Relating Adversarially Robust Generalization to Flat Minima

David Stutz1  Matthias Hein2  Bernt Schiele1
1Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken
2University 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 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...
Robust Loss training examples
Robust Error (RErr)

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).

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 a 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 measure RLoss and RErr. We show that RLoss given a local neighborhood Bξ(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).

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].

#### 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 $\delta$ 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:

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

(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., $\epsilon = \delta / 255$ on CIFAR10. For evaluation (at test time), we consider both robust loss ($\text{RLoss}$) $\max_{\|\delta\|_{\infty} \leq \epsilon} \mathcal{L}(f(x + \delta; w), y)$, approximated using PGD, and robust test error ($\text{RErr}$), 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...
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

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

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 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 damps 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].
Average-Case / Random Flatness: Considering random weight perturbations $\nu \in B_\xi(w)$ within the $\xi$-neighborhood of $w$, average-case flatness is computed as
\[
\mathbb{E}_\nu \max_{||\delta|| \leq \epsilon} \mathcal{L}(f(x+\delta; w+\nu), y) - \max_{||\delta|| \leq \epsilon} \mathcal{L}(f(x+\delta; w), y)
\]
(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)):
\[
B_\xi(w) = \{w+\nu : ||\nu(l)||_2 \leq \xi ||w(l)||_2 \forall \text{ layers } l\}.
\]
(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, $\xi$ 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
\[
\max_{\nu \in B_\xi(w)} \left[ \max_{||\delta|| \leq \epsilon} \mathcal{L}(f(x+\delta; w+\nu), y) \right] - \max_{||\delta|| \leq \epsilon} \mathcal{L}(f(x+\delta; w), y)
\]
(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 $\xi$. 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 $\nu$ and $\delta$ (for each batch individually) using PGD. As illustrated in Fig. 4, RLoss increases quickly along adversarial directions, even for very small values of $\xi$, 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 1900 which reduces to 505 when up-scaling the model and increases to 7936 when down-scaling, without affecting 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).
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 $L_\infty$ 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 $\xi=0.5$. For worst-case flatness, we maximize RLoss jointly over adversarial examples and adversarial weights with $\xi=0.00075$, 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 [35] 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 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 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 $\xi$ of adversarial weight perturbations for AT-AWP (orange).](https://example.com/figure7.png)

![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.](https://example.com/table1.png)
translate to improved RErr if RLoss is high to begin with, i.e., “above” $-\ln(\frac{y}{\kappa}) \approx 2.3$ 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 a 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 $\text{RLoss} \approx 2.3$, 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
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=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 <1×10⁻²¹), 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 ϵ 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


[16] Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 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
[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
[59] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In ICLR, 2019. 3
[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
[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
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