Relating Adversarially Robust Generalization to Flat Minima

David Stutz[1] Matthias Hein[2] Bernt Schiele[1]
[1]Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken
[2]University of Tübingen, Tübingen
{david.stutz,schiele}@mpi-inf.mpg.de, matthias.hein@uni-tuebingen.de

Abstract

Adversarial training (AT) has become the de-facto standard to obtain models robust against adversarial examples. However, AT exhibits severe robust overfitting: cross-entropy loss on adversarial examples, so-called robust loss, decreases continuously on training examples, while eventually increasing on test examples. In practice, this leads to poor robust generalization, i.e., adversarial robustness does not generalize well to new examples. In this paper, we study the relationship between robust generalization and flatness of the robust loss landscape in weight space, i.e., whether robust loss changes significantly when perturbing weights. To this end, we propose average- and worst-case metrics to measure flatness in the robust loss landscape and show a ***correlation between good robust generalization and flatness****. For example, throughout training, flatness reduces significantly during overfitting such that early stopping effectively finds flatter minima in the robust loss landscape. Similarly, AT variants achieving higher adversarial robustness also correspond to flatter minima. This holds for many popular choices, e.g., AT-AWP, TRADES, MART, AT with self-supervision or additional unlabeled examples, as well as simple regularization techniques, e.g., AutoAugment, weight decay or label noise. For fair comparison across these approaches, our flatness measures are specifically designed to be scale-invariant and we conduct extensive experiments to validate our findings.*

1. Introduction

In order to obtain robustness against adversarial examples [56], *adversarial training (AT)* [37] augments training with adversarial examples that are generated on-the-fly. While many different variants have been proposed, AT is known to require more training data [29, 49], generally leading to generalization problems [17]. In fact, *robust overfitting* [46] has been identified as the main problem in AT: adversarial robustness on test examples eventually starts to decrease, while robustness on training examples

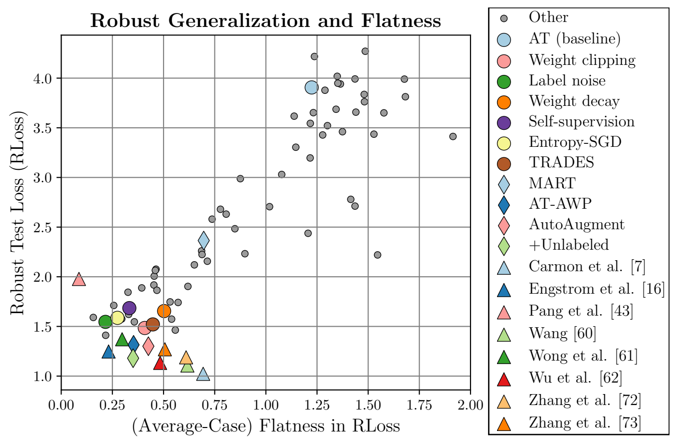

Figure 1: **Robust Generalization and Flatness:** Robust loss (RLoss, lower is more robust, y-axis), i.e., cross-entropy loss on PGD adversarial examples [37], against our *average-case flatness* measure of RLoss in weight space (lower is "flatter", x-axis). Popular AT variants improving adversarial robustness on CIFAR10, e.g., TRADES [72], AT-AWP [62], MART [60] or AT with self-supervision [22]/unlabeled examples [7], also correspond to flatter minima. Vice-versa, regularization explicitly improving flatness, e.g., Entropy-SGD [8], weight decay or weight clipping [52], also improve robustness. Across all models, there is a **clear relationship between good robust generalization and flatness in RLoss.** ●,♦ Our models, <u>without</u> early stopping. ▲ RobustBench [10] models *with* early stopping.

continues to increase (cf. Fig. 2). This is typically observed as increasing *robust loss (RLoss)* or *robust test error (RErr)*, i.e., (cross-entropy) loss and test error on adversarial examples. As a result, the *robust generalization gap*, i.e., the difference between test and training robustness, tends to be very large. In [46], early stopping is used as a simple and effective strategy to avoid robust overfitting. However, despite recent work tackling robust overfitting [51, 62, 25], it remains an open and poorly understood problem.

In *"clean"* generalization (i.e., on natural examples), overfitting is well-studied and commonly tied to flatness of the loss landscape in weight space, both visually [34] and

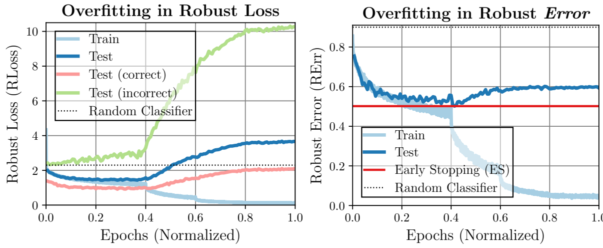

Figure 2: **Robust Overfitting:** Robust (cross-entropy) loss (RLoss) and robust error (RErr) over epochs (normalized by 150 epochs) for AT, using a ResNet-18 on CIFAR10 (cf. Sec. 4), to illustrate *robust* overfitting. **Left:** Training RLoss (light blue) reduces continuously throughout training, while test RLoss (dark blue) eventually increases again. We also highlight that robust overfitting is *not* limited to incorrectly classified examples (green), but also affects correctly classified ones (rose). **Right:** Similar behavior, but less pronounced, can be observed considering RErr. We also show RErr obtained through early stopping (red).

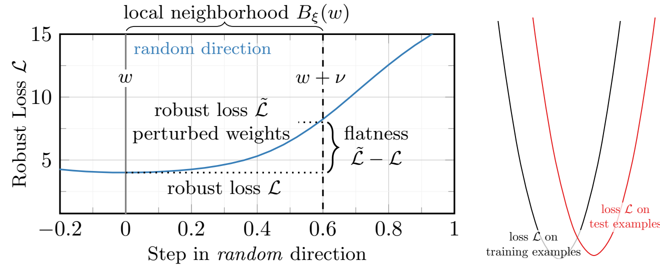

Figure 3: **Measuring Flatness. Left:** Illustration of measuring flatness in a random (i.e., average-case, blue) direction by computing the difference between RLoss $\tilde{\mathcal{L}}$ *after* perturbing weights (i.e., $w + \nu$) and the "reference" RLoss $\mathcal{L}$ given a local neighborhood $B_\xi(w)$ around the found weights w, see Sec. 3.3. In practice, we average across/take the worst of several random/adversarial directions. **Right:** Large changes in RLoss around the "sharp" minimum causes poor generalization from training (black) to test examples (red).

empirically [41, 28, 27]. In general, the optimal weights on test examples do not coincide with the minimum found on training examples. Flatness ensures that the loss does *not* increase significantly in a neighborhood around the found minimum. Therefore, flatness leads to good generalization because the loss on test examples does not increase significantly (i.e., small generalization gap, cf. Fig. 3, right). [34] showed that *visually* flatter minima correspond to better generalization. [41] and [28] formalize this idea by measuring the change in loss within a local neighborhood around the minimum considering random [41] or "adversarial" weight perturbations [28]. These measures are shown to be effective in predicting generalization in a recent large-scale empirical study [27] and explicitly encouraging flatness during training has been shown to be successful in practice [74, 9, 35, 8, 26].

Recently, [62] applied the idea of flat minima to AT: through *adversarial weight perturbations*, AT is regularized to find flatter minima of the *robust* loss landscape. This reduces the impact of robust overfitting and improves robust generalization, but does not *avoid* robust overfitting. As result, early stopping is still necessary. Furthermore, flatness is only assessed *visually* and it remains unclear whether flatness does actually improve in these adversarial weight directions. Similarly, [18] shows that weight averaging [26] can improve robust generalization, indicating that flatness might be beneficial in general. This raises the question whether other "tricks" [42, 18], e.g., different activation functions [51] or label smoothing [55], or approaches such as AT with self-supervision [22]/unlabeled examples [7] are successful *because of* finding flatter minima.

Contributions: In this paper, we study **whether flatness of the robust loss (RLoss) in weight space improves robust generalization**. To this end, we propose both average- and worst-case flatness measures for the *robust* case, thereby addressing challenges such as scale-invariance [14], estimation of RLoss on top or jointly with weight perturbations, and the discrepancy between RLoss and RErr. We show that **robust generalization generally improves alongside flatness** and vice-versa: Fig. 1 plots RLoss (lower is more robust, y-axis) against our average-case flatness in RLoss (lower is flatter, x-axis), showing a clear relationship. In contrast to [62], not providing empirical flatness measures, our results show that this relationship is stronger for average-case flatness. This trend covers a wide range of AT variants on CIFAR10, e.g., AT-AWP [62], TRADES [72], MART [60], AT with self-supervision [22] or additional unlabeled examples [7, 2], as well as various regularization schemes, including AutoAugment [12], label smoothing [55] and noise or weight clipping [52]. Furthermore, we consider hyper-parameters, e.g., learning rate schedule, weight decay, batch size, or different activation functions [15, 39, 21], and methods explicitly improving flatness, e.g., Entropy-SGD [8] or weight averaging [26].

2. Related Work

Adversarial Training (AT): Despite a vast amount of work on adversarial robustness, e.g., see [50, 69, 1, 5, 65], adversarial training (AT) has become the de-facto standard for (empirical) robustness. Originally proposed in different variants in [56, 40, 24], it received considerable attention in [37, 16] and has been extended in various ways: [33, 7, 2] utilize interpolated or unlabeled examples, [57, 38] achieve robustness against multiple threat models, [54, 32, 64] augment AT with a reject option, [67, 36] use Bayesian networks, [58, 19] build ensembles, [4, 13] adapt the threat model for each example, [61, 3, 47] perform AT with single-step attacks, [22] uses self-supervision and [43] additionally regularizes features – to name a few directions. However, AT is slow [71] and suffers from increased sample complexity [49] as well as reduced (clean) accuracy [59, 53, 72, 45]. Furthermore, progress is slowing down. In fact, "standard" AT is shown to perform surprisingly well on recent benchmarks [11, 10] when tuning hyper-parameters properly [42, 18]. In our experiments, we consider several popular variants [62, 60, 72, 7, 22].

Robust Overfitting: Recently, [46] identified *robust* overfitting as a crucial problem in AT and proposed early stopping as an effective mitigation strategy. This motivated work [51, 62] trying to mitigate robust overfitting. While [51] studies the use of different activation functions, [62] proposes AT with *adversarial weight perturbations* (AT-AWP) explicitly aimed at finding flatter minima in order to reduce overfitting. While the results are promising, early stopping is still necessary. Furthermore, flatness is merely assessed visually, leaving open whether AT-AWP *actually* improves flatness in adversarial weight directions. We consider both average- and worst-case flatness, i.e., random and adversarial weight perturbations, to answer this question.

Flat Minima in the loss landscape, w.r.t. changes in the weights, are generally assumed to improve *standard* generalization [23]. [34] shows that residual connections in ResNets [20] or weight decay lead to *visually* flatter minima. [41, 28] formalize this concept of flatness in terms of *average-case* and *worst-case* flatness. [28, 27] show that worst-case flatness correlates well with better generalization, e.g., for small batch sizes, while [41] argues that generalization can be explained using both an average-case flatness measure and an appropriate capacity measure. Similarly, batch normalization is argued to improve generalization by allowing to find flatter minima [48, 6]. These insights have been used to explicitly regularize flatness [74], improve semi-supervised learning [9] and develop novel optimization algorithms such as Entropy-SGD [8], local SGD [35] or weight averaging [26]. [14], in contrast, criticizes some of these flatness measures as not being scale-invariant. We transfer the intuition of flatness to the *robust* loss landscape, showing that flatness is desirable for adversarial robustness, while using scale-invariant measures.

3. Robust Generalization and Flat Minima

We study robust generalization and overfitting in the context of flatness of the *robust* loss landscape in weight space, i.e., w.r.t. changes in the weights. While flat minima have consistently been linked to standard generalization [23, 34, 41, 28], this relationship remains unclear for adversarial robustness. We start by briefly introducing the robust overfitting phenomenon (Sec. 3.1). Then, we discuss problems in judging flatness visually [34] (Sec. 3.2). Thus, we are inspired by [28, 41] and introduce average- and worst-case flatness measures based on the change in robust loss along random or adversarial weight directions in a local neighborhood (Sec. 3.3), cf. Fig. 3. We also discuss the connection of flatness to the Hessian eigenspectrum [66] and the importance of scale-invariance as outlined in [14].

Model	RErr ↓
AT (baseline) ●	62.8
Scaled $\times 0.5$ ●	62.8
Scaled $\times 2$ ●	62.8
MiSH ●	59.8
Batch size 8 ●	58.2
Adam ●	57.5
Label smoothing ●	61.2
MART ●	61
Entropy-SGD ●	58.6
Self-supervision ●	57.1
TRADES ●	56.7
AT-AWP ●	54.3

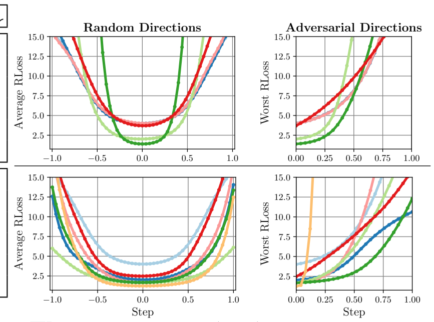

Figure 4: **Visualizing Flatness:** RLoss landscape across 10 random or adversarial directions. **Top:** Our AT baseline (ResNet-18) and scaled variants ($\times 2$ and $\times 0.5$). Training with smaller batch size or Adam [30] improves adversarial robustness (lower RErr vs. AutoAttack [11]) but does *not* result in *visually* flatter minima. **Bottom:** AT-AWP [62] or Entropy-SGD [8] improve robustness *and* visual flatness in random directions. In adversarial directions, however, AT-AWP looks very sharp. Overall, visual inspection does *not* provide a clear, objective picture of flatness.

3.1. Background

Adversarial Training (AT): Let f be a (deep) neural network taking input $x \in [0,1]^D$ and weights $w \in \mathbb{R}^W$ and predicting a label $f(x;w)$. Given a true label y, an adversarial example is a perturbation $\tilde{x} = x + \delta$ such that $f(\tilde{x};w) \neq y$. The perturbation δ is intended to be nearly invisible which is, in practice, enforced using a L_p constraint: $\|\delta\|_p \leq \epsilon$. To obtain robustness against these perturbations, AT injects adversarial examples during training:

$$\min_w \mathbb{E}_{x,y}\left[\max_{\|\delta\|_p \leq \epsilon} \mathcal{L}(f(x+\delta;w),y)\right] \quad (1)$$

where $\mathcal{L}$ denotes the cross-entropy loss. The outer minimization problem can be solved using regular stochastic gradient descent (SGD) on mini-batches. To compute adversarial examples, the inner maximization problem is tackled using projected gradient descent (PGD) [37]. Here, we focus on $p = \infty$ as this constrains the maximum change per feature/pixel, e.g., $\epsilon = {}^{8}/_{255}$ on CIFAR10. For evaluation (at test time), we consider both robust loss (RLoss) $\max_{\|\delta\|_\infty \leq \epsilon} \mathcal{L}(f(x+\delta;w),y)$, approximated using PGD, and robust test error (RErr), which we approximate using AutoAttack [11]. Note that AutoAttack stops when adversarial examples are found and does *not* maximize cross-entropy loss, rendering it unfit to estimate RLoss.

Robust Overfitting: Following [46], Fig. 2 illustrates the problem of *robust* overfitting, plotting RLoss (left) and RErr (right) over epochs, which we normalize by the total

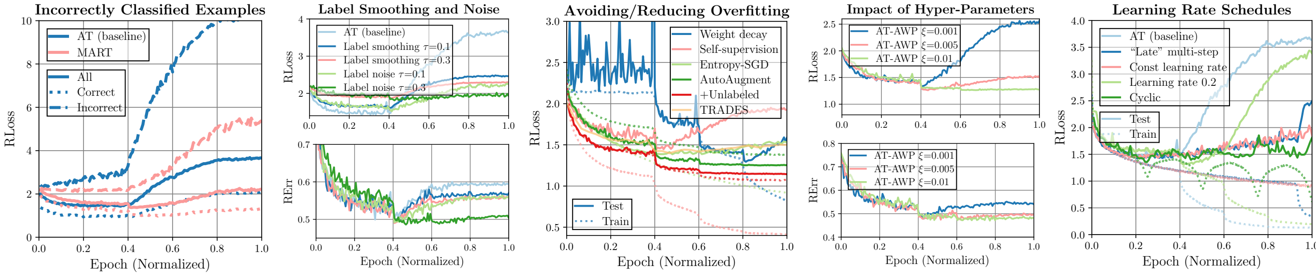

Figure 5: **Understanding Robust Overfitting:** Training curves plotted over (normalized) epochs, see Sec. 3.4 for detailed discussion. **First column:** RLoss, split for correct/incorrect test examples, for AT and MART, which successfully dampens the effect of overfitting using a weighted loss on incorrectly classified examples. **Second column:** Both label smoothing and label noise reduce robust overfitting w.r.t. RLoss. However, the reduction in RLoss does not translate to a similar reduction of RErr. **Third to fifth column:** RLoss (test solid and train dotted) for various approaches improving adversarial robustness and different learning rate schedules. While some approaches avoid robust overfitting altogether (e.g., AT-AWP), others (e.g., weight decay) merely reduce its impact (third column). But the success depends strongly on hyper-parameters (fourth column). Robust overfitting occurs using all tested learning rate schedules (fifth column), confirming [46].

number of epochs for clarity. Shortly after the first learning rate drop (at epoch 60, i.e., 40% of training), test RLoss and RErr start to increase significantly, while robustness on training examples continues to improve. Robust overfitting was shown to be independent of the learning rate schedule [46] and, as we show (Sec. 4.1), occurs across various different activation functions as well as many popular AT variants. In contrast to [46], mostly focusing on RErr, Fig. 2 shows that RLoss overfits more severely, indicating a "disconnectedness" between RLoss and RErr that we consider in detail later. For now, RLoss and RErr do clearly not move "in parallel" and RLoss, reaching values around 4, is higher than for a random classifier (which is possible considering *adversarial* examples). This is primarily due to an extremely high RLoss on incorrectly classified test examples (which are "trivial" adversarial examples). We emphasize, however, that robust overfitting also occurs on correctly classified test examples.

3.2. Intuition and Visualizing Flatness

For judging robust flatness, we consider how RLoss changes w.r.t. random or adversarial perturbations in the weights w. Generally, we expect flatter minima to generalize better as the loss does not change significantly within a small neighborhood around the minimum, i.e., the found weights. Then, even if the loss landscape on test examples does not coincide with the loss landscape on training examples, loss remains small, ensuring good generalization. The contrary case, i.e., that sharp minima generalize poorly is illustrated in Fig. 3 (right). Before considering to *measure* flatness, we discuss the easiest way to "judge" flatness: visual inspection of the RLoss landscape along random or adversarial directions in weight space.

In [34], loss landscape is visualized along *normalized* random directions. Normalization is important to handle different scales, i.e., weight distributions, and allow comparison across models. We follow [62] and perform *per-layer* normalization: Letting $\nu \in \mathbb{R}^W$ be a direction in weight space, it is normalized as

$$\hat{\nu}^{(l)} = \frac{\nu^{(l)}}{\|\nu^{(l)}\|_2}\|w^{(l)}\|_2 \quad \text{for layer } l. \tag{2}$$

In contrast to [34], we also consider biases and treat them as individual layer, but we exclude batch normalization parameters. Then, the loss landscape is visualized in discrete steps along this direction, i.e., $w + s\hat{\nu}$ for $s \in [-1, 1]$. Adversarial examples are computed "on-the-fly", i.e., for each $w + s\hat{\nu}$ individually, to avoid underestimating RLoss as in [68, 44]. The result is indeed scale-invariant: Fig. 4 (top) shows that the loss landscapes for scaled versions (factors 0.5 or 2, see supplementary material) of our AT baseline coincide with the original landscape. However, Fig. 4 also illustrates that judging flatness visually is difficult: Considering random weight directions, AT with Adam [30] or small batch size improves adversarial robustness, but the found minima look less flat (top). For other approaches, e.g., TRADES [72] or AT-AWP [62], results look indeed flatter while also improving robustness (bottom). In adversarial directions, in contrast, AT-AWP looks particularly sharp. Furthermore, not only flatness but also the vertical "height" of the loss landscape matters and it is impossible to tell "how much" flatness is necessary.

3.3. Average- and Worst-Case Flatness Measures

In order to objectively measure and compare flatness, we draw inspiration from [41, 28] and propose average- and worst-case flatness measures adapted to the robust loss. We emphasize that measuring flatness in RLoss is non-trivial and flatness in (clean) Loss *cannot* be expected to correlate with robustness (see supplementary material). For example, we need to ensure scale-invariance [14] and estimate RLoss *on top* of random or adversarial weight perturbations:

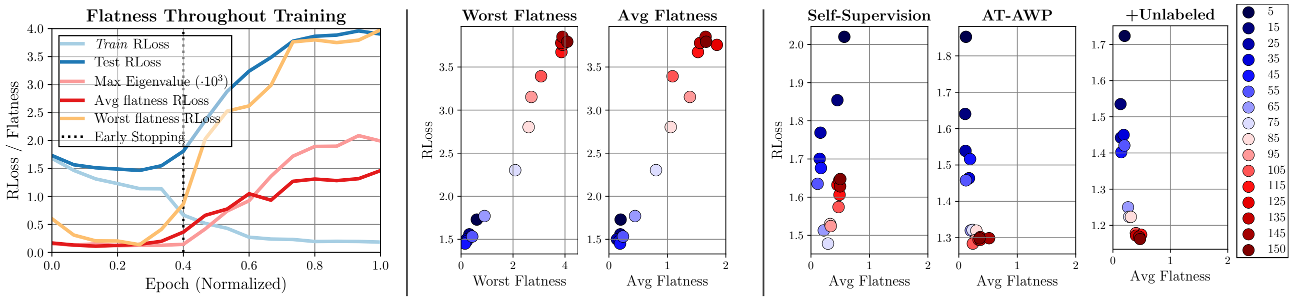

Figure 6: **Flatness Throughout Training. Left:** Flatness in RLoss throughout training, showing that flatness reduces when the model overfits (i.e., test RLoss increases, while train RLoss decreases). **Middle:** Test RLoss (y-axis) plotted against flatness in RLoss (x-axis) during training (early epochs in dark blue, late epochs in dark red), showing a clear correlation, for both average- and worst-case flatness. **Right:** AT with self-supervision reduces the impact of robust overfitting (RLoss increases less) and simultaneously favors flatter minima. This behavior is pronounced for AT-AWP, explicitly optimizing flatness, and AT with additional unlabeled examples, generally resulting in the highest adversarial robustness, cf. Tab. 1.

Average-Case / Random Flatness: Considering random weight perturbations $\nu \in B_\xi(w)$ within the ξ-neighborhood of w, average-case flatness is computed as

$$\mathbb{E}_\nu[\max_{\|\delta\|_\infty \le \epsilon} \mathcal{L}(f(x+\delta; w+\nu), y)] - \max_{\|\delta\|_\infty \le \epsilon} \mathcal{L}(f(x+\delta; w), y) \quad (3)$$

averaged over test examples x, y, as illustrated in Fig. 3. We define $B_\xi(w)$ using *relative* L_2-balls per layer (cf. Eq. (2)):

$$B_\xi(w) = \{w + \nu : \|\nu^{(l)}\|_2 \le \xi \|w^{(l)}\|_2 \forall \text{ layers } l\}. \quad (4)$$

This ensures scale-invariance w.r.t. the weights as $B_\xi(w)$ scales with the weights on a *per-layer* basis. Note that the second term in Eq. (3), i.e., the "reference" robust loss, is important to make the measure independent of the absolute loss (i.e., corresponding to the vertical shift in Fig. 4, left). In practice, ξ can be as large as 0.5. We refer to Eq. (3) as **average-case flatness in RLoss**.

Worst-Case / Adversarial Flatness: [62] explicitly optimizes flatness in *adversarial weight* directions and shows that average-case flatness is not sufficient to improve adversarial robustness. As it is unclear whether [62] actually improves worst-case flatness, we define

$$\max_{\nu \in B_\xi(w)} \left[\max_{\|\delta\|_\infty \le \epsilon} \mathcal{L}(f(x+\delta; w+\nu), y)\right] - \max_{\|\delta\|_\infty \le \epsilon} \mathcal{L}(f(x+\delta; w), y) \quad (5)$$

as **worst-case flatness in RLoss**. Here, we use the same definition of $B_\xi(w)$ as above (aligned with [62]), but for smaller values of ξ. Regarding *standard* performance, this worst-case notion of flatness has been shown to be a reliable predictor of generalization [27, 28]. For computing Eq. (5) in practice, we jointly optimize over ν and δ (for each batch individually) using PGD. As illustrated in Fig. 4, RLoss increases quickly along adversarial directions, even for very small values of ξ, e.g., $\xi = 0.005$.

3.4. Discussion

In the context of flatness, there has also been some discussion concerning the meaning of Hessian eigenvalues [34, 66] as well as concerns regarding the scale-invariance of flatness measures [14]. First, regarding the Hessian eigenspectrum, [66] shows that large Hessian eigenvalues indicate poor adversarial robustness. However, Hessian eigenvalues are generally *not* scale-invariant (which is acknowledged in [66]): Our AT baseline has a maximum eigenvalue of 1990 which reduces to 505 when *up-scaling* the model and increases to 7936 when *down-scaling*, without affecting robustness (cf. $\times 0.5$ and $\times 2$ in Fig. 4). We also found that the largest eigenvalue is *not* correlated with adversarial robustness. Second, following a similar train of thought, [14] criticizes the flatness measures of [41, 28] as not being scale-invariant. That is, through clever scaling of weights, without changing predictions, arbitrary flatness values can be "produced". However, the analysis in [14] does not take into account the relative neighborhood as defined in [28], which renders the measure explicitly scale-invariant. This also applies to our definition of $B_\xi(w)$ in Eq. (4) and is shown in Fig. 4 where normalization is performed relative (per-layer) to the weights; empirical validation can be found in the supplementary material.

4. Experiments

We start with a closer look at RLoss in robust overfitting (Sec. 4.1, Fig. 5). Then, we show a strong correlation between good robust generalization and flatness (Sec. 4.2). For example, robust overfitting causes sharper minima (Fig. 6). More importantly, more robust models generally find flatter minima and, vice-versa, methods encouraging flatness improve adversarial robustness (Fig. 7, 8). In fact, flatness improves robust generalization by *both* lowering the robust generalization gap (incl. a reduction in robust overfitting, cf. Fig. 9).

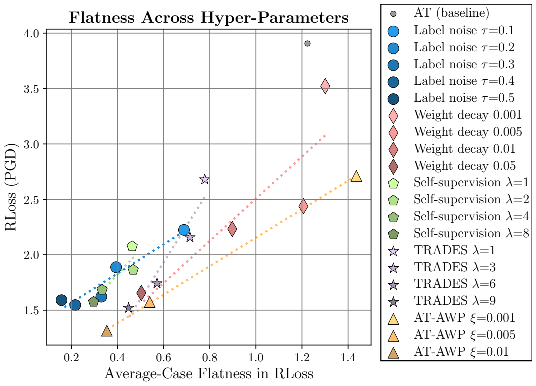

Figure 7: **Flatness Across Hyper-Parameters:** RLoss (y-axis) vs. average-case flatness (x-axis) for selected methods and hyper-parameters (cf. supplementary material). For example, we consider different strengths of weight decay (rose) or sizes ξ of adversarial weight perturbations for AT-AWP (orange). For clarity, we plot (dotted) lines representing the trend per method. Clearly, improved adversarial robustness, i.e., low RLoss, is related to improved flatness.

Setup: On CIFAR10 [31], our *AT baseline* uses ResNet-18 [20] and is trained for 150 epochs, batch size 128, learning rate 0.05, reduced by factor 0.1 at 60, 90 and 120 epochs, using weight decay 0.005 and momentum 0.9 with standard SGD. We use random flips and cropping as data augmentation. During training, we use 7 iterations PGD, with learning rate 0.007, signed gradient and $\epsilon = {}^8/_{255}$ for L_∞ adversarial examples. PGD-7 is also used for early stopping (every 5th epoch) on the last 500 test examples. We do *not* use early stopping by default. For evaluation on the first 1000 test examples, we run PGD with 20 iterations, 10 random restarts to estimate RLoss and AutoAttack [11] to estimate RErr (cf. Sec. 3.1). For *average-case flatness of RLoss*, we take the average of 10 random weight perturbations with ξ=0.5. For *worst-case flatness*, we maximize RLoss jointly over adversarial examples and adversarial weights with ξ=0.00075, taking the worst of 10 restarts.

Methods: Besides our AT baseline, we consider AT-AWP [63], TRADES [72], MART [60], AT with self-supervision [22] or additional unlabeled examples [7, 2], weight averaging [26] and AT with "early-stopped" PGD [73]. We investigate different hyper-parameters and "tricks" recently studied in [42, 18]: learning rate schedules, batch size, weight decay, label smoothing [55] as well as SiLU/Mish/GeLU [15, 39, 21] activation functions. Furthermore, we consider Entropy-SGD [8], label noise, weight clipping [52] and AutoAugment [12]. We emphasize that weight averaging, Entropy-SGD and weight clipping are known to improve flatness of the (clean) loss. *If not stated otherwise, these methods are applied on top or as replacement of our AT baseline.* We report results using the best hyper-parameters per method. Finally, we also

Model	**Robustness ↓**		**Flatness ↓**		**Early Stop.**
(sorted asc. by test RErr) (split at 70%/30% percentiles)	RErr (test)	RErr (train)	Avg (RLoss)	Worst (RLoss)	RErr ↓ (early stop)
+Unlabeled	48.9	43.2 (-5.7)	0.32	1.20	48.9 (-0.0)
Cyclic	53.6	35.4 (-18.2)	0.35	1.50)	53.6 (-0.0)
AutoAugment	54.0	47.9 (-6.1)	0.49	0.69	53.5 (-0.5)
AT-AWP	54.3	43.1 (-11.2)	0.35	2.68	53.6 (-0.7)
Label noise	56.2	30.0 (-26.2)	0.33	0.93)	55.5 (-0.7)
Weight clipping	56.5	39.0 (-17.5)	0.41	4.57	56.5 (-0.0)
TRADES	56.7	15.8 (-40.9)	0.57	2.25	53.4 (-3.3)
Self-supervision	57.1	45.0 (-12.1)	0.33	2.63	56.8 (-0.3)
Weight decay	58.1	32.8 (-25.3)	0.50	3.93	54.8 (-3.3)
Entropy-SGD	58.6	46.1 (-12.5)	0.28	1.80	56.9 (-1.7)
MiSH	59.8	5.3 (-54.5)	1.56	3.54	53.7 (-6.1)
"Late" multi-step	59.8	18.4 (-41.4)	0.80	2.96	57.8 (-2.0)
SiLU	60.0	5.6 (-54.4)	1.71	4.20	53.7 (-6.3)
Weight averaging	60.0	10.0 (-50.0)	1.28	5.98	53.0 (-7.0)
Larger ϵ=${}^9/_{255}$	60.9	11.1 (-49.8)	1.33	5.84	53.8 (-7.1)
MART	61.0	20.8 (-40.2)	0.73	3.17	54.7 (-6.3)
GeLU	61.1	3.2 (-57.9)	1.55	4.12	56.7 (-4.4)
Label smoothing	61.2	8.0 (-53.2)	0.65	2.72	54.0 (-7.2)
AT (baseline)	62.8	10.7 (-52.1)	1.21	6.48	54.6 (-8.2)

Robustness	**Averages (across models)**				
Good (RErr$<$57%$\approx$30% percentile)	54.3	36.3 (-18.0)	0.40	2.00	53.6 (-0.7)
Average (57%$\geq$RErr $<$ 60%)	58.7	29.5 (-29.2)	0.69	2.9	56.0 (-2.7)
Poor (RErr$\geq$60%$\approx$70% percentile)	61.0	9.9 (-51.1)	1.21	4.67	54.4 (-6.6)

Table 1: **Robustness and Flatness, Quantitative Results:** Test and train RErr (first, second column, early stopping in fifth column) as well as average-/worst-case flatness in RLoss (third, fourth column) for selected methods, cf. Fig. 8. We split methods into good, average, and poor robustness using the 30% and 70% percentiles. Most methods improve adversarial robustness alongside both average- and worst-case flatness.

use pre-trained models from RobustBench [10], which were obtained using early stopping.

Our **supplementary material** includes additional details on the experimental setup and the evaluated methods. Furthermore, it contains an ablation regarding our average- and worst-case flatness measure and hyper-parameter ablation for individual methods, including training curves.

4.1. Understanding Robust Overfitting

In contrast to related work [46, 62], we take a closer look at RLoss during robust overfitting because RErr is "blind" to many improvements in RLoss, especially on incorrectly classified examples. Fig. 5 shows training curves for various methods, i.e., RLoss/RErr over (normalized) epochs. For example, explicitly handling incorrectly classified examples during training, using MART, helps but does not *prevent* overfitting: RLoss for MART reduces compared to AT (first column). Unfortunately, this improvement does *not* translate to significantly better RErr, cf. Tab. 1. This discrepancy between RLoss and RErr can be reproduced for other methods, as well: label smoothing and label noise enforce, in expectation, the same target distribution. Thus, both reduce RLoss during overfitting (second column, top, rose and dark green). Label smoothing, however, does not improve RErr as significantly as label noise, i.e., does not *prevent* misclassification. This illustrates an important aspect: against adversarial examples, "merely" improving RLoss does not

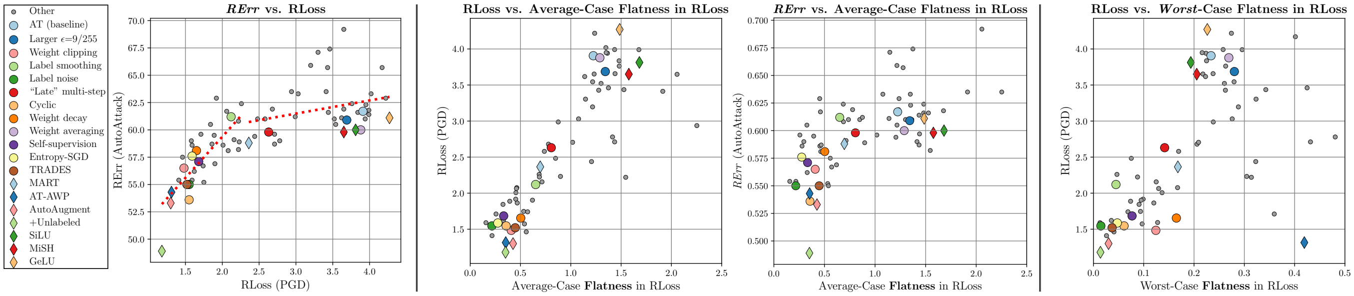

Figure 8: **Robustness and Flatness: Left:** RErr plotted against RLoss, showing that improved RLoss does not directly translate to reduced RErr for large RLoss. In these cases, reducing RLoss mainly means reducing the confidence of adversarial examples, which is necessary to improve adversarial robustness. **Middle:** RLoss or RErr (y-axis) plotted against our *average-case* flatness in RLoss. We highlight selected models, as in Tab. 1. Considering RLoss, we reveal a striking correlation between adversarial robustness and flatness. Popular AT variants improving robustness (e.g., TRADES, MART, etc.) also correspond to flatter minima. Vice versa, methods improving flatness (e.g., Entropy-SGD, weight decay, etc.) improve robustness obtained through AT. Subject to the non-trivial interplay between RErr and RLoss (cf. left), this relationship is also visible using RErr to quantify robustness. **Right:** RLoss (y-axis) plotted against *worst-case* flatness (x-axis) shows a less clear relationship. Still, improved flatness remains a necessity for better robust generalization, see Sec. 4.2 for discussion.

translate to improved RErr if RLoss is high to begin with, i.e., "above" $-\ln(1/K)\approx 2.3$ for $K=10$ classes. However, this is usually the case during robust overfitting. RErr, on the other hand, does not take into account the confidence of wrong predictions, i.e., it is "blind" for these improvements in RLoss. Label noise, in contrast, also improves RErr, which might be due to the additional randomness.

Similar to established methods, many "simple" regularization schemes prove surprisingly effective in tackling robust overfitting. For example, strong weight decay delays robust overfitting and AutoAugment prevents overfitting entirely, cf. Fig. 5 (third column). This indicates that popular AT variants, e.g., TRADES, AT with self-supervision or unlabeled examples, improve adversarial robustness by avoiding robust overfitting through regularization. This is achieved by preventing convergence on training examples (dotted). In regularization, however, hyper-parameters play a key role: even AT-AWP does not prevent robust overfitting if regularization is "too weak" (blue, fourth column). This is particularly prominent in terms of RLoss (top). Finally, learning rate schedules play an important role in how and *when* robust overfitting occurs (fifth column). However, as in [46], all schedules are subject to robust overfitting.

4.2. Robust Generalization and Flatness in RLoss

As robust overfitting is primarily avoided through strong regularization, we hypothesize that this is because strong regularization finds flatter minima in the RLoss landscape. These flat minima help to improve robust generalization.

Flatness in RLoss "Explains" Overfitting: Using our average- and worst-case flatness measures in RLoss, we find that flatness reduces significantly during robust overfitting. Namely, flatness "explains" the increased RLoss caused by overfitting very well. Fig. 6 (left) plots RLoss, alongside average- and worst-case flatness and the maximum Hessian eigenvalue throughout training of our AT baseline. Clearly, flatness increases alongside (test) RLoss as soon as robust overfitting occurs. Note that the best epoch is 60, meaning 0.4 (black dotted). For further illustration, Fig. 6 (middle) explicitly plots RLoss (y-axis) against flatness in RLoss (x-axis) across epochs (dark blue to dark red): RLoss and flatness clearly worsen "alongside" each other during overfitting, for both average- and worst-case flatness. Methods such as AT with self-supervision, AT-AWP or AT with unlabeled examples avoid both robust overfitting *and* sharp minima (right). This relationship generalizes to different hyper-parameter choices of these methods: Fig. 7 plots RLoss (y-axis) vs. average-case flatness (x-axis) across different hyper-parameters. Again, e.g., for TRADES or AT-AWP, hyper-parameters with lower RLoss also correspond to flatter minima. In fact, Fig. 7 indicates that the connection between robustness and flatness also generalizes *across* different methods (and individual models).

Improved Robustness Through Flatness: Indeed, across all trained models, we found a **strong correlation between robust generalization and flatness**, using RLoss as measure for robust generalization. As discussed in Sec. 4.1, we mainly consider RLoss to assess robust generalization as improvements in RLoss above $\sim$2.3 have, on average, only small impact on RErr. Pushing RLoss below 2.3, in contrast, directly translates to better RErr. This is illustrated in Fig. 8 (left) which plots RErr vs. RLoss for all evaluated models. To avoid this "kink" in the dotted red lines around RLoss$\approx$2.3, Fig. 8 (middle left) plots *RLoss* (y-axis) against *average-case* flatness in RLoss (x-axis), highlighting selected models. This reveals a *clear correlation between robustness and flatness*: More robust methods, e.g., AT with unlabeled examples or AT-AWP, correspond to

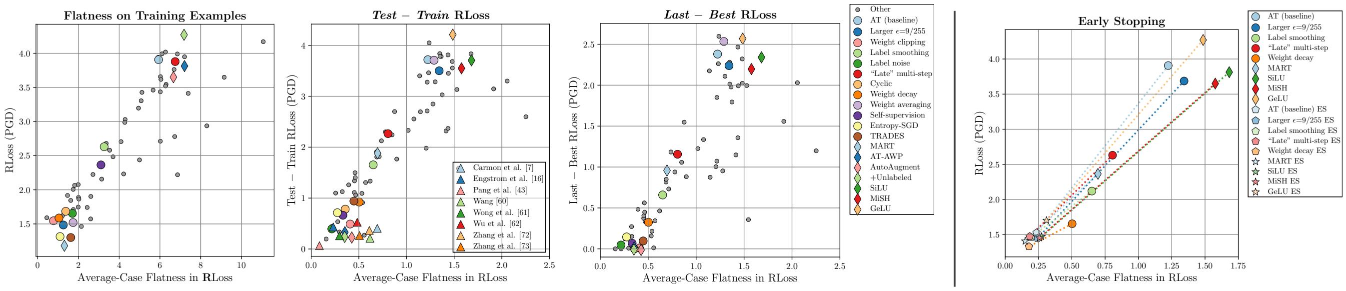

Figure 9: **Robust Generalization and Early Stopping. Left:** RLoss plotted vs. average-case flatness measured *on training examples*. Even on training examples, flatness is a good indicator for robust generalization. **Middle:** Robust generalization (RLoss) decomposed into the test-train difference and the last-best (epoch) improvement (y-axis), both plotted against average-case flatness in RLoss (x-axis). In both cases, flatness seems to play an important role, i.e., flatness clearly reduces both the robust generalization gap *and* robust overfitting. **Right:** RLoss vs. average-case flatness in RLoss for selected models (all in supplementary material) with and without early stopping ("ES"). Early stopping consistently leads to improved adversarial robustness *and* better flatness.

flatter minima. Methods improving flatness, e.g., Entropy-SGD, weight decay or weight clipping, improve adversarial robustness. This also translates to RErr (middle right), subject to the described bend at RLoss$\approx$2.3. While many robust methods still obtain better flatness, activation functions such as SiLU, MiSH or GeLU also seem to improve flatness, without clear advantage in terms of robustness. Similarly, weight decay or clipping improve robustness considerably. Overall, with Pearson/Spearman correlation coefficients of $0.85/0.87$ (p-values $<10^{-21}$), we revealed a strong relationship between robustness and flatness.

Fig. 8 (right) shows that this relationship is less clear when considering *worst-case* flatness in RLoss (Pearson coefficient 0.54). This is in contrast to [62] suggesting that worst-case flatness, in particular, is important to improve robustness of AT. However, worst-case flatness is more sensitive to ξ and, thus, less comparable across methods. Note that worst-case robustness is still a good indicator for overfitting, cf. Fig. 6. All results are summarized in tabular form in Tab. 1: Grouping methods by good, average or poor robustness, we find that methods need at least "some" flatness, average- *or* worst-case, to be successful.

Decomposing Robust Generalization: So far, we used (absolute) RLoss on test examples as proxy of robust generalization. This is based on the assumption that deep models are generally able to obtain nearly zero *train* RLoss. However, this is not the case for many methods in Tab. 1 (second column). Thus, we also consider the robust generalization *gap* and the RLoss difference between last and best (early stopped) epoch. First, however, Fig. 9 (left) shows that flatness, when measured on *training examples*, is also a good predictor of (test) robustness. Then, Fig. 9 (middle left) explicitly plots the RLoss generalization gap (test−train RLoss, y-axis) against average-case flatness in RLoss (x-axis). Robust methods generally reduce this gap by *both* reducing test RLoss *and* avoiding convergence in train RLoss. Furthermore, Fig. 9 (middle right) considers the difference between last and best epoch, essentially quantifying the extent of robust overfitting. Again, methods with small difference, i.e., little robust overfitting, generally correspond to flatter minima. This is also confirmed in Fig. 9 (right) showing that early stopping essentially finds flatter minima along the training trajectory, thereby improving adversarial robustness. Altogether, flatness improves robust generalization by reducing both the robust generalization gap *and* the impact of robust overfitting.

More Results: Fig. 1 shows that the pre-trained models from RobustBench [10] confirm our observations so far (also see Fig. 9, middle left). While detailed analysis is not possible as only early stopped models are provided, they are consistently more robust *and* correspond to flatter minima compared to our models. This is despite using different architectures (commonly Wide ResNets [70]).

5. Conclusion

In this paper, we studied the relationship between adversarial robustness, specifically considering robust overfitting [46], and flatness of the robust loss (RLoss) landscape w.r.t. perturbations in the weight space. We introduced both average- and worst-case measures for flatness in RLoss that are scale-invariant and allow comparison across models. Considering adversarial training (AT) and several popular variants, including TRADES [72], AT-AWP [62] or AT with additional unlabeled examples [7], we show a **clear relationship between adversarial robustness and flatness** in RLoss. More robust methods predominantly find flatter minima. Vice versa, approaches known to improve flatness, e.g., Entropy-SGD [8] or weight clipping [52] can help AT become more robust, as well. Moreover, even simple regularization methods such as AutoAugment [12], weight decay or label noise, are effective in increasing robustness by improving flatness. These observations also generalize to pre-trained models from RobustBench [10].

References

[1] Naveed Akhtar and Ajmal S. Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6, 2018. 2

[2] Jean-Baptiste Alayrac, Jonathan Uesato, Po-Sen Huang, Alhussein Fawzi, Robert Stanforth, and Pushmeet Kohli. Are labels required for improving adversarial robustness? In *NeurIPS*, 2019. 2, 6

[3] Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *NeurIPS*, 2020. 2

[4] Yogesh Balaji, Tom Goldstein, and Judy Hoffman. Instance adaptive adversarial training: Improved accuracy tradeoffs in neural nets. *arXiv.org*, abs/1910.08051, 2019. 2

[5] Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. In *CCS*, 2018. 2

[6] Johan Bjorck, Carla P. Gomes, Bart Selman, and Kilian Q. Weinberger. Understanding batch normalization. In *NeurIPS*, 2018. 3

[7] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C. Duchi, and Percy Liang. Unlabeled data improves adversarial robustness. In *NeurIPS*, 2019. 1, 2, 3, 6, 8

[8] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer T. Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. In *ICLR*, 2017. 1, 2, 3, 6, 8

[9] Safa Cicek and Stefano Soatto. Input and weight space smoothing for semi-supervised learning. In *ICCV Workshops*, 2019. 2, 3

[10] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv.org*, abs/2010.09670, 2020. 1, 3, 6, 8

[11] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020. 3, 6

[12] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data. *arXiv.org*, abs/1805.09501, 2018. 2, 6, 8

[13] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. MMA training: Direct input space margin maximization through adversarial training. In *ICLR*, 2020. 2

[14] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *ICML*, 2017. 2, 3, 4, 5

[15] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *NN*, 107, 2018. 2, 6

[16] Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. 2

[17] Farzan Farnia, Jesse M. Zhang, and David Tse. Generalizable adversarial training via spectral normalization. In *ICLR*, 2019. 1

[18] Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy A. Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv.org*, abs/2010.03593, 2020. 2, 3, 6

[19] Edward Grefenstette, Robert Stanforth, Brendan O'Donoghue, Jonathan Uesato, Grzegorz Swirszcz, and Pushmeet Kohli. Strength in numbers: Trading-off robustness and computation via adversarially-trained ensembles. *arXiv.org*, abs/1811.09300, 2018. 2

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 6

[21] Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *arXiv.org*, abs/1606.08415, 2016. 2, 6

[22] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In *NeurIPS*, 2019. 1, 2, 3, 6

[23] S. Hochreiter and J. Schmidhuber. Flat minima. *NC*, 9, 1997. 3

[24] Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. Learning with a strong adversary. *arXiv.org*, abs/1511.03034, 2015. 2

[25] J. Hwang, Youngwan Lee, Sungchan Oh, and Yu-Seok Bae. Adversarial training with stochastic weight average. *arXiv.org*, abs/2009.10526, 2020. 1

[26] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *UAI*, 2018. 2, 3, 6

[27] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *ICLR*, 2020. 2, 3, 5

[28] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *ICLR*, 2017. 2, 3, 4, 5

[29] Marc Khoury and Dylan Hadfield-Menell. On the geometry of adversarial examples. *arXiv.org*, abs/1811.00525, 2018. 1

[30] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 3, 4

[31] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 6

[32] Cassidy Laidlaw and Soheil Feizi. Playing it safe: Adversarial robustness with an abstain option. *arXiv.org*, abs/1911.11253, 2019. 2

[33] Alex Lamb, Vikas Verma, Juho Kannala, and Yoshua Bengio. Interpolated adversarial training: Achieving robust neural networks without sacrificing too much accuracy. In *AISec*, 2019. 2

[34] Hao Li, Zheng Xu, G. Taylor, and T. Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, 2018. 1, 2, 3, 4, 5

[35] Tao Lin, Sebastian U. Stich, Kumar Kshitij Patel, and Martin Jaggi. Don't use large mini-batches, use local SGD. In *ICLR*, 2020. 2, 3

[36] Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. Adv-bnn: Improved adversarial defense through robust bayesian neural network. In *ICLR*, 2019. 2

[37] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *ICLR*, 2018. 1, 2, 3

[38] Pratyush Maini, Eric Wong, and J. Zico Kolter. Adversarial robustness against the union of multiple perturbation models. *ICML*, 2020. 2

[39] Diganta Misra. Mish: A self regularized non-monotonic activation function. In *BMVC*, 2020. 2, 6

[40] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional smoothing with virtual adversarial training. *ICLR*, 2016. 2

[41] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *NeurIPS*, 2017. 2, 3, 4, 5

[42] Tianyu Pang, Xian Yang, Yinpeng Dong, Hang Su, and Jun Zhu. Bag of tricks for adversarial training. *arXiv.org*, abs/2010.00467, 2020. 2, 3, 6

[43] Tianyu Pang, Xiao Yang, Yinpeng Dong, Taufik Xu, Jun Zhu, and Hang Su. Boosting adversarial training with hypersphere embedding. In *NeurIPS*, 2020. 2

[44] Vinay Uday Prabhu, Dian Ang Yap, Joyce Xu, and J. Whaley. Understanding adversarial robustness through loss landscape geometries. *arXiv.org*, abs/1907.09061, 2019. 4

[45] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C. Duchi, and Percy Liang. Adversarial training can hurt generalization. *arXiv.org*, abs/1906.06032, 2019. 3

[46] Leslie Rice, Eric Wong, and J. Zico Kolter. Overfitting in adversarially robust deep learning. In *ICML*, 2020. 1, 3, 4, 6, 7, 8

[47] Vivek B. S. and R. Venkatesh Babu. Single-step adversarial training with dropout scheduling. In *CVPR*, 2020. 2

[48] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *NeurIPS*, 2018. 3

[49] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *NeurIPS*, 2018. 1, 3

[50] Samuel Henrique Silva and Peyman Najafirad. Opportunities and challenges in deep learning adversarial robustness: A survey. *arXiv.org*, abs/2007.00753, 2020. 2

[51] Vasu Singla, Sahil Singla, David Jacobs, and Soheil Feizi. Low curvature activations reduce overfitting in adversarial training. *arXiv.org*, abs/2102.07861, 2021. 1, 2, 3

[52] David Stutz, Nandhini Chandramoorthy, Matthias Hein, and Bernt Schiele. Bit error robustness for energy-efficient dnn accelerators. In *MLSys*, 2021. 1, 2, 6, 8

[53] David Stutz, Matthias Hein, and Bernt Schiele. Disentangling adversarial robustness and generalization. *CVPR*, 2019. 3

[54] David Stutz, Matthias Hein, and Bernt Schiele. Confidence-calibrated adversarial training: Generalizing to unseen attacks. In *ICML*, 2020. 2

[55] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 2, 6

[56] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. 1, 2

[57] Florian Tramèr and Dan Boneh. Adversarial training and robustness for multiple perturbations. In *NeurIPS*, 2019. 2

[58] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick D. McDaniel. Ensemble adversarial training: Attacks and defenses. *ICLR*, 2018. 2

[59] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *ICLR*, 2019. 3

[60] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *ICLR*, 2020. 1, 2, 3, 6

[61] Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020. 2

[62] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *NeurIPS*, 2020. 1, 2, 3, 4, 5, 6, 8

[63] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *NeurIPS*, 2016. 6

[64] Xi Wu, Uyeong Jang, Jiefeng Chen, Lingjiao Chen, and Somesh Jha. Reinforcing adversarial robustness using model confidence induced by adversarial training. In *ICML*, 2018. 2

[65] Han Xu, Yao Ma, Haochen Liu, Debayan Deb, Hui Liu, Jiliang Tang, and Anil K. Jain. Adversarial attacks and defenses in images, graphs and text: A review. *arXiv.org*, abs/1909.08072, 2019. 2

[66] Zhewei Yao, Amir Gholami, Qi Lei, Kurt Keutzer, and Michael W. Mahoney. Hessian-based analysis of large batch training and robustness to adversaries. In *NeurIPS*, 2018. 3, 5

[67] Nanyang Ye and Zhanxing Zhu. Bayesian adversarial learning. In *NeurIPS*, 2018. 2

[68] F. Yu, C. Liu, Yanzhi Wang, Liang Zhao, and X. Chen. Interpreting adversarial robustness: A view from decision surface in input space. *arXiv.org*, abs/1810.00144, 2018. 4

[69] Xiaoyong Yuan, Pan He, Qile Zhu, Rajendra Rana Bhat, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *arXiv.org*, abs/1712.07107, 2017. 2

[70] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 8

[71] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. In *NeurIPS*. 3

[72] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically

principled trade-off between robustness and accuracy. In *ICML*, 2019. 1, 2, 3, 4, 6, 8

[73] Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan S. Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In *ICML*, 2020. 6

[74] Yaowei Zheng, Richong Zhang, and Yongyi Mao. Regularizing neural networks via adversarial model perturbation. *arXiv.org*, abs/2010.04925, 2020. 2, 3