Adversarial Weight Perturbation Helps Robust Generalization

Dongxian Wu[1,3] **Shu-Tao Xia**[1,3] **Yisen Wang**[2†]
[1]Tsinghua University
[2]Key Lab. of Machine Perception (MoE), School of EECS, Peking University
[3]PCL Research Center of Networks and Communications, Peng Cheng Laboratory

Abstract

The study on improving the robustness of deep neural networks against adversarial examples grows rapidly in recent years. Among them, adversarial training is the most promising one, which flattens the *input loss landscape* (loss change with respect to input) via training on adversarially perturbed examples. However, how the widely used *weight loss landscape* (loss change with respect to weight) performs in adversarial training is rarely explored. In this paper, we investigate the weight loss landscape from a new perspective, and identify a clear correlation between the flatness of weight loss landscape and robust generalization gap. Several well-recognized adversarial training improvements, such as early stopping, designing new objective functions, or leveraging unlabeled data, all implicitly flatten the weight loss landscape. Based on these observations, we propose a simple yet effective *Adversarial Weight Perturbation (AWP)* to explicitly regularize the flatness of weight loss landscape, forming a *double-perturbation* mechanism in the adversarial training framework that adversarially perturbs both inputs and weights. Extensive experiments demonstrate that AWP indeed brings flatter weight loss landscape and can be easily incorporated into various existing adversarial training methods to further boost their adversarial robustness.

1 Introduction

Although deep neural networks (DNNs) have been widely deployed in a number of fields such as computer vision [13], speech recognition [47], and natural language processing [10], they could be easily fooled to confidently make incorrect predictions by adversarial examples that are crafted by adding intentionally small and human-imperceptible perturbations to normal examples [42, 12, 52, 4]. As DNNs penetrate almost every corner in our daily life, ensuring their security, *e.g.,* improving their robustness against adversarial examples, becomes more and more important.

There have emerged a number of defense techniques to improve adversarial robustness of DNNs [33, 25, 48]. Across these defenses, Adversarial Training (AT) [12, 25] is the most effective and promising approach, which not only demonstrates moderate robustness, but also has thus far not been comprehensively attacked [2]. AT directly incorporates adversarial examples into the training process to solve the following optimization problem:

$$\min_{\mathbf{w}} \rho(\mathbf{w}), \quad \text{where} \quad \rho(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \max_{\|\mathbf{x}_i'-\mathbf{x}_i\|_p \le \epsilon} \ell(f_{\mathbf{w}}(\mathbf{x}_i'), y_i), \tag{1}$$

where n is the number of training examples, $\mathbf{x}_i'$ is the adversarial example within the ϵ-ball (bounded by an L_p-norm) centered at natural example $\mathbf{x}_i$, $f_{\mathbf{w}}$ is the DNN with weight $\mathbf{w}$, $\ell(\cdot)$ is the standard

†Corresponding author: Yisen Wang (yisen.wang@pku.edu.cn)

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

classification loss (*e.g.*, the cross-entropy (CE) loss), and $\rho(\mathbf{w})$ is called the "adversarial loss" following Madry et al. [25]. Eq. (1) indicates that AT restricts the change of loss when its input is perturbed (*i.e.*, flattening the **input loss landscape**) to obtain a certain of robustness, but its robustness is still far from satisfactory because of the huge robust generalization gap [39, 37], for example, an adversarially trained PreAct ResNet-18 [14] on CIFAR-10 [20] only has 43% test robustness, even it has already achieved 84% training robustness after 200 epochs (see Figure 1). Its robust generalization gap reaches 41%, which is very different from the standard training (on natural examples) whose standard generalization gap is always lower than 10%. Thus, how to mitigate the robust generalization gap becomes essential for the robustness improvement of adversarial training methods.

Recalling that **weight loss landscape** is a widely used indicator to characterize the standard generalization gap in standard training scenario [31, 21], however, there are few explorations under adversarial training, among which, Prabhu et al. [35] and Yu et al. [55] tried to use the pre-generated adversarial examples to explore but failed to draw the expected conclusions. In this paper, we explore the weight loss landscape under adversarial training using on-the-fly generated adversarial examples, and identify a strong connection between the flatness of weight loss landscape and robust generalization gap. Several well-recognized adversarial training improvements, *i.e.*, AT with early stopping [37], TRADES [58], MART [49] and RST [6], all implicitly flatten the weight loss landscape to narrow the robust generalization gap. Motivated by this, we propose an explicit weight loss landscape regularization, named *Adversarial Weight Perturbation (AWP)*, to directly restrict the flatness of weight loss landscape. Different from random perturbations [15], AWP injects the strongest worst-case weight perturbations, forming a *double-perturbation* mechanism (*i.e.*, inputs and weights are both adversarially perturbed) in the adversarial training framework. AWP is generic and can be easily incorporated into existing adversarial training approaches with little overhead. Our main contributions are summarized as follows:

- We identify the fact that flatter weight loss landscape often leads to smaller robust generalization gap in adversarial training via characterizing the weight loss landscape using adversarial examples generated on-the-fly.
- We propose *Adversarial Weight Perturbation (AWP)* to explicitly regularize the weight loss landscape of adversarial training, forming a double-perturbation mechanism that injects the worst-case input and weight perturbations.
- Through extensive experiments, we demonstrate that AWP consistently improves the adversarial robustness of state-of-the-art methods by a notable margin.

2 Related Work

2.1 Adversarial Defense

Since the discovery of adversarial examples, many defensive approaches have been developed to reduce this type of security risk such as defensive distillation [33], feature squeezing [53], input denoising [3], adversarial detection [24], gradient regularization [34, 43], and adversarial training [12, 25, 48]. Among them, adversarial training has been demonstrated to be the most effective method [2]. Based on adversarial training, a number of new techniques are introduced to enhance its performance further.

TRADES [58]. TRADES optimizes an upper bound of adversarial risk that is a trade-off between accuracy and robustness:

$$\rho^{\text{TRADES}}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\left\{\text{CE}\big(f_{\mathbf{w}}(\mathbf{x}_i), y_i\big) + \beta \cdot \max \text{KL}\big(f_{\mathbf{w}}(\mathbf{x}_i) \| f_{\mathbf{w}}(\mathbf{x}_i')\big)\right\}, \tag{2}$$

where KL is the Kullback–Leibler divergence, CE is the cross-entropy loss, and β is the hyperparameter to control the trade-off between natural accuracy and robust accuracy.

MART [49]. MART incorporates an explicit differentiation of misclassified examples as a regularizer of adversarial risk:

$$\rho^{\text{MART}}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\left\{\text{BCE}\big(\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i'), y_i\big) + \lambda \cdot \text{KL}\big(\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i) \| \mathbf{f}_{\mathbf{w}}(\mathbf{x}_i')\big) \cdot \big(1 - [\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i)]_{y_i}\big)\right\}, \tag{3}$$

where $[\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i)]_{y_i}$ denotes the y_i-th element of output vector $\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i)$ and $\mathrm{BCE}\big(\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i), y_i\big) = -\log\big([\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i')]_{y_i}\big) - \log\big(1 - \max_{k \neq y_i}[\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i')]_k\big)$.

Semi-Supervised Learning (SSL) [6, 46, 28, 57]. SSL-based methods utilize additional unlabeled data. They first generate pseudo labels for unlabeled data by training a natural model on the labeled data. Then, adversarial loss $\rho(\mathbf{w})$ is applied to train a robust model based on both labeled and unlabeled data:

$$\rho^{\text{SSL}}(\mathbf{w}) = \rho^{\text{labeled}}(\mathbf{w}) + \lambda \cdot \rho^{\text{unlabeled}}(\mathbf{w}), \tag{4}$$

where λ is the weight on unlabeled data. $\rho^{\text{labeled}}(\mathbf{w})$ and $\rho^{\text{unlabeled}}(\mathbf{w})$ are usually the same adversarial loss. For example, RST in Carmon et al. [6] uses TRADES loss and semi-supervised MART in Wang et al. [49] uses MART loss.

2.2 Robust Generalization

Compared with standard generalization (on natural examples), training DNNs with robust generalization (on adversarial examples) is particularly difficult [25], and often possesses significantly higher sample complexity [19, 54, 26] and needs more data [39]. Nakkiran [29] showed that a model requires more capacity to be robust. Tsipras et al. [44] and Zhang et al. [58] demonstrated that adversarial robustness may be inherently at odds with natural accuracy. Moreover, there are a series of works studying the robust generalization from the view of loss landscape. In adversarial training, there are two types of loss landscape: 1) *input loss landscape* which is the loss change with respect to the input. It depicts the change of loss in the vicinity of training examples. AT explicitly flattens the input loss landscape by training on adversarially perturbed examples, while there are other methods doing this by gradient regularization [23, 38], curvature regularization [27], and local linearity regularization [36]. These methods are fast on training but only achieve comparable robustness with AT. 2) *weight loss landscape* which is the loss change with respect to the weight. It reveals the geometry of loss landscape around model weights. Different from the standard training scenario where numerous studies have revealed the connection between the weight loss landscape and their standard generalization gap [17, 31, 7], whether the connection exists in adversarial training is still under exploration. Prabhu et al. [35] and Yu et al. [55] tried to establish this connection in adversarial training but failed due to the inaccurate weight loss landscape characterization.

Different from these studies, we characterize the weight loss landscape from a new perspective, and identify a clear relationship between weight loss landscape and robust generalization gap.

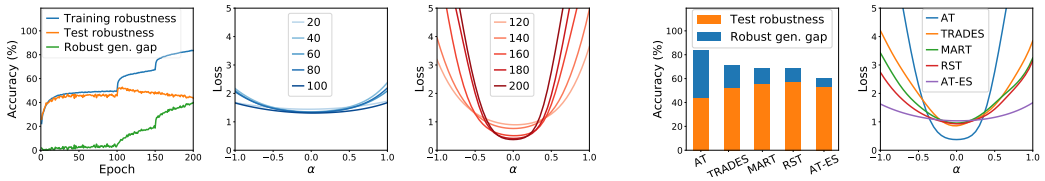
3 Connection of Weight Loss Landscape and Robust Generalization Gap

In this section, we first propose a new method to characterize the weight loss landscape, and then investigate it from two perspectives: 1) in the training process of adversarial training, and 2) across different adversarial training methods, which leads to a clear correlation between weight loss landscape and robust generalization gap. To this end, some discussions about the weight loss landscapes are provided.

Visualization. We visualize the weight loss landscape by plotting the adversarial loss change when moving the weight $\mathbf{w}$ along a random direction $\mathbf{d}$ with magnitude α:

$$g(\alpha) = \rho(\mathbf{w} + \alpha\mathbf{d}) = \frac{1}{n}\sum_{i=1}^{n} \max_{\|\mathbf{x}_i' - \mathbf{x}_i\|_p \leq \epsilon} \ell(f_{\mathbf{w}+\alpha\mathbf{d}}(\mathbf{x}_i'), y_i), \tag{5}$$

where $\mathbf{d}$ is sampled from a Gaussian distribution and filter normalized by $\mathbf{d}_{l,j} \leftarrow \frac{\mathbf{d}_{l,j}}{\|\mathbf{d}_{l,j}\|_F}\|\mathbf{w}_{l,j}\|_F$ ($\mathbf{d}_{l,j}$ is the j-th filter at the l-th layer of $\mathbf{d}$ and $\|\cdot\|_F$ denotes the Frobenius norm) to eliminate the scaling invariance of DNNs following Li et al. [21]. The adversarial loss ρ is usually approximated by the cross-entropy loss on adversarial examples following Madry et al. [25]. Here, we generate adversarial examples on-the-fly by PGD (attacks are reviewed in Appendix A) for the current perturbed model $f_{\mathbf{w}+\alpha\mathbf{d}}$ and then compute their cross-entropy loss (refer to Appendix B.1 for details). The key difference to previous works lies on the adversarial examples used for visualization. Prabhu et al. [35] and Yu et al. [55] used a fixed set of pre-generated adversarial examples on the original model $f_{\mathbf{w}}$ in the visualization process, which will severely underestimate the adversarial loss due to the inconsistency between the source model (original model $f_{\mathbf{w}}$) and the target model (current perturbed model $f_{\mathbf{w}+\alpha\mathbf{d}}$). Considering $\mathbf{d}$ is randomly selected, we repeat the visualization 10 times



(a) In the training process of vanilla AT (*Left*: Learning curve; *Mid*: Landscape before "best"; *Right*: Landscape after "best")

(b) Across different methods (*Left*: Generalization gap; *Right*: Landscape)

Figure 1: The relationship between weight loss landscape and robust generalization gap is investigated a) in the training process of vanilla AT; and b) across different adversarial training methods on CIFAR-10 using PreAct ResNet-18 and L_∞ threat model. ("Landscape" is a abbr. of weight loss landscape)

with different $\mathbf{d}$ in Appendix B.2 and their shapes are similar and stable. Thus, the visualization method is valid to characterize the weight loss landscape, based on which, we can carefully investigate the connection between weight loss landscape and robust generalization gap.

The Connection in the Learning Process of Adversarial Training. We firstly show how the weight loss landscape changes along with the robust generalization gap in the learning process of adversarial training. We train a PreAct ResNet-18 [14] on CIFAR-10 for 200 epochs using vanilla AT with a piece-wise learning rate schedule (initial learning rate is 0.1, and divided by 10 at the 100-th and 150-th epoch). The training and test attacks are both 10-step PGD (PGD-10) with step size $2/255$ and maximum L_∞ perturbation $\epsilon = 8/255$. The learning curve and weight loss landscape are shown in Figure 1(a) where the "best" (highest test robustness) is at the 103-th epoch. Before the "best", the test robustness is close to the training robustness, thus the robust generalization gap (green line) is small. Meanwhile, the weight loss landscape (plotted every 20 epochs) before the "best" is also very flat. After the "best", the robust generalization gap (green line) becomes larger as the training continues, while the weight loss landscape becomes sharper simultaneously. The trends also exist on other model architectures (VGG-19 [40] and WideResNet-34-10 [56]), datasets (SVHN [30] and CIFAR-100 [20]), threat model (L_2), and learning rate schedules (cyclic [41], cosine [22]), as shown in Appendix C. Thus, the flatness of weight loss landscape is well-correlated with the robust generalization gap during the training process.

The Connection across Different Adversarial Training Methods. Furthermore, we explore whether the relationship between weight loss landscape and robust generalization gap still exists across different adversarial training methods. Under the same settings as above, we train PreAct ResNet-18 using several state-of-the-art adversarial training methods like TRADES [58], MART [49], RST [6], and AT with Early Stopping (AT-ES) [37]. Figure 1(b) demonstrates their training/test robustness and weight loss landscape. Compared with vanilla AT, all methods have a smaller robust generalization gap and a flatter weight loss landscape. Although these state-of-the-art methods improve adversarial robustness using various techniques, they all implicitly flatten the weight loss landscape. It can be also observed that the smaller generalization gap one method achieves, the flatter weight loss landscape it has. This observation is consistent with that in the training process, which verifies that weight loss landscape has a strong correlation with robust generalization gap.

Does Flatter Weight Loss Landscape Certainly Lead to Higher Test Robustness? Revisiting Figure 1(b), AT-ES has the flattest weight loss landscape (also the smallest robust generalization gap), but does not obtain the highest test robustness. Since the robust generalization gap is defined as the difference between training and test robustness, the low test robustness of AT-ES is caused by the low training robustness. It indicates that early stopping technique does not make full use of the whole training process, *e.g.*, it stops training around 100-th epoch only with 60% training robustness which is 20% lower than that of 200-th epoch. Therefore, a flatter weight loss landscape does directly lead to a smaller robust generalization gap but is only beneficial to the final test robustness on condition that the training process is sufficient (*i.e.*, training robustness is high).

Why Do We Need Weight Loss Landscape? As aforementioned, adversarial training has already optimized the input loss landscape via training on adversarial examples. However, the adversarial example is generated by injecting input perturbation on each individual example to obtain the highest adversarial loss, which is an example-wise "local" worst-case that does not consider the overall effect on multiple examples. The weight of DNNs can influence the losses of all examples such that it could be perturbed to obtain a model-wise "global" worst-case (highest adversarial loss over

multiple examples). Weight perturbations can serve as a good complement for input perturbations. Also, optimizing on perturbed weights (*i.e.*, making the loss remains small even if perturbations are added on the weights) could lead to a flat weight loss landscape, which further will narrow the robust generalization gap. In the next section, we will propose such a weight perturbation for adversarial training.

4 Proposed Adversarial Weight Perturbation

In this section, we propose *Adversarial Weight Perturbation (AWP)* to explicitly flatten the weight loss landscape via injecting the worst-case weight perturbation into DNNs. As discussed above, in order to improve the test robustness, we need to focus on both the training robustness and the robust generalization gap (delivered by the flatness of weight loss landscape). Thus, we have the objective:

$$\min_{\mathbf{w}} \big\{\rho(\mathbf{w}) + \big(\rho(\mathbf{w}+\mathbf{v}) - \rho(\mathbf{w})\big)\big\} \rightarrow \min_{\mathbf{w}} \rho(\mathbf{w}+\mathbf{v}), \tag{6}$$

where $\rho(\mathbf{w})$ is the original adversarial loss in Eq. (1), $\rho(\mathbf{w}+\mathbf{v}) - \rho(\mathbf{w})$ is a term to characterize the flatness of weight loss landscape, and $\mathbf{v}$ is weight perturbation that needs to be carefully selected.

4.1 Weight Perturbation

Perturbation Direction. Different from the commonly used random weight perturbation (sampling a random direction) [50, 18, 15], we propose the *Adversarial Weight Perturbation* (AWP), along which the adversarial loss increases dramatically. That is,

$$\min_{\mathbf{w}} \max_{\mathbf{v}\in\mathcal{V}} \rho(\mathbf{w}+\mathbf{v}) \rightarrow \min_{\mathbf{w}} \max_{\mathbf{v}\in\mathcal{V}} \frac{1}{n}\sum_{i=1}^{n} \max_{\|\mathbf{x}_i'-\mathbf{x}_i\|_p\leq\epsilon} \ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i), \tag{7}$$

where $\mathcal{V}$ is a feasible region for the perturbation $\mathbf{v}$. Similar to the adversarial input perturbation, AWP also injects the worst-case on weights in a small region around $\mathbf{f}_{\mathbf{w}}$. Note that the maximization on $\mathbf{v}$ depends on the entire examples (at least the batch examples) to make the whole loss (not the loss on each example) maximal, thus these two maximizations are not exchangeable.

Perturbation Size. Following the weight perturbation direction, we need to determine how much perturbation should be injected. Different from the fixed value constraint ϵ on adversarial inputs, we restrict the weight perturbation $\mathbf{v}_l$ using its relative size to the weights of l-th layer $\mathbf{w}_l$:

$$\|\mathbf{v}_l\| \leq \gamma\|\mathbf{w}_l\|, \tag{8}$$

where γ is the constraint on weight perturbation size. The reasons for using relative size to constrain weight perturbation lie on two aspects: 1) the numeric distribution of weights is different from layer to layer, so it is impossible to constrain weights of different layers using a fixed value; and 2) there is scale invariance on weights, *e.g.*, when nonlinear ReLU is used, the network remains unchanged if we multiply the weights in one layer by 10, and divide by 10 at the next layer.

4.2 Optimization

Once the direction and size of weight perturbation are determined, we propose an algorithm to optimize the double-perturbation adversarial training problem in Eq. (7). For the two maximization problems, we circularly generate adversarial example $\mathbf{x}_i'$ and then update weight perturbation $\mathbf{v}$ both empirically using PGD*. The procedure of AWP-based vanilla AT, named AT-AWP, is as follows.

Input Perturbation. We craft adversarial examples $\mathbf{x}'$ using PGD attack on $\mathbf{f}_{\mathbf{w}+\mathbf{v}}$:

$$\mathbf{x}_i' \leftarrow \Pi_\epsilon\big(\mathbf{x}_i' + \eta_1 \text{sign}(\nabla_{\mathbf{x}_i'}\ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i))\big) \tag{9}$$

where $\Pi(\cdot)$ is the projection function and $\mathbf{v}$ is 0 for the first iteration.

Weight Perturbation. We calculate the adversarial weight perturbation based on the generated adversarial examples $\mathbf{x}'$:

$$\mathbf{v} \leftarrow \Pi_\gamma\big(\mathbf{v} + \eta_2 \frac{\nabla_{\mathbf{v}}\frac{1}{m}\sum_{i=1}^{m}\ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i)}{\|\nabla_{\mathbf{v}}\frac{1}{m}\sum_{i=1}^{m}\ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i)\|}\|\mathbf{w}\|\big), \tag{10}$$

*We find it works well in our experiments. With regard to the results on theoretical measurements or guarantees for the maximization problem like Wang et al. [48], we leave it for further work.

where m is the batch size, and $\mathbf{v}$ is layer-wise updated (refer to Appendix D for details). Similar to generating adversarial examples $\mathbf{x}'$ via FGSM (one-step) or PGD (multi-step), $\mathbf{v}$ can also be solved by one-step or multi-step methods. Then, we can alternately generate $\mathbf{x}'$ and calculate $\mathbf{v}$ for a number of iterations A. As shortly will be shown in Section 5.2, one iteration for A and one-step for $\mathbf{v}$ (default settings) are enough to get good robustness improvements.

Model Training. Finally, we update the parameters of the perturbed model $\mathbf{f}_{\mathbf{w}+\mathbf{v}}$ using SGD. Note that after optimizing the loss of a perturbed point on the landscape, we should come back to the center point again for the next start. Thus, the actual parameter update follows:

$$\mathbf{w} \leftarrow (\mathbf{w}+\mathbf{v}) - \eta_3 \nabla_{\mathbf{w}+\mathbf{v}} \frac{1}{m} \sum_{i=1}^{m} \ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i', y_i)) - \mathbf{v}. \tag{11}$$

The complete pseudo-code of AT-AWP and extensions of AWP to other adversarial training approaches like TRADES, MART and RST are shown in Appendix D.

4.3 Theoretical Analysis

We also provide a theoretical view on why AWP works. Based on previous work on PAC-Bayes bound [31], in adversarial training, let $\ell(\cdot,\cdot)$ be 0-1 loss, then $\rho(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \max_{\|\mathbf{x}_i'-\mathbf{x}_i\|_p \leq \epsilon} \ell(f_{\mathbf{w}}(\mathbf{x}_i'), y_i) \in [0,1]$. Given a "prior" distribution P (a common assumption is zero mean, σ^2 variance Gaussian distribution) over the weights, the expected error of the classifier can be bounded with probability at least $1-\delta$ over the draw of n training data:

$$\mathbb{E}_{\{\mathbf{x}_i,y_i\}_{i=1}^n,\mathbf{u}}[\rho(\mathbf{w}+\mathbf{u})] \leq \rho(\mathbf{w}) + \big\{\mathbb{E}_{\mathbf{u}}[\rho(\mathbf{w}+\mathbf{u})] - \rho(\mathbf{w})\big\} + 4\sqrt{\frac{1}{n}KL(\mathbf{w}+\mathbf{u}\|P) + \ln\frac{2n}{\delta}}. \tag{12}$$

Following Neyshabur et al. [31], we choose $\mathbf{u}$ as a zero mean spherical Gaussian perturbation with variance σ^2 in every direction, and set the variance of the perturbation to the weight with respect to its magnitude $\sigma = \alpha\|\mathbf{w}\|$, which makes the third term of Eq. (12) become a constant $4\sqrt{\frac{1}{n}(\frac{1}{2\alpha} + \ln\frac{2n}{\delta})}$. Thus, the robust generalization gap is bounded by the second term that is the expectation of the flatness of weight loss landscape. Considering the optimization efficiency and effectiveness on expectation, $\mathbb{E}_{\mathbf{u}}[\rho(\mathbf{w}+\mathbf{u})] \leq \max_{\mathbf{u}}[\rho(\mathbf{w}+\mathbf{u})]$. AWP exactly optimizes the worst-case of the flatness of weight loss landscape $\{\max_{\mathbf{u}}[\rho(\mathbf{w}+\mathbf{u})] - \rho(\mathbf{w})\}$ to control the above PAC-Bayes bound, which theoretically justifies why AWP works.

4.4 A Case Study on Vanilla AT and AT-AWP

In this part, we conduct a case study on vanilla AT and AT-AWP across three benchmark datasets (SVHN [30], CIFAR-10 [20], CIFAR-100 [20]) and two threat models (L_∞ and L_2) using PreAct ResNet-18 for 200 epochs. We follow the same settings in Rice et al. [37]: for L_∞ threat model, $\epsilon = 8/255$, step size is $1/255$ for SVHN, and $2/255$ for CIFAR-10 and CIFAR-100; for L_2 threat model, $\epsilon = 128/255$, step size is $15/255$ for all datasets. The training/test attacks are PGD-10/PGD-20 respectively. For AT-AWP, $\gamma = 1 \times 10^{-2}$. The test robustness is reported in Table 1 (natural accuracy is in Appendix E) where "best" means the highest robustness that ever achieved at different checkpoints for each dataset and threat model while "last" means the robustness at the last epoch checkpoint. We can see that AT-AWP consistently improves the test robustness for all cases. It indicates that AWP is generic and can be applied on various threat models and datasets.

Table 1: Test robustness (%) of AT and AT-AWP across different datasets and threat models. We omit the standard deviations of 5 runs as they are very small ($< 0.40\%$), which hardly effect the results.

Threat Model	Method	SVHN		CIFAR-10		CIFAR-100	
		Best	Last	Best	Last	Best	Last
L_∞	AT	53.36	44.49	52.79	44.44	27.22	20.82
	AT-AWP	**59.12**	**55.87**	**55.39**	**54.73**	**30.71**	**30.28**
L_2	AT	66.87	65.03	69.15	65.93	41.33	35.27
	AT-AWP	**72.57**	**67.73**	**72.69**	**72.08**	**45.60**	**44.66**

Table 2: Test robustness (%) on CIFAR-10 using WideResNet under L_∞ threat model. We omit the standard deviations of 5 runs as they are very small ($< 0.40\%$), which hardly effect the results.

Defense	Natural	FGSM	PGD-20	PGD-100	CW_∞	SPSA	AA
AT	**86.07**	61.76	56.10	55.79	54.19	61.40	52.60 [¶]
AT-AWP	85.57	**62.90**	**58.14**	**57.94**	**55.96**	**62.65**	**54.04**
TRADES	84.65	61.32	56.33	56.07	54.20	61.10	53.08
TRADES-AWP	**85.36**	**63.49**	**59.27**	**59.12**	**57.07**	**63.85**	**56.17**
MART	84.17	61.61	58.56	57.88	54.58	58.90	51.10
MART-AWP	**84.43**	**63.98**	**60.68**	**59.32**	**56.37**	**62.75**	**54.23**
Pre-training	87.89	63.27	57.37	56.80	55.95	62.55	54.92
Pre-training-AWP	**88.33**	**66.34**	**61.40**	**61.21**	**59.28**	**65.55**	**57.39**
RST	**89.69**	**69.60**	62.60	62.22	60.47	67.60	59.53
RST-AWP	88.25	67.94	**63.73**	**63.58**	**61.62**	**68.72**	**60.05**

5 Experiments

In this section, we conduct comprehensive experiments to evaluate the effectiveness of AWP including its benchmarking robustness, ablation studies and comparisons to other regularization techniques.

5.1 Benchmarking the State-of-the-art Robustness

In this part, we evaluate the robustness of our proposed AWP on CIFAR-10 to benchmark the state-of-the-art robustness against white-box and black-box attacks. Two types of adversarial training methods are considered here: One is only based on original data: 1) AT [25]; 2) TRADES [58]; and 3) MART [49]. The other uses additional data: 1) Pre-training [16]; and 2) RST [6].

Experimental Settings. For CIFAR-10 under L_∞ attack with $\epsilon = 8/255$, we train WideResNet-34-10 for AT, TRADES, and MART, while WideResNet-28-10 for Pre-training and RST, following their original papers. For pre-training, we fine-tune 50 epochs using a learning rate of 0.001 as [16]. Other defenses are trained for 200 epochs using SGD with momentum 0.9, weight decay 5×10^{-4}, and an initial learning rate of 0.1 that is divided by 10 at the 100-th and 150-th epoch. Simple data augmentations such as 32×32 random crop with 4-pixel padding and random horizontal flip are applied. The training attack is PGD-10 with step size $2/255$. For AWP, we set $\gamma = 5 \times 10^{-3}$. Other hyper-parameters of the baselines are configured as per their original papers.

White-box/Black-box Robustness. Table 2 reports the "best" test robustness (the highest robustness ever achieved at different checkpoints for each defense against each attack) against white-box and black-box attacks. "Natural" denotes the accuracy on natural test examples. First, for white-box attack, we test FGSM, PGD-20/100, and CW_∞ (L_∞ version of CW loss optimized by PGD-100). AWP almost improves the robustness of state-of-the-art methods against all types of attacks. This is because AWP aims at achieving a flat weight loss landscape, which is generic across different methods. Second, for black-box attack, we test the query-based attack SPSA [45] (100 iterations with perturbation size 0.001 (for gradient estimation), learning rate 0.01, and 256 samples for each gradient estimation). Again, the robustness improved by AWP is consistent amongst different methods. In addition, we test AWP against Auto Attack (AA) [9], which is a strong and reliable attack to verify the robustness via an ensemble of diverse parameter-free attacks including three white-box attacks (APGD-CE [9], APGD-DLR [9], and FAB [8]) and a black-box attack (Square Attack [1]). Compared with their leaderboard results[†], AWP can further boost their robustness, ranking the 1st on both with and without additional data. Even some AWP methods without additional data can surpass the results under additional data[‡]. This verifies that AWP improves adversarial robustness reliably rather than improper tuning of hyper-parameters of attacks, gradient obfuscation or masking.

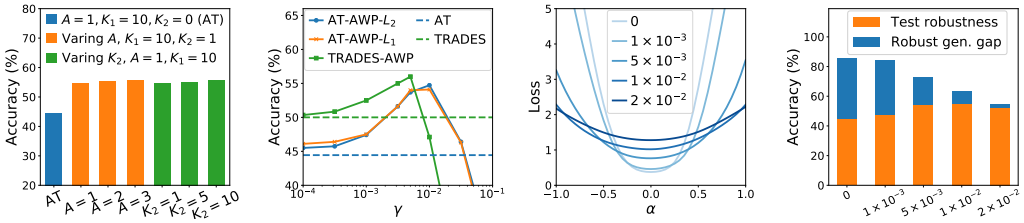
[¶]Here is the result on WideResNet-34-10 while the leaderborder one is on WideResNet-34-20.

[†]`https://github.com/fra31/auto-attack`

[‡]`https://github.com/csdongxian/AWP/tree/main/auto_attacks`

5.2 Ablation Studies on AWP

In this part, we delve into AWP to investigate its each component. We train PreAct ResNet-18 using vanilla AT and AT-AWP with L_∞ threat model with $\epsilon = 8/255$ for 200 epochs following the same setting in Section 5.1. The training/test attacks are PGD-10/PGD-20 (step size $2/255$) respectively.



(a) Optimization (b) Weight perturbation (c) Weight loss landscape (d) Generalization gap

Figure 2: The ablation study experiments on CIFAR-10 using AT-AWP unless otherwise specified.

Analysis on Optimization Strategy. Recalling Section 4.2, there are 3 parameters when optimizing AWP, *i.e.*, step number K_1 in generating adversarial example $\mathbf{x}'$, step number K_2 in solving adversarial weight perturbation $\mathbf{v}$, and alternation iteration A between $\mathbf{x}'$ and $\mathbf{v}$. For step number K_1 in generating $\mathbf{x}'$, previous work has showed that PGD-10 based AT usually obtains good robustness [48], so we set $K_1 = 10$ by default. For step number K_2 in solving $\mathbf{v}$, we assess AT-AWP with $K_2 \in \{1, 5, 10\}$ while keeping $A = 1$. The green bars in Figure 2(a) show that varying K_2 achieves almost the same test robustness. For alternation iteration A, we test $A \in \{1, 2, 3\}$ while keeping $K_2 = 1$. The orange bars show that one iteration ($A = 1$) already has 55.39% test robustness, and extra iterations only bring few improvements but with much overhead. Based on these results, the default setting for AWP is $A = 1, K_1 = 10, K_2 = 1$ whose training time overhead is $\sim 8\%$.

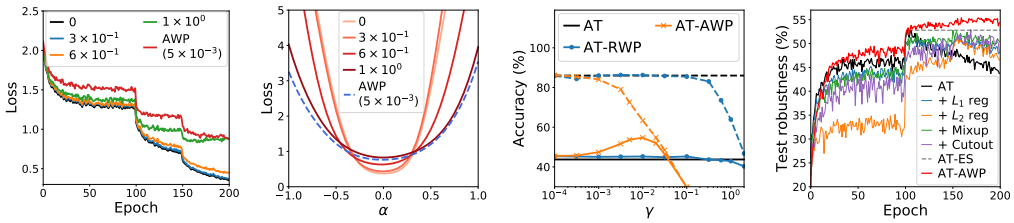
Analysis on Weight Perturbation. Here, we explore the effect of weight perturbation size (direction will be analyzed in Section 5.3) from two aspects: size constraint γ and size measurement norm. The test robustness with varying γ on AT-AWP and TRADES-AWP are shown in Figure 2(b). We can see that both methods can achieve notable robustness improvements in a certain range $\gamma \in [1 \times 10^{-3}, 5 \times 10^{-3}]$. It implies that the perturbation size cannot be too small to ineffectively regularize the flatness of weight loss landscape and also cannot be too large to make DNNs hard to train. Once γ is properly selected, it has a relatively good transferability across different methods (improvements of AT-AWP and TRADES-AWP have an overlap on γ, though their highest points are not the same). As for the size measurement norm, L_1 and L_2 (also called Frobenius norm L_F) almost have no difference on test robustness.

Effect on Weight Loss Landscape and Robust Generalization Gap. We visualize the weight loss landscape of AT-AWP with different γ in Figure 2(c) and present its corresponding training/test robustness in Figure 2(d). The gray line of $\gamma = 0$ is the vanilla AT (without AWP). As γ grows, the regularization becomes stronger, thus the weight loss landscape becomes flatter. Accordingly, the robust generalization gap becomes smaller. This verifies that AWP indeed brings flatter weight loss landscape and smaller robust generalization gap. In addition, the flattest weight loss landscape (smallest robust generalization gap) is obtained at a large $\gamma = 2 \times 10^{-2}$ but its training/test robustness decreases, which implies that γ should be properly selected by balancing the training robustness and the flatness of weight loss landscape to obtain the test robustness improvement.

5.3 Comparisons to Other Regularization Techniques

In this part, we compare AWP with other regularizations using the same setting as Section 5.2.

Comparison to Random Weight Perturbation (RWP). We evaluate the difference of AWP and RWP from the following 3 views: 1) Adversarial loss of AT pre-trained model perturbed by RWP and AWP. As shown in Figure 3(a), RWP only has an obvious increase of adversarial loss at a extremely large $\gamma = 1$ (others are similar to pre-trained AT ($\gamma = 0$)), while AWP (red line) has much higher adversarial loss than others just using a very small perturbation ($\gamma = 5 \times 10^{-3}$). Therefore, AWP can find the worst-case perturbation in a small region while RWP needs a relatively large perturbation. 2) Weight loss landscape of models trained by AT-RWP and AT-AWP. As shown in Figure 3(b), RWP only flattens the weight loss landscape at a large $\gamma \geq 0.6$. Even, RWP under $\gamma = 1$ can only obtain a



(a) Loss curve (b) Weight loss landscapes (c) Robustness (d) Learning curve

Figure 3: Comparisons of AWP and other regularization techniques (the values in (a)/(c) legend are γ in RWP unless otherwise specified) on CIFAR-10 using PreAct ResNet-18 and L_∞ threat model.

similar flatter weight loss landscape as AWP under $\gamma = 5 \times 10^{-3}$. 3) Robustness. We test AT-AWP and AT-RWP with a large range $\gamma \in [1 \times 10^{-4}, 2.0]$. Figure 3(c) (solid/dashed lines are test/training robustness respectively) shows that AWP can significantly improve the test robustness at a small $\gamma \in [1 \times 10^{-3}, 1 \times 10^{-2}]$. For RWP, the test robustness almost does not improve at $\gamma \leq 0.3$ because of the unchanged weight loss landscape, even begins to decrease when $\gamma \geq 0.6$. This is because such a large weight perturbation makes DNNs hard to train and severely reduces the training robustness (dashed blue line), which in turns reduces the test robustness though the weight loss landscape is flattened. In summary, AWP is much better than RWP for weight perturbation.

Comparison to Weight Regularization and Data Augmentation. Here, we compare AWP ($\gamma = 5 \times 10^{-3}$) with L_1/L_2 weight regularization and data augmentation of mixup [59]/cutout [11]. We follow the *best* hyper-parameters tuned in Rice et al. [37]: $\lambda = 5 \times 10^{-6}/5 \times 10^{-3}$ for L_1/L_2 regularization respectively, patch length 14 for cutout, and $\alpha = 1.4$ for mixup. We show the test robustness (natural accuracy is in Appendix F) of all checkpoints for different methods in Figure 3(d). The vanilla AT achieves the best robustness after the first learning rate decay and starts overfitting. Other techniques, except of AWP, do not obtain a better robustness than early stopped AT (AT-ES), which is consistent with the observations in Rice et al. [37]. However, AWP (red line) behaves very differently from the others: it does improve the best robustness (52.79% of vanilla AT $\rightarrow$ 55.39% of AT-AWP). AWP shows its superiority over other weight regularization and data augmentation, and improves the best robustness further compared with early stopping. More experiments under L_2 threat model could be found in Appendix F, which also demonstrates the effectiveness of AWP.

5.4 A Closer Look at the Weights Learned by AWP

In this part, we explore how the distribution of weights changes when we apply AWP on it. We plot the histogram of weight values in different layers, and find that AT-AWP and vanilla AT are similar in shallower layers, while AT-AWP has smaller magnitudes and a more symmetric distribution in deeper layers. Figure 4 demonstrates the distribution of weight values in the last convolutional layer of PreAct ResNet-18 on CIFAR-10 dataset.

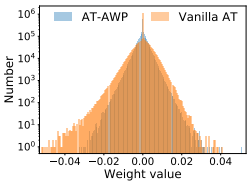


Figure 4: Weight distribution

6 Conclusion

In this paper, we characterized the weight loss landscape using the on-the-fly generated adversarial examples, and identified that the weight loss landscape is closely related to the robust generalization gap. Several well-recognized adversarial training variants all introduce a flatter weight loss landscape though they use different techniques to improve adversarial robustness. Based on these findings, we proposed *Adversarial Weight Perturbation (AWP)* to directly make the weight loss landscape flat, and developed a double-perturbation (adversarially perturbing both inputs and weights) mechanism in the adversarial training framework. Comprehensive experiments show that AWP is generic and can improve the state-of-the-art adversarial robustness across different adversarial training approaches, network architectures, threat models, and benchmark datasets.

Broader Impact

Adversarial training is the currently most effective and promising defense against adversarial examples. In this work, we propose AWP to improve the robustness of adversarial training, which may help to build a more secure and robust deep learning system in real world. At the same time, AWP introduces extra computation, which probably has negative impacts on the environmental protection (*e.g.*, low-carbon). Further, the authors do not want this paper to bring overoptimism about AI safety to the society. The majority of adversarial examples are based on known threat models (*e.g.* L_p in this paper), and the robustness is also achieved on them. Meanwhile, the deployed machine learning system faces attacks from all sides, and we are still far from complete model robustness.

Acknowledgments and Disclosure of Funding

Yisen Wang is partially supported by the National Natural Science Foundation of China under Grant 62006153, and CCF-Baidu Open Fund. Shu-Tao Xia is partially supported by the National Key Research and Development Program of China under Grant 2018YFB1800204, the National Natural Science Foundation of China under Grant 61771273, the R&D Program of Shenzhen under Grant JCYJ20180508152204044, and the project PCL Future Greater-Bay Area Network Facilities for Large-scale Experiments and Applications (LZC0019).

References

[1] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. *arXiv preprint arXiv:1912.00049*, 2019.

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

[3] Yang Bai, Yan Feng, Yisen Wang, Tao Dai, Shu-Tao Xia, and Yong Jiang. Hilbert-based generative defense for adversarial examples. In *ICCV*, 2019.

[4] Yang Bai, Yuyuan Zeng, Yong Jiang, Yisen Wang, Shu-Tao Xia, and Weiwei Guo. Improving query efficiency of black-box adversarial attack. In *ECCV*, 2020.

[5] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *S&P*, 2017.

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

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

[8] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. *arXiv preprint arXiv:1907.02044*, 2019.

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

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

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

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

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

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

[15] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. In *CVPR*, 2019.

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

[17] 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.

[18] Mohammad Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. In *ICML*, 2018.

[19] Justin Khim and Po-Ling Loh. Adversarial risk bounds for binary classification via function transformation. *arXiv preprint arXiv:1810.09519*, 2018.

[20] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical Report, University of Toronto*, 2009.

[21] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, 2018.

[22] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

[23] Chunchuan Lyu, Kaizhu Huang, and Hai-Ning Liang. A unified gradient regularization family for adversarial examples. In *ICDM*, 2015.

[24] Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In *ICLR*, 2018.

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

[26] Omar Montasser, Steve Hanneke, and Nathan Srebro. Vc classes are adversarially robustly learnable, but only improperly. In *COLT*, 2019.

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

[28] Amir Najafi, Shin-ichi Maeda, Masanori Koyama, and Takeru Miyato. Robustness to adversarial perturbations in learning from incomplete data. In *NeurIPS*, 2019.

[29] Preetum Nakkiran. Adversarial robustness may be at odds with simplicity. *arXiv preprint arXiv:1901.00532*, 2019.

[30] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[31] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *NeurIPS*, 2017.

[32] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *EuroS&P*, 2016.

[33] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *S&P*, 2016.

[34] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Asia CCS*, 2017.

[35] Vinay Uday Prabhu, Dian Ang Yap, Joyce Xu, and John Whaley. Understanding adversarial robustness through loss landscape geometries. *arXiv preprint arXiv:1907.09061*, 2019.

[36] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. In *NeurIPS*, 2019.

[37] Leslie Rice, Eric Wong, and J Zico Kolter. Overfitting in adversarially robust deep learning. In *ICML*, 2020.

[38] Andrew Slavin Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *AAAI*, 2018.

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

[40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[41] Leslie N Smith. Cyclical learning rates for training neural networks. In *WACV*, 2017.

[42] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

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

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

[45] Jonathan Uesato, Brendan O'Donoghue, Pushmeet Kohli, and Aaron Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *ICML*, 2018.

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

[47] Yisen Wang, Xuejiao Deng, Songbai Pu, and Zhiheng Huang. Residual convolutional ctc networks for automatic speech recognition. *arXiv preprint arXiv:1702.07793*, 2017.

[48] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *ICML*, 2019.

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

[50] Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. In *ICLR*, 2018.

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

[52] Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, and Xingjun Ma. Skip connections matter: On the transferability of adversarial examples generated with resnets. In *ICLR*, 2020.

[53] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

[54] Dong Yin, Ramchandran Kannan, and Peter Bartlett. Rademacher complexity for adversarially robust generalization. In *ICML*, 2019.

[55] Fuxun Yu, Chenchen Liu, Yanzhi Wang, Liang Zhao, and Xiang Chen. Interpreting adversarial robustness: A view from decision surface in input space. *arXiv preprint arXiv:1810.00144*, 2018.

[56] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[57] Runtian Zhai, Tianle Cai, Di He, Chen Dan, Kun He, John Hopcroft, and Liwei Wang. Adversarially robust generalization just requires more unlabeled data. *arXiv preprint arXiv:1906.00555*, 2019.

[58] 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.

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

A Adversarial Attack

Given a natural example $\mathbf{x}_i$ with class label y_i and a DNN model $\mathbf{f}_\mathbf{w}$, the goal of an adversary is to find an adversarial example $\mathbf{x}_i'$ that fools the network to make incorrect predictions while still remains in the ϵ-ball centered at $\mathbf{x}_i$ ($\|\mathbf{x}_i' - \mathbf{x}_i\|_p \leq \epsilon$). A lot of attacking methods have been proposed for the crafting of adversarial examples. Here, we only name a few.

Fast Gradient Sign Method (FGSM) [12]. FGSM perturbs the natural example $\mathbf{x}_i$ for one step by the amount of ϵ along the gradient direction:

$$\mathbf{x}_i' = \mathbf{x}_i + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}_i} \ell(\mathbf{f}_\mathbf{w}(\mathbf{x}_i), y_i)). \tag{13}$$

Projected Gradient Descent (PGD) [25]. PGD perturbs the natural example $\mathbf{x}_i$ for K_1 steps with small step size η_1. After each step of perturbation, PGD projects the adversarial example back onto the ϵ-ball of $\mathbf{x}_i$, if it goes beyond the ϵ-ball:

$$\mathbf{x}_i'^{(k+1)} = \Pi_\epsilon\big(\mathbf{x}_i'^{(k)} + \eta_1 \cdot \text{sign}(\nabla_{\mathbf{x}_i'} \ell(\mathbf{f}_\mathbf{w}(\mathbf{x}_i'^{(k)}), y_i))\big), \tag{14}$$

where $\Pi(\cdot)$ is the projection operation, and $\mathbf{x}_i'^{(k)}$ is the adversarial example at the k-th step. There are also other types of attacks including Jacobian-based Saliency Map Attack (JSMA) [32], Carlini and Wagner (CW) [5], and so on.

B Details for the Weight Loss Landscape Visualization Method

In this section, we first provide the pseudo-code of our proposed visualization method for the weight loss landscape in the adversarial training, and then verify its reliability.

B.1 Pseudo-code of the Visualization Method

As shown in Algorithm 1 for the visualization of weight loss landscape, we firstly sample a random direction $\mathbf{d}$ from a Gaussian distribution. Then, we apply the "filter normalization" technique (Line 3-7) from Li et al. [21] to avoid the scaling effect[§] of DNNs. Next, we calculate the adversarial loss for a series of perturbed weights independently, *i.e.*, $\rho(\mathbf{w} + \alpha\mathbf{d}), \alpha \in \{\alpha_{min}, \cdots, \alpha_{max}\}$. For a given perturbed weights $\mathbf{w} + \alpha\mathbf{d}$, we generate its own adversarial examples using PGD following Madry et al. [25] (Line 9-14). Then, we approximate the adversarial loss of the current perturbed model $f_{\mathbf{w}+\alpha\mathbf{d}}$ by the cross-entropy loss on these on-the-fly generated adversarial examples (Line 15). Finally, we plot the weight loss landscape (Line 17).

B.2 Reliability of the 1-D Visualization

To improve the trustworthiness of our results, we check the reliability of our visualization method from two perspectives: 1) repeatability; 2) comparisons to 2-D visualization.

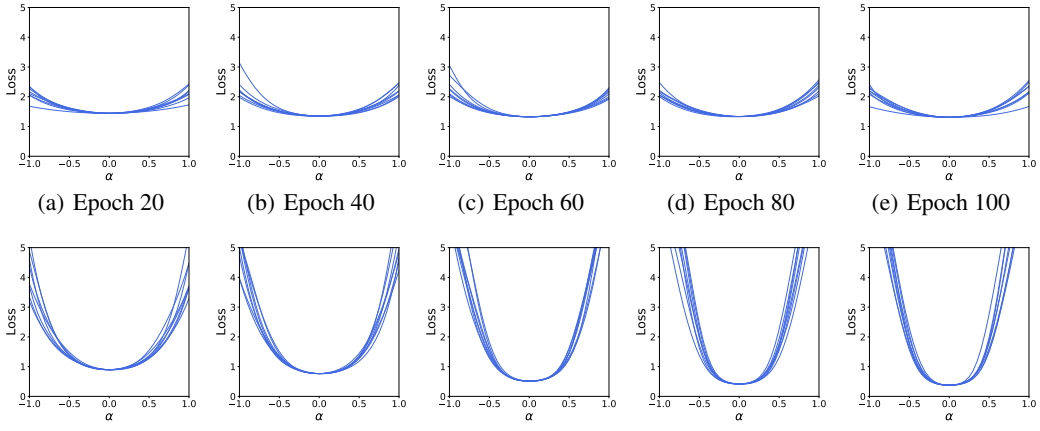
Repeatability. We first investigate whether different random directions produce dramatically different plots. We show the weight loss landscape of PreAct ResNet-18 during vanilla adversarial training along 10 randomly selected directions in Figure 5. We find the plots of the same checkpoint are very close in shape, which indicates the stability of our visualization method.

Comparisons to 2-D Visualization. Next, we explore whether 1-D visualization obtains similar results to 2-D visualization (a much time-consuming method). The 2-D visualization introduces an extra random filter-normalized direction $\mathbf{e}$, and plots $g(\alpha, \beta) = \rho(\mathbf{w} + \alpha\mathbf{d} + \beta\mathbf{e})$. Different from the standard training scenario where near-zero loss on the training set can be always achieved, the adversarial loss on the training set is usually larger than zero, *i.e.*, the center point $g(0, 0)$ in the weight loss landscape is often larger than zero, which hampers the comparison on the flatness of weight loss landscape. Therefore, we visualize the relative weight loss landscape $|g(\alpha, \beta) - g(0, 0)|$ instead. Figure 6 presents the 2-D visualization of the same model as Figure 5 at the 100-th, 140-th and 200-th epoch checkpoint. We can see that the weight loss landscape is flatter at the 100-th epoch and sharper at the 200-th epoch, which is compatible to the findings from the 1-D visualization. For time costs, it consumes ~ 4 days for the 2-D visualization of a single adversarially trained PreAct

[§]In DNNs with ReLU activation, the network remains unchanged if we multiply the weights in one layer by 10, and divide by 10 at the next layer.

Algorithm 1 Visualization of Weight Loss Landscape

Input: Network $\mathbf{f_w}$ with L-layer (F_l filters in the l-th layer), training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, PGD step size η_1, PGD step number K_1, the scalar parameter $\alpha \in [\alpha_{min}, \alpha_{max}]$.
Sample a random direction $\mathbf{d} \sim \mathcal{N}(0, 1)$
for $l = 1, \ldots, L$ **do**
  for $j = 1, \ldots, F_l$ **do**
    $\mathbf{d}_{l,j} \leftarrow \frac{\mathbf{d}_{l,j}}{\|\mathbf{d}_{l,j}\|_F} \|\mathbf{w}_{l,j}\|_F$
  end for
end for
for $\alpha = \alpha_{min}, \cdots, \alpha_{max}$ **do**
  for $i = 1, \ldots, n$ (in parallel) **do**
    $\mathbf{x}_i' \leftarrow \mathbf{x}_i + \epsilon\delta$, where $\delta \sim$ Uniform$(-1, 1)$
    for $k = 1, \cdots, K_1$ **do**
      $\mathbf{x}_i' \leftarrow \Pi_\epsilon\big(\mathbf{x}_i' + \eta_1 \text{sign}(\nabla_{\mathbf{x}_i'} \ell(\mathbf{f}_{\mathbf{w}+\alpha\mathbf{d}}(\mathbf{x}_i'), y_i))\big)$
    end for
  end for
  $\rho(\mathbf{w} + \alpha\mathbf{d}) \leftarrow \frac{1}{n} \sum_{i=1}^n \ell(f_{\mathbf{w}+\alpha\mathbf{d}}(\mathbf{x}_i'), y_i)$
end for
Plot $(\alpha, \rho(\mathbf{w} + \alpha\mathbf{d})), \forall \alpha \in [\alpha_{min}, \alpha_{max}]$



(a) Epoch 20 (b) Epoch 40 (c) Epoch 60 (d) Epoch 80 (e) Epoch 100

(f) Epoch 120 (g) Epoch 140 (h) Epoch 160 (i) Epoch 180 (j) Epoch 200

Figure 5: Repeatability of the 1-D visualization along 10 different random directions.

ResNet-18 using one GeForce RTX 2080Ti, while it is $\sim$ 2 hours for the 1-D visualization of the same model. Thus, we adopt the 1-D visualization in most cases.

From the above experiments, we can conclude that our 1-D visualization method can characterize the property of the high-dimensional weight loss landscape reliably and efficiently.

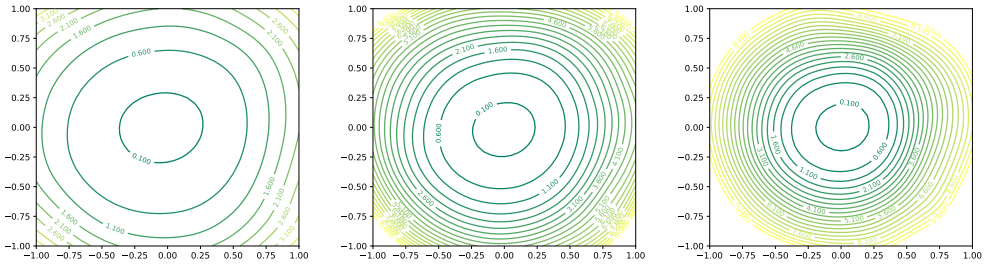
C More Evidence for the Connection of Weight Loss Landscape and Robust Generalization Gap

In this section, we provide more empricial evidence to identify the connection of the weight loss landscape and the robust generalization gap across learning rate schedules, model architectures, datasets, and threat models.

C.1 The Connection across Learning Rate Schedules

To investigate whether the learning rate schedule affects the connection of weight loss landscape and robust generalization gap, we test another two commonly used learning rate schedules:

- Cosine schedule [22]: We decrease the learning rate lr using the cosine function from 0.1 to 0 over 200 epochs, *i.e.*, $lr = 0.05(\cos(\pi t/200) + 1))$ at the t-th epoch [6];

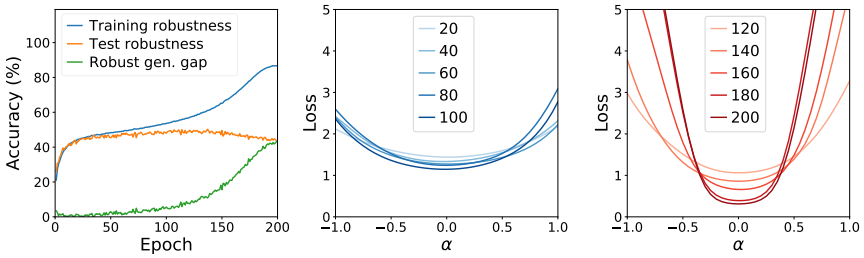


(a) Epoch 100 (b) Epoch 140 (c) Epoch 200

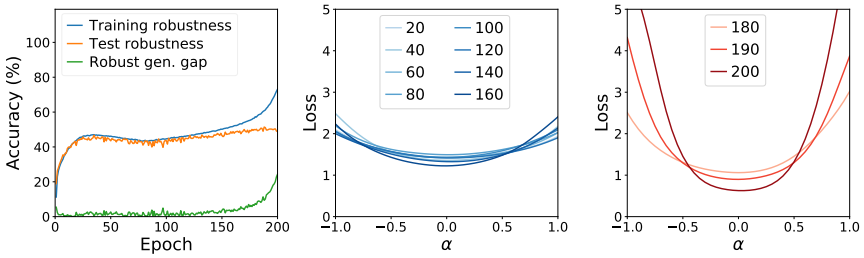
Figure 6: 2-D visualization of weight loss landscape at the different epoch checkpoints.

- Cyclic schedule [41]: We increase lr linearly from 0 to some maximum (0.2 at the 80-th epoch), and then decrease it linearly to 0 until 200-th epoch [51].

We adversarially train PreAct ResNet-18 with different learning rate schedules using the same experimental settings in Section 3. The learning curves are shown on the left column in Figure 7, where the whole training process can be split into two stages: the early stage with small robust generalization gap ($\leq 10\%$) and the late stage with large robust generalization gap ($> 10\%$). In Figure 7, the weight loss landscapes of checkpoints at different epochs from the early stage (blue curves) are on the middle column, while the weight loss landscapes from the late stage (red curves) are on the right column. Due to the different learning rate schedules, the robust generalization gap becomes large at different epochs. The cosine schedule enlarges the robust generalization gap after the 120-th epoch with $lr < 0.34$. The weight loss landscape becomes sharper correspondingly. The cyclic schedule starts to significantly enlarge the gap much later, almost after the 175-th epoch with $lr < 0.16$. Meanwhile, the weight loss landscape also becomes sharp much later. Generally, no matter which epoch and learning rate it is, we can always find that the weight loss landscape becomes sharper immediately after the robust generalization gap increases, which implies a clear correlation between weight loss landscape and robust generalization gap.



(a) Cosine schedule

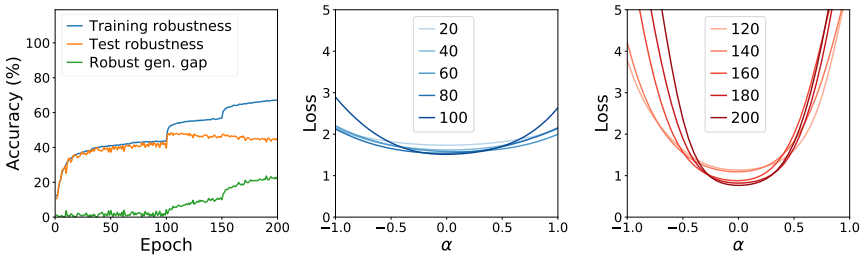


(b) Cyclic schedule

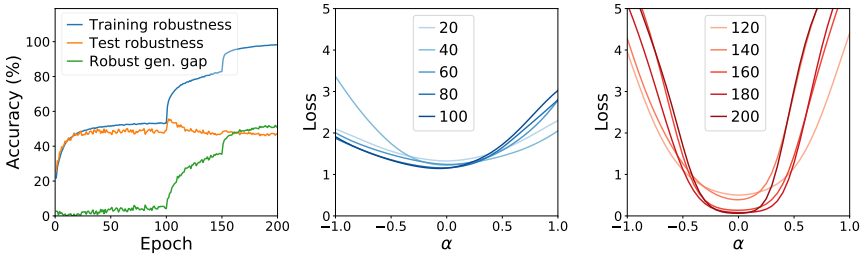
Figure 7: The relationship between weight loss landscape and robust generalization gap across learning rate schedules (cosine and cyclic) with PreAct ResNet-18 on CIFAR-10 under L_∞ attack.

C.2 The Connection across Model Architectures

The previous experiments are all based on PreAct ResNet-18. Here we additionally conduct experiments with VGG-19 [40] and WideResNet-34-10 [56] to verify the connection across network architectures. The same experimental settings as Section 3 are adopted and the results are shown in Figure 8. They all behave similarly: once the robust generalization gap increases (after the first learning rate decay), the weight loss landscape becomes sharper. This indicates that the connection of weight loss landscape and robust generalization gap still exists across architectures.



(a) VGG-19



(b) WideResNet-34-10

Figure 8: The relationship between weight loss landscape and robust generalization gap across model architectures (VGG-19 and WideResNet-34-10) on CIFAR-10 using piece-wise learning rate schedule and L_∞ attack.

C.3 The Connection across Datasets

Next, we demonstrate that the connection is an universal phenomenon across datasets on CIFAR-100 [20] and SVHN [30]. We adversarially train PreAct ResNet-18 on different datasets with the same settings as Section 3. The results are shown in Figure 9. The phenomenon on CIFAR-100 is similar to that on CIFAR-10, *i.e.*, after the first learning rate decay, the robust generalization gap becomes larger and the weight loss landscape becomes sharper as well. For SVHN, the robust generalization gap increases significantly even earlier: it almost achieves its highest robustness around 10-th epoch, and starts overfitting. Meanwhile, the weight loss landscape also keeps flat at 10-th epoch and starts to become sharper. In conclusion, the strong connection of weight loss landscape and robust generalization gap is universal across datasets.

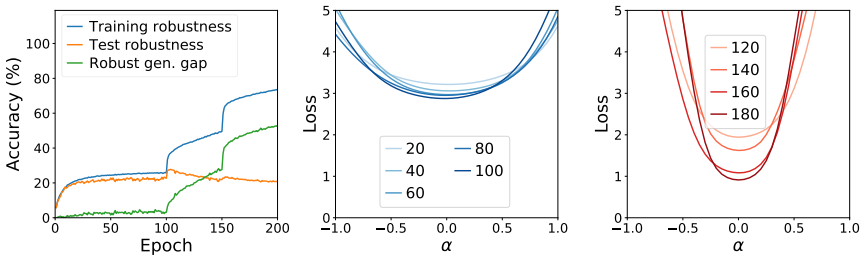
C.4 The Connection on L_2 Threat Model

To further explore the universality of the connection, we additionally conduct experiments on L_2 threat model in Figure 10. The other experimental settings are the same as Section 3. Under the L_2 threat model, the connection still exists: once the robust generalization gap increases (after the first learning rate decay), the weight loss landscape becomes sharper.

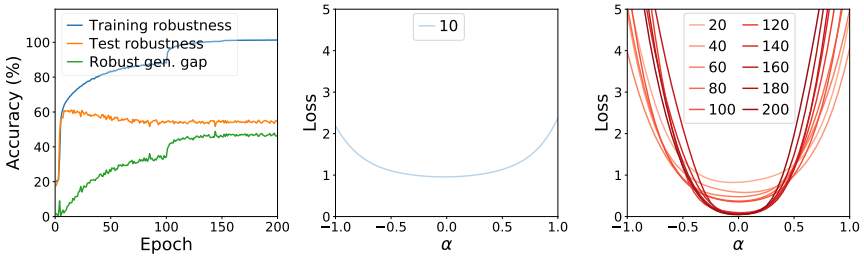
D Algorithms for the AWP-based Defense

In this section, we first provide the pseudo-code of the AWP-based vanilla advesraial trainig (AT-AWP), and then describe how to satisfy the constraint of the perturbation size in Eq. (8) via the weight update in Eq. (10) and the extensions to other AT variants (TRADES, MART, and RST).

D.1 Pseudo-code of AT-AWP

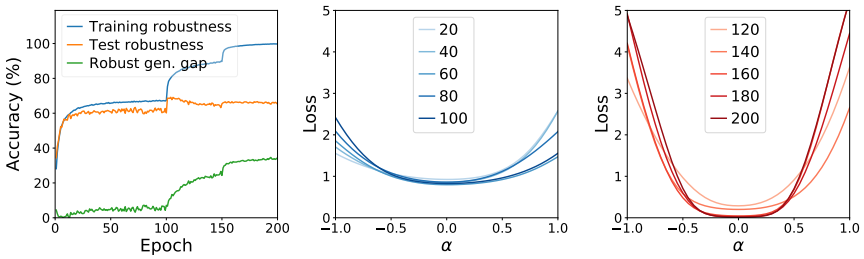


(a) CIFAR-100



(b) SVHN

Figure 9: The relationship between weight loss landscape and robust generalization gap across datasets (CIFAR-100 and SVHN) with PreAct ResNet-18 using piece-wise learning rate schedule and L_∞ attack.



(a) L_2 threat model

Figure 10: The relationship between weight loss landscape and robust generalization gap with PreAct ResNet-18 on CIFAR-10 using piece-wise learning rate schedule and L_2 attack.

Algorithm 2 AT-AWP

Input: Network $\mathbf{f}_\mathbf{w}$, training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, batch size m, learning rate η_3, PGD step size η_1, PGD steps K_1, AWP constraint γ, AWP step size η_2, AWP steps K_2, alternate iteration A.
Output: Robust model $\mathbf{f}_\mathbf{w}$.
for $t = 0$ to $T-1$ **do**
  for $a = 0$ to $A-1$ **do**
    for $i = 1, \ldots, m$ (in parallel) **do**
      $\mathbf{x}_i' \leftarrow \mathbf{x}_i + \epsilon\delta$, where $\delta \sim \text{Uniform}(-1, 1)$
      for $k = 1, \cdots, K_1$ **do**
        $\mathbf{x}_i' \leftarrow \Pi_\epsilon\big(\mathbf{x}_i' + \eta_1 \text{sign}(\nabla_{\mathbf{x}_i'} \ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i))\big)$
      end for
    end for
    for $k = 1, \ldots, K_2$ **do**
      $\mathbf{v} \leftarrow \Pi_\gamma\big(\mathbf{v} + \eta_2 \frac{\nabla_\mathbf{v} \frac{1}{m}\sum_i \ell(f_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i)}{\|\nabla_\mathbf{v} \frac{1}{m}\sum_i \ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i)\|}\|\mathbf{w}\|\big)$
    end for
  end for
  $\mathbf{w} \leftarrow (\mathbf{w}+\mathbf{v}) - \eta_3 \nabla_{\mathbf{w}+\mathbf{v}} \frac{1}{m}\sum_i \ell(f_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i', y_i)) - \mathbf{v}$
end for

D.2 Details for the Weight Update under the Constraint

Recall that we restrict the weight perturbation $\mathbf{v}_l$ in the l-th layer using its relative size to the corresponding weight $\mathbf{w}_l$, *i.e.*, $\|\mathbf{v}_l\| \leq \gamma\|\mathbf{w}_l\|$. To implement this restriction on weight perturbation, we apply a layer-wise projection operation, *i.e.*, once the weight perturbation is out of the ball, we project it back onto the surface of the ball. We have the following layer-wise projection operator,

$$\Pi_\gamma(\mathbf{v}) = \begin{cases} \gamma\dfrac{\|\mathbf{w}_l\|}{\|\mathbf{v}_l\|}\mathbf{v}_l, & \text{if } \|\mathbf{v}_l\| > \gamma\|\mathbf{w}_l\|, \forall l \in \{1, \cdots, L\} \\ \mathbf{v}_l, & \text{if } \|\mathbf{v}_l\| \leq \gamma\|\mathbf{w}_l\|, \forall l \in \{1, \cdots, L\}. \end{cases} \tag{15}$$

By default, we set $\eta_2 = \frac{\gamma}{A \cdot K_2}$ (the notations refer to Algorithm 2).

D.3 Extensions of AWP to Other Adversarial Training Methods

Our proposed AWP is a general method and can be easily extended to other well-recognized adversarial training variants including TRADES, MART and RST where the only difference is the method-specific adversarial loss in Eq. (1).

Specifically, for AWP-based TRADES (TRADES-AWP), we also first generate adversarial examples following TRADES method,

$$\mathbf{x}_i' \leftarrow \Pi_\epsilon\big(\mathbf{x}_i' + \eta_1 \text{sign}(\nabla_{\mathbf{x}_i'} \text{KL}(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i) \| \mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i')))\big), \tag{16}$$

and then the AWP $\mathbf{v}$ and the DNN parameter $\mathbf{w}$ of TRADES are updated similarly following Eq. (10) and Eq. (11) respectively, where $\ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i)$ is TRADES-specific as $\text{CE}(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i), y_i) + \beta \cdot \text{KL}(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i) \| f_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'))$.

Similarly, for AWP-based MART (MART-AWP), we generate adversarial examples following MART method,

$$\mathbf{x}_i' \leftarrow \Pi_\epsilon\big(\mathbf{x}_i' + \eta_1 \text{sign}(\nabla_{\mathbf{x}_i'} \ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i))\big), \tag{17}$$

and then update the AWP $\mathbf{v}$ follow Eq. (10). Next, the DNN parameter $\mathbf{w}$ of MART is updated using Eq. (11), where $\ell(\mathbf{f}_{\mathbf{w}+\mathbf{v}}(\mathbf{x}_i'), y_i)$ is MART-specific loss as $\text{BCE}(\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i'), y_i) + \lambda \cdot \text{KL}(\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i) \| \mathbf{f}_{\mathbf{w}}(\mathbf{x}_i')) \cdot (1 - [\mathbf{f}_{\mathbf{w}}(\mathbf{x}_i)]_{y_i})$.

RST, as an SSL-based method, first generates pseudo labels for unlabeled data, and then adversarially train DNNs using TRADES loss on the new dataset which consists of labeled data and unlabeled data with pseudo labels. Thus, we can incorporate AWP into RST just like TRADES-AWP.

E More Results for Section 4.3: A Case Study on Vanilla AT and AT-AWP

Following Section 4.3, we provide the complete results (test robustness and natural accuracy) of AT and AT-AWP in Table 3. Under L_2 threat model, AWP improves both the robustness and natural accuracy on all datasets. While under L_∞ threat model, AWP improves the robustness on condition of sacrificing the natural accuracy on CIFAR-10 and CIFAR-100. This maybe because natural images (CIFAR) are more complicated than color digits (SVHN). However, for robustness, AWP demonstrates a general behaviour that consistently improves the best and last robustness by a recognizable gain across datasets and threat models.

F More Experiments on Comparison to Other Regularization Techniques

Following Section 5.3, here we provide the complete results (test robustness and natural accuracy) of AWP and several regularization techniques under both L_∞ and L_2 threat models on CIFAR-10. Under the L_∞ threat model, we follow the best hyper-parameters tuned in Rice et al. [37]: $\lambda = 5 \times 10^{-6}/5 \times 10^{-3}$ for L_1/L_2 regularization respectively, patch length 14 for cutout, and $\alpha = 1.4$ for mixup. Under the L_2 threat model, we use the same hyper-parameters since we find that they almost have the same trends after parameter tuning. For AWP, we all set $\gamma = 5 \times 10^{-3}$. We use the same training settings as Section 5.2 and the test attack is PGD-20. We show test robustness and test natural accuracy in Table 4 (L_∞ threat model) and Table 5 (L_2 threat model). We find AWP indeed improves the test robustness of both the best checkpoint and the last checkpoint by a notable margin. Besides, we visualize the learning curves in Figure 11. We find that AWP can constantly

Table 3: Performance (%) of AT and AT-AWP on PreAct ResNet-18 across different datasets and threat models over 5 random runs.

Dataset	Norm	Method	Robustness		Natural accuracy	
			Best	Last	Best	Last
SVHN	L_∞	AT	53.36 ± 0.09	44.49 ± 0.27	92.18 ± 0.15	89.85 ± 0.33
		AT-AWP	$\mathbf{59.12 \pm 0.26}$	$\mathbf{55.87 \pm 0.39}$	$\mathbf{93.85 \pm 0.11}$	$\mathbf{92.59 \pm 0.52}$
	L_2	AT	66.87 ± 0.25	65.03 ± 0.24	93.69 ± 0.12	93.25 ± 0.28
		AT-AWP	$\mathbf{72.57 \pm 0.40}$	$\mathbf{67.73 \pm 0.21}$	$\mathbf{95.95 \pm 0.36}$	$\mathbf{95.22 \pm 0.27}$
CIFAR-10	L_∞	AT	52.79 ± 0.21	44.44 ± 0.39	$\mathbf{85.57 \pm 0.16}$	$\mathbf{84.56 \pm 0.19}$
		AT-AWP	$\mathbf{55.39 \pm 0.39}$	$\mathbf{54.73 \pm 0.16}$	82.00 ± 0.19	81.11 ± 0.39
	L_2	AT	69.15 ± 0.13	65.93 ± 0.35	89.57 ± 0.09	88.96 ± 0.18
		AT-AWP	$\mathbf{72.69 \pm 0.19}$	$\mathbf{72.08 \pm 0.39}$	$\mathbf{90.18 \pm 0.31}$	$\mathbf{89.71 \pm 0.17}$
CIFAR-100	L_∞	AT	27.22 ± 0.16	20.82 ± 0.20	$\mathbf{56.33 \pm 0.23}$	$\mathbf{54.61 \pm 0.33}$
		AT-AWP	$\mathbf{30.71 \pm 0.25}$	$\mathbf{30.28 \pm 0.30}$	54.19 ± 0.39	54.39 ± 0.29
	L_2	AT	41.33 ± 0.09	35.27 ± 0.29	62.65 ± 0.11	60.50 ± 0.17
		AT-AWP	$\mathbf{45.60 \pm 0.23}$	$\mathbf{44.66 \pm 0.22}$	$\mathbf{65.07 \pm 0.31}$	$\mathbf{64.40 \pm 0.41}$

improve the robustness under both L_∞ and L_2 threat models throughout the entire training process, which demonstrates the superiority of AWP over other regularization methods.

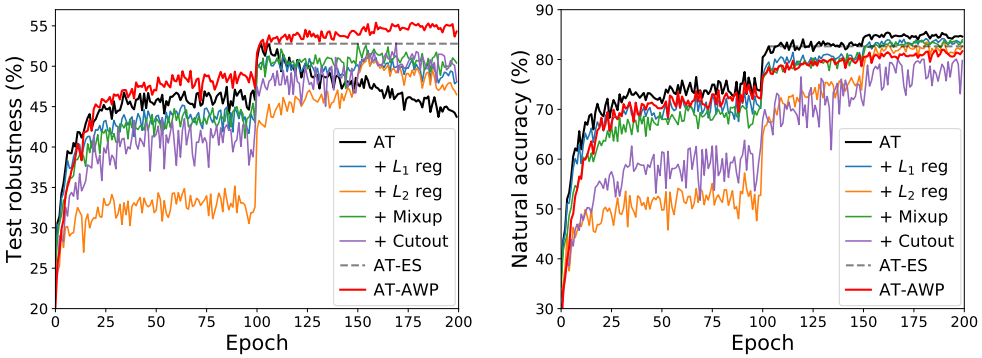
In addition, we find AWP sacrifices the natural accuracy throughout the whole training on CIFAR-10 under L_∞ threat model, which indicates AWP is still affected by the trade-off between robustness and natural accuracy [58]. For L_2 threat model on CIFAR-10, AWP just has similar natural accuracy to vanilla AT. This is because L_2 threat model ($\epsilon = 128/255$) is easier than L_∞ threat model and many techniques have the similar natural accuracy.

Table 4: Performance (%) of AT and AT with other regularization techniques on CIFAR-10 using PreAct ResNet-18 under L_∞ threat model ($\epsilon = 8/288$) over 5 random runs.

Method	Robustness		Natural accuracy	
	Best	Last	Best	Last
AT	52.79 ± 0.21	44.44 ± 0.39	$\mathbf{85.57 \pm 0.16}$	$\mathbf{84.56 \pm 0.19}$
+ L_1 regularization	51.95 ± 0.51	48.76 ± 0.61	82.77 ± 0.37	83.42 ± 0.26
+ L_2 regularization	51.60 ± 0.41	47.37 ± 0.52	81.05 ± 0.44	81.97 ± 0.50
+ Cutout	52.78 ± 0.14	50.37 ± 0.41	80.99 ± 0.19	83.46 ± 0.33
+ Mixup	52.87 ± 0.47	49.76 ± 0.81	78.76 ± 0.61	78.50 ± 1.21
AT-AWP	$\mathbf{55.39 \pm 0.39}$	$\mathbf{54.73 \pm 0.16}$	82.00 ± 0.19	81.11 ± 0.39

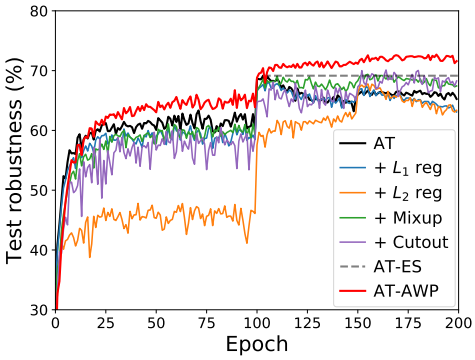
Table 5: Performance (%) of AT and AT with other regularization techniques on CIFAR-10 using PreAct ResNet-18 under L_2 threat model ($\epsilon = 128/255$) over 5 random runs.

Method	Robustness		Natural accuracy	
	Best	Last	Best	Last
AT	69.15 ± 0.13	65.93 ± 0.35	89.57 ± 0.09	88.96 ± 0.18
+ L_1 regularization	67.99 ± 0.27	63.75 ± 0.40	88.04 ± 0.19	88.49 ± 0.27
+ L_2 regularization	67.78 ± 0.33	63.62 ± 0.46	88.57 ± 0.14	87.75 ± 0.23
+ Cutout	69.38 ± 0.27	67.68 ± 0.30	88.36 ± 0.31	88.01 ± 0.26
+ Mixup	70.11 ± 0.50	68.20 ± 0.46	87.29 ± 0.71	86.91 ± 0.94
AT-AWP	$\mathbf{72.69 \pm 0.19}$	$\mathbf{72.08 \pm 0.39}$	$\mathbf{90.18 \pm 0.31}$	$\mathbf{89.71 \pm 0.17}$

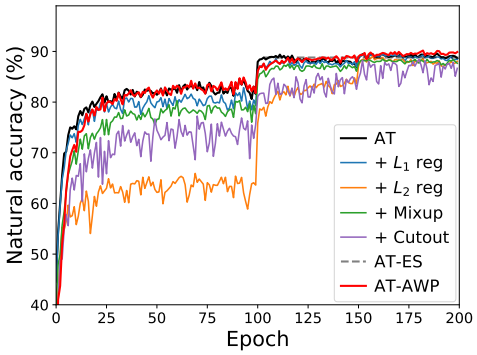


(a) Test robustness under ℓ_∞ threat model

(b) Natural accuracy under ℓ_∞ threat model



(c) Test robustness under ℓ_2 threat model



(d) Natural accuracy under ℓ_2 threat model

Figure 11: Performance (%) on the test set of CIFAR-10 during the training for AT, AT-AWP, and AT with other regularization techniques.