(x), y)}_{\text{loss-input gradient of model } f_\theta \text{ on example } x \text{ with groundtruth class } y}, \tag{1}$$

where $\mathcal{L}_{\text{CE}}$ is the cross entropy loss. The quantity $\nabla_x \mathcal{L}$ is related to the first-order Taylor expansion of model loss. It is known that a smaller *Gradient Norm* $\|\nabla_x \mathcal{L}\|$ is correlated to model smoothness

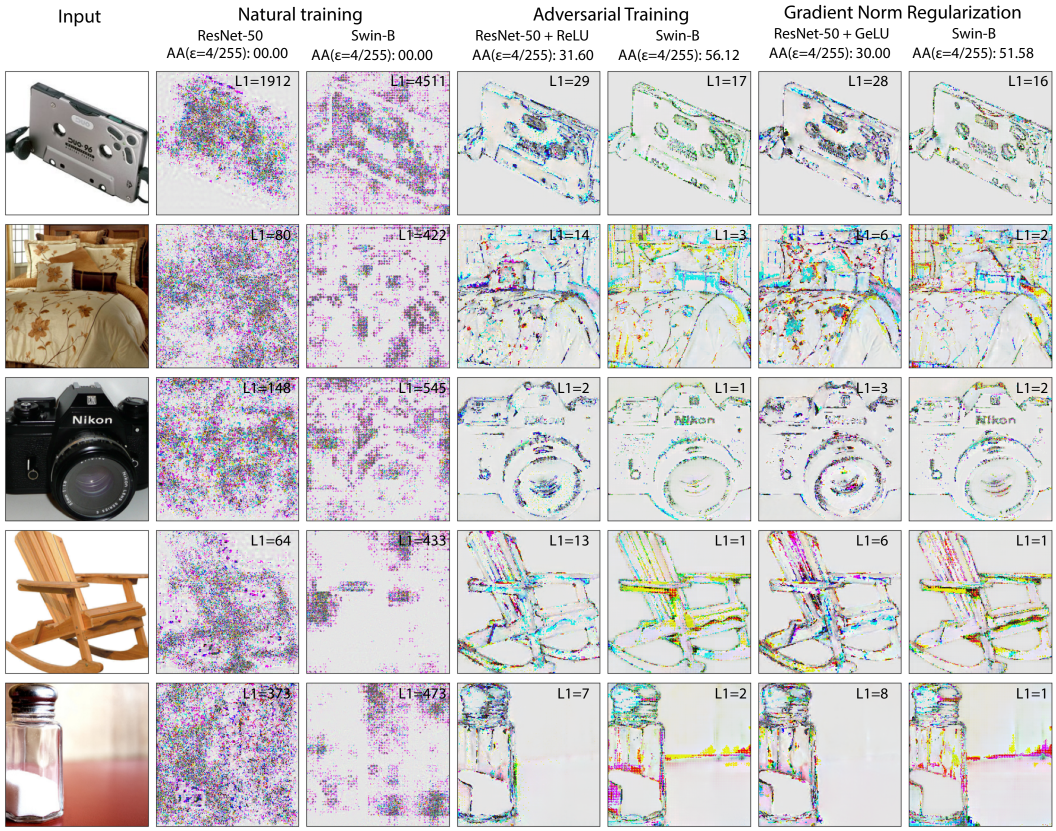

Figure 1: Comparison of loss-input gradients of non-robust and robust models across architectures for a set of images. Non-robust models are taken from the open-source repository `timm` [56]. Adversarial training is from the work of Liu *et al*. [33]. Gradient norm regularization optimizes Equation (2). As can be seen, a model can be easily identified as vulnerable or robust simply by looking at clean input gradients. Gradients of robust models (adversarial training and gradient norm regularization) highly resemble the input images, and look visually similar to each other to the human eye. By contrast, gradients of vulnerable models are noise-like, bearing apparently little resemblance to each other or the input images. Numerically, the norm of the input gradient (top right for each gradient) is also highly discriminative of vulnerability or robustness. Gradients normalized to [0, 1] then shifted by 0.4 for visualization purposes only.

and thus robustness [48, 44]. Figure 1 highlights the substantial visual differences between the loss-input gradients of vulnerable and robust models of the same architecture. Such differences reveal the regions of the image that the model uses to make predictions. Notably, the gradients for robust models are more "image-like". In robust models, the gradients are generally focused on the edges of input images, and in fact, have much smaller magnitudes (Table 1). In this paper, we analyze if these properties contribute to robustness, or are simply irrelevant byproducts.

Previous research [47] showed that simply regularizing $\|\nabla_x \mathcal{L}\|$ only yielded limited robustness compared to *Adversarial Training*, the current state of the art. We revisit this regularized training objective, and find it very effective on model architectures that use smooth activation functions, including modern vision transformers [35]. In contrast to prior beliefs, our results on ImageNet [11] show that more than 90% of robustness (compared to the state of the art) can be obtained by simply regularizing input gradients using 63% of the computational cost (see Table 3) and no inner maximization. Beyond regularizing magnitudes, we also obtain non-trivial robustness gains by enforcing gradients to concentrate on image edges in Section 5.3. Our work is the first to show the importance of input gradients $\nabla_x \mathcal{L}$ in both understanding and improving the robustness of modern neural network architectures.

In summary, our contributions are as follows:

1. We are the first to achieve near state-of-the-art robustness on ImageNet by only regularizing the norm of *natural* input gradients (Section 4.1).
2. We demonstrate that the effectiveness of gradient norm regularization critically depends on the smoothness of activation functions (Section 4.2).
3. We show that significant robustness appears when input gradients are more perceptually "image-like" and enforced to concentrate on image edges without using any other regularization (Section 5.3).

2 Related Works

Adversarial examples. Szegedy *et al*. [53] first identified the existence of *adversarial examples*, small perturbations imperceptible to humans but that completely fool networks. Since then, extensive researches have been conducted on the subject of adversarial examples, both on defending against such attacks [30, 54, 39, 20, 60, 57, 32, 5, 63, 41], as well as stronger attacks [18, 29, 4, 13, 61, 8], developing into a race between between attackers and defenders. Currently, the attack algorithm AutoAttack [8] is used by the community to empirically evaluate adversarial robustness. While it does not provide a certification of robustness, it has been shown to bypass gradient masking in practice [8], and requires no hyper-parameter tuning. This makes it a strong public progress benchmark [6]. Following the current SOTA work in adversarial robustness [33], we use it in our work for all main evaluations.

Training robust models. *Adversarial Training* emerged as the predominant paradigm to train robust models in practice [39, 2, 12, 45, 58, 33]. It is a complex bi-level algorithm, where at each iteration we generate a strong attack (via iterative optimization) and train the network to classify it correctly. In practice, such approaches, including the accelerated single-step Fast Gradient Sign Method (FGSM), require tuning many hyper-parameters and can be difficult to train [58, 31, 1, 27]. Alternative approaches attempted to regularize models to have small input gradient norms [47, 34, 44, 25, 15, 48, 38]. Despite strong theoretical arguments, no previous works have shown competitive performance on ImageNet from such regularizations. To our best knowledge, our work is the first to show its strong performance on modern architectures, and pinpoints activation function smoothness as the deciding factor of its effectiveness.

Perceptually aligned gradients. Previous works noted that robust models tend to have *perceptually aligned gradients*, *i.e.*, class gradients $\nabla_x f_\theta(x)_{y_t}$ that align with human perception [26, 46, 51] (see also Figure 1). In this work, we consider the reverse implication and ask: do perceptually aligned gradients imply robustness? The work of [17] aligned model class gradients $\nabla_x f_\theta(x)_{y_t}$ with several notions of class-representative images (defined via real images and generative models), and observed improved robustness on small datasets, but only small benefits on TinyImageNet. Our paper shows that simply aligning gradients to image edges yields much stronger robustness gains on the more challenging full ImageNet-1k. In addition to perceptual alignment, we also highlight several other properties that separate gradients of robust and non-robust models.

3 Experimental Settings for Evaluating Robustness

For all our experiments, we use the standard settings for large-scale experiments with modern architectures used by the community, following the work of Liu *et al*. [33], the current SOTA work in adversarial robustness, and the public RobustBench benchmark [6]. Specifically, we focus on a setting consisting of (1) the ImageNet-1K classification task, (2) Swin Vision Transformers architecture [35], (3) the training recipe of Liu *et al*. [33], and (4) a full end-to-end knowledge adversary model with an L_∞ constraint of $\frac{4}{255}$, empirically benchmarked with AutoAttack [8]. See more details below.

Dataset Our work aims to provide insights into real-world, large-scale classification scenarios. We choose to work with the challenging ImageNet-1K [11] dataset for all experiments. Additionally, the 224x224 resolution images of ImageNet enable higher-resolution visualization and easier analyses.

Architecture We use the current state-of-the-art architecture for Adversarial Robustness on ImageNet, Swin Transformers [35], albeit in base size. We will also start from a standard pre-trained checkpoint from `timm` [56] in all experiments to reduce computational expense from 300 to 100 epochs per full training run (roughly 3 weeks to 1 week reduction on 8 V100 GPUs). From the work of Liu *et al*. [33], we know the impact of using pre-trained initializations and base size models is measurable but qualitatively equivalent, so we argue it is a sufficiently strong setting for our experiments. When analysing the effect of architectural choices in Section 4.2, we run experiments on ResNet50's [21] to allow for more extensive ablations, due to the aforementioned computational cost of training transformers.

Adversarial Training skyline and training recipe We use the work of Liu *et al*. [33], the current state-of-the-art for L_∞ robust accuracy on ImageNet, both as a skyline to compare performance as well as for their strong training recipes. Unless otherwise stated, as per the work of Liu *et al*. [33] all training recipes for Swin Transformers last 100 epochs, and use standard training tricks and augmentations like RandAugment [10, 9], mixup with label smoothing and random erasing [62], the AdamW optimizer with weight decay [36], and model averaging [24, 19, 3]. Recipes used for ResNets are equivalent but halved in length. The full recipe details for all experiments are included in Appendix G of the appendix, and the code can be found on our github.

Attack benchmark We generally evaluate robustness using AutoAttack-L_∞ [8] with perturbation strength $\epsilon = \frac{4}{255}$, following the RobustBench standard [7]. AutoAttack has access to the end-to-end model, and contains a mix of white-box and black-box attacks. The white-box attacks themselves also use a mix of loss functions. When evaluating robustness for a dense set of perturbations strengths ϵ, we use the PGD100 attack due to computational cost. Implementation-wise, we use the code and test set of Liu *et al*. [33]. In addition to testing with AutoAttack, the work of Carlini *et al*. [4] also provides a checklist of important practices, common pitfalls, and sanity checks in empirical evaluations of robustness, which we verify for Gradient Norm Regularization in the Appendix C.

4 Small L_1 Gradient Norm Makes a Model Robust

Mathematically, the stability of a function to small perturbations is captured by the norm of its derivative. Hence, it is a necessary condition for adversarial robustness to have small gradient norm [1]. However, previous works encountered that minimizing gradient norm was not competitive with adversarial training on large benchmarks [47, 34, 44, 25, 15, 48, 38]. Is this still the case with modern vision transformers [35]? Why did this performance gap exist? We take a deep look in this section.

A function stable to perturbations of the input has a small gradient w.r.t. the input. Empirically, it has been observed that adversarial robustness (obtained with adversarial training) [39] correlates with small gradient norm [48]. As we can see in Table 1, this continues to hold with state-of-the-art robust transformers. The expected L_1 norm of the loss-input gradient $\|\nabla_x \mathcal{L}\| := \|\nabla_{\mathbf{x}} \mathcal{L}_{\text{CE}}(f_\theta(\mathbf{x}, y)\|_1$ is around two orders of magnitude smaller for robust models than for their non-robust counterparts of the same architecture. Furthermore, fixing the models and taking expectations over inputs conditioning on PGD10 attack success, the gradient is much smaller when the attack fails than when it succeeds (last two columns of Table 1).

Despite the large amounts of theoretical and empirical evidence supporting low norms for robustness, the results of extensive previous work studying low gradient norms as a regularizer [47, 34, 44, 25, 15, 48, 38] have so far been either inconclusive, or non-performant on large datasets like ImageNet. In the following section, we show how a simplified version of the gradient norm L_1 penalty, trained through double back-propagation [14], is within 5% of the state-of-the-art on adversarial robustness on ImageNet despite seeing only natural examples, showing how small gradients have a larger driving role in robustness than previously thought.

4.1 Regularizing for Small Gradient Norms

Works in the literature studying gradient norm regularization have presented numerous formulations with differing details, such as optimizing Jacobian gradients [25, 34, 47], regularizing gradients of both natural and adversarial examples [47], or regularizing through a discrete scheme [15]. In this work, we study the effect of the cleanest formulation possible of the objective, as presented in equation 4 of [48] and optimized in [44], which we restate in Equation (2)

Table 1: Accuracy, robustness, and gradient norm statistics on 10k ImageNet validation images for publicly available vulnerable and robust models from `timm` [56] and RobustBench [7] respectively. The quantities Standard, AA, and PGD10 refer to clean accuracy, and AutoAttack and PGD10 robust accuracy respectively. The quantities $\mathbf{E}[L_1|\checkmark]$ and $\mathbf{E}[L_1|\times]$ are the conditional expectations of the loss input-gradient L_1 norm conditioned on the PGD10 attack failing and succeeding respectively.

		Accuracy			**Gradient Norm**		
Architecture	**Training**	Standard	AA	PGD10	$\mathbf{E}[L_1]$	$\mathbf{E}[L_1\|\checkmark]$	$\mathbf{E}[L_1\|\times]$
Resnet-50	Std. (He *et al.*)	76.35	00.00	00.25	3013	0.030	3021
	Adv. (Salman *et al.*)	63.99	34.96	39.98	56.82	11.78	86.82
Swin-B	Std. (Liu *et al.*)	84.84	00.00	02.77	3892	546	3983
	Adv. (Liu *et al.*)	76.86	56.16	59.25	37.42	10.96	75.90
Swin-L	Std. (Liu *et al.*)	86.13	00.00	02.07	2408	285.9	2453
	Adv. (Liu *et al.*)	78.53	59.56	61.33	33.73	7.08	75.64

Table 2: Robustness of a Swin Transformer trained with gradient norm regularization compared to natural training and state-of-the-art adversarial training on AutoAttack-L_∞. Adversarial training performed from pretrained `timm` [56] checkpoint using the recipe of [33].

	Clean	**AutoAttack-L_∞**		
Method	-	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
Natural Training	84.19	00.00	00.00	00.00
Grad. Norm ($\lambda_{\text{CE}} = 0.8, \lambda_{\text{GN}} = 1.2$)	77.78	72.04	66.20	51.58
Adv. Train. (PGD-3, $\epsilon = 4$)	77.20	72.46	67.38	56.12

$$\mathcal{L}_{\text{GradNorm}}(\mathbf{x}, y) = \lambda_{\text{CE}} \mathcal{L}_{\text{CE}}(f_\theta(\mathbf{x}), y) + \lambda_{\text{GN}} \frac{\epsilon}{\sigma} \|\nabla_{\mathbf{x}} \mathcal{L}_{\text{CE}}(f_\theta(\mathbf{x}), y))\|_1 \tag{2}$$

where $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss, $\epsilon = \frac{4}{255}$ is the adversarial strength, $\sigma = 0.225$ is the standard deviation used for normalization on ImageNet, and $\lambda_{\text{CE}}, \lambda_{\text{GN}}$ are weighing hyper-parameters set 0.8 and 1.2 respectively.

As we can see Table 2, training on the above objective yields a highly competitive model despite the constraints of seeing only natural examples and having 60% of the computational budget. On AutoAttack L_∞ with $\epsilon = \frac{4}{255}$, the standard benchmark for ImageNet [7], gradient norm regularization obtains 51.58% robust performance compared to the 56.12% obtained by state-of-the-art adversarial training (also starting from a pretrained checkpoint) [33]. For smaller epsilons, the gap shrinks to 1.18% and 0.42% for ϵ of 2 and 1 respectively, though it may be possible that the gap would be larger if the Adversarial Training was performed with a lower ϵ than $\frac{4}{255}$.

More interestingly, we also evaluate behaviour for higher values of ϵ. In Figure 2, we plot robust performance on PGD100 for a dense ϵ interval from 0 to $\frac{32}{255}$. As we can see, while the gap to adversarial training grows larger as a function of the adversarial strength ϵ, it is always less than 10%. Additionally, for $\epsilon < \frac{12}{255}$, accuracy is always two orders of magnitude above chance. This is an incredibly surprising result: despite gradient norm regularization working on only natural examples, it remains strong even for very large ϵ. Moreover, while single-step adversarial training with FGSM will fail without injection of Gaussian noise prior to attack calculation [58] (a technique called random initialization), this is not necessary for gradient norm. Moreover, Figure 2 also verifies the sanity check that robustness approaches 0% as the adversarial strength ϵ grows unboundedly.

This showcases both the strong bias towards smoothness endowed by modern architectural choices, as well as the strong robustness driving effect of small gradient norms. Table 3 displays computational cost comparisons between natural and adversarial training compared to gradient norm regularization. Roughly speaking, it means that for smooth swin transformers, adversarial training with PGD-3 is expending 40% of its computational budget improving results by 5.1% accuracy points, or about an 8.8% relative increase. Similarly, minimizing the loss input-gradient L_1 norm is responsible for 92% of state-of-the-art robust accuracy.

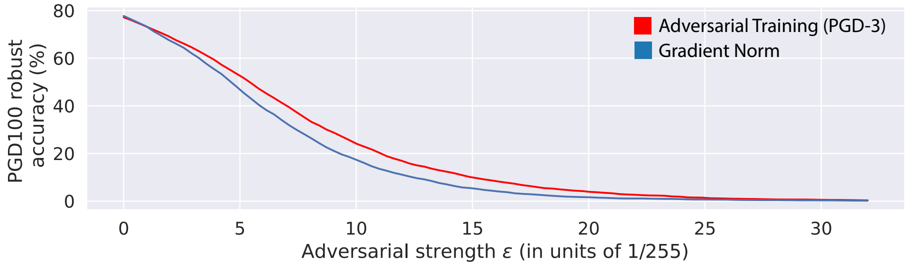

Figure 2: Robust accuracy vs epsilon for the PGD100 attack on ImageNet for Swin Transformer trained on Gradient Norm Regularization and state-of-the-art Adversarial Training. Gradient Norm Regularization achieves slightly better accuracy on clean images ($\epsilon = 0$) and good robust performance ($\epsilon > 0$), despite seeing only natural examples and having 60% of the computational cost of Adversarial Training with PGD-3. Robust accuracy for both models smoothly converges towards 0% as the adversarial strength grows.

Table 3: Computational cost per batch comparison between natural training, adversarial training, and gradient norm regularization for Swin Transformer (base size). Theoretical cost measured in number of network passes per batch, and empirical cost measured in seconds per batch. Experiments conducted on the same set of 8 V100 GPUs without mixed precision. Averages and standard deviations reported for the average batch time over three separate runs.

Method	# passes	Rel. to Adv. Train.	Empirical cost (s)	Rel to Adv. Train.
Nat. Train.	2	0.250	0.749 ± 0.00216	0.255
Grad. Norm.	5	0.625	1.848 ± 0.01108	0.628
Adv. Train. (PGD-3)	8	1.000	2.943 ± 0.00141	1.000

In the following section, we empirically show the drastic effect of smooth non-linearities on the performance of gradient norm regularization, even on smaller architectures such as ResNets.

4.2 Smooth Activation Functions Make Gradient Norm Regularization Effective

The formulation of the gradient norm objective in Equation (2) is extremely similar to that of previous works [47, 15, 44, 25], so why are results so different now? As was openly discussed by previous works [15, 48], ReLU networks are non-smooth, *i.e.*, they have non-differentiable gradients. While they raised concerns regarding the effect of these non-differentiable gradients on gradient norm regularization, they did not conduct tests to measure the size of this effect, which we perform in this section.

We set up the following controlled comparison. First, we take a pre-trained ResNet-50 [56] and replace all the ReLU non-linearities with smooth GeLUs and SiLUs [23]. This change causes clean accuracy to decrease to 0%, so we finetune over three epochs. For consistency, we also finetune the ResNet-50 with ReLUs with the same recipe and seed. Next, we make a copy of each network and train each copy on the same recipe (a halved version of the recipe used for transformers), with both adversarial training with PGD-3 and gradient norm regularization. Throughout all the trainings we keep the batch-normalizations in evaluation mode in order to isolate the effect of the non-linearity. The evaluations on L_∞-AutoAttack($\epsilon = \frac{4}{255}$) are shown on Table 4. We show validation accuracies as functions of epoch in Figure 4.

As we can see in Table 4 and Figure 4, the ResNet with ReLU is completely incapable of properly fitting the objective at the appropriate strength, with clean performance sharply decaying and robust performance barely increasing, compared to the ResNet with GeLU, despite training on the same recipe with the same regularization objective and weights. In contrast, the gradient norm regularized GeLU ResNet displays similar convergence behaviour to the adversarially trained model, obtaining extremely similar final clean and robust accuracies.

The work of [59] conducted a similar analysis for Adversarial Training, observing small increases in performance from using smooth non-linearities. As we can see from Table 4, for Gradient Norm

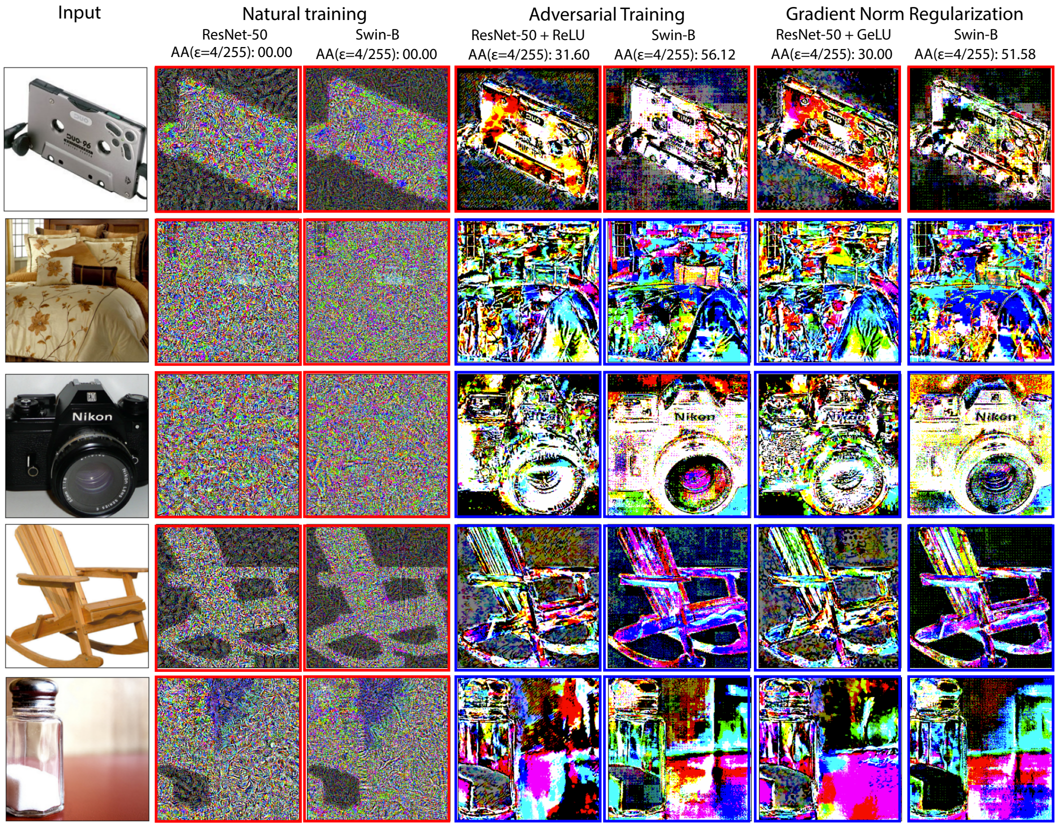

Figure 3: Comparison of PGD10 $L_\infty(\epsilon = \frac{4}{255})$ perturbations of non-robust and robust models across architectures for a set of images (same as in Figure 1). The border is blue if the model is robust to the perturbation, and red if it is not robust. As with clean input gradients, models can again be easily identified as vulnerable or robust simply by looking at the perturbations. Perturbations coming from robust models (adversarial training and gradient norm regularization) highly resemble the input images, though the visual similarity has decreased w.r.t. the input gradients. Perturbations originating from vulnerable models are now even more noise-like, with the exception of images with very flat backgrounds, potentially because the gradient may oscillate around zero in those areas. Perturbations normalized to [0, 1] for visualization purposes only.

regularization the effect is more than 20 times larger: robust performance on AutoAttack with adversarial training increases by 1%, while for gradient norm the increase is roughly 23%. In essence, the gradient norm regularization objective minimizes a penalty on the gradients of the network; since for ReLU networks the latter is non-differentiable, Taylor's theorem does not necessarily hold on the gradient loss, so there is no guarantee that gradient descent will work.

Additionally, we found the usage of adaptive optimizers like Adam [28], as well as a relatively small warm-up learning rate of 10^{-5}, to be extremely important. Especially at the beginning, the size of the norm of the gradient changes drastically; using a non-adaptive optimizer or too high a learning rate also causes performance to similarly crash, even on smooth networks, for the regularization weights required to reach performance comparable to adversarial training. This may have been another reason behind the lack of conclusive success of previous works.

In Figure 3, we visualize PGD10 perturbations of the same vulnerable and robust models from Figure 1. Similarly as in Figure 1, obvious visual differences exist between the perturbations of vulnerable and robust models.

Table 4: Clean and L_∞-AutoAttack accuracy for ResNet50 with ReLU, GeLU, and SiLU non-linearities trained with both Adversarial Training and GradNorm for 50 epochs using a shortened version of the Adversarial Transformer recipe of [33].

Method		**Accuracy**	
Training	Non-linearity	Standard	AutoAttack-L_∞
GradNorm	ReLU	16.94	6.82
	GeLU	60.34	30.00
	SiLU	61.84	30.58
Adv. Train.	ReLU	59.46	31.60
	GeLU	59.34	32.64
	SiLU	60.58	33.40

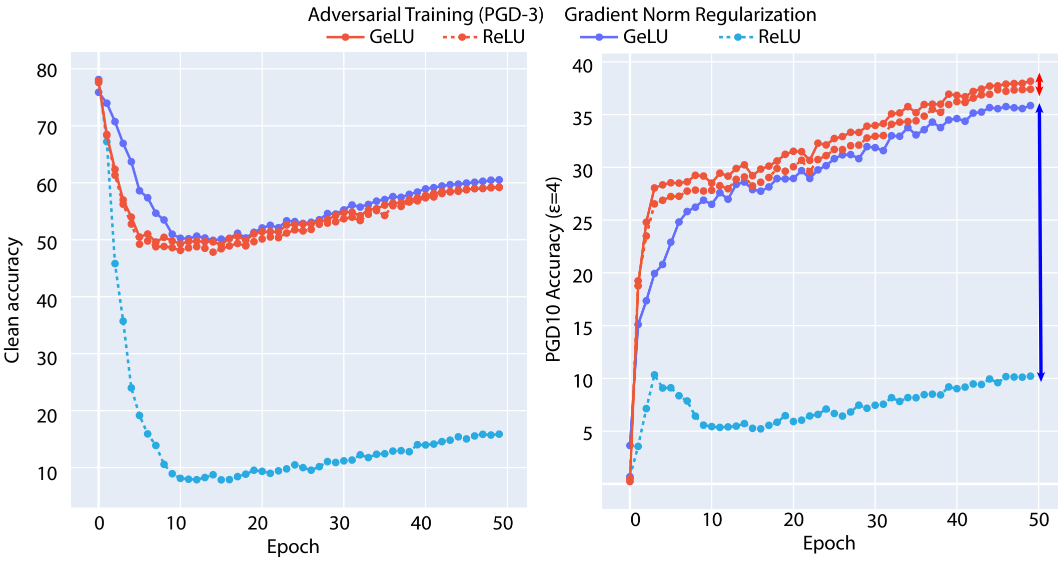

Figure 4: Clean and PGD10 ($\epsilon = 4$) robust accuracy vs epoch for ResNet50 with ReLU and GeLU trained with Adversarial Training and Gradient Norm Regularization. We observe how the ReLU ResNet is not capable of handling the regularization objective at the appropriate strength.

4.3 Trading Off Smoothness and Performance

Figure 5 plots clean and AutoAttack [8] accuracy as a function of GradNorm weight λ for Swin Transformer. Increasing the GradNorm weight from 1.2 to 1.4 increases AutoAttack accuracy at $\epsilon = \frac{4}{255}$ by almost 1%, from 51.58% to 52.48%. While the gap to Adversarial Training robust accuracy is reduced to 3.64%, clean accuracy degrades by 1.5%, from 77.78% to 76.20%. Hence, the "optimal" value depends on the user-specified trade-off between clean accuracy and robustness.

5 Beyond Small L_1 Norm: Other Properties of Robust Gradients

In this section, we investigate properties of adversarially robust gradients beyond small average L_1 gradient norm: (1) differences between gradients of correctly and incorrectly classified clean examples, (2) differences between RGB channels of gradients, and (3) alignment of gradients to image edges.

5.1 Image Gradient Norm Correlates with Clean Classification Accuracy

Figure 6 plots the L_1 norm distribution of loss-input gradients for clean images for the adversarially trained model, and the most robust GradNorm model of Figure 5 ($\lambda = 1.4$). The adversarially trained model exhibits strong bimodality (Figure 6.a). This is due to differing behaviour on wrongly-classified

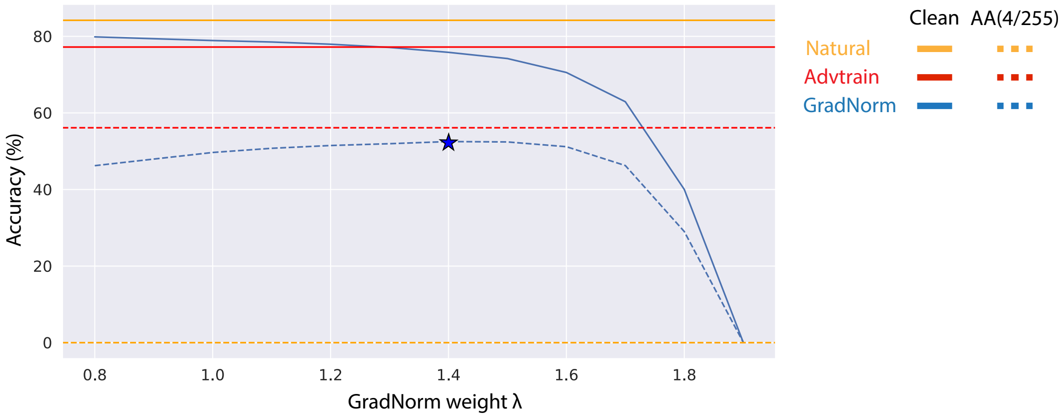

Figure 5: Clean and AutoAttack (AA; $\epsilon = \frac{4}{255}$) accuracy for optimizing $(2 - \lambda)\mathcal{L}_{\text{CE}} + \lambda \|\nabla_x \mathcal{L}\|_1$ for Swin Transformer. Models obtained by finetuning original GradNorm model from Table 2 for 30 epochs due to computational cost of training from scratch. Maximum robust accuracy obtained for $\lambda = 1.4$, which has clean accuracy 76.28 and AutoAttack accuracy 52.48. The robust accuracy gap to Adversarial Training is 3.64%.

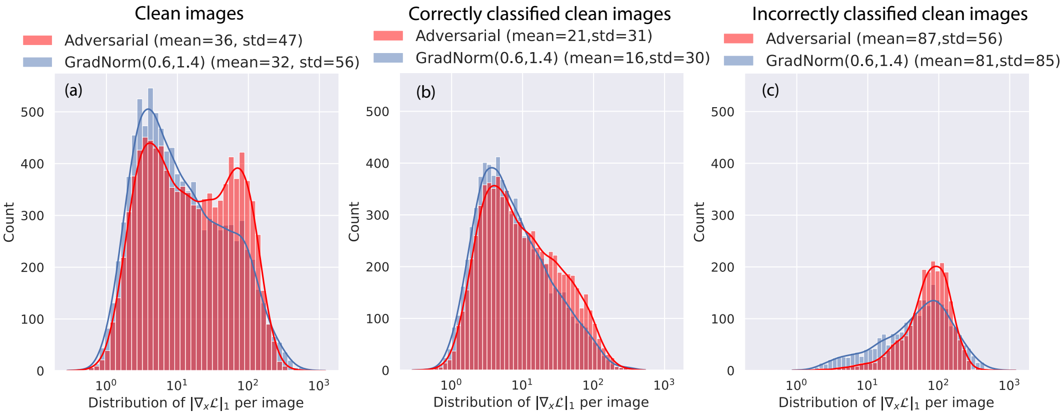

Figure 6: L_1 loss-input gradient norm ($\|\nabla_x \mathcal{L})\|_1$) distribution of AdvTrain and best GradNorm Swin Transformer models for different sets of clearn images. On all images (**left**), the adversarially trained model has a bimodal distribution, while the GradNorm model does not. This is mostly due to differing behaviour on incorrectly classified examples (**right**). While the AdvTrain model has a highly concentrated distribution on large gradient norms for incorrect examples, the GradNorm model has a much higher proportion of low gradient norms.

clean examples, as we can see in Figure 6.c. The gradient norm distribution for incorrectly classified clean examples is highly concentrated around large values for the adversarially trained model, but is more distributed around low values in the GradNorm model.

5.2 Not All Color Channels Have Equally Small Gradient Norms

Figure 7 plots the distribution of the absolute values of gradients across 1000 validation images. In Figure 7 left, we see that adversarially trained gradients have more small values than gradient norm regularized gradients, even though the mean is slightly higher. The distribution for the latter tapers off at 10^{-9}, while the former has a bump between 10^{-9} and 10^{-10}. In other words, the adversarially trained gradients are sparser, with more dimensions closer to zero. But what are these additional sparse dimensions?

The right three plots of Figure 7 show the gradient magnitude distributions *for each channel*. we observe that the difference observed in above is almost entirely due to the green channel of the gradients. This asymmetric role of the green channel is seemingly unique to adversarial training.

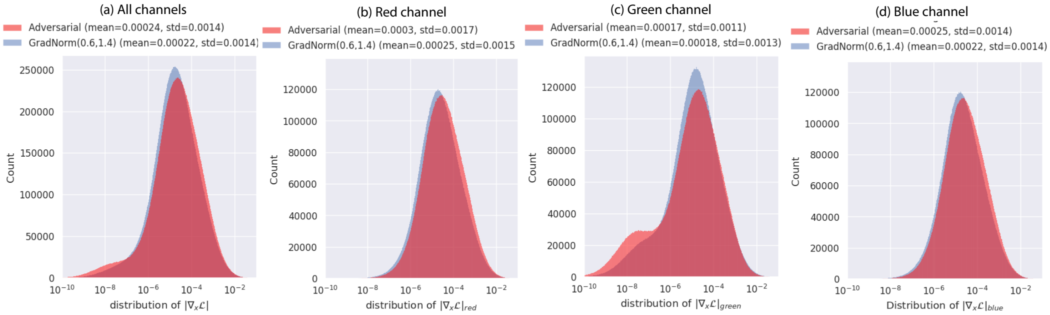

Figure 7: Distribution of absolute value of gradients over 1k images over all channels (**a**), and each RGB color channel (**b,c,d**), for AdvTrain and best GradNorm Swin Transformer models. In **a**, we see adversarial gradients have a significantly fatter left tail; essentially, the small values are much lower. From the **b,c,d** channel-specifcic plots, we observe that most of this difference is owed to the green channel: the small values of the green channel of adversarially trained gradients are very small. This asymmetric behaviour is partially missing from the gradient norm regularized models.

It's possible this may be due to the special role that the green channel plays in image coding, such as sampling using Bayer filters [37], but we are not able to test it experimentally without access to ImageNet in RAW format. Instead, we conducted alternative experiments where GradNorm is penalized stronger for the green channel, but didn't obtain positive results. Doubling the weight of the green channel reduced both clean and robust accuracy, from 77.74% to 77.38% and 51.66% to 51.38% respectively.

5.3 Aligning Gradient to Image Edges Improves Robustness

Adversarially robust models are known to be perceptually aligned—class gradients $\nabla_x f_\theta(x)_{y_t}$ are image-like and align with human perception [26, 46, 51]. In particular, the saliency maps (the per-pixel maximum absolute value of the target class input-gradients) of vulnerable and robust models display opposite correlations with the image edges despite highlighting the same object. This is seen visually in Figure 8 and numerically in Table 5. The image edges, also known as the oriented energy of the image, measure of the amount of local change at each pixel. We calculate them using Sobel filters [16] as is standard in signal processing. In simple terms, horizontal and vertical derivatives are calculated using a convolution, squared, and added.

Table 5 reports the log-log correlation between both the saliency map and the loss input-gradient absolute value with the edges of the input. For the naturally trained model, both values are significantly negative at around -0.45, for both robust models, they are significantly positive at around +0.56.

In Figure 8, this similarity between saliency maps and image edges (*i.e.*, oriented energy) is visualized, showing the correlation between log saliency map and log oriented energy for individual images. Like in Table 5, we observe strong positive correlation for robust models, and weaker negative correlations for non-robust models.

A natural question that arises is whether this property is merely a consequence of robustness or actually induces robust behavior. We find that, to a significant extent, it is actually the latter. Aligning class gradients ($\nabla_x f_\theta(x)_{y_t}$, *i.e.*, the gradient of target-class logit w.r.t. input) with image edges yields 60% of the robustness of SOTA Adversarial Training. This is reported in Table 5 under the name "Edge Regularization". The exact form of the regularizer is given by the following Equation (3).

$$\mathcal{L}_{\text{edge}}(\mathbf{x}, y) := \mathcal{L}_{\cos}(\nabla_{\mathbf{x}} f_\theta(x)_t, |g_u * x|^2 + |g_v * x|^2) \tag{3}$$

$$t \sim \text{softmax}\left(\frac{f_\theta(x)}{0.5}\right) \tag{4}$$

where $\mathcal{L}_{\cos}$ is the cosine similarity loss and $\nabla_{\mathbf{x}} f_\theta(x)_y$ is a class gradient. The class is chosen according to the probabilities predicted by the model modulated by a temperature of 0.5, expressed mathematically with softmax$(f_\theta(x)/0.5)$. This is done to bias selections towards the likely classes. Additionally, $|g_u * x|^2 + |g_v * x|^2$ are the image edges: $*$ denotes 2D convolution, and g_u, g_v denote gaussian horizontal and vertical derivative filters respectively.

Table 5: Average and standard deviation of Pearson correlation between log saliency map magnitude and log oriented energy across 10k validation images. We remark a positive correlation of $+0.56$ for the robust models and a negative correlation of -0.45 for vulnerable vanilla models. This shows that their gradients are spatially located in opposite regions of the image, the edge regions vs the flat regions respectively, despite accurately classifying the same object. Moreover, we believe these values undersell the relationship, as the edges are naively calculated over the whole image and thus highlight irrelevant objects. Edges / oriented energy calculated as $\text{edge}(x) = |g_u * x|^2 + |g_v * x|^2$, where $*$ denotes the convolution operation, and g_u, g_v denote Sobel horizontal and vertical derivative filters respectively.

	Accuracy		Pearson corr. of log w/ log oriented energy	
Training	Clean	AutoAttack	Saliency map	$\lvert\nabla_x \mathcal{L}_{\text{CE}}(f_\theta(x), y)\rvert$
Natural	$\mathbf{84.19}$	00.00	$-0.4510 \pm (0.1467)$	$-0.4428 \pm (0.1498)$
Gradient Norm	77.78	51.58	$+0.5627 \pm (0.1640)$	$+0.5679 \pm (0.1647)$
Adversarial (PGD-3)	77.20	$\mathbf{56.16}$	$+0.5627 \pm (0.1569)$	$+\mathbf{0.5692} \pm (\mathbf{0.1569})$
Edge regularization	76.80	35.02	$+\mathbf{0.6055} \pm (\mathbf{0.1644})$	$+0.3882 \pm (0.2020)$

In addition to optimizing the class gradient, we also attempted other forms of edge regularization, such as using loss gradients or the saliency map, but were not able to obtain good results. Please see Appendix E for results with alternate losses and other training details.

Unlike gradient norm regularization, Equation (3) does nothing to a-priori regularize the norm of the gradients, only the direction. While not at the level of models regularized purposefully for robustness, this serves as significant evidence to support the claim that perceptual similarity directly causes robustness. Moreover, the true value of the result may lie in the following: it is much easier to conceptualize devising an architecture that natively focuses on edges than one that has low gradient norm or is resistant to arbitrary perturbations. That is, the success of edge regularization could potentially provide a start towards a *structurally robust* architecture *i.e.* one that displays adversarial robustness even when trained normally. We leave this direction to future work.

6 Implications

What properties of input gradients characterize model robustness? In our work, we find that, on architectures with smooth non-linearities, cleanly minimizing the L_1 norm of the loss input gradients achieves close to state-of-the-art robust performance, despite never training on perturbed examples. This implies that (1) model robustness is significantly characterizable by behaviour on natural inputs, and (2) architectural changes like non-linearity choice may drastically change the effectiveness of alternate approaches towards robustness. We also find an additional characterization of robust gradients based on image edges, independent of norm, that achieves 60% performance of state-of-the-art. While numerically weaker than the gradient norm result, the implications are still potentially significant. Specifically, it provides a possible hint towards a *naturally robust architecture* that, through directly enforcing dependence only on the edge regions of the image, is resistant to perturbations despite being naturally trained.

Our results show that input gradients provide a useful lens to both analyze and improve model robustness. This approach suggests several interesting future research directions to explore: (1) Are there other model properties that quantify robustness? (2) How can we characterize the performance gap between Gradient Norm Regularization and Adversarial Training? (3) How do these properties explain the relationship between robustness and architecture in general (beyond activation smoothness)? Based on our results, we believe that answering these questions will lead to further insights and advances in building more general and robust deep models.

Broader Impact Deep neural networks have become the gold standard for computer vision tasks, but are also extremely brittle under small distribution shifts, including adversarial attacks. In this work, we highlighted deep connections between the robustness of neural networks and the statistics of their gradients, and proposed several approaches to train robust and highly-performant neural networks. We believe that our findings are important for applying such models in real-world safety-critical tasks.

Acknowledgements

Adrián Rodríguez-Muñoz is supported by the LaCaixa fellowship. Tongzhou Wang is supported by the ONR MURI program.

References

[1] Andriushchenko, M., Flammarion, N.: Understanding and Improving Fast Adversarial Training (Oct 2020), http://arxiv.org/abs/2007.02617, arXiv:2007.02617 [cs, stat]

[2] Athalye, A., Carlini, N., Wagner, D.: Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples (Jul 2018), http://arxiv.org/abs/1802.00420, arXiv:1802.00420 [cs]

[3] Bolme, D.S., Draper, B.A., Beveridge, J.R.: Average of synthetic exact filters. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2105–2112 (2009)

[4] Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I., Madry, A.: On evaluating adversarial robustness. arXiv preprint arXiv:1902.06705 (2019)

[5] Cohen, J.M., Rosenfeld, E., Kolter, J.Z.: Certified adversarial robustness via randomized smoothing. In: International Conference on Machine Learning. pp. 1310–1320 (2019)

[6] Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., Hein, M.: Robustbench: a standardized adversarial robustness benchmark. arXiv preprint arXiv:2010.09670 (2020)

[7] Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., Hein, M.: RobustBench: a standardized adversarial robustness benchmark. In: Neural Information Processing Systems Datasets and Benchmarks Track (2021)

[8] Croce, F., Hein, M.: Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In: International Conference on Machine Learning. pp. 2206–2216 (2020)

[9] Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation policies from data. arXiv preprint arXiv:1805.09501 (2018)

[10] Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.: RandAugment: Practical Automated Data Augmentation with a Reduced Search Space. In: Advances in Neural Information Processing Systems. pp. 18613–18624 (2020)

[11] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009)

[12] Dong, Y., Fu, Q.A., Yang, X., Pang, T., Su, H., Xiao, Z., Zhu, J.: Benchmarking adversarial robustness on image classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 321–331 (2020)

[13] Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., Li, J.: Boosting adversarial attacks with momentum. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9185–9193 (2018)

[14] Drucker, H., Le Cun, Y.: Improving generalization performance using double backpropagation. IEEE Transactions on Neural Networks **3**(6), 991–997 (Nov 1992). https://doi.org/10.1109/72.165600, https://ieeexplore.ieee.org/document/165600/authors#authors, conference Name: IEEE Transactions on Neural Networks

[15] Finlay, C., Oberman, A.M.: Scaleable input gradient regularization for adversarial robustness (Oct 2019), http://arxiv.org/abs/1905.11468, arXiv:1905.11468 [cs, stat]

[16] Freeman, W., Adelson, E.: The design and use of steerable filters. IEEE Transactions on Pattern Analysis and Machine Intelligence **13**(9), 891–906 (Sep 1991). https://doi.org/10.1109/34.93808, https://ieeexplore.ieee.org/document/93808, conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence

[17] Ganz, R., Kawar, B., Elad, M.: Do Perceptually Aligned Gradients Imply Adversarial Robustness? (Aug 2023), http://arxiv.org/abs/2207.11378, arXiv:2207.11378 [cs]

[18] Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: International Conference on Learning Representations (2015)

[19] Gowal, S., Qin, C., Uesato, J., Mann, T., Kohli, P.: Uncovering the limits of adversarial training against norm-bounded adversarial examples. arXiv preprint arXiv:2010.03593 (2020)

[20] Guo, C., Rana, M., Cisse, M., Maaten, L.V.D.: Countering Adversarial Images using Input Transformations. In: International Conference on Learning Representations (2018)

[21] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)

[22] He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European Conference on Computer Vision. pp. 630–645 (2016)

[23] Hendrycks, D., Gimpel, K.: Gaussian Error Linear Units (GELUs) (Jun 2023). https://doi.org/10.48550/arXiv.1606.08415, http://arxiv.org/abs/1606.08415, arXiv:1606.08415 [cs]

[24] Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., Wilson, A.G.: Averaging weights leads to wider optima and better generalization. In: 34th Conference on Uncertainty in Artificial Intelligence. pp. 876–885 (2018)

[25] Jakubovitz, D., Giryes, R.: Improving DNN Robustness to Adversarial Attacks using Jacobian Regularization (May 2019), http://arxiv.org/abs/1803.08680, arXiv:1803.08680 [cs, stat]

[26] Kaur, S., Cohen, J., Lipton, Z.C.: Are Perceptually-Aligned Gradients a General Property of Robust Classifiers? (Oct 2019), http://arxiv.org/abs/1910.08640, arXiv:1910.08640 [cs, stat]

[27] Kim, H., Lee, W., Lee, J.: Understanding Catastrophic Overfitting in Single-step Adversarial Training (Dec 2020). https://doi.org/10.48550/arXiv.2010.01799, http://arxiv.org/abs/2010.01799, arXiv:2010.01799 [cs, eess, stat]

[28] Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization (Jan 2017). https://doi.org/10.48550/arXiv.1412.6980, http://arxiv.org/abs/1412.6980, arXiv:1412.6980 [cs]

[29] Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial examples in the physical world. arXiv preprint arXiv:1607.02533 (2016)

[30] Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial Machine Learning at Scale. In: International Conference on Learning Representations (2017)

[31] Li, Z., Liu, L., Wang, Z., Zhou, Y., Xie, C.: Bag of Tricks for FGSM Adversarial Training (Sep 2022), http://arxiv.org/abs/2209.02684, arXiv:2209.02684 [cs]

[32] Liao, F., Liang, M., Dong, Y., Pang, T., Hu, X., Zhu, J.: Defense against adversarial attacks using high-level representation guided denoiser. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1778–1787 (2018)

[33] Liu, C., Dong, Y., Xiang, W., Yang, X., Su, H., Zhu, J., Chen, Y., He, Y., Xue, H., Zheng, S.: A Comprehensive Study on Robustness of Image Classification Models: Benchmarking and Rethinking (Feb 2023). https://doi.org/10.48550/arXiv.2302.14301, http://arxiv.org/abs/2302.14301, arXiv:2302.14301 [cs]

[34] Liu, D., Wu, L., Zhao, H., Boussaid, F., Bennamoun, M., Xie, X.: Jacobian Norm with Selective Input Gradient Regularization for Improved and Interpretable Adversarial Defense (Nov 2022), http://arxiv.org/abs/2207.13036, arXiv:2207.13036 [cs]

[35] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10012–10022 (2021)

[36] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)

[37] Lukac, R., Plataniotis, K.: Color filter arrays: design and performance analysis. IEEE Transactions on Consumer Electronics **51**(4), 1260–1267 (Nov 2005). https://doi.org/10.1109/TCE.2005.1561853, https://ieeexplore.ieee.org/document/1561853, conference Name: IEEE Transactions on Consumer Electronics

[38] Lyu, C., Huang, K., Liang, H.N.: A Unified Gradient Regularization Family for Adversarial Examples (Nov 2015), http://arxiv.org/abs/1511.06385, arXiv:1511.06385 [cs, stat]

[39] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards Deep Learning Models Resistant to Adversarial Attacks. In: International Conference on Learning Representations (2018)

[40] Moosavi-Dezfooli, S.M., Fawzi, A., Uesato, J., Frossard, P.: Robustness via curvature regularization, and vice versa (Nov 2018). https://doi.org/10.48550/arXiv.1811.09716, http://arxiv.org/abs/1811.09716, arXiv:1811.09716 [cs, stat]

[41] Pang, T., Yang, X., Dong, Y., Su, H., Zhu, J.: Bag of Tricks for Adversarial Training (Mar 2021), http://arxiv.org/abs/2010.00467, arXiv:2010.00467 [cs, stat]

[42] Qin, C., Martens, J., Gowal, S., Krishnan, D., Dvijotham, K., Fawzi, A., De, S., Stanforth, R., Kohli, P.: Adversarial Robustness through Local Linearization. In: Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019), https://proceedings.neurips.cc/paper_files/paper/2019/hash/0defd533d51ed0a10c5c9dbf93ee78a5-Abstract.html

[43] Rocamora, E.A., Liu, F., Chrysos, G., Olmos, P.M., Cevher, V.: Efficient local linearity regularization to overcome catastrophic overfitting (Oct 2023), https://openreview.net/forum?id=SZzQz8ikwg

[44] Ross, A.S., Doshi-Velez, F.: Improving the Adversarial Robustness and Interpretability of Deep Neural Networks by Regularizing their Input Gradients (Nov 2017), http://arxiv.org/abs/1711.09404, arXiv:1711.09404 [cs]

[45] Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., Madry, A.: Do adversarially robust imagenet models transfer better? In: Advances in Neural Information Processing Systems. vol. 33, pp. 3533–3545 (2020)

[46] Santurkar, S., Tsipras, D., Tran, B., Ilyas, A., Engstrom, L., Madry, A.: Image Synthesis with a Single (Robust) Classifier (Aug 2019). https://doi.org/10.48550/arXiv.1906.09453, http://arxiv.org/abs/1906.09453, arXiv:1906.09453 [cs, stat]

[47] Seck, I., Loosli, G., Canu, S.: L 1-norm double backpropagation adversarial defense (Mar 2019). https://doi.org/10.48550/arXiv.1903.01715, http://arxiv.org/abs/1903.01715, arXiv:1903.01715 [cs]

[48] Simon-Gabriel, C.J., Ollivier, Y., Bottou, L., Schölkopf, B., Lopez-Paz, D.: First-order Adversarial Vulnerability of Neural Networks and Input Dimension (Jun 2019), http://arxiv.org/abs/1802.01421, arXiv:1802.01421 [cs, stat]

[49] Simonyan, K., Vedaldi, A., Zisserman, A.: Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps (Apr 2014), http://arxiv.org/abs/1312.6034, arXiv:1312.6034 [cs]

[50] Singla, V., Singla, S., Jacobs, D., Feizi, S.: Low Curvature Activations Reduce Overfitting in Adversarial Training (Aug 2021), http://arxiv.org/abs/2102.07861, arXiv:2102.07861 [cs]

[51] Srinivas, S., Bordt, S., Lakkaraju, H.: Which Models have Perceptually-Aligned Gradients? An Explanation via Off-Manifold Robustness (May 2023), http://arxiv.org/abs/2305.19101, arXiv:2305.19101 [cs]

[52] Srinivas, S., Lakkaraju, H., Matoba, K., Fleuret, F.: Efficient Training of Low-Curvature Neural Networks

[53] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks (Feb 2014). https://doi.org/10.48550/arXiv.1312.6199, http://arxiv.org/abs/1312.6199, arXiv:1312.6199 [cs]

[54] Tramèr, F., Kurakin, A., Papernot, N., Boneh, D., McDaniel, P.: Ensemble Adversarial Training: Attacks and Defenses. In: International Conference on Learning Representations (2018)

[55] Tsiligkaridis, T., Roberts, J.: Second Order Optimization for Adversarial Robustness and Interpretability (Sep 2020), http://arxiv.org/abs/2009.04923, arXiv:2009.04923 [cs, stat]

[56] Wightman, R.: PyTorch Image Models (2019). https://doi.org/10.5281/zenodo.4414861, https://github.com/rwightman/pytorch-image-models, publication Title: GitHub repository

[57] Wong, E., Kolter, Z.: Provable defenses against adversarial examples via the convex outer adversarial polytope. In: International Conference on Machine Learning. pp. 5286–5295 (2018)

[58] Wong, E., Rice, L., Kolter, J.Z.: Fast is better than free: Revisiting adversarial training (Jan 2020), http://arxiv.org/abs/2001.03994, arXiv:2001.03994 [cs, stat]

[59] Xie, C., Tan, M., Gong, B., Yuille, A., Le, Q.V.: Smooth Adversarial Training (Jul 2021), http://arxiv.org/abs/2006.14536, arXiv:2006.14536 [cs]

[60] Xie, C., Wang, J., Zhang, Z., Ren, Z., Yuille, A.: Mitigating Adversarial Effects Through Randomization. In: International Conference on Learning Representations (2018)

[61] Xie, C., Zhang, Z., Zhou, Y., Bai, S., Wang, J., Ren, Z., Yuille, A.L.: Improving transferability of adversarial examples with input diversity. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2730–2739 (2019)

[62] Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond Empirical Risk Minimization. In: International Conference on Learning Representations (2018)

[63] Zhang, Z., Jung, C., Liang, X.: Adversarial defense by suppressing high-frequency components. arXiv preprint arXiv:1908.06566 (2019)

Table 6: Geometry statistics as presented in Srinivas *et al.* [52]. Due to the high variation of the statistic scale across images standard deviations are very high, especially for the natural model. Hence, we also show the distribution of the curvature in Figure 9. Robust training (both gradient norm regularization and adversarial training) leads to an improvement in normalized curvature of over three orders of magnitude w.r.t. to natural training. Surprisingly, amongst the two robust models adversarial training has the highest curvature, though the numbers are quite close. Statistics calculated over 3200 ImageNet validation images.

	Accuracy		Geometry		
Training	Clean	AutoAttack	$\mathbb{E}_{\mathbf{x}} \lVert \nabla \mathcal{L}(\mathbf{x}) \rVert_2$	$\mathbb{E}_{\mathbf{x}} \left\lVert \nabla^2 \mathcal{L}(\mathbf{x}) \right\rVert_2$	$\mathbb{E}_{\mathbf{x}} C_{\mathcal{L}}(\mathbf{x})$
Natural training	$\mathbf{84.19}$	00.00	$28.7 \pm (93.8)$	$10^6 \pm (10^7)$	$10^4 \pm (10^5)$
Gradient norm regularization	77.78	51.58	$0.31 \pm (0.67)$	0.79 ± 3.25	$2.15 \pm (1.72)$
Adversarial training (PGD-3)	77.20	$\mathbf{56.16}$	$0.31 \pm (0.44)$	0.75 ± 1.71	$2.17 \pm (1.83)$

A Second-Order Analysis

Our main paper focuses on analyses of first-order statistics involving input-gradients. In this section, we provide additional analyses focusing on second-order behaviors, following previous work [43, 52, 40, 55, 42, 50]. In particular:

- Appendix A.1 measures the geometry statistics introduced by Srinivas *et al.* [52].
- Appendix A.2 measures the local linearity error of Rocamora *et al.* [43].
- Appendix A.3 measures loss $\mathcal{L}$, L_1 gradient norm $|\nabla \mathcal{L}|_1$, and normalized curvature $\frac{|\nabla^2 \mathcal{L}|_2}{|\nabla \mathcal{L}|_2}$ [52] as we move in the attack direction.

In contrast to previous works on smaller datasets, our analyses focus on large-scale models trained on ImageNet.

A.1 Geometry Statistics

Table 6 plots average and standard deviation of the geometry statistics introduced by Srinivas *et al.* [52]: the loss-input gradient L_2 norm $|\nabla \mathcal{L}|_2$, the loss-input hessian spectral norm $|\nabla^2 \mathcal{L}|_2$, and the normalized curvature $C_{\mathcal{L}} := \frac{|\nabla^2 \mathcal{L}|_2}{|\nabla \mathcal{L}|_2}$.

Standard deviations are very high due to a high variability in scale across examples, which we think is a feature unique to ImageNet as this was not observed by Srinivas *et al.*, though it could also be due to the much larger transformer model. Hence, we plot the distribution of curvature across examples in log scale in Figure 9.

As we can see, robust training leads to a decrease in normalized curvature of over three orders of magnitude w.r.t. natural training; hessian spectral norms drop by more than six, while gradient L_2 norms by around two. Between the two robust models, all geometry statistic averages are surprisingly very similar, with the normalized curvature of the adversarially trained model actually being slightly higher. However, we also observe that the standard deviation of gradient and hessian norm are significantly higher for gradient norm regularization, despite normalized curvature standard deviation being lower; this means that gradient and hessian norms vary on the same examples, such that normalized curvature remains small. Through visual examination of Figure 9.b, we see that the curvature distribution is also very similar across the two robust models. A possible explanation is that the gradient norm regularized model overfits to the clean examples, and thus displays similar numbers to the adversarially trained model despite its lower performance.

A.2 Local Linearity Error

As defined by Rocamora *et al.* in [43], minimizing the local linearity error optimizes the linearity of the model w.r.t. uniformly distributed perturbations. We rewrite their objective in Equation (5)

$$\mathbf{E}_{\alpha,\eta_1,\eta_2}\left[|\alpha\mathcal{L}(x+\eta_1)+(1-\alpha)\mathcal{L}(x+\eta_2)-\mathcal{L}(x+\alpha\eta_1+(1-\alpha)\eta_2)|^2\right] \tag{5}$$

$$\alpha \sim U(0,1) \quad \eta_1,\eta_2 \sim U(-\epsilon,\epsilon) \tag{6}$$

Table 7: Average and standard deviation of the local linearity error of Rocamora *et al*. [43]. Due to the high variation of the scale across images, standard deviations are larger than the mean. Hence, we also report mean and standard deviation of the base 10 logarithm of the error. We add 10^{-17} before computing log statistics for numerical stability. Robust training (both gradient norm regularization and adversarial training) leads to an improvement in local linearity error of two orders of magnitude w.r.t. to natural training. Within the robust models, adversarial training has significantly lower local linearity error, reflecting its superior robustness and second-order stability. Statistics calculated for 10000 ImageNet validation images. Architecture is Swin Transformer.

	Accuracy		Local linearity error	
Training	Clean	AutoAttack	linerr	$\log_{10}$(linerr)
Natural Training	$\mathbf{84.19}$	00.00	$3.55 \times 10^{-3} \pm (2.69 \times 10^{-2})$	$-4.57 \pm (1.63)$
Gradient norm regularization	77.78	51.58	$1.86 \times 10^{-5} \pm (1.65 \times 10^{-4})$	$-6.60 \pm (1.43)$
Adversarial training (PGD-3)	77.20	$\mathbf{56.16}$	$\mathbf{8.2 \times 10^{-6} \pm (5.08 \times 10^{-5})}$	$\mathbf{-6.84 \pm (1.48)}$

where $\epsilon = 4./255$.

Table 7 reports mean and standard deviation of the local linearity error across 10000 ImageNet validation images. As we can see, the two robust models have significantly smaller local linearity errors by about two orders of magnitude compared to the naturally trained model. Within the two robust models, adversarial training has lower average local linearity error, consistent with its superior robustness.

A.3 Loss, L_1 Gradient Norm, and Normalized Curvature Along Attack Direction

In this section, we measure how statistics change along the attack direction by linearly interpolating between a clean and attacked input. While we use the relatively weak PGD-5 attack to reduce computational expense, as we can see in the legend of Figure 11 that it is enough to clearly separate the three models in terms of robustness. The interpolated attack is calculated as follows

$$x(\epsilon) = x + \frac{\epsilon}{4}(\text{PGD-5}_{\epsilon=4}(x) - x) \tag{7}$$

Figure 11.a1 plots average classification loss as function of interpolation ϵ for the three models. As we can see, the loss of the naturally trained model quickly grows to worse-than-random (6.9). Figure 11.a2 zooms into the two robust models, where we observe two key things: (1) adversarial training has higher loss very close to the origin but lower loss away, and (2) despite their robustness loss still significantly increases as we move in the attack direction.

Figure 11.b1 and Figure 11.b2 plots L_1 loss-input gradient norm as a function of interpolation ϵ in the same manner, with similar conclusions. Despite having very similar gradient norms at the origin, gradient norm regularization quickly becomes worse than adversarial training away from the origin. Moreover, both models display a high increase in gradient norm along the attack direction w.r.t. the value at the origin.

Figure 11.c1 and Figure 11.c2 plot normalized curvature. In this particular sample of 1000 images (and using less iterations when calculating hessian spectral norm), normalized curvature at the origin is slightly higher for gradient norm regularization, though they are still quite similar. However, as we step away from the origin in the attack direction, curvature for gradient norm regularization spikes relative to adversarial training. This is consistent with the relative increase in gradient norm w.r.t. the adversarially trained model seen in Figure 11.b2.

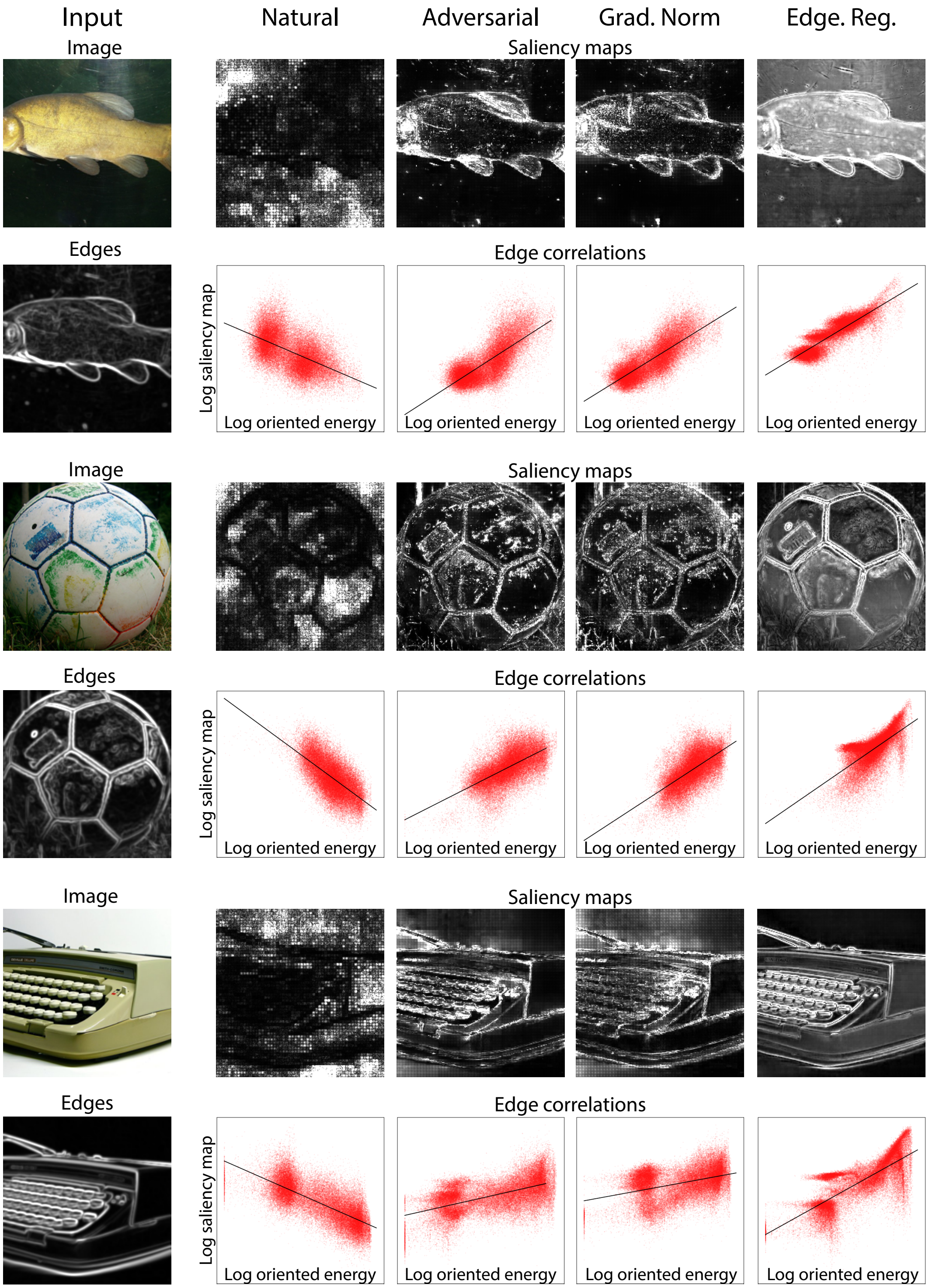

Figure 8: Scatter plots of log gradient magnitude vs log oriented energy for a non-robust and robust Swin Transformer. Oriented energy is calculated as $\text{edge}(x) = |g_u * x|^2 + |g_v * x|^2$, where $*$ denotes the convolution operation, and g_u, g_v denote horizontal and vertical Sobel derivative filters respectively. Oriented energy is clamped at 1e-3 to eliminate extremely low outliers. Saliency maps are clipped to percentile 0.95 for visualization purposes only.

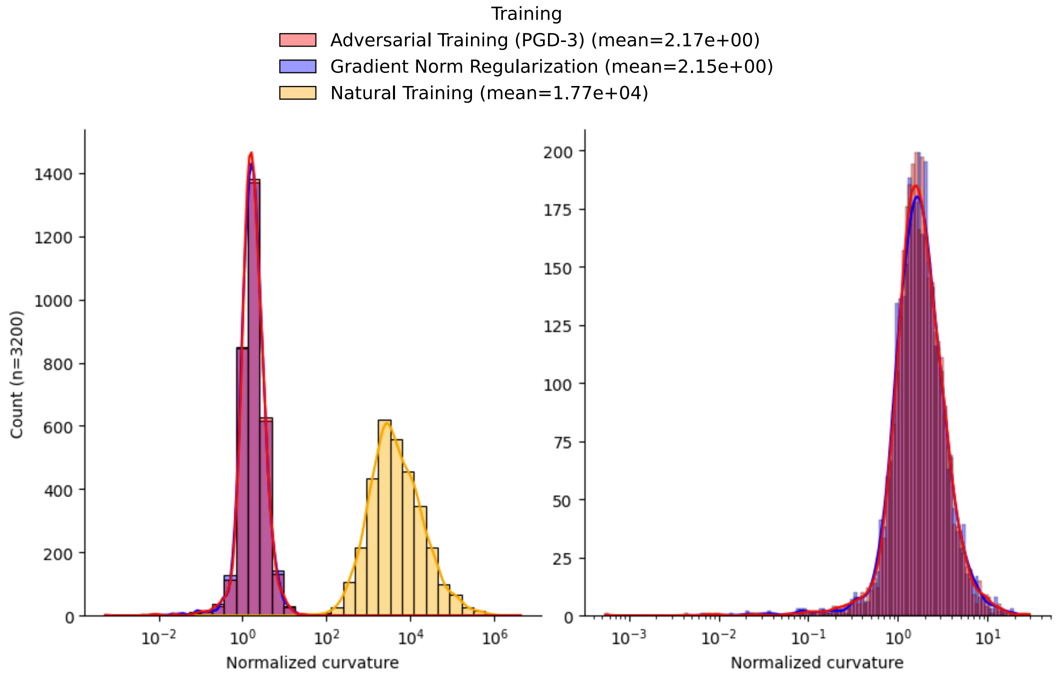

Figure 9: Distribution of normalized curvature [52] over 3200 ImageNet validation images, (a) plots all three models and (b) plots just the robust models. Due to the high variation in scale over images, the graph is plotted with the x-axis in log scale. In (a) we see robust training (both gradient norm regularization and adversarial training) leads to an improvement in normalized curvature of over three orders of magnitude w.r.t. to natural training. Surprisingly, amongst the robust models adversarial training has the slightly higher average curvature, though the distributions are so close they cannot be distinguished in the graph. Architecture is Swin Transformer.

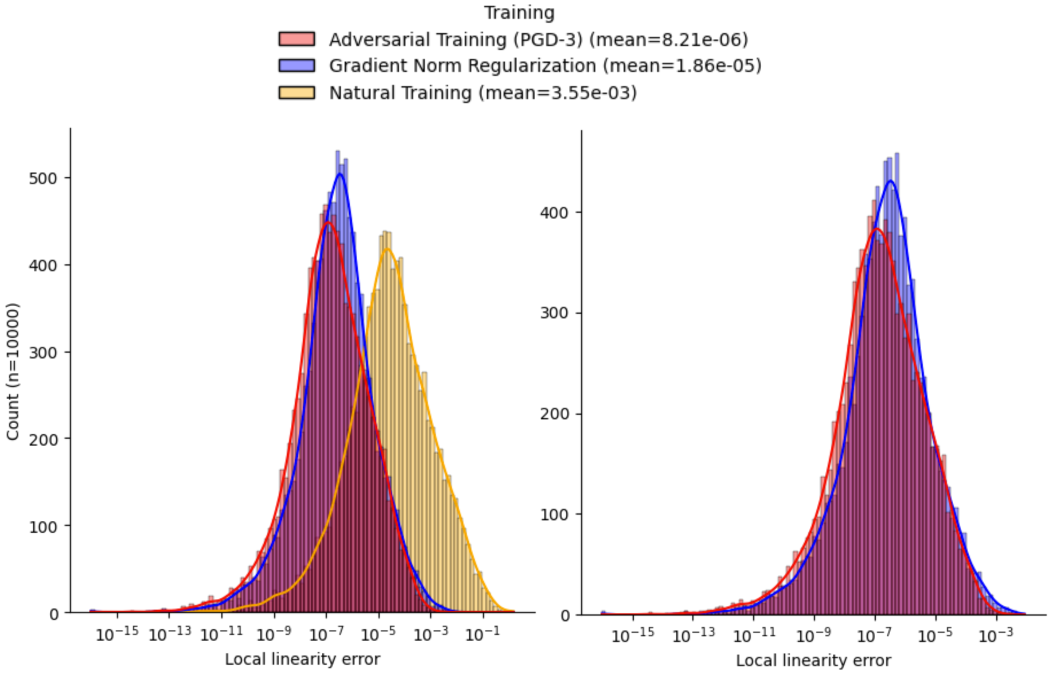

Figure 10: Distribution of local linearity error [43] over 10000 ImageNet validation images, (a) plots all three models and (b) plots just the robust models. Due to the high variation in scale over images, the graph is plotted with the x-axis in log scale. In (a) we see robust training (both gradient norm regularization and adversarial training) leads to a visible leftward shift of the error distribution by about two orders of magnitude w.r.t. natural training. Within the robust models, adversarial training has the lowest local linearity error. The error distribution of adversarial training is shifted slightly to the left and has a much lower peak, meaning that it has a greater number of images with very low local linearity error. Architecture is Swin Transformer.

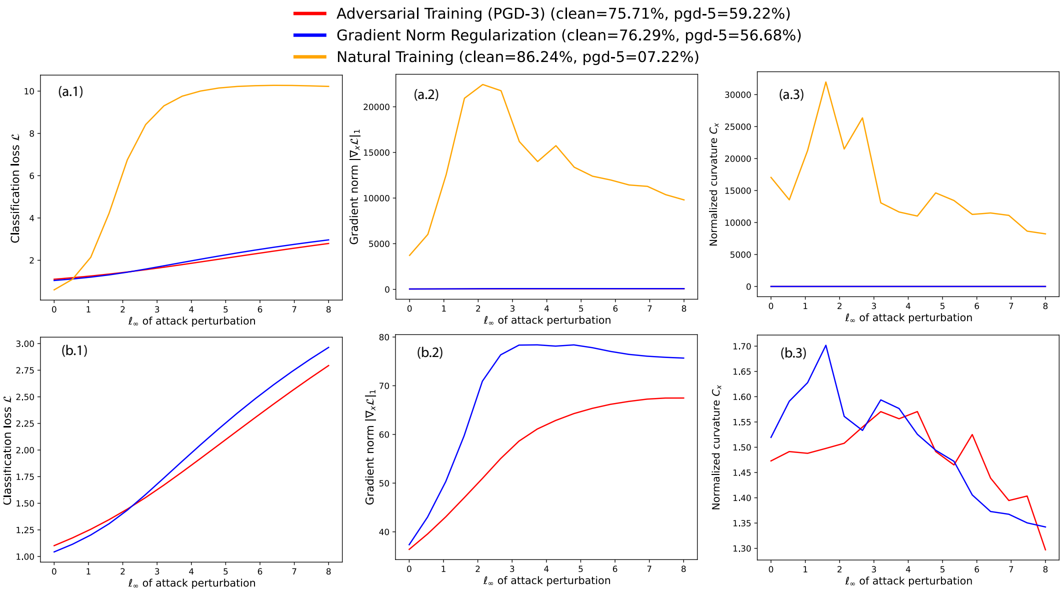

Figure 11: Average loss (left), L_1 gradient norm (middle), and normalized curvature (right) [52] along the PGD-5 ($\epsilon = 4$) attack direction. The top row shows all three models, where we see the extreme brittleness of natural training. The bottom row zooms into the two robust models. Despite their robustness, loss and gradient norm significantly increase along the attack direction for both models. Comparing the two, L_1 gradient norm and curvature at the origin are similar, but gradient norm regularization has significantly higher curvature slightly away from the origin, resulting in higher losses and gradient norms along the attack direction. Statistics calculated on 1000 ImageNet validation examples. Power iteration for the normalized curvature done with 2 iterations and initialized from gradients in order to reduce computational cost. Architecture is Swin Transformer.

B Comparison with FGSM Training

Figure 12 plots PGD10 accuracy vs epoch for Adversarial, GradNorm, and FGSM training with random inititialization [58] for Swin Transformer over 100 epochs. Consistent with Sec. 5.4 of [58], we also find FGSM training to catastrophically overfit in longer training, and fail to reach a comparable performance. In other words, FGSM-trained models become robust to FSGM attacks but vulnerable to stronger ones (e.g., at epoch 57: FGSM acc. 52%, PGD100 acc. 9.78%). In contrast, GradNorm training does not suffer from such overfitting (Figure 12). Just before FGSM training's collapse (epoch 43), FGSM is also the worst among the three (0.8% less robust than GradNorm on PGD10).

C Sanity Checks For Empirical Evaluation of Robustness

In addition to using the community standard AutoAttack [8] for our main robustness evaluations, we performed the suggested checks for rigorous evaluations presented in [4], and some others, which we present here.

C.1 Threat Model

As stated in Section 3, the attacker has access to the model end-to-end, and operates under an L_∞ constraint of $\frac{4}{255}$, the standard used by the RobustBench public benchmark [6] and prior works [33].

C.2 Adaptive Attacks

AutoAttack [8] is the currently known strongest L_∞ attack, and is the standard in adversarial robustness. Gradient norm regularization minimizes the norm of the cross entropy loss gradient (CE), and AutoAttack uses targeted attacks (FAB-T, APGD-T), boundary attacks (FAB-T), and black-box/gradient-free attacks (Square).

Moreover, the "plus" version of AutoAttack adds attacks with alternative losses (APGD-DLR) and the un-targeted boundary attack FAB, and only increased attack success by 0.08%. The latter took over 5x the computational cost to evaluate than the standard AutoAttack, so we only evaluated it for the main GradNorm model of Table 2.

C.3 Release Models and Source Code

These can be found on our github.

C.4 Clean Model Accuracy

Found in all tables and figures such as Table 2 and Figures 2 and 5.

C.5 Accuracy versus Number of Steps and Perturbation Strength

See Figure 2 for accuracy vs perturbation strength, where we confirm that accuracy smoothly reaches 0% as perturbation strength is increased. The following Figure 13 shows smooth convergence of accuracy and loss as the number of PGD($L_\infty, \epsilon = \frac{4}{255}$, step size $= \frac{1}{255}$) steps is increased.

C.6 Transfer Attack from Adversarially Trained Model

We computed a transfer attack using PGD100($L_\infty, \frac{4}{255}$) as the attack algorithm, the adversarially trained model of Table 2 as the source model, and the Grad. Norm model of Table 2 as the target model. The accuracy obtained was 63.59%, significantly higher than the 54% obtained against the same PGD100 attack but using the Grad.Norm model as a source (*i.e.* without transfer). 54% is itself higher than the accuracy achieved by the model against AutoAttack (51.58%), also as expected.

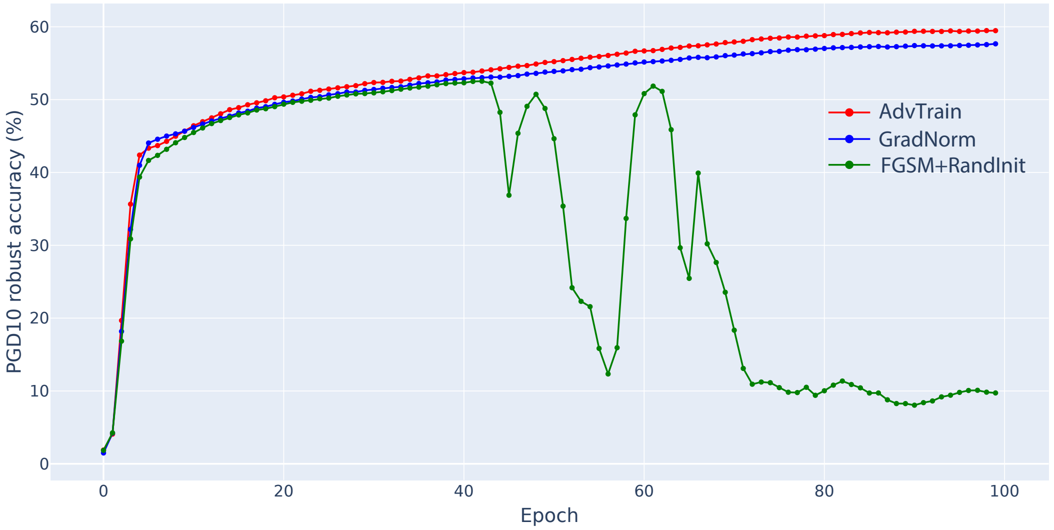

Figure 12: PGD10 ($\epsilon = \frac{4}{255}$) accuracy comparison with FGSM training [58] for Swin Transformer over 100 epochs. FSGM training catastrophically overfits in longer training, as also noted in [58]. Just before FGSM training's collapse (epoch 43), FGSM is also the worst among the three (0.8% less robust than GradNorm on PGD10).

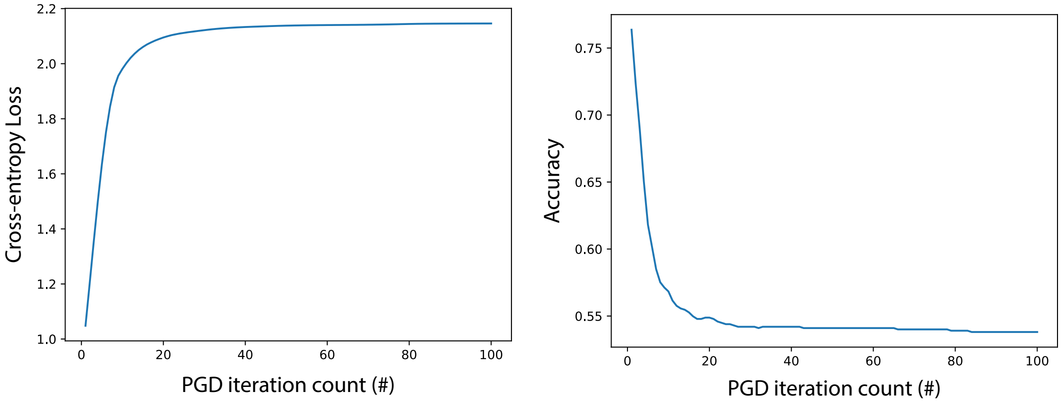

Figure 13: Cross-entropy loss and accuracy as function of PGD iterations ($L_\infty, \epsilon = \frac{4}{255}$, step size $\frac{1}{255}$).

C.7 Convergence of Attacks

Doubling the iterations of all AutoAttack attacks from 100 to 200 (with the exception of Square, which already uses 5000 queries, for computational reasons), only increased the success rate by 0.02%, showing convergence of the attack suite.

Additionally, the full AutoAttack, which adds attacks with alternative losses (APGD-DLR) and an untargeted boundary attack (FAB), only increased attack success by 0.08%.

C.8 L2 Direction with L_∞ Constraint

We also performed a sanity check whereby we used the L_2 descent direction for PGD100, but keeping the L_∞ constraint, by forgoing the sign operation and scaling appropriately. The accuracy obtained by the model against this attack was 65%, higher than that obtained against PGD100 using the L_∞ direction (54%) as expected.

Table 8: Results with no noise, gaussian noise ($\sigma = \frac{4}{255}$, 10 epoch finetuned from noiseless GradNorm), and adversarial noise (PGD3, $\epsilon = \frac{4}{255}$, 1 epoch finetune from AdvTrain).

Method		**Accuracy**	
Training	Noise	Standard	AutoAttack-L_∞
Grad. Norm ($\lambda_{\text{CE}} = 0.8, \lambda_{\text{GN}} = 1.2$)	None	77.76	51.66
	Gaussian	**77.84**	52.04
	Adversarial	76.62	**56.49**
Adv. Train. (PGD-3, $\epsilon = 4$)	-	77.20	56.12

Table 9: Clean and robust accuracies under PGD-50-5 (5 random restarts) and AutoAttack with $\epsilon = \frac{8}{255}$ on Cifar-10 with PRN-18 [22]. Natural training for 50 epochs, Adversarial and GradNorm training for 300.

	Clean	**Robust-($L_\infty, \epsilon = \frac{8}{255}$)**	
Method	-	PGD-50-5	AA
Natural Training	89.38	00.05	00.00
Grad. Norm ($\lambda_{\text{CE}} = 0.8, \lambda_{\text{GN}} = 1.2$)	77.74	46.66	39.60
Adv. Train. (PGD-3, $\epsilon = 4$)	69.81	48.85	42.05

D Gradient Norm Regularization on Non-Natural Inputs

While the focus of our work is on robustness using naturaly inputs only, we also investigate the effect of using non-natural inputs for completeness. Table 8 shows regularizing adversarial input slightly improves over AdvTrain; and regularizing random noise input slightly improves over noiseless GradNorm.

E Edge Regularization Alternate Approaches and Training Details

We first tried to optimize cosine similarity between the loss-input gradient ($\nabla_x \mathcal{L}$)) and oriented energy directly. However, optimization did not converge: even past 27 epochs, the cosine similarity loss was still at 0.00. As we can see in Figure 14.G, gradients become neither image-like nor smooth, and robustness is not achieved.

Next, we attempted to optimize cosine similarity between the saliency map (per-pixel maximum absolute value of the target class input-gradients) [46, 49] and oriented energy. While the optimization converged in this case, a robust model was not obtained. As we can directly visualize in Figure 14.E, this is because the usage of the absolute value allowed for a shortcut solution where the gradient sign-flops at high frequency. Robustness is also not achieved.

Instead, we obtained significant results by directly aligning the class gradients ($\nabla_x f_\theta(x)_{y_t}$, *i.e.*, the gradient of target-class logit w.r.t. input), without the absolute value present in saliency map calculation. The work of [26] also regularized this quantity in a different fashion, but did not obtain strong results on large benchmarks. With this form of edge regularization, gradients become both image-like and smooth, and a non-trivial robustness of 60% of SOTA Adversarial Training on AutoAttack is obtained, as can be seen in Figure 14.D. Convergence takes slightly longer than gradient norm regularization, requiring 128 epochs rather than 100.

F Cifar-10 Experiments

We obtained similar results on CIFAR-10+PreActResnet-18 with $\epsilon = \frac{8}{255}$. As seen in Table 9, over 300 epochs AdvTrain (PGD-3) obtained clean acc. 70%, PGD-50-5 acc. 49%, and AutoAttack acc. 42%. GradNorm got clean acc. 78%, PGD-50-5 acc. 47%, and AutoAttack acc 39.60%. The AutoAttack robust accuracy gap is 2.45% in favor of Adversarial Training, but GradNorm has signficantly higher clean accuracy 78% vs 70%.

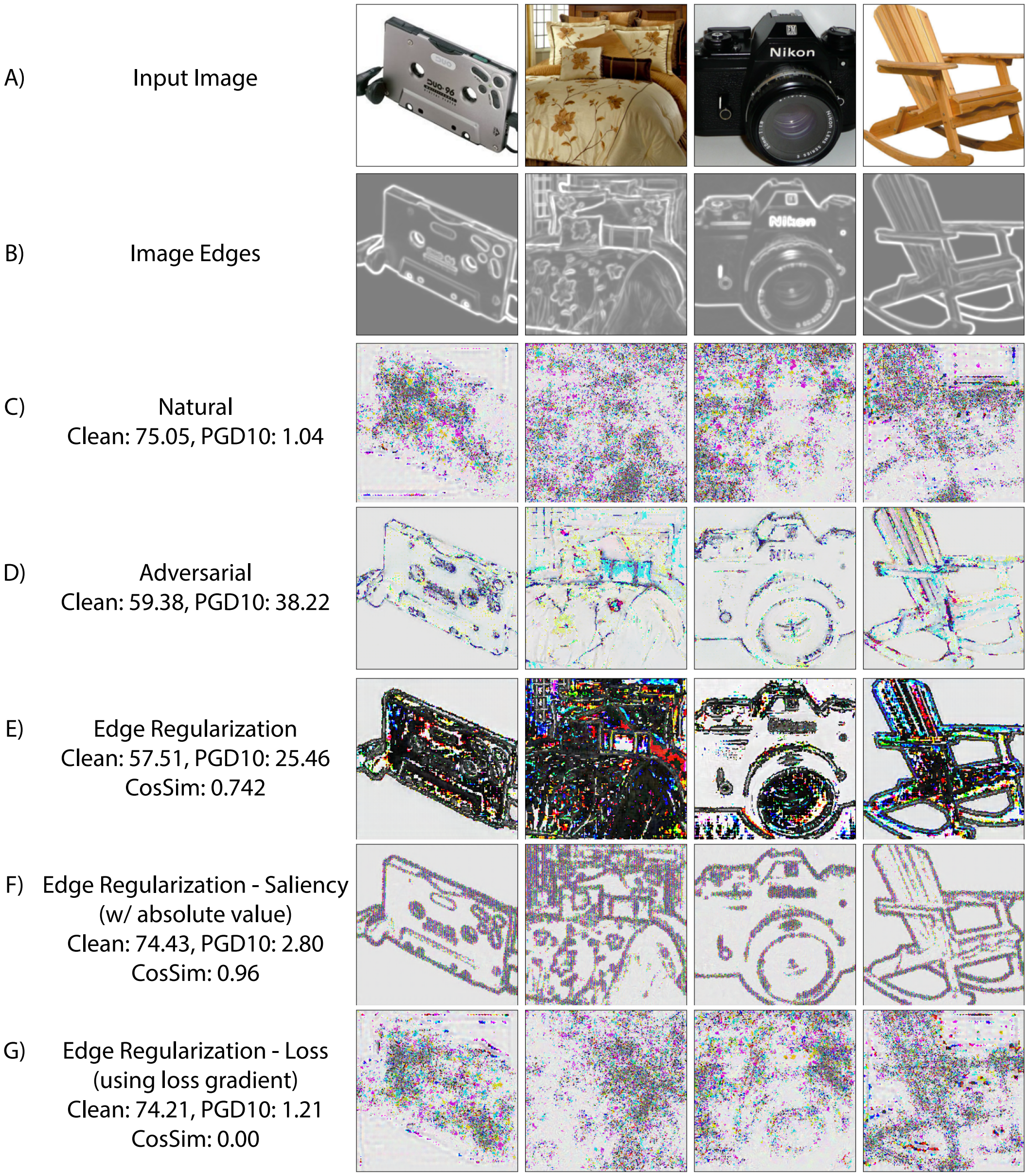
A) Input Image
B) Image Edges
C) Natural
Clean: 75.05, PGD10: 1.04
D) Adversarial
Clean: 59.38, PGD10: 38.22
E) Edge Regularization
Clean: 57.51, PGD10: 25.46
CosSim: 0.742
F) Edge Regularization - Saliency
(w/ absolute value)
Clean: 74.43, PGD10: 2.80
CosSim: 0.96
G) Edge Regularization - Loss
(using loss gradient)
Clean: 74.21, PGD10: 1.21
CosSim: 0.00

Figure 14: Input image, image edges, and target-class logit w.r.t. input gradients of ResNet50+GeLU trained with different regularizations. Adversarial training (row D) has both image-like and smooth gradients. Edge Regularization (row E) makes gradients image-like and smooth, and non-trivial robustness is obtained. Regularizing saliency maps (row F) leads to a shortcut solution that, despite being image-like, is not smooth due to high frequency sign-flops, and robustness is not obtained. Lastly, regularizing loss gradients (row G), does not optimize, and has non-image-like and non-smooth gradients akin to the naturally trained model (row C). PGD10 attack performed with $\epsilon = \frac{4}{255}$. Gradients normalized to [0, 1] then shifted by 0.4 for visualization purposes only.

G Training Details

We used the code of Liu *et al*. [33] as the basis for our experiments. The training configuration used in our experiments is found below in yaml format.

G.1 ResNet50

G.1.1 Adversarial Training (PGD-3)

Same as the transformer recipe of Liu *et al*. [33], but shortened to 50 epochs. Training done in evaluation mode.

```
# optimizer parameters
opt: adamw
opt_eps: 1.0e-8
opt_betas: null
momentum: 0.9
weight_decay: 0.05
clip_grad: null
clip_mode: norm
layer_decay: null

# lr schedule
epochs: 50
sched: cosine
lrb: 1.25e-3
warmup_lr: 1.0e-6
min_lr: 1.0e-5
epoch_repeats: 0
start_epoch: null
decay_epochs: 15
warmup_epochs: 5
cooldown_epochs: 0
patience_epochs: 0
decay_rate: 0.1

# dataset parameters
batch_size: 256

# augmentation
no_aug: False
color_jitter: 0.4
aa: rand-m9-mstd0.5-inc1
aug_repeats: 0
aug_splits: 0
jsd_loss: False
# random erase
reprob: 0.25
remode: pixel
recount: 1
resplit: False
mixup: 0.8
cutmix: 1.0
cutmix_minmax: null
mixup_prob: 1.0
mixup_switch_prob: 0.5
mixup_mode: batch
mixup_off_epoch: 0
smoothing: 0.1
train_interpolation: bicubic
# drop connection
drop: 0.0
drop_path: 0.0
drop_block: null
```

```
# ema
model_ema: True
model_ema_force_cpu: False
model_ema_decay: 0.9998

# adversarial training
attack_eps: 0.01568627450980392 # 4./255.
attack_it: 3
attack_step: 0.01045751633986928 # 8./255./3.
```

Additionally, the attack step is warmed up linearly over 5 epochs as per Liu *et al*. [33].

G.1.2 Gradient Norm Regularization

Same as above, but changing adversarial training for gradient norm regularization with the following weights.

```
# gradient norm regularization
ce_weight: 0.5
gradnorm_weight: 0.5
```

Additionally, `gradnorm_weight` is warmed up linearly over 5 epochs.

G.2 Swin Transformers

G.2.1 Adversarial Training (PGD-3)

Exactly the recipe of Liu *et al*. [33].

```
# optimizer parameters
opt: adamw
opt_eps: 1.0e-8
opt_betas: null
momentum: 0.9
weight_decay: 0.05
clip_grad: null
clip_mode: norm
layer_decay: null

# lr schedule
epochs: 100
sched: cosine
lrb: 1.25e-3
warmup_lr: 1.0e-6
min_lr: 1.0e-5
epoch_repeats: 0
start_epoch: null
decay_epochs: 30
warmup_epochs: 5
cooldown_epochs: 0
patience_epochs: 0
decay_rate: 0.1

# dataset parameters
batch_size: 512

# augmentation
no_aug: False
color_jitter: 0.4
aa: rand-m9-mstd0.5-inc1
```

```
aug_repeats: 0
aug_splits: 0
jsd_loss: False
# random erase
reprob: 0.25
remode: pixel
recount: 1
resplit: False
mixup: 0.8
cutmix: 1.0
cutmix_minmax: null
mixup_prob: 1.0
mixup_switch_prob: 0.5
mixup_mode: batch
mixup_off_epoch: 0
smoothing: 0.1
train_interpolation: bicubic
# drop connection
drop: 0.0
drop_path: 0.0
drop_block: null

# ema
model_ema: True
model_ema_force_cpu: False
model_ema_decay: 0.9998
```

Additionally, the attack step is warmed up linearly over 5 epochs as per Liu *et al*. [33].

G.2.2 Gradient Norm Regularization

Same as above, except for changing adversarial training for gradient norm regularization (see Equation (2)) with weights ($\lambda_{\text{CE}} = 0.8, \lambda_{\text{GN}} = 1.2$), a halving of the learning rate to 0.625×10^{-3}, an increase in the batch size from 512 to 532, and an increase in the warm-up learning rate from 1×10^{-6} to 1×10^{-5}.

```
# lr schedule
lrb: 0.625e-3
warmup_lr: 1.0e-5

# dataset parameters
batch_size: 532

# gradient norm regularization
ce_weight: 0.8
gradnorm_weight: 1.2
```

Additionally, `gradnorm_weight` is warmed up linearly over 5 epochs.