IN-TIME REFINING OPTIMIZATION TRAJECTORIES TOWARD IMPROVED ROBUST GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the fact that adversarial training has become the de facto method for improving robustness of deep neural networks, it is well-known that vanilla adversarial training suffers from daunting robust overfitting, resulting in unsatisfactory robust generalization. A number of approaches have been proposed to address these drawbacks such as extra regularization, adversarial weights perturbation, and training with more data over the last few years. However, the robust generalization improvement is yet far from satisfactory. In this paper, we approach this challenge with a brand new perspective – refining historical optimization trajectories. We propose a new method named **Weighted Optimization Trajectories (WOT)** that leverages the optimization trajectories of adversarial training in time. We have conducted extensive experiments to demonstrate the effectiveness of WOT under various state-of-the-art adversarial attacks. Our results show that WOT integrates seamlessly with the existing adversarial training methods and consistently overcomes the robust overfitting issue, resulting in better adversarial robustness. For example, WOT boosts the robust accuracy of AT-PGD under AA-L_∞ attack by $1.53\% \sim 6.11\%$ and meanwhile increases the clean accuracy by $0.55\%\sim5.47\%$ across SVHN, CIFAR-10, CIFAR-100, and Tiny-ImageNet datasets. Codes are included in the supplementary.

1 INTRODUCTION

Deep neural networks (DNNs) have achieved enormous breakthroughs in various fields, e.g., image classification (Hinton et al., 2012; He et al., 2016), speech recognition (Hinton et al., 2012), object detection (Girshick et al., 2014) and etc. However, it has been shown that they are vulnerable to adversarial examples, i.e., carefully crafted imperceptible perturbations on inputs can easily change the prediction of the model (Szegedy et al., 2013; Goodfellow et al., 2014). The vulnerability of DNNs hinders their applications in risk-sensitive tasks such as face recognition, autonomous driving and medical diagnostics. While various methods have been proposed to obtain robustness against adversarial perturbations, adversarial training (Madry et al., 2017b) is the most leading approach to achieve adversarial robustness.

However, the vanilla adversarial training usually suffers from daunting robust overfitting, resulting in poor robust generalization[1] (Rice et al., 2020). To tackle this issue, a number of methods from different perspectives have been proposed including but not limited to training with more data (Schmidt et al., 2018; Rebuffi et al., 2021; Sehwag et al., 2021; Carmon et al., 2019; Alayrac et al., 2019), adversarial weights perturbation (Wu et al., 2020b; Yu et al., 2021), and knowledge distillation and stochastic weights averaging (SWA) (Chen et al., 2020). Recently, Stutz et al. (2021) empirically show that the improved adversarial robustness can be attributed to the flatter loss landscape at the minima.

Although the generalization properties of SGD-based optimizer under standard training setting have been well studied (Zhang et al., 2017; Elisseeff et al., 2005; Zhou et al., 2018; Hardt et al., 2016), the corresponding robust generalization property under adversarial setting has not been fully explored. Among previous studies, Chen et al. (2020) heuristically adopts stochastic weight averaging (SWA) and average model weights along the optimization trajectory, which potentially mitigates robust

[1]Robust generalization refers to the gap between the adversarial accuracy of training set and test set, following previous work (Chen et al., 2020; Wu et al., 2020b; Stutz et al., 2021).

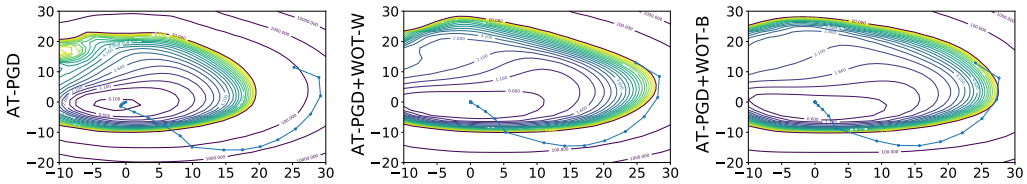


Figure 1: Visualization of loss contours and optimization trajectories for AT-PGD, AT-PGD+WOT-W, and AT-PGD+WOT-B, respectively. The experiments are conducted on CIFAR-10 with PreRN-18.

overfitting. However, it has been shown that naive weight averaging is not general enough to fundamentally address this problem, still prone to robust overfitting (Rebuffi et al., 2021). Instead of simply averaging weights, we propose a new approach - **Weighted Optimization Trajectories** (briefly **WOT)** for the first time showing that we can largely improve the flatness of solutions of existing adversarial training variants by periodically refining a set of historical optimization trajectories. Compared with the existing approaches, our method has three unique design contributions: ❶ our refinement is obtained by maximizing the robust accuracy on the unseen hold-out set, which is naturally advantageous to address the overfitting issue; ❷ our refinement is performed on a set of previous optimization trajectories rather than solely on previous weights; ❸ we further propose a block-wise trajectory refinement, which significantly enlarges the optimization space of refinement, leading to better robust performance. We conduct rigorous experiments to demonstrate the effectiveness of these design novelties in Section 4.1 as well as the ablation study in Section 4.3. Simple as it looks in Figure 1, the optimization trajectories after refining converge to a flatter loss valley compared to the vanilla AT-PGD, indicating the improved robust generalization (Wu et al., 2020b;a; Stutz et al., 2021).

Extensive experiments on different architectures and datasets show that WOT seamlessly mingles with the existing adversarial training methods with consistent robust accuracy improvement. For example, WOT-B directly boosts the robust accuracy over AT-PGD (early stops) under AA-L_∞ attack by 6.11%, 1.53%, 1.57%, and 4.38% on SVHN, CIFAR-10, CIFAR-100, and Tiny ImageNet, respectively; meanwhile improves the corresponding clean accuracy by $0.55\% \sim 5.47\%$. Moreover, we show that WOT can completely prevent robust overfitting across different attack approaches, including the strongest one off-the-shelf - AA-L_∞ attack.

2 Related Work

Adversarial Attacks. Adversarial examples were first illustrated in Szegedy et al. (2013). Following Szegedy et al. (2013), many adversarial attacks have been proposed and can be categorized into white-box and black-box attacks. White-box attacks have full access to the model when crafting adversarial examples. Popular white-box attacks are FGSM attack (Goodfellow et al., 2014), PGD attack (Madry et al., 2017a), Deepfool (Moosavi-Dezfooli et al., 2016) and C&W (Carlini & Wagner, 2017). Black-box attacks generate adversarial examples without any knowledge about the model. They are query-based attacks, e.g., SPSA attack (Uesato et al., 2018), Square attack (Andriushchenko et al., 2020), and transferability-based attacks, e.g., DIM (Xie et al., 2019), TIM (Dong et al., 2019) and DA attack (Huang et al., 2022). Recently, Croce & Hein (2020b) proposed Autoattack (AA) for reliable adversarial robustness evaluation which is an ensembled adversarial attacks containing white-box and black-box attacks. AA attack has been recognized as the most reliable method for evaluating model's adversarial robustness (Croce & Hein, 2020b) and will be used as the main evaluation method in this paper.

Adversarial Robustness. Many methods have been proposed to improve the model's robustness such as gradient regularization (Ross & Doshi-Velez, 2018), curvature regularization (Moosavi-Dezfooli et al., 2018), randomized smoothing (Cohen et al., 2019), local linearization (Qin et al., 2019), adversarial training methods (Goodfellow et al., 2014; Madry et al., 2017a; Zhang et al., 2019; Wu et al., 2020b; Wang et al., 2020; Zhang et al., 2020a; Huang et al., 2021; Balaji et al., 2019; Zhang et al., 2020b; Pang et al., 2022), and etc. Among all these methods, adversarial training has been the de facto method for achieving adversarial robustness. We briefly introduce four commonly used adversarial training methods that we use as baselines in this study.

Given a C-class dataset $S = \{(x_i, y_i)|x_i \in \mathbb{R}^d, y_i \in \mathbb{R}\}_{i=1}^n$, the cross-entropy loss $L(\cdot)$ and the DNN function $f_w : \mathbb{R}^d \to \mathbb{R}^C$.

AT-PGD (Madry et al., 2017a) is formalized as *min-max* optimization problem.

$$\min_w \rho^{AT}(w), \quad \rho^{AT}(w) = \frac{1}{n}\sum_i^n \{\max_{\|\Delta x\| \leq \epsilon} L(f_w(x_i + \Delta x), y_i)\},$$

where the *inner maximization* finds the adversarial examples and ϵ is the allowed perturbation magnitude. We use by default AT to denote AT-PGD in the following sections.

Trades (Zhang et al., 2019) separates training loss into cross-entropy loss (CE) and Kullback-Leibler ($\mathbf{KL}$) divergence loss to control clean accuracy and adversarial robustness respectively.

$$\rho^{TRADES}(w) = \frac{1}{n}\sum_i^n \{CE(f_w(x_i, y_i) + \beta \cdot \max_{\|\Delta x\| \leq \epsilon} \mathbf{KL}(f_w(x_i)||f_w(x_i + \Delta x)\}$$

MART (Wang et al., 2020) designs the training loss as the binary cross-entropy loss (BCE) and an explicit regularization for misclassified examples. Given that $[f_w(x_i)]_{y_i}$ denotes the y_i-th element of $f_w(x_i)$, the objective function is expressed as follows:

$$\rho^{MART}(w) = \frac{1}{n}\sum_i^n \{BCE(f_w(x_i + \Delta x, y_i) + \lambda \cdot \mathbf{KL}(f_w(x_i)||f_w(x_i + \Delta x)) \cdot (1 - [f_w(x_i)]_{y_i})\}$$

Adversarial Weights Perturbation (AWP) (Wu et al., 2020b) explicitly flattens the loss landscape by injecting the worst weight perturbations. Given that $\mathcal{V}$ is the feasible perturbation region for weights, the objective function is formalized as follows:

$$\rho^{AWP}(w) = \min_w \max_{v \in \mathcal{V}} \frac{1}{n}\sum_1^n \max_{\|\Delta x\| \leq \epsilon} CE(f_{w+v}(x_i + \Delta x), y_i)$$

In essence, AWP is equivalent to the combination of sharpness-aware optimizer (Foret et al., 2020) and adversarial training.

Robust Overfitting and its Mitigation. Rice et al. (2020) first identified the robust overfitting issue in AT that robust accuracy in test set degrades severely after the first learning rate decay and found that early stop is an effective strategy for mitigating the robust overfitting issue. Following Rice et al. (2020), several studies have been proposed to explain and mitigate the robust overfitting issue (Wu et al., 2020b; Singla et al., 2021; Chen et al., 2020; Dong et al., 2021; Chen et al., 2022; Stutz et al., 2021). Chen et al. (2020) showed that stochastic weight averaging (SWA) and knowledge distillation can mitigate the robust overfitting issue decently and Singla et al. (2021) found that low curvature activation function helps to mitigate the robust overfitting problem. Dong et al. (2021) took a step further to explain that robust overfitting issue may be caused by the memorization of hard samples in the final phase of training. Wu et al. (2020b); Yu et al. (2021); Stutz et al. (2021) demonstrated that a flat loss landscape improves robust generalization and reduces the robust overfitting problem, which is in line with the sharpness studies in standard training setting (Foret et al., 2020; Jiang et al., 2019; Dziugaite & Roy, 2017). Among these studies, SWA can be seen as a special case of our method and it equates our method under specific conditions (see more details in Appendix D).

3 METHODOLOGY

In this section, we will introduce **weighted optimization trajectories (WOT)**, a carefully designed method that refines the optimization trajectory of adversarial training towards a flatter region in the training loss landscape, to avoid robust overfitting. Specifically, WOT collects a set of historical optimization trajectories and further learns a weighted combination of them explicitly on the unseen set. The sketch map of WOT is shown in Figure 2. Concretely, WOT contains two steps: (1) collect optimization trajectories of adversarial training. (2) re-weight collected optimization trajectories and optimize weights according to the robust loss on an unseen set. Two unanswered problems of this process are how to collect optimization trajectories and how to construct the objective function of optimizing weights. We give detailed solutions as follows.

3.1 WOT: OPTIMIZATION TRAJECTORIES

We denote optimization trajectories as the consecutively series status of weights in weight space after n steps optimization. Formally, given a deep neural network f with the parameter $w \in \mathcal{W}$. n steps of optimization trajectories of adversarial training are denoted as $\{w^1, w^2 ... w^i, ..., w^n\}$ where w^i is the weight after i-th optimization step. This process can also be simplified as follows: $\{w^1, \Delta w^1 ... \Delta w^i, ..., \Delta w^{n-1}\}$ where $\Delta w^i = w^{i+1} - w^i$. In practice, it is time-consuming and space-consuming to collect the weights of each batch optimization step and it is also not necessary to collect the weights at a high frequency (see details in Figure 5). Therefore, we propose to collect weights for every m batch optimization step and the collected trajectories with n optimization steps are re-denoted as follows:

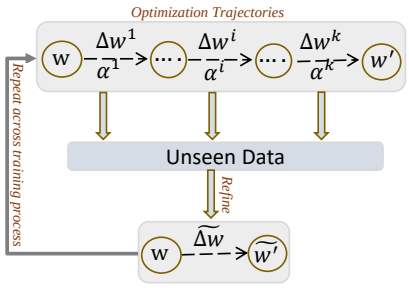


Figure 2: Sketch map of WOT.

$$\Delta W = \{w^1, \Delta w^1, ..., \Delta w^i, ..., \Delta w^k\}, \tag{1}$$

where $k = \frac{n}{m}$. For brevity, we call m the Gaps that controls the length between two consecutively collections and k the number of Gaps that controls the number of weights that are collected.

3.2 WOT: OBJECTIVE FUNCTION

We design the objective function based on historical optimization trajectories of model training. From the description of optimization trajectories introduced above, the weights w' with n batch optimization steps from w can be written as $w' = w + \Delta w^1 + ... + \Delta w^i + .. + \Delta w^k$. Since WOT refines the optimization trajectories by re-weighting them, the new model weights $\widetilde{w'}$ after refining optimization trajectories can be expressed as follows:

$$\widetilde{w'} = w + \widetilde{\Delta w}, \quad \widetilde{\Delta w} = \alpha^1 \Delta w^1 + ... + \alpha^i \Delta w^i + ... + \alpha^k \Delta w^k, \tag{2}$$

where $\alpha^1, ..., \alpha^i, ..., \alpha^k$ are optimizable variables. Considering that we expect to find the model with better robust generalization via optimizing α, a straightforward idea is to optimize α^i with respect to improving its robust performance on a small unseen set. The philosophy for this idea is that if a model can generalize robustness better to an unseen set, it would probably generalize robustness well to the target unseen set.

Formally, the **objective function** of optimizing α^i is defined as follows:

$$\min_{0 \le \alpha^i \le 1} \max_{\|\Delta x_{uns}\| \le \epsilon} L(f_{w + \widetilde{\Delta w}}(x_{uns} + \Delta x_{uns}), y_{uns}), \tag{3}$$

where (x_{uns}, y_{uns}) is from an unseen set and Δx_{uns} is the corresponding adversarial perturbations. We constrain α^i to [0,1] (see Appendix G for the results of other constraints for α^i).

Update α^i. α^i can be optimized by any SGD-based optimizers according to the objective function (Eq. 3) described above. In this study, we update α^i by SGD optimizer with momentum buffer.

$$m^t = m^{t-1} \cdot \gamma + \nabla_{\alpha^i} L(f_{w^{i-1} + \widetilde{\Delta w}}(x_{uns} + \Delta x_{uns}), y_{uns}) \tag{4}$$

$$\alpha^i = \alpha^i - lr \cdot m^t, \tag{5}$$

where m^t is the momentum buffer of α^i at the t-th step and lr is the learning rate.

3.3 WOT: IN-TIME REFINING OPTIMIZATION TRAJECTORIES

WOT reconstructs a set of historical optimization trajectories in time during the course of training on unseen set to avoid overfitting. A naive strategy is treating each optimization trajectory w^i as a whole and learning an individual weight for each trajectory shown as Eq. 2. This simple method naturally has limited learning space for refinement especially when we only have few historical trajectories. To improve the learning space of WOT, we further propose blockwise WOT which breaks down

Table 1: Robust accuracy of WOT under multiple adversarial attacks with various adversarial training variants. The experiments are conducted on CIFAR-10 with the PreRN-18 architecture. The best results are marked in bold.

MODELS	FGSM	PGD-20	PGD-100	CW_∞	AA-L_∞
AT+EARLY STOP	57.30	52.90	51.90	50.90	47.43
AT+SWA	58.89	53.02	51.86	52.32	48.61
AT+WOT-W (OURS)	58.50	53.19	51.90	51.74	48.36
AT+WOT-B (OURS)	**59.67**	**54.85**	**53.77**	**52.56**	**48.96**
TRADES	58.16	53.14	52.17	51.24	48.90
TRADES+SWA	58.07	53.17	52.22	50.91	49.07
TRADES+WOT-W (OURS)	**58.95**	**54.07**	**53.29**	51.74	49.95
TRADES+WOT-B (OURS)	58.50	53.73	52.95	**52.12**	**50.19**
MART	59.93	54.07	52.30	50.16	47.01
MART+SWA	58.19	54.21	53.56	49.39	46.86
MART+WOT-W (OURS)	58.13	53.79	52.66	50.24	47.43
MART+WOT-B (OURS)	**59.95**	**55.13**	**54.09**	**50.56**	**47.49**
AT+AWP	59.11	55.45	54.88	52.50	49.65
AT+AWP+SWA	58.23	55.54	54.91	51.88	49.39
AT+AWP+WOT-W (OURS)	59.05	**55.95**	54.96	52.70	49.84
AT+AWP+WOT-B (OURS)	**59.26**	55.69	**55.09**	**52.82**	**50.00**

each trajectory into multiple blocks based on the original block designing of the model itself. For convenience, we dot these two methods WOT-W and WOT-B, respectively.

WOT-W takes one trajectory as whole and assigns a single α for each trajectory. Hence the number of α that need to be learned exactly equals to the number of Gaps:k.

WOT-B in contrast learns a vector of α whose length is determined by the number of model's block. Therefore, Eq. 2 can be reformulated as:

$$\widetilde{\Delta w} = \begin{bmatrix} \widetilde{\Delta w_1} \\ \cdots \\ \widetilde{\Delta w_j} \\ \cdots \\ \widetilde{\Delta w_t} \end{bmatrix}, \quad \widetilde{\Delta w_j} = \alpha_j^1 \Delta w_j^1 + \alpha_j^2 \Delta w_j^2 + ... + \alpha_j^k \Delta w_j^k \tag{6}$$

where j denotes the j-th block. Optimizing α for blockwise WOT is exactly the same as the description in Eq. 4 and Eq. 5.

The main difference between WOT-W and WOT-B is that WOT-W learns one α for each cached Δw (the difference of parameters in two checkpoints) whereas WOT-B further breakdowns each cached Δw into several blocks (each block contains several layers depending on the specific architectures as explained in detail in Appendix B) and learns an individual α value for each block. Therefore, the learning space of WOT-B is larger than WOT-W, leading to better performance in general. The pseudocode of WOT can be found in Appendix C.

4 EXPERIMENTS

We perform extensive experiments to show the effectiveness of our method in improving adversarial robustness as well as addressing the robust overfitting issue.

Datasets. Four datasets are considered in our experiments: CIFAR-10, CIFAR-100 (Krizhevsky et al., 2010), Tiny-ImageNet (Deng et al., 2009) and SVHN (Netzer et al., 2011). For experiments of WOT, we randomly split 1000 samples from the original CIFAR-10 training set, 10000 samples from Tiny-ImageNet, and 2000 samples from the original CIFAR-100 and SVHN training set as the unseen hold-out sets.

Baselines. Five baselines are included: AT (Rice et al., 2020), Trades (Zhang et al., 2019), AWP+AT (Wu et al., 2020b), MART (Wang et al., 2020) and SWA (Chen et al., 2020). Three architectures including VGG-16 (Simonyan & Zisserman, 2014), PreActResNet-18 (PreRN-18) (He et al., 2016), WideResNet-34-10 (WRN-34-10) (Zagoruyko & Komodakis, 2016).

Experimental Setting. For WOT, we adopt an SGD optimizer with a momentum of 0.9, weight decay of 5e-4 and a total epoch of 200 with a batch size of 128 following Rice et al. (2020). By default, we start to refine optimization trajectories after 100 epochs. For WOT-B, we set each block in PreRN-18 and WRN-34-10 architectures as the independent weight space. We set the layers with the same width as a group and set each group as an independent block for VGG-16 (see details in Appendix B). We by default set the gaps m to 400, the number of gaps k to 4 and initialize α as zero. For all baselines, we use the training setups and hyperparameters exactly the same as their papers (see details in Appendix A).

Evaluation Setting. We use AA attack (Croce & Hein, 2020b) as our main adversarial robustness evaluation method. AA attack is a parameter-free ensembled adversarial attack which contains three white-box attacks: APGD-CE (Croce & Hein, 2020b), APGD-T (Croce & Hein, 2020b), FAB-T (Croce & Hein, 2020a) and one black-box attack: Square attack (Andriushchenko et al., 2020). To the best of our knowledge, AA attack is currently the most reliable adversarial attack for evaluating adversarial robustness. We also adopt three other commonly used white-box adversarial attacks: FGSM (Goodfellow et al., 2014), PGD-20/100 (Madry et al., 2017b) and C&W$_\infty$ attack (Carlini & Wagner, 2017). Besides, we also report the performance of query-based SPSA black-box attack (Uesato et al., 2018) (100 iterations with a learning rate of 0.01 and 256 samples for each gradient estimation). By default, we report the mean of three random runs for all experiments of our method and omit the standard deviation since it is very small ($\leq 0.3\%$). We by default set $\epsilon = 8/255$ for L_∞ version adversarial attack and $\epsilon = 64/255$ for L_2 version adversarial attack.

Table 2: Test robustness under multiple adversarial attacks based on VGG-16/WRN-34-10 architectures. The experiments are conducted on CIFAR-10 with AT and Trades. The bold denotes the best performance.

ARCHITECTURE	METHOD	CW$_\infty$	PGD-20	PGD-100	AA-L_∞
VGG16	AT+EARLY STOP	46.87	49.95	46.87	43.63
VGG16	AT+SWA	47.01	49.58	49.13	43.89
VGG16	AT+WOT-W(OURS)	47.42	49.96	49.36	44.01
VGG16	AT+WOT-B(OURS)	**47.52**	**50.28**	**49.58**	**44.10**
VGG16	TRADES	45.47	48.24	47.54	43.64
VGG16	TRADES+SWA	45.92	48.64	47.86	44.12
VGG16	TRADES+WOT-W(OURS)	**46.75**	**49.19**	**48.28**	**44.82**
VGG16	TRADES+WOT-B(OURS)	46.21	48.81	47.85	44.17
WRN-34-10	AT+EARLY STOP	53.82	55.06	53.96	51.77
WRN-34-10	AT+SWA	**88.45**	55.34	53.61	52.25
WRN-34-10	AT+WOT-W(OURS)	56.05	58.21	57.11	52.88
WRN-34-10	AT+WOT-B(OURS)	**57.13**	**60.15**	**59.38**	**53.89**
WRN-34-10	TRADES	54.20	56.33	56.07	53.08
WRN-34-10	TRADES+SWA	54.55	54.95	53.08	51.43
WRN-34-10	TRADES+WOT-W(OURS)	56.10	57.56	56.20	53.68
WRN-34-10	TRADES+WOT-B(OURS)	**56.62**	**57.92**	**56.80**	**54.33**

4.1 SUPERIOR PERFORMANCE IN IMPROVING ADVERSARIAL ROBUSTNESS

Table 3: Test robustness under AA-L_2 and AA-L_∞ attacks across various datasets. The experiments are based on PreRN-18 and AT. The bold denotes the best performance.

ATTACK	METHOD	SVHN		CIFAR-10		CIFAR-100		TINY-IMAGENET	
		CLEAN	ROBUST	CLEAN	ROBUST	CLEAN	ROBUST	CLEAN	ROBUST
L_∞	AT+EARLY STOP	89.05	45.72	81.72	47.43	53.84	23.69	42.76	14.39
L_∞	AT+SWA	90.36	40.24	**85.23**	48.61	**58.51**	23.90	49.19	17.94
L_∞	AT+WOT-W(OURS)	**93.25**	50.42	84.47	48.36	55.07	24.41	**49.31**	17.10
L_∞	AT+WOT-B(OURS)	92.95	**51.83**	83.84	**48.96**	54.39	**25.26**	48.83	**18.77**
L_2	AT+EARLY STOP	89.05	72.13	81.72	71.30	53.84	42.75	42.76	36.61
L_2	AT+SWA	90.36	67.76	**85.23**	73.28	**58.51**	43.10	49.19	42.40
L_2	AT+WOT-W(OURS)	**93.25**	72.75	84.47	73.20	55.07	**43.88**	**49.31**	42.43
L_2	AT+WOT-B(OURS)	92.95	**72.80**	83.84	**73.39**	54.39	43.32	48.83	**42.54**

We evaluate the effectiveness of WOT in improving adversarial robustness across AT and three of its variants, four popular used datasets, i.e., SVHN, CIFAR-10, CIFAR-100 and Tiny-ImageNet, and three architectures, i.e., VGG16, PreRN-18, and WRN-34-10.

WOT consistently improves the adversarial robustness of all adversarial training variants. In Table 1, we applied WOT-B and WOT-W to AT+early stop, Trades, MART, AWP variants and compare them with their counterpart baselines. Besides, we add the combination of SWA and these adversarial training variants as one of the baselines. The results show: ① WOT consistently improves adversarial robustness among the four adversarial training variants under both weak attacks, e.g. FGSM, PGD-20, and strong attacks, e.g., C&W$_\infty$, AA-L_∞ attacks. ② WOT-B as the WOT variant confirms our hypothesis and consistently performs better than WOT-W. WOT-B improves the robust accuracy over their counterpart baselines by $0.35\% \sim 1.53\%$ under AA-L_∞ attack. ③ WOT boosts robust accuracy with a larger margin on AT and Trades than MART and AWP under AA-L_∞ attack. One reason might be that MART and AWP themselves enjoy good ability in mitigating robust overfitting (Stutz et al., 2021; Wu et al., 2020b), leading to the less space for WOT to further boost the performance.

WOT can generalize to different architectures and datasets. Table 2 and Table 3 show that WOT consistently outperforms the counterpart baseline under AA-L_∞ attack, which indicates that the effectiveness of WOT generalizes well to different architectures and datasets. In Table 2, WOT boosts robust accuracy by $0.47\% \sim 2.12\%$ on VGG16 and WRN-34-10 architectures. In Table 3, WOT improves robust accuracy with $1.53\% \sim 6.11\%$ among SVHN, CIFAR-10, CIFAR-100 and Tiny-ImageNet under AA-L_∞ attack. Besides, the success of WOT can also be extended to AA-L_2 attack with the improvement by $0.67\% \sim 5.93\%$.

Excluding Obfuscated Gradients. Athalye et al. (2018) claims that obfuscated gradients can also lead to the "counterfeit" of improved robust accuracy under gradients-based white-box attacks. To exclude this possibility, we report the performance of different checkpoints under transfer attack and SPSA black-box attack over epochs. In Figure 3, the left figure shows robust accuracy of the unseen robust model on the adversarial examples generated by the PreRN-18 model trained by AT, AT+WOT-B, AT+WOT-W respectively with PGD-10 attack on CIFAR-10. A higher robust accuracy on the unseen robust model corresponds to a weaker attack. It can be seen that both AT+WOT-B and AT+WOT-W generate more transferable adversarial examples than AT. Similarly, the middle figure shows the robust accuracy of the PreRN-18 model trained by AT, AT+WOT-B, AT+WOT-W on the adversarial examples generated by the unseen robust model. It can be seen that AT+WOT-B, AT+WOT-W can better defend the adversarial examples from the unseen model. What's more, in the right figure, we observe again that both AT+WOT-B and AT+WOT-W outperform AT under SPSA black-box attack over different checkpoints during training. All these empirical results sufficiently suggest that the improved robust accuracy of WOT is not caused by obfuscated gradients.

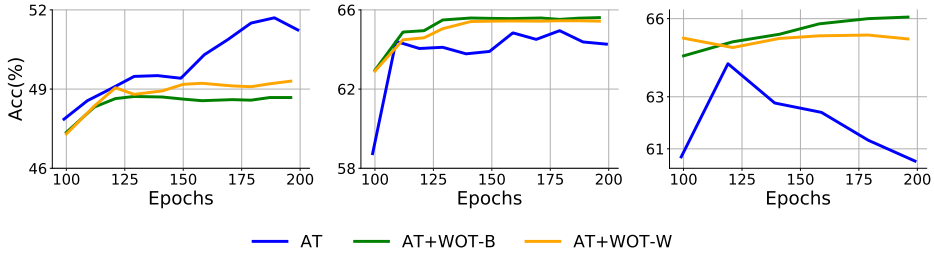


Figure 3: Robust accuracy under black-box attack over epochs. **(Left)** Robust accuracy on the unseen robust model transfer attacked from checkpoints of AT, AT+WOT-W/B. **(Middle)** Robust accuracy on checkpoints of AT, AT+WOT-W/B transfer attacked from the unseen model. **(Right)** Robust accuracy on checkpoints of AT, AT+WOT-W/B under SPSA black-box attack. The experiments are conducted on PreRN-18 and CIFAR-10. The unseen robust model is WRN-34-10 trained by AT.

4.2 Ability to Prevent Robust Overfitting

We report the robust accuracy under AA-L_∞ attack for the best checkpoint and the last checkpoint based on PreRN-18 and WRN-34-10 architectures on CIFAR-10 (Table 4). Besides, we show the robust accuracy curve under PGD-10 attack on different checkpoints over epochs (Figure 4).

In Figure 4, the third and fourth figures show that after the first learning rate decay (at 100 epoch), there is a large robust accuracy drop for AT between the best checkpoint and the last checkpoint on both PreRN-18 and WRN-34-10 architectures. In comparison, there is completely no robust accuracy

drop for AT+WOT-W/B between the best checkpoint and the last checkpoint on both PreRN-18 and WRN-34-10 architectures. In Table 4, we further show the evidence that there is no robust accuracy drop for AT+WOT-B/W under stronger attack, i.e., AA-L_∞ attack. From the first and second figures of Figure 4, we observe that the mean of α decreases to a very small value after 150,100 epochs for PreRN-18, WRN-34-10 respectively. The small mean of α indicates that WOT stops the model's weights updating with unexpected magnitudes, which prevents the occurrence of robust overfitting.

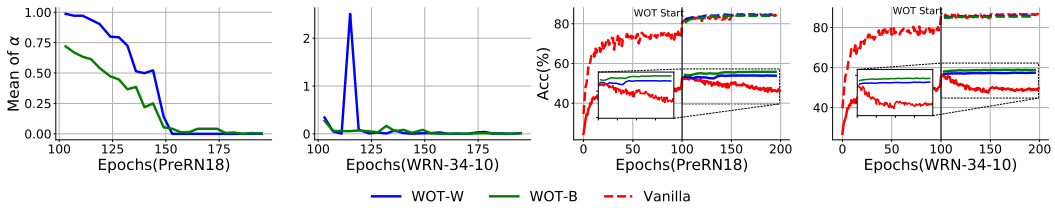


Figure 4: Mean value of α and results of test robust/clean accuracy over epochs. The experiments are conducted on CIFAR-10 with PreRN-18 based on AT.

Table 4: Test robustness under AA-L_∞ attack to show the robust overfitting issue in AT and the effectiveness of WOT in overcoming it. The difference between the best and final checkpoints indicates performance degradation during training and the best checkpoint is chosen by PGD-10 attack on the validation set. The experiments are conducted on CIFAR-10 with PreRN-18/WRN-34-10 architectures.

Architectures	Method	Robust Accuracy(RA)			Standard Accuracy(SA)		
		Best	Final	Diff.	Best	Final	Diff.
PreRN-18	AT	48.02	42.48	-5.54	81.33	84.40	+3.07
PreRN-18	AT+SWA	48.93	48.61	-0.32	84.19	85.23	+1.04
PreRN-18	AT+WOT-W(Ours)	48.04	48.36	+0.32	84.05	84.47	-0.42
PreRN-18	AT+WOT-B(Ours)	48.90	48.96	+0.06	83.84	83.83	-0.01
WRN-34-10	AT	51.77	46.78	-4.99	85.74	86.34	+0.6
WRN-34-10	AT+SWA	53.38	52.25	-1.13	87.14	88.45	+1.31
WRN-34-10	AT+WOT-W(Ours)	52.84	52.88	+0.04	84.83	84.88	+0.05
WRN-34-10	AT+WOT-B(Ours)	52.23	53.89	+1.66	83.46	85.50	+2.04

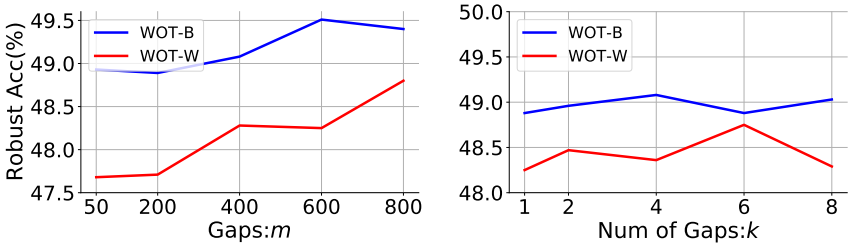


Figure 5: The impact of gaps m and the number of gaps k on robust accuracy under AA-L_∞ attack. The experiments are conducted on CIFAR-10 with PreRN-18 based on AT. k is fixed to 4 for the left figure and m is fixed to 400 for the right figure.

4.3 Ablations and Visualizations

In this section, we first conduct ablation studies to show the effectiveness of the designed optimization trajectories and the unseen hold-out set in WOT. Then we investigate the impact of gaps:m and the number of gaps:k, the effect of WOT on the loss landscapes w.r.t weight space and the visualization of α for blocks. The results are shown in Table 9, Figure 5, and Figure 6. All experiments in the two figures are conducted on CIFAR-10 with PreRN-18 based on AT except for Figure 6 where Trades is also included. The robust accuracy is evaluated under AA-L_∞ attack for all the three figures.

Ablation studies To demonstrate the effectiveness of the designed optimization trajectories and the unseen hold-out set in boosting adversarial robustness, we designed the following baselines: 1) Keep the same unseen hold-out set and training strategy with WOT, but optimize model weights

instead of α on the unseen hold-out set (Abbreviated as "B1"); 2) Keep the same unseen hold-out set and optimize the hyperparameter of SWA by the hold-out set (Abbreviated as "B2"); 3) Replace the unseen hold-out set with a seen set, i.e. keep the same number of samples from training set (Abbreviated as "B3"). Results in Table 5 show that ① AT+WOT-W/B outperforms AT+B1 and AT+B2, indicating the designed optimization trajectories play key roles in WOT. ② AT+WOT-W/B outperforms AT+B3 with large margin, indicating the unseen hold-out set is crucial for WOT.

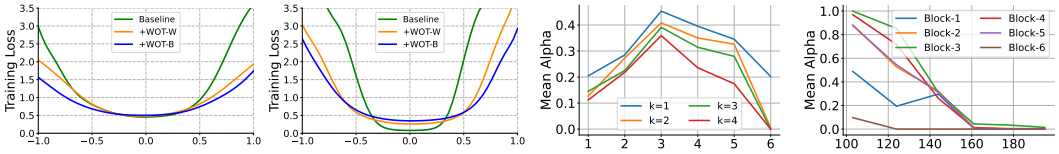
Table 5: Robust Accuracy of ablation experiments on CIFAR-10 with PreRN-18.

METHODS	PGD-20	PGD-100	CW_∞	AA-L_∞
AT+B1	49.68	47.44	49.04	45.26
AT+B2	52.74	51.28	51.31	48.22
AT+WOT-B+B3 (M=400,K=4)	47.14	44.23	43.87	41.02
AT+WOT-W (M=400,K=4)	53.19	51.90	51.74	48.36
AT+WOT-B (M=400,K=4)	54.85	53.77	52.56	48.96

Impact of m and k. Figure 5 shows the impact of gaps m and number of gaps k on robust accuracy under AA-L_∞ attack. In the left figure, we observe that robust accuracy increases with an increase of m. Besides, we find that WOT-W is more sensitive to m than WOT-B. The right figure shows that both WOT-W and WOT-B are not sensitive to the number of gaps k.

Averaged α for Blocks. To shed insights on why WOT-B outperforms WOT-W, we plot the learned α for each block. Experiments are conducted on CIFAR-10 with PreRN-18 based on WOT-B (K=4, m=400). Results in Figure 6c and Figure 6d show that the magnitude of learned α are different among blocks. Specifically, WOT-B assigns large value of α for middle blocks, i.e., Block-2,3,4,5 and small value of α for bottom and top blocks, i.e., Block-1,6. This indicates that assigning different weights for different blocks may play an crucial role in boost adversarial robustness.

Visualizing loss landscape. We expect WOT to search flatter minima for adversarial training to boost its robust generalization. We demonstrate that it indeed happens via visualizing the loss landscape with respect to weight space (Figure 6a and Figure 6b). Figure 6a and Figure 6b show that WOT+baseline obtains flatter minima than baseline, which indicates an improved robust generalization (Stutz et al., 2021; Wu et al., 2020b). The visualization of the loss landscape with respect to input space can be found in Appendix J.



(a) Trades (b) AT (c) Blocks (X-axis) (d) Epochs (X-axis)

Figure 6: Loss landscape w.r.t weight space (Figure 6a and Figure 6b). z axis denotes the loss value. We plot the loss landscape following the setting in Wu et al. (2020b). The averaged α by averaging along training process (Figure 6c). The k-averaged α during training process. (Figure 6d). The experiments are conducted on CIFAR-10 with PreRN-18.

5 CONCLUSION

In this paper, we propose a new method named weighted optimization trajectories (WOT) for improving adversarial robustness and avoiding robust overfitting. We re-weight the optimization trajectories in time by maximizing the robust performance on an unseen hold-out set during the training process. The comprehensive experiments demonstrate: (1) WOT can effectively improve adversarial robustness across various adversarial training variants, model architectures and benchmark datasets. (2) WOT enjoys superior performance in mitigating robust overfitting. Moreover, visualizing analysis validates that WOT flattens the loss landscape with respect to input and weight space, showing an improved robust generalization.

REPRODUCIBILITY STATEMENT

The implementation code is public (in the supplementary) and our results can be reproducible.

ETHIC STATEMENT

This work does not present any foreseeable negative societal impacts.

REFERENCES

Jean-Baptiste Alayrac, Jonathan Uesato, Po-Sen Huang, Alhussein Fawzi, Robert Stanforth, and Pushmeet Kohli. Are labels required for improving adversarial robustness? *Advances in Neural Information Processing Systems*, 32, 2019.

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision*, pp. 484–501. Springer, 2020.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Yogesh Balaji, Tom Goldstein, and Judy Hoffman. Instance adaptive adversarial training: Improved accuracy tradeoffs in neural nets. *arXiv preprint arXiv:1910.08051*, 2019.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57. IEEE, 2017.

Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. *Advances in Neural Information Processing Systems*, 32, 2019.

Tianlong Chen, Zhenyu Zhang, Sijia Liu, Shiyu Chang, and Zhangyang Wang. Robust overfitting may be mitigated by properly learned smoothening. In *International Conference on Learning Representations*, 2020.

Tianlong Chen, Zhenyu Zhang, Pengjun Wang, Santosh Balachandra, Haoyu Ma, Zehao Wang, and Zhangyang Wang. Sparsity winning twice: Better robust generaliztion from more efficient training. *arXiv preprint arXiv:2202.09844*, 2022.

Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. *arXiv preprint arXiv:1902.02918*, 2019.

Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pp. 2196–2205. PMLR, 2020a.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pp. 2206–2216. PMLR, 2020b.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4312–4321, 2019.

Yinpeng Dong, Ke Xu, Xiao Yang, Tianyu Pang, Zhijie Deng, Hang Su, and Jun Zhu. Exploring memorization in adversarial training. *arXiv preprint arXiv:2106.01606*, 2021.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

Andre Elisseeff, Theodoros Evgeniou, Massimiliano Pontil, and Leslie Pack Kaelbing. Stability of randomized learning algorithms. *Journal of Machine Learning Research*, 6(1), 2005.

Logan Engstrom, Andrew Ilyas, and Anish Athalye. Evaluating and understanding the robustness of adversarial logit pairing. *arXiv preprint arXiv:1807.10272*, 2018.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014. ISBN 9781479951178. doi: 10.1109/CVPR.2014.81.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. dec 2014. URL http://arxiv.org/abs/1412.6572.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International conference on machine learning*, pp. 1225–1234. PMLR, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, and Brian Kingsbury. Deep Neural Networks for Acoustic Modeling in Speech Recognition. *Ieee Signal Processing Magazine*, 2012. ISSN 1053-5888. doi: 10.1109/MSP.2012.2205597.

Tianjin Huang, Vlado Menkovski, Yulong Pei, and Mykola Pechenizkiy. calibrated adversarial training. In *Asian Conference on Machine Learning*, pp. 626–641. PMLR, 2021.

Tianjin Huang, Vlado Menkovski, Yulong Pei, Yuhao Wang, and Mykola Pechenizkiy. Direction-aggregated attack for transferable adversarial examples. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 18(3):1–22, 2022.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. *arXiv preprint arXiv:1912.02178*, 2019.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). *URL http://www. cs. toronto. edu/kriz/cifar. html*, 5, 2010.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. jun 2017a. URL http://arxiv.org/abs/1706.06083.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017b.

Seyed Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pp. 2574–2582. IEEE Computer Society, dec 2016. ISBN 9781467388504. doi: 10.1109/CVPR.2016.282.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. nov 2018. URL http://arxiv.org/abs/1811.09716.

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

Tianyu Pang, Min Lin, Xiao Yang, Jun Zhu, and Shuicheng Yan. Robustness and accuracy could be reconcilable by (proper) definition. *arXiv preprint arXiv:2202.10103*, 2022.

Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy, Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial Robustness through Local Linearization. jul 2019. URL http://arxiv.org/abs/1907.02610.

Sylvestre-Alvise Rebuffi, Sven Gowal, Dan A Calian, Florian Stimberg, Olivia Wiles, and Timothy Mann. Fixing data augmentation to improve adversarial robustness. *arXiv preprint arXiv:2103.01946*, 2021.

Minsoo Rhu, Natalia Gimelshein, Jason Clemons, Arslan Zulfiqar, and Stephen W Keckler. vdnn: Virtualized deep neural networks for scalable, memory-efficient neural network design. In *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 1–13. IEEE, 2016.

Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pp. 8093–8104. PMLR, 2020.

Andrew Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems*, pp. 5014–5026, 2018.

Vikash Sehwag, Saeed Mahloujifar, Tinashe Handina, Sihui Dai, Chong Xiang, Mung Chiang, and Prateek Mittal. Robust learning meets generative models: Can proxy distributions improve adversarial robustness? *arXiv preprint arXiv:2104.09425*, 2021.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Vasu Singla, Sahil Singla, Soheil Feizi, and David Jacobs. Low curvature activations reduce overfitting in adversarial training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16423–16433, 2021.

David Stutz, Matthias Hein, and Bernt Schiele. Relating adversarially robust generalization to flat minima. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7807–7817, 2021.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. dec 2013. URL http://arxiv.org/abs/1312.6199.

Jonathan Uesato, Brendan O'donoghue, Pushmeet Kohli, and Aaron Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning*, pp. 5025–5034. PMLR, 2018.

Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*, 2020.

Dongxian Wu, Yisen Wang, and Shu-tao Xia. Revisiting loss landscape for adversarial robustness. *arXiv preprint arXiv:2004.05884*, 2020a.

Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33, 2020b.

Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2730–2739, 2019.

Chaojian Yu, Bo Han, Mingming Gong, Li Shen, Shiming Ge, Bo Du, and Tongliang Liu. Robust weight perturbation for adversarial training. 2021.

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

Chiyuan Zhang, Qianli Liao, Alexander Rakhlin, Karthik Sridharan, Brando Miranda, Noah Golowich, and Tomaso Poggio. Musings on deep learning: Properties of sgd. Technical report, Center for Brains, Minds and Machines (CBMM), 2017.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pp. 7472–7482. PMLR, 2019.

Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In *International conference on machine learning*, pp. 11278–11287. PMLR, 2020a.

Jingfeng Zhang, Jianing Zhu, Gang Niu, Bo Han, Masashi Sugiyama, and Mohan Kankanhalli. Geometry-aware instance-reweighted adversarial training. *arXiv preprint arXiv:2010.01736*, 2020b.

Yi Zhou, Yingbin Liang, and Huishuai Zhang. Generalization error bounds with probabilistic guarantee for sgd in nonconvex optimization. *arXiv preprint arXiv:1802.06903*, 2018.

A IMPLEMENTATIONS DETAILS

The implementation of WOT is based on PyTorch library and all experiments in this study are run in single A100 GPU. Following Chen et al. (2020), SWA starts after the first learning decay. All models used in this study are trained with $\epsilon = 8/255$.

All baselines are achieved by running the code provided by their authors (Table 6).

Table 6: Download Links for baseline codes.

METHODS	DOWNLOAD LINKS
AT+EARLY STOP	HTTPS://GITHUB.COM/LOCUSLAB/ROBUST_OVERFITTING.GIT
TRADES	HTTPS://GITHUB.COM/YAODONGYU/TRADES.GIT
MART	HTTPS://GITHUB.COM/YISENWANG/MART.GIT
AWP	HTTPS://GITHUB.COM/CSDONGXIAN/AWP.GIT
AT+SWA	HTTPS://GITHUB.COM/VITA-GROUP/ALLEVIATE-ROBUST-OVERFITTING.GIT
FAT	HTTPS://GITHUB.COM/ZJFHEART/FRIENDLY-ADVERSARIAL-TRAINING.GIT

B BLOCK DETAILS FOR WOT-B

We naturally divide ResNet kinds of architectures according to their original block design. For VGG-16, we group the layers with the same number of channels as a block. We share the details in Table 7.

Table 7: The corresponding relationship between layers and blocks for PreActResNet-18, WRN-34-10 and VGG-16.

PREACTRESNET-18		WRN-34-10		VGG-16	
LAYERS	BLOCK INDEX	LAYERS	BLOCK INDEX	LAYERS	BLOCK INDEX
1	1	1	1	1-2	1
2-5	2	2-12	2	3-4	2
6-9	3	13-23	3	5-7	3
10-13	4	24-34	4	8-13	4
14-17	5	FINAL LINEAR LAYER	5	13-16	5
FINAL LINEAR LAYER	6	-	-	-	-

C PSEUDOCODE

The Psedocode can be found in Algorithm 1.

Algorithm 1 WOT for AT-PGD

Input: A model $f_w(x)$ with parameters w, training set $(x, y) \in S$, unseen set $(x_{uns}, y_{uns}) \in S_{uns}$, total training steps T, WOT starting epoch:p, Gaps: m, Number of Gaps:k, and adversarial training loss $L(\cdot)$.
Output: $f_w(x)$ with trained parameters.
initialize model weights w, $\Delta W = \emptyset$.
for $t = 1$ **to** T **do**
  # Normal adversarial training for f_w.
  $w_{(t)} = w_{(t-1)} - lr \cdot \nabla_w L(f_{w_{(t-1)}}(x + \Delta x), y)$
  # WOT starts after p epochs (steps).
  if $t > p$ **then**
    if $(t - p)\%m$=0 **then**
      #Collect optimization trajectories.
      $\Delta w = w_{(t)} - w_{(t-m)}$
      $\Delta W = \Delta W \cup \{\Delta w\}$
    end if
    if $(t - p)\%(m * k)$=0 **then**
      #Optimize α based on Eq. 4 and Eq. 5
      Initialize α with zero.
      for $i = 1$ **to** n **do**
        $m^i = m^{i-1} \cdot \gamma + \nabla_\alpha L(f_{\widetilde{w'}}(x_{uns} + \Delta x_{uns}), y_{uns})$
        $\alpha = \alpha - lr \cdot m^i$
      end for
      $w_{(t)} = w_{(t-mk)} + \widetilde{\Delta w}$
      $\Delta W = \emptyset$
    end if
  end if
end for
Return $f_w(x)$

D SWA: A SPECIAL CASE OF WOT

Given a deep model f_w with parameters w. The parameters of f are denoted as w_0, w_1, w_2,...,w_n at T_0, T_1, T_2,..,T_n steps respectively. The gradients of the training loss w.r.t w are denoted as g_0, g_1, g_2,...,g_n at T_0, T_1, T_2,..,T_n steps. The learning rate and momentum decay factor are denoted as λ and α respectively. The momentum buffer at T_0-th step is given M_0.

Momentum can be expressed as follows:

$$M_t = M_{t-1} \cdot \alpha + g_{t-1} \tag{7}$$

Then, parameters updating can be formulated as follows:

$$w_1 = w_0 - \lambda \cdot M_1 \tag{8}$$
$$w_2 = w_0 - \lambda \cdot (M_1 + M_2) \tag{9}$$
$$w_t = w_0 - \lambda \cdot (M_1 + ... + M_t) \tag{10}$$

SWA: Averaging model weights. We expand it by gradients:

$$w_{swa} = \frac{\sum_{t=0}^{t=n-1} w_t}{n} \tag{11}$$
$$w_{swa} = \frac{n \cdot w_0 - \lambda \cdot \{(n-1) \cdot M_1 + (n-2) \cdot M_2 + ... + M_{n-1}\}}{n} \tag{12}$$
$$= w_0 - \lambda \cdot (\frac{n-1}{n} \cdot M_1 + \frac{n-2}{n} \cdot M_2 + ... \frac{1}{n} M_{n-1}) \tag{13}$$

$$
\begin{aligned}
w_{swa} - w_0 &= -\lambda \cdot (\frac{n-1}{n} \cdot M_1 + \frac{n-2}{n} \cdot M_2 + ... \frac{1}{n} M_{n-1}) \\
&= -\lambda \cdot (M_0 (\frac{n-1}{n} \cdot \alpha + \frac{n-2}{n} \cdot \alpha^2 + \frac{n-3}{n} \cdot \alpha^3 + ... + \frac{1}{n} \cdot \alpha^{n-1}) \\
&+ g_0 \cdot (\frac{n-1}{n} + \frac{n-2}{n} \cdot \alpha + \frac{n-3}{n} \cdot \alpha^2 + ... + \frac{1}{n} \alpha^{n-2}) \\
&+ g_1 \cdot (\frac{n-2}{n} + \frac{n-3}{n} \cdot \alpha + \frac{n-4}{n} \cdot \alpha^2 + ... + \frac{1}{n} \alpha^{n-3}) + ... \\
&+ \frac{1}{n} g_{n-2}) \qquad (14)
\end{aligned}
$$

From the analysis above, SWA heuristically re-weights the gradients in essence. It is a special case of WOT and it equals to WOT only if the gaps m is set to 1, the number of gaps k is set to the length from the checkpoint that SWA starts to the checkpoint that SWA ends and the learned weights exactly equal to the coefficient of the gradient described in Eq. 14.

E More Experiments

The experiments are conducted on CIFAR-10 based on FAT, FAT+WOT-W and FAT+WOT-B with PreActResNet-18 architecture. The results in Table 8 consistently verify that WOT can effectively improve adversarial robustness across multiple attacks without sacrifice of clean accuracy. Besides, we also report the robust performance of AT+EMA and compare it with AT+WOT-B/W. The results in Table 9 show that AT+WOT-B/W outperforms AT+EMA as well.

Table 8: Test robustness under multiple adversarial attacks based on FAT. The experiments are conducted on CIFAR-10 with PreActResNet-18 architecture. The bold denotes the best performance.

Models	Clean	FGSM	PGD-20	PGD-100	CW_∞	AA-L_∞
FAT	86.90	54.186	44.69	42.35	44.84	40.71
FAT+WOT-W(**Ours**)	87.25	55.97	47.81	45.71	46.87	43.14
FAT+WOT-B(**Ours**)	**87.59**	**57.22**	**48.27**	**46.03**	**47.62**	**43.40**

Table 9: The robust accuracy of EMA VS WOT on CIFAR-10 with PreActResNet-18 architecture

	PGD-20	PGD-100	CW_∞	AA-L_∞
AT+EMA	52.88	52.24	50.75	47.94
AT+WOT-B (M=400,K=4)	54.85	53.77	52.56	48.96
AT+WOT-W (M=400,K=4)	53.19	51.90	51.74	48.36

F Memory cost and Time cost

WOT takes more GPU memory space since it needs to record the optimization trajectories. We report the memory cost of WOT (Gaps:m=400, Number of Gaps:K=4) based on VGG-16 and PreActResNet-18 respectively in Table 10. It shows that even though we need extra memory to cache Δw, the memory overhead in practice is quite small since the majority of memory usage is concentrated on the intermediate feature maps and gradient maps, accounting for 81% of memory usage on AlexNet and 96% on VGG-16 for an example (Rhu et al., 2016).

Besides, WOT consumes more computation time since it needs to optimize the α by maximizing the robust performance on an unseen set. However, the extra computation time cost is negligible since the unseen set is very small. For example, AT+WOT takes **135** minutes to train PreActResNet-18 model for 100 epochs while AT takes **129** minutes to train the model for 100 epochs.

Table 10: Momory cost of WOT-B (Gaps:400, Number of Gaps:K=4)

	WOT-B	VANILLA AT
VGG	3771M	3409M
PREACRESNET-18	4635M	4213M

G THE RATIONALITY FOR CONSTRAINING α TO $[0, 1]$

We constrain α to [0, 1] such that the refined optimization trajectories will not go too far away from the optimization trajectories. In contrast, if we do not constrain α, refining optimization trajectories could lead to worse performance or even cause collapse.

We conduct experiments on CIFAR-10 and CIFAR-100 with PreActResNet-18 model to show that constraining α to $[0, 1]$ is a reasonable choice. The experimental settings are as follows:

- WOT-B without constraining α. (Abbreviated as WOT-B (No constraints))
- WOT-B with sum(α) =1.
- WOT-B with max(α)=0.5, 0.8, and $\alpha \in$ [0,max(α)].

The results are reported in Table 11 where we can see that without our constraint, there is a worse performance. We empirically find that neither constraining sum(α) =1 nor constraining max(α) with smaller value (< 1) can match the performance of our default setting.

Table 11: Robust Accuracy under PGD-10 attack for WOT, WOT without constraints for α, WOT with the constraint by setting sum(α) =1 or max(α)=0.5, 0.8. Number of Gaps:k=4. NaN denotes the refining process leads to a NaN loss.

METHODS	CIFAR-10	CIFAR-100
WOT-B (GAPS:M=400)	55.83	30.22
WOT-B (GAPS:M=800)	**56.22**	**30.47**
WOT-B (SUM(α)=1, GAPS:M=400)	53.12	28.43
WOT-B (MAX(α)=0.8, GAPS:M=400)	55.18	29.83
WOT-B (MAX(α)=0.5, GAPS:M=400)	55.34	29.66
WOT-B (NO CONSTRAINTS, GAPS:M=400)	55.68	NAN

H THE EFFECT OF THE SIZE OF THE HOLD-OUT SET

We conduct experiments to show the effect of the size of the hold-out set (validation set) based on CIFAR-10 with PreActResNet-18 and WOT-B (Gaps:400, Number of Gaps:k=4). The results are reported below in Table 12 and show that the robust accuracy decreases with the size increasing from 1000 to 8000. We conjecture that a large size of hold-out set reduces the size of training set, which would deteriorate the optimizing trajectories, leading to a worse performance.

Table 12: Robust and clean accuracy under PGD-10 on CIFAR-10 with various sizes of the unseen hold-out set.

SIZE	CLEAN ACCURACY	ROBUST ACCURACY
1000	83.97	**55.83**
2000	**84.30**	55.56
4000	83.90	55.10
8000	84.17	54.58

I THE INFLUENCE OF THE CHOICE OF THE VALIDATION SET

We report the standard deviation of robust accuracy among the three repeated runs based on different validation sets on CIFAR-10 and CIFAR-100 respectively with WOT-B. The robust accuracy is calculated under CW_∞ attack. The different validation sets are randomly sampled from CIFAR-10/100 with different seed.

The results (In Table 13) show that the standard deviations are less than 0.3%, in line with the statement in our paper.

Table 13: The standard deviations on the robust accuracy with three runs using different samples in the validation set.

VALIDATION SET	ROBUST ACCURACY(CIFAR-10)	ROBUST ACCURACY (CIFAR-100)
SET 1	52.01	27.04
SET 2	52.36	27.14
SET 3	52.11	27.3
MEAN	52.16	27.16
STD	0.18	0.13

J VISUALIZATIONS

Figure 7 shows that WOT+AT enjoys a loss landscape with low curvature compared with AT, which is in line with the robust generalization claim in Moosavi-Dezfooli et al. (2018).

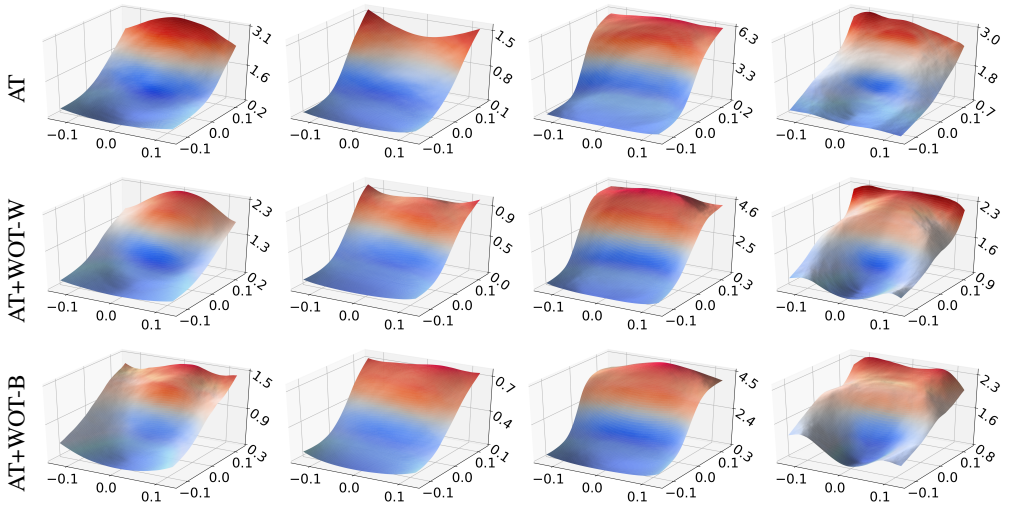


Figure 7: Comparison of loss landscapes of PreRN-18 models trained by AT (the first row) and our methods (the second and third row). Loss plots in each column are generated from the same original image randomly chosen from the CIFAR-10 test set. z axis denotes the loss value. Following the setting in Engstrom et al. (2018), we plot the loss landscape function: $z = loss(x \cdot r_1 + y \cdot r_2)$ where $r_1 = sign(\nabla_x f(x))$ and $r_2 \sim Rademacher(0.5)$.