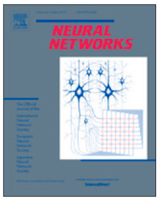
Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet



Bridged adversarial training

Hoki Kim [a], Woojin Lee [b], Sungyoon Lee [c], Jaewook Lee [d,*]

[a] *Institute of Engineering Research, Seoul National University, Gwanak-gu 08826, Republic of Korea*
[b] *School of AI Convergence, Dongguk University-Seoul, Jung-gu 04620, Republic of Korea*
[c] *Department of Computer Science, Hanyang University, Seongdong-gu 04763, Republic of Korea*
[d] *Department of Industrial Engineering, Seoul National University, Gwanak-gu 08826, Republic of Korea*

ARTICLE INFO

Article history:
Received 30 August 2021
Received in revised form 7 June 2023
Accepted 13 August 2023
Available online 22 August 2023

Keywords:
Neural networks
Adversarial robustness
Adversarial training
Adversarial defense

ABSTRACT

Adversarial robustness is considered a required property of deep neural networks. In this study, we discover that adversarially trained models might have significantly different characteristics in terms of margin and smoothness, even though they show similar robustness. Inspired by the observation, we investigate the effect of different regularizers and discover the negative effect of the smoothness regularizer on maximizing the margin. Based on the analyses, we propose a new method called bridged adversarial training that mitigates the negative effect by bridging the gap between clean and adversarial examples. We provide theoretical and empirical evidence that the proposed method provides stable and better robustness, especially for large perturbations.

1. Introduction

Deep neural networks are vulnerable to adversarial examples, which are intentionally perturbed to cause misclassification (Szegedy et al., 2013). Since deep neural networks can be applied to various fields, defense techniques against adversarial attacks are now considered an important research area. To improve the robustness of neural networks against adversarial attacks, many defense methods have been proposed (Goodfellow, Shlens, & Szegedy, 2014; Kim, Lee, & Lee, 2021; Madry, Makelov, Schmidt, Tsipras, & Vladu, 2017; Tramèr et al., 2017; Zhang et al., 2019). Among these, adversarial training (AT) (Madry et al., 2017) and TRADES (Zhang et al., 2019) are considered powerful base methods to achieve high adversarial robustness (Gowal, Qin, Uesato, Mann, & Kohli, 2020; Wu, Xia, & Wang, 2020). In this paper, while AT and TRADES have similar robustness, we discover that they have totally different *margin* and *smoothness*.

Margin, in general, corresponds to the distance from an example to the decision boundary. For example, given a clean example $\boldsymbol{x}$ and its probability output $\boldsymbol{p}(\boldsymbol{x})$, the adversarial margin can be defined as the difference between the probability with respect to the true label y and the other most probable class, $\boldsymbol{p}(\boldsymbol{x})_y - \max_i \boldsymbol{p}(\boldsymbol{x})_{i \neq y}$ (Liu, Han et al., 2021) as shown in Fig. 1. A Larger distance indicates a better margin. AT tries to maximize the margin of an adversarial example $\boldsymbol{x}^*$, which corresponds to the red arrow in Fig. 1.

Smoothness corresponds to the insensitiveness of the output to the input perturbation. The $\mathcal{L}_2$ distance $\|\boldsymbol{p}(\boldsymbol{x}) - \boldsymbol{p}(\boldsymbol{x}^*)\|_2$ (Kannan, Kurakin, & Goodfellow, 2018) or the Kullback–Leibler divergence $\mathrm{KL}(\boldsymbol{p}(\boldsymbol{x}) \| \boldsymbol{p}(\boldsymbol{x}^*))$ (Zhang et al., 2019) can be easily used to estimate smoothness. TRADES tries to maximize the margin of an clean example $\boldsymbol{x}$ (green arrow), while minimizes the smoothness between $\boldsymbol{p}(\boldsymbol{x})$ and $\boldsymbol{p}(\boldsymbol{x}^*)$ (red and blue arrows).

Inspired by the observation that AT and TRADES have totally different margin and smoothness, we investigate the characteristics of the regularizers of AT and TRADES, and find that there exists the negative effect of the smoothness regularizer (blue arrow) on maximizing the margin. From the analyses, we propose a novel method to mitigate the negative effect and provide stable performance by bridging the gap between clean and adversarial examples.

2. Related work and background

2.1. Notations

We consider a c-class classification task with a neural network $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathbb{R}^c$. The network $f_{\boldsymbol{\theta}}$ classifies a sample $\boldsymbol{x} \in \mathcal{X}$ as $\arg\max_{i \in \mathcal{Y}} [f_{\boldsymbol{\theta}}(\boldsymbol{x})]_i$, where $\mathcal{Y} = \{0, \ldots, c-1\}$. We denote the true label with respect to $\boldsymbol{x}$ by y and the corresponding one-hot representation by $\boldsymbol{y} \in \{0, 1\}^c$. That is, $y_i = \mathbf{1}\{i = y\}, \forall i \in \mathcal{Y}$, with an indicator function $\mathbf{1}\{C\}$ which outputs 1 if the condition C is true and 0 otherwise. Then the probability function $\boldsymbol{p}_{\boldsymbol{\theta}} = \mathrm{softmax} \circ f_{\boldsymbol{\theta}} : \mathcal{X} \to [0, 1]^c$ outputs a c-dimensional probability vector whose elements sum to 1.

Given two probability vectors $\boldsymbol{p}, \boldsymbol{q}$ in the c-dimensional probability simplex, we define the following values: $H_{\boldsymbol{p}}(\boldsymbol{q}) = -\boldsymbol{p}^T \log \boldsymbol{q}$

[*] Corresponding author.
E-mail address: jaewook@snu.ac.kr (J. Lee).

https://doi.org/10.1016/j.neunet.2023.08.024
0893-6080/© 2023 Elsevier Ltd. All rights reserved.

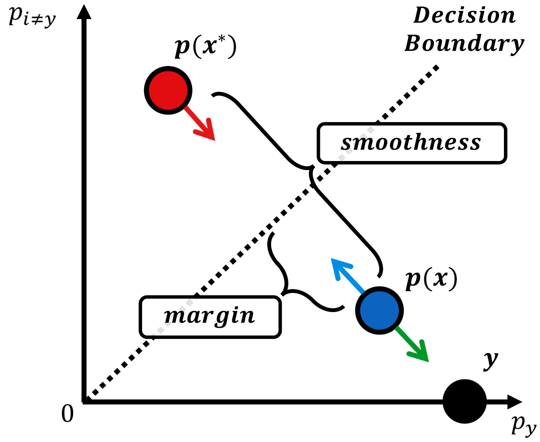


Fig. 1. Simple illustration of margin and smoothness in adversarial training. The distance to the decision boundary from an example corresponds to margin. In contrast, the distance between outputs of a clean example $\boldsymbol{x}$ and an adversarial example $\boldsymbol{x}^*$ corresponds to smoothness. A smaller distance indicates a better smoothness.

and $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{q}) = \boldsymbol{p}^T \log \frac{\boldsymbol{p}}{\boldsymbol{q}}$. These are called the cross-entropy and Kullback–Leibler (KL) divergence between $\boldsymbol{p}$ and $\boldsymbol{q}$, respectively. In addition, we denote the entropy of $\boldsymbol{p}$ as $H(\boldsymbol{p}) = H_{\boldsymbol{p}}(\boldsymbol{p}) = -\boldsymbol{p}^T \log \boldsymbol{p}$. Note that for a one-hot vector $\boldsymbol{y} \in \{0, 1\}^c$, $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{q}) = \sum_{i\in\mathcal{Y}} y_i \log \frac{y_i}{q_i} = -\boldsymbol{y}^T \log \boldsymbol{q}$ is equivalent to the well-known cross-entropy, $H_{\boldsymbol{y}}(\boldsymbol{q})$.

2.2. Adversarial robustness

Since Szegedy et al. (2013) identified the existence of adversarial examples, most defenses are broken by adaptive attacks (Athalye, Carlini, & Wagner, 2018; Tramer, Carlini, Brendel, & Madry, 2020) and the state-of-art performance is still observed from variants of adversarial training (Madry et al., 2017) and TRADES (Zhang et al., 2019) utilizing the training tricks (Gowal et al., 2020; Pang, Yang, Dong, Su, & Zhu, 2020), weight averaging (Wu et al., 2020), and using more data (Carmon, Raghunathan, Schmidt, Duchi, & Liang, 2019; Rebuffi et al., 2021).

Adversarial Training (AT) (Madry et al., 2017) is one of the most effective defense methods. Given a perturbation set $\mathbb{B}(\boldsymbol{x}, \epsilon)$, which denotes a ball around an example $\boldsymbol{x}$ with a maximum perturbation ϵ, it encourages the worst-case probability output over the perturbation set $\mathbb{B}(\boldsymbol{x}, \epsilon)$ to directly match the label $\boldsymbol{y}$ by minimizing the following loss:

$$\begin{aligned}\ell_{AT}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) &= \max_{\boldsymbol{x}'\in\mathbb{B}(\boldsymbol{x},\epsilon)} H_{\boldsymbol{y}}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}')) \\ &= \max_{\boldsymbol{x}'\in\mathbb{B}(\boldsymbol{x},\epsilon)} \mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}')).\end{aligned} \tag{1}$$

TRADES (Zhang et al., 2019) was proposed based on the analysis of the trade-off between adversarial robustness and standard accuracy. TRADES minimizes the following loss:

$$\begin{aligned}&\ell_{TRADES}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) \\ =&\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})) + \beta \max_{\boldsymbol{x}'\in\mathbb{B}(\boldsymbol{x},\epsilon)} \mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}')).\end{aligned} \tag{2}$$

where β is the regularization hyper-parameter. Here, the first term aims to maximize the margin of clean examples, while the second term encourages the model to be smooth.

To solve this highly non-concave optimization in (1) and (2), an iterative projected gradient descent (PGD) (Madry et al., 2017) with n steps is widely used:

$$\boldsymbol{x}^{t+1} = \Pi_{\mathbb{B}(\boldsymbol{x},\epsilon)}\big(\boldsymbol{x}^t + \alpha \cdot \mathrm{sign}(\nabla_{\boldsymbol{x}}\ell_{inner}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}))\big) \text{ for } t = 1, \ldots, n-1 \tag{3}$$

where $\Pi_{\mathbb{B}(\boldsymbol{x},\epsilon)}$ refers the projection to the $\mathbb{B}(\boldsymbol{x}, \epsilon)$ and α is a step-size for each step. Here, $\boldsymbol{x}^0$ is the original example and $\boldsymbol{x}^n$ is used an adversarial example $\boldsymbol{x}^*$. We denote this as PGD^n. For example, AT aims to minimize the loss in (1) so that $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^t))$ is used as $\ell_{inner}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})$.

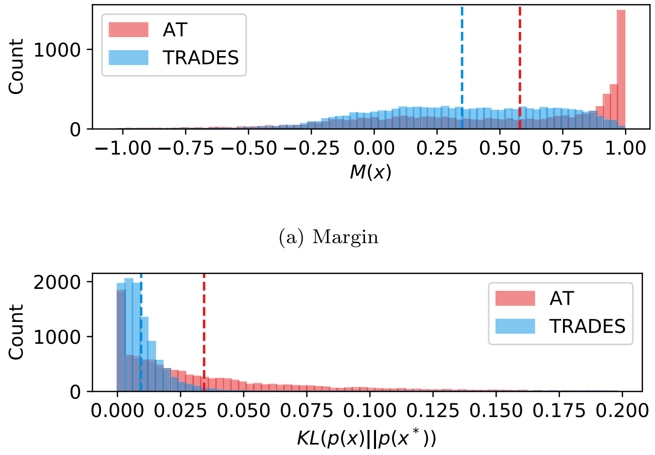
2.3. Margin and smoothness

To achieve a higher accuracy, margin and smoothness have been considered as important characteristics of deep neural networks (Anil, Lucas, & Grosse, 2019; Elsayed, Krishnan, Mobahi, Regan, & Bengio, 2018; Fazlyab, Robey, Hassani, Morari, & Pappas, 2019; Sokolić, Giryes, Sapiro, & Rodrigues, 2017). Following prior works, the concept of margin and smoothness also have been adopted in the adversarial training framework.

In the case of margin, max-margin adversarial training (MMA) (Ding, Sharma, Lui, & Huang, 2019) trains adversarial examples for the correctly classified examples, and clean examples for the misclassified examples to maximize the input space margin, which is the distance to the decision boundary in the input space. Misclassification aware adversarial training (MART) (Wang et al., 2019) outperforms MMA by emphasizing the regularization of the misclassified examples. Note that Wang et al. (2019) used the output space margin (i.e., the difference between the probability with respect to the true label and the other most probable class) that is the distance to the decision boundary in the output space, which we use in this paper. Sanyal, Dokania, Kanade, and Torr (2020) discovered that adversarial trained models have a larger margin than naturally trained models, and explained it with consideration of the complexity of decision boundaries. Yang, Khanna et al. (2020) focused on the boundary thickness, an extended concept of the margin, and discovered the correlation with their adversarial robustness.

In the case of smoothness, there has been a line of work that forces to decrease the distance between logits (or probabilities) of benign and adversarial examples such as Parseval network (Cisse, Bojanowski, Grave, Dauphin, & Usunier, 2017), input gradient regularization (Ross & Doshi-Velez, 2018), and adversarial logit pairing (Kannan et al., 2018). Hein and Andriushchenko (2017) attempted to explain the adversarial robustness with the instance-based local Lipschitz. Although, Zhang et al. (2019) initially proposed the KL divergence loss in TRADES inspired by the mathematical proofs on the bound of adversarial robustness, but it is now also reinterpreted as a smoothness-related method as described in (Yang, Rashtchian, Zhang, Salakhutdinov and Chaudhuri, 2020). Yang, Rashtchian et al. (2020) also attempted to find connection between local Lipschitzness and adversarial robustness.

However, while the correlation between the margin and smoothness in standard training has been discussed (von Luxburg & Bousquet, 2004; Xu, Caramanis, & Mannor, 2009), none of the works analyzed both margin and smoothness together in the adversarial training framework. Thus, we analyze the margin and smoothness of different adversarial training frameworks, and explain the difference between their margin and smoothness by analysis on their regularization terms. Although some provable defensive methods have discussed the trade-off between the margin and smoothness (Chen, Cheng, Gan, Gu, & Liu, 2020; Lee, Lee, & Park, 2020; Lee, Lee, Park, & Lee, 2021; Salman et al., 2019), they are not our main focus here because they are figuratively orthogonal from the above adversarial training frameworks.



(a) Margin

(b) Smoothness

Fig. 2. (CIFAR10) Margin and smoothness of AT and TRADES. (a) $M(\boldsymbol{x})$ for estimating margin (higher is better). (b) $\mathrm{KL}(\boldsymbol{p}_\theta(\boldsymbol{x})\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$ for estimating smoothness (lower is better). Each vertical line indicates the average value of each measure. Each plot used 10,000 test examples.

3. Understanding margin and smoothness in adversarial training

We will start by introducing the difference in margin and smoothness between AT and TRADES. Then, we explore the cause of the difference by analyzing their regularizers.

To illustrate the difference between AT and TRADES in terms of margin and smoothness, we first define a measure for margin and smoothness. To estimate margin, we adopt the probabilistic margin $M(\cdot)$ that is commonly used by prior studies (Carlini & Wagner, 2017; Gowal et al., 2020; Liu, Han et al., 2021):

$$M(\boldsymbol{x}) := [\boldsymbol{p}_\theta(\boldsymbol{x})]_y - \max_{i\neq y}[\boldsymbol{p}_\theta(\boldsymbol{x})]_i \tag{4}$$

Thus, $M(\boldsymbol{x}) > 0$ indicates that the model correctly predicts the label of $\boldsymbol{x}$. On the contrary, the model outputs a wrong prediction when $M(\boldsymbol{x}) < 0$. To estimate smoothness, we use $\mathrm{KL}(\boldsymbol{p}_\theta(\boldsymbol{x})\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$, where $\boldsymbol{x}^*$ is an adversarial example of a clean example $\boldsymbol{x}$. It is worth noting that the KL divergence is primarily used in TRADES (Zhang et al., 2019) and has been further investigated as a measure related to smoothness in (Yang, Rashtchian et al., 2020).

Fig. 2 illustrates the difference between AT and TRADES in terms of margin and smoothness. First, we trained models with the maximum perturbation $\epsilon = 8/255$ on CIFAR10. Then, for each model, we measured the margin and smoothness on an adversarial dataset $(\boldsymbol{x}^*, \boldsymbol{y})$ generated by using PGD^{50} with the same maximum perturbation $\epsilon = 8/255$. More detailed settings are presented in Section 5. At the end of the training, although AT and TRADES have similar robustness (53.94% and 52.98%), they show totally different characteristics. AT shows a larger margin that is distributed close to 1, whereas it has a poor smoothness than TRADES. On the contrary, TRADES shows a smaller margin with only a few examples around 1, whereas it has a better smoothness than AT. The vertical lines (the average values of their margin and smoothness) also show the difference between AT and TRADES. For various settings, we observed similar results in Appendix C.1. Thus, we can conclude that AT and TRADES have different characteristics in terms of margin and smoothness.

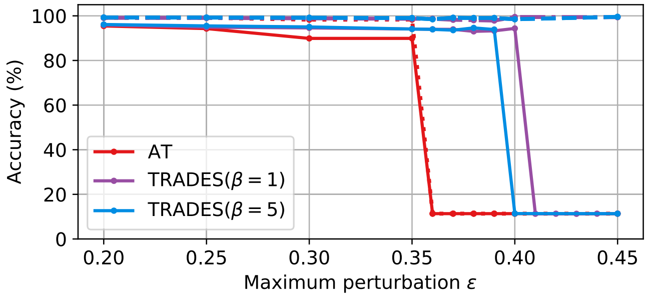


Fig. 3. (MNIST) Stability of each method for a wide range of the maximum perturbation ϵ during training. Each model is trained on the given maximum perturbation ϵ and evaluated by using PGD^{50} with the same ϵ used during training. The dotted and solid lines indicate the clean accuracy and robust accuracy, respectively.

3.1. Same training settings, different optimization difficulty

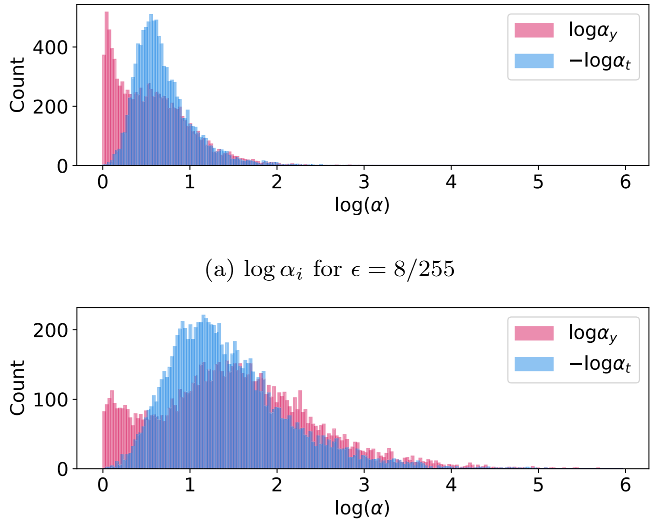
To push further, we evaluate their clean and robust accuracy under various maximum perturbations during the training on MNIST. As shown in Fig. 3, AT fails to achieve both standard and robust accuracy for $\epsilon > 0.35$. This is consistent with the observation that the regularization term for maximizing the margin of adversarial examples has some drawbacks in convergence (Dong et al., 2021; Liu, Salzmann, Lin, Tomioka, & Süsstrunk, 2020; Shaeiri, Nobahari, & Rohban, 2020; Sitawarin, Chakraborty, & Wagner, 2020). In contrast, even for $\epsilon > 0.35$, TRADES holds its robustness better than AT. Interestingly, TRADES shows high clean accuracy for $\epsilon \geq 0.4$ even it fails to gain robustness. However, TRADES also fails to achieve the robustness for $\epsilon \geq 0.4$. The result is similar even with a smaller weight ($\beta = 1$) on $\mathrm{KL}(\boldsymbol{p}_\theta(\boldsymbol{x})\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$.

To explain this phenomenon, we take a closer look at the regularization terms of AT and TRADES. First, AT directly increases the margin of adversarial examples $\boldsymbol{x}^*$ as in (1) by minimizing $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$. However, due to lack of considering the connection between $\boldsymbol{x}$ and $\boldsymbol{x}^*$, AT has difficulty attaining smoothness, which is observed in Fig. 2. In addition, minimizing $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$ yields the drawbacks in convergence (Dong et al., 2021; Liu et al., 2020; Shaeiri et al., 2020; Sitawarin et al., 2020), which can be observed in Fig. 3. In contrast, TRADES adopts the regularization term $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_\theta(\boldsymbol{x}))$ and $\mathrm{KL}(\boldsymbol{p}_\theta(\boldsymbol{x})\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$ in (2) instead of $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$. Due to this loss function, TRADES shows much more stable performance than AT for $\epsilon < 0.4$ as shown in Fig. 3. However, TRADES still fails to optimize $\mathrm{KL}(\boldsymbol{p}_\theta(\boldsymbol{x})\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$ for $\epsilon \geq 0.4$, while it successfully optimizes $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_\theta(\boldsymbol{x}))$. This implies that TRADES has trouble converging to the optimal point during maximizing the margin and minimizing the KL divergence simultaneously.

3.2. Negative effect of smoothness regularizer on maximizing the margin

The degraded margin of TRADES and its failure cases for large perturbations lead us to postulate the hypothesis that there is a conflict between $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_\theta(\boldsymbol{x}))$ and $\mathrm{KL}(\boldsymbol{p}_\theta(\boldsymbol{x})\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$. Now, we mathematically prove that the regularizer for smoothness $\mathrm{KL}(\boldsymbol{p}_\theta(\boldsymbol{x})\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))$ in (2) has a negative effect on training a large margin.

For simplicity, we first consider the binary case. Without loss of generality, we may assume that the correct label for $\boldsymbol{x}$ is $y = 0$ and $\boldsymbol{p}_\theta(\boldsymbol{x}) = [p, 1-p]$. Then, the margin becomes $M(\boldsymbol{x}) = 2p - 1$. Under this binary case, the following proposition holds.



(a) $\log\alpha_i$ for $\epsilon = 8/255$

(b) $\log\alpha_i$ for $\epsilon = 16/255$

Fig. 4. (CIFAR10) Distribution of $\log\alpha_y = \log(p_y/p_y^*)$ for the true class y (red) and $-\log\alpha_t = -\log(p_t/p_t^*)$ for the target class $t = \arg\max_{i\neq y} p_i^*$ (blue). Each plot shows the distribution of $\log\alpha_y$ and $\log\alpha_t$ for whole test adversarial examples generated with different maximum perturbation ϵ. $\log\alpha_y$ is always positive and $\log\alpha_t$ is always negative. In addition, for a larger ϵ during the attack process, both $|\log\alpha_y|$ and $|\log\alpha_t|$ have a larger deviation from 0.

Proposition 1. *Let $\boldsymbol{p} = \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})$, $\boldsymbol{p}^* = \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*)$, $\nabla = \nabla_{\boldsymbol{\theta}}$. Assume the binary case, where $\boldsymbol{p} = [p, 1-p]$ and $\boldsymbol{p}^* = [p^*, 1-p^*]$. Then, $-\nabla KL(\boldsymbol{p}\|\boldsymbol{p}^*)$ has a gradient direction opposite to $\nabla M(\boldsymbol{x})$.*

$$\nabla KL(\boldsymbol{p}\|\boldsymbol{p}^*) = \frac{1}{2}\Big(\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}\Big)\cdot\nabla M(\boldsymbol{x}) + \boldsymbol{c}$$

Here, $\boldsymbol{c}$ is a linear combination of other gradient directions. Since $p > p^*$, $\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}$ is always positive. Thus, minimizing $KL(\boldsymbol{p}\|\boldsymbol{p}^*)$ hinders the increase in $M(\boldsymbol{x})$.

We can easily extend the binary case to the multi-class problem by introducing an element-wise division vector $\boldsymbol{\alpha} = \boldsymbol{p}/\boldsymbol{p}^*$, i.e., the ith element of $\boldsymbol{\alpha}$ is $\alpha_i = p_i/p_i^*$.

Proposition 2. *Let the correct label y, $t = \arg\max_{i\neq y} p_i^*$ and $\alpha_i = p_i/p_i^*$. Assume that $p_y > p_y^*$ and $p_t < p_t^*$. Then, $-\nabla KL(\boldsymbol{p}\|\boldsymbol{p}^*)$ is aligned with a gradient direction that penalizes p_i with the scale of $\log\alpha_i$, which minimizes the margin $M(\boldsymbol{x})$.*

$$\nabla KL(\boldsymbol{p}\|\boldsymbol{p}^*) = (\nabla\boldsymbol{p})^T\log\boldsymbol{\alpha} + \boldsymbol{c}$$

Here, $\boldsymbol{c}$ is a linear combination of other gradient directions. By the assumption, $\log\alpha_y > 0$ and $\log\alpha_t < 0$ so that $-(\nabla p_y)^T\log\alpha_y - (\nabla p_t)^T\log\alpha_t$ minimizes the margin $M(\boldsymbol{x})$.

The proofs are provided in Appendix A. Note that the assumption, $p_y > p_y^*$ and $p_t < p_t^*$, is generally acceptable under the characteristic of adversarial attack, which reduces p_y and increase $p_{i\neq y}$. We empirically proved this assumption in Fig. 4. From the model trained with TRADES under $\epsilon = 8/255$, we generated an adversarial dataset $(\boldsymbol{x}^*, \boldsymbol{y})$ from the whole test dataset by using PGD^{50} with the same maximum perturbation $\epsilon = 8/255$. Then, we plot the distribution of $\log\alpha_i$ for y and $t = \arg\max_{i\neq y} p_i^*$. As shown in Fig. 4, $\log\alpha_y$ is always positive and $\log\alpha_t$ is always negative for all test examples. Thus, we can conclude that the regularization term for smoothness $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ hinders the model from maximizing the margin.

Now, to provide empirical evidence of the negative effect, we visualize the effect of each loss term of TRADES on the margin in Fig. 5. The x-axis denotes the gradient direction that maximizes the margin, $\nabla M(\boldsymbol{x})$. The blue and green points represent the effect of each term's gradients, $\nabla\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p})$ and $\nabla\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$, on maximizing the margin. While the gradient descent direction of $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p})$ is aligned with $\nabla M(\boldsymbol{x})$, the gradient descent of the other term $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ is in the opposite direction. This is consistent with the intuition that the blue arrow in Fig. 1 is opposite to increases the margin $\boldsymbol{p}_y - \boldsymbol{p}_{i\neq y}$. Thus, the result confirms that minimizing the regularization term $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ has a negative effect on maximizing the margin.

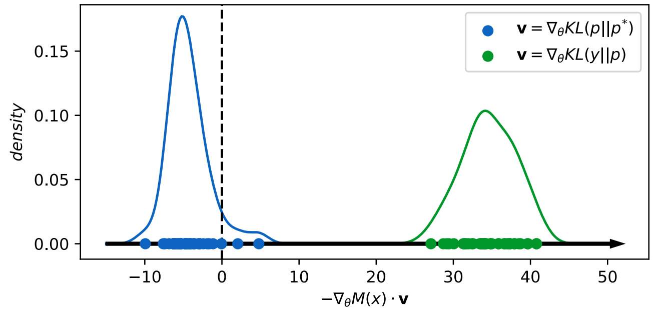


Fig. 5. (CIFAR10) Projected gradients of each loss term of TRADES in (2) on the direction of increasing the margin. At the end of each epoch, we calculated $\nabla\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p})$ and $\nabla\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$, and $\nabla M(\boldsymbol{x})$. Then, we plotted their inner products. Green and blue points correspond to the colored arrows in Fig. 1. Minimizing the regularization term $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ (blue points) has a negative effect on maximizing the margin.

Moreover, we prove that this negative effect gets more pronounced as the maximum perturbation ϵ increases. This can be otherwise said, if $\epsilon_1 < \epsilon_2$, then $\mathbb{B}(\boldsymbol{x}, \epsilon_1) \subset \mathbb{B}(\boldsymbol{x}, \epsilon_2)$ so that

$$\max_{\boldsymbol{x}_1^*\in\mathbb{B}(\boldsymbol{x},\epsilon_1)} |\log[\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})/\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_1^*)]_i| \le \max_{\boldsymbol{x}_2^*\in\mathbb{B}(\boldsymbol{x},\epsilon_1)} |\log[\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})/\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_2^*)]_i| \tag{5}$$

for $i = y, t$. Thus, minimizing $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ with a larger ϵ comes with a larger $|\alpha_i| = |[\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})/\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*)]_i|$. In summary, because the negative effect is proportional to the scale of $\log\alpha_i$ as proved in Proposition 2, a larger ϵ leads to a prohibitive negative effect on maximizing the margin.

Indeed, if we set a larger maximum perturbation $\epsilon = 16/255$, the values of $\log\alpha_y$ and $\log\alpha_t$ tends to have a larger deviation from 0 than that of $\epsilon = 8/255$ (see Fig. 4). This is consistent with our observation that TRADES suffers the convergence problem and fails to achieve decent robustness for a larger perturbation in Fig. 3 and Section 5.3, respectively. Thus, from the above observations and analyses, we expect the model to converge to a better local minimum by mitigating the negative effect.

4. Bridged adversarial training

In the previous sections, we theoretically and experimentally confirm that there exists the negative effect of the smoothness regularizer on maximizing the margin. Now, we enable the model to converge to a better optimal point by mitigating the negative effect.

4.1. Mitigating the negative effect by bridging

To mitigate the negative effect of $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ on maximizing the margin, we have to minimize the absolute value of $\log\boldsymbol{\alpha}$ as proved in Proposition 2. The key idea to control the absolute value of $\log\boldsymbol{\alpha}$ is bridging the gap between $\boldsymbol{p}$ and $\boldsymbol{p}^*$. Let us consider a new probability vector $\tilde{\boldsymbol{p}}$. If we use a new loss function $\mathrm{KL}(\boldsymbol{p}\|\tilde{\boldsymbol{p}})+$

Algorithm 1 Generalized Bridged Adversarial Training

```
Input: training data, $\mathcal{D}$; a continuous path, $\gamma(\cdot)$; a model parameter, $\boldsymbol{\theta}$; a maximum perturbation, $\epsilon$; an adversarial attack, $\mathcal{A}_{\boldsymbol{\theta},\epsilon} : \mathcal{X} \times \mathcal{Y} \to \mathcal{X}$; a hyper-parameter for regularizer, $\beta$; the number of bridges, $m$.
for $(\boldsymbol{x}, \boldsymbol{y}) \sim \mathcal{D}$ do
    $\boldsymbol{x}^* \leftarrow \mathcal{A}_{\boldsymbol{\theta},\epsilon}(\boldsymbol{x}, \boldsymbol{y})$
    $\gamma(0) \leftarrow \boldsymbol{x}$ and $\gamma(1) \leftarrow \boldsymbol{x}^*$
    $\ell \leftarrow \mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}))$
          $+ \sum_{k=0}^{m-1} \beta \mathrm{KL}\big(\boldsymbol{p}_{\boldsymbol{\theta}}(\gamma(\frac{k}{m}))||\boldsymbol{p}_{\boldsymbol{\theta}}(\gamma(\frac{k+1}{m}))\big)$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \nabla_{\boldsymbol{\theta}} \ell$
end for
```

$\mathrm{KL}(\tilde{\boldsymbol{p}}\|\boldsymbol{p}^*)$ instead of $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$, the gradient of $\mathrm{KL}(\boldsymbol{p}\|\tilde{\boldsymbol{p}}) + \mathrm{KL}(\tilde{\boldsymbol{p}}\|\boldsymbol{p}^*)$ can be formalized as follows:

$$\begin{aligned}&\nabla(\mathrm{KL}(\boldsymbol{p}\|\tilde{\boldsymbol{p}}) + \mathrm{KL}(\tilde{\boldsymbol{p}}\|\boldsymbol{p}^*))\\ &\quad=(\nabla \boldsymbol{p})^T \log \boldsymbol{\alpha}^{(1)} + (\nabla \tilde{\boldsymbol{p}})^T \left(-\boldsymbol{\alpha}^{(1)} + \log \boldsymbol{\alpha}^{(2)}\right) - (\nabla \boldsymbol{p}^*)^T \boldsymbol{\alpha}^{(2)} \qquad (6)\end{aligned}$$

where $\alpha_i^{(1)} = p_i/\tilde{p}_i$ and $\alpha_i^{(2)} = \tilde{p}_i/p_i^*$. This is trivial by the proof of Proposition 2 in Appendix A.

As shown in (6), $\nabla \boldsymbol{p}$ is now effected by $\log \boldsymbol{\alpha}^{(1)} = \log(\boldsymbol{p}/\tilde{\boldsymbol{p}})$ instead of $\log \boldsymbol{\alpha} = \log(\boldsymbol{p}/\boldsymbol{p}^*)$. Thus, now we can control the degree of the negative effect by introducing $\tilde{\boldsymbol{p}}$, while achieving the smoothness. For example, if $\tilde{\boldsymbol{p}}$ satisfies $p_y > \tilde{p}_y > p_y^*$ for a correct label y and $p_i < \tilde{p}_i < p_i^*$ for all $i \neq y$, the negative effect can be reduced because $|\log \alpha_i^{(1)}| < |\log \alpha_i|$ for all i. Considering this property, we name a probability vector $\tilde{\boldsymbol{p}}$ that reduces the negative effect as an *intermediate probability*.

In summary, minimizing the new loss $\mathrm{KL}(\boldsymbol{p}\|\tilde{\boldsymbol{p}}) + \mathrm{KL}(\tilde{\boldsymbol{p}}\|\boldsymbol{p}^*)$ can provide the smoothness between $\boldsymbol{p}$ and $\boldsymbol{p}^*$ with the reduced negative effect on maximizing the margin by introducing an intermediate probability $\tilde{\boldsymbol{p}}$ as a bridge. Thus, we name a new adversarial training method, which minimizes $\mathrm{KL}(\boldsymbol{p}\|\tilde{\boldsymbol{p}})+\mathrm{KL}(\tilde{\boldsymbol{p}}\|\boldsymbol{p}^*)$ instead of $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$, *bridged adversarial training (BAT)*.

Intuitively, more than one intermediate probability can induce the less negative effect of $\mathrm{KL}(\boldsymbol{p}\|\tilde{\boldsymbol{p}})$. For a given sample $\boldsymbol{x}$, let $\gamma : [0, 1] \to \mathcal{X}$ be a continuous path from $\gamma(0) = \boldsymbol{x}$ to $\gamma(1) = \boldsymbol{x}^*$, where $\boldsymbol{x}^*$ is an adversarial example of $\boldsymbol{x}$. Now, we minimize the bridged loss $\sum_{k=0}^{m-1} \mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\gamma(\frac{k}{m}))\|\boldsymbol{p}_{\boldsymbol{\theta}}(\gamma(\frac{k+1}{m})))$ instead of $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$. Here, m is a hyper-parameter for the number of intermediate probabilities. The generalized bridged adversarial training procedure is presented in Algorithm 1.

4.2. Bound on the robust error

Here, we provide theoretical evidence that the proposed loss serves as an upper bound on the robust error of the model under the binary classification setting. To give a self-contained overview, we follow (Zhang et al., 2019):

In the binary classification case, a model can be denoted as $f : \mathcal{X} \to \mathbb{R}$. Given a sample $\boldsymbol{x} \in \mathcal{X}$ and a label $y \in \{-1, 1\}$, we use $\mathrm{sign}(f(\boldsymbol{x}))$ as a prediction value of y.

Formally, given a surrogate loss ϕ and $\eta \in [0, 1]$, the conditional ϕ-risk can be denoted as $H(\eta) := \inf_{\alpha\in\mathbb{R}}(\eta\phi(\alpha) + (1 - \eta)\phi(-\alpha))$. Similarly, we can define $H^-(\eta) := \inf_{\alpha(2\eta-1)\leq 0}(\eta\phi(\alpha) + (1 - \eta)\phi(-\alpha))$. Now, we assume the surrogate loss ϕ is classification-calibrated, so that $H^-(\eta) > H(\eta)$ for any $\eta \neq 1/2$. Then, the ψ-transform of a loss function ϕ, which is the convexified version of $\hat{\psi}(\theta) = H^-(\frac{1+\theta}{2}) - H(\frac{1+\theta}{2})$, is continuous convex function on $\theta \in [-1, 1]$.

Then, $\mathcal{R}_{rob}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\exists \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } f(\boldsymbol{x}')y \leq 0\}$ is the robust error. Similarly, $\mathcal{R}_{nat}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{f(\boldsymbol{x})y \leq 0\}$ is the natural classification error. Then, $\mathcal{R}_{bdy}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{f(\boldsymbol{x})y > 0, \exists \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } f(\boldsymbol{x})f(\boldsymbol{x}') \leq 0\}$ is the boundary error by (8). Given a classification-calibrated surrogate loss function ϕ and a surrogate loss $\mathcal{R}_\phi(f) := \mathbb{E}_{(\boldsymbol{x},y)}\phi(f(\boldsymbol{x})y)$, the following theorem is demonstrated.

Theorem 1. *Given a sample $\boldsymbol{x}$ and a positive β, let $\gamma : [0, 1] \to \mathcal{X}$ be a continuous path from $\gamma(0) = \boldsymbol{x}$ to $\gamma(1) = \boldsymbol{x}^*$ where $\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}'\in\mathbb{B}(\boldsymbol{x},\epsilon)} \mathbf{1}\{\beta f(\boldsymbol{x}')f(\boldsymbol{x}) < 0\}$. Then, for any non-negative classification-calibrated loss function ϕ such that $\phi(0) \geq 1$, we have*

$$\begin{aligned}\mathcal{R}_{rob}(f) - \mathcal{R}^{\star}_{nat} &\leq \psi^{-1}(\mathcal{R}_\phi(f) - \mathcal{R}^{\star}_\phi)\\ &\quad + \mathbb{E}_{(\boldsymbol{x},y)} \sum_{k=0}^{m-1} \phi(\beta f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})))\end{aligned}$$

where $\mathcal{R}^{\star}_{nat} := \inf_f \mathcal{R}_{nat}(f)$, $\mathcal{R}^{\star}_\phi := \inf_f \mathcal{R}_\phi(f)$ and ψ^{-1} is the inverse function of the ψ-transform of ϕ.

The proof is presented in Appendix A. Theorem 1 tells us that our proposed method provides an upper bound on the robust error of the model. To push further, we prove that the suggested loss is tighter than that of TRADES under a weak assumption on the path $\gamma(\cdot)$ in Appendix A.

5. Experiments

In this section, we describe a set of experiments conducted to verify the advantages of the proposed method. Finally, we evaluate the robustness and boost its performance using recent techniques.

5.1. Reduced negative effect and its benefits

Here, we investigate the effect of the proposed method in terms of the negative effect, focusing on comparison with AT and TRADES.

Experimental setups. We train Wide-ResNet-28-10 (WRN-28-10) (Zagoruyko & Komodakis, 2016) with AT, TRADES and the proposed method on CIFAR10. During training, horizontal flip and random cropping with padding of 4 are used for data augmentation. All the models are trained 100 epochs using SGD with momentum of 0.9 and weight decay of 5×10^{-4}. We use PGD^{10} with $\epsilon = 8/255$ and the step-size $\alpha = 2/255$ to generate adversarial examples in the training session. For both TRADES and the proposed method, we use $\beta = 5$. For the proposed method, we use $m = 2$, a simple linear path $\gamma(t) = (1 - t)\boldsymbol{x} + t\boldsymbol{x}^*$, and the cross-entropy loss as the inner maximization objective. Sensitivity analysis on the hyper-parameter of the proposed method is in Section 5.5.

Effect of gradient and actual margin. Here, we remark that the gradient of $-\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$ has a direction opposite to the gradient of the margin $M(\boldsymbol{x})$ as described in Proposition 2. Moreover, we experimentally confirmed this negative effect in Fig. 5. In this paragraph, we further measure the actual effect of the gradient on margin and its value during the training. First, we observe the effect of the gradient descent $\nabla_{\boldsymbol{\theta}}\ell$ on the margin maximization, measured by $-\nabla_{\boldsymbol{\theta}} M(\boldsymbol{x}) \cdot \nabla_{\boldsymbol{\theta}}\ell$. This indicates the expected margin increase by the weight update with the loss ℓ. Then, we measure the actual margin $M(\boldsymbol{x})$.

Fig. 6 shows that the proposed method mitigates the negative effect of the regularization term during training. Compared to TRADES, the proposed method shows a higher expected increase in the margin, and this enables the model to achieve a large margin. Thus, by introducing the intermediate probability $\tilde{\boldsymbol{p}}$, we successfully encourage the model to reduce the negative effect of the regularization term on maximizing the margin. Although AT shows a larger margin of adversarial examples $\boldsymbol{x}^*$ by minimizing $\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$, however, AT has difficulty attaining smoothness simultaneously, which is further analyzed in Section 5.2.

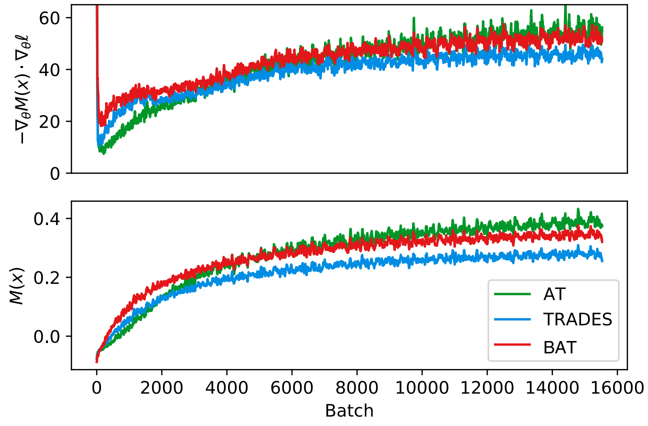


Fig. 6. (CIFAR10) Analysis on the margin during the first 40 epochs. Top: the expected margin increase $-\nabla_\theta M(\boldsymbol{x}) \cdot \nabla_\theta \ell$ of each method. Bottom: the actual margin $M(\boldsymbol{x})$ of each method. The proposed method shows a larger margin than TRADES by mitigating the negative effect.

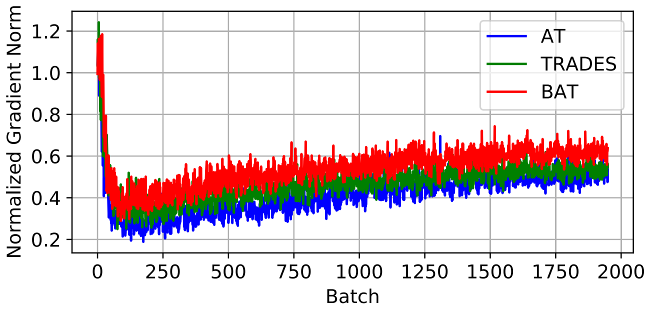


Fig. 7. (CIFAR10) Normalized gradients of each loss term during the first 2000 batches. The proposed method (BAT) shows the largest gradient magnitude, which can help the model quickly escapes the initial suboptimal region (Dong et al., 2021).

Gradient magnitude. A larger norm of gradient also serves to explain the advantage of the proposed method. As prior works discovered (Dong et al., 2021; Liu et al., 2020), a larger gradient norm in the initial training phase enables the model to escape a suboptimal region. To provide a fair comparison for different training methods, we normalize the norm of gradient by the norm of the loss value as follows:

$$\left\| \frac{\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})}{\ell(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})} \right\|_2 \tag{7}$$

As shown in Fig. 7, the gradients of AT exhibit the smallest normalized gradient norm compared to other training methods. This suggests that AT faces challenges in escaping from the initial suboptimal region (Liu et al., 2020). This observation is supported by the experiments conducted under the large maximum perturbation setting in Fig. 3. In contrast, TRADES demonstrates a higher norm of gradients, which aligns with the finding that TRADES provides greater gradient stability within the continuous loss landscape (Dong et al., 2021). However, as depicted in Fig. 3, TRADES encounters difficulties in achieving both high clean accuracy and robustness under the large maximum perturbation setting. On the other hand, the proposed method exhibits the highest normalized gradient norm, indicating its ability to mitigate the negative effects illustrated in Fig. 7. We believe that this supports the stable performance of the proposed method under a higher ϵ in Section 5.3.

5.2. Balanced margin and smoothness

In the previous subsection, we verified that the proposed method successfully mitigates the negative effect during the initial training phase. Now, we investigate whether the proposed method achieves sufficient smoothness while mitigates the negative effect until the end of training.

First, for each trained model, we generate a corresponding adversarial dataset $(\boldsymbol{x}^*, \boldsymbol{y})$ by using PGD^{50} with $\epsilon = 8/255$ and the step-size $\alpha = 2/255$. Then, we plot pairs of the margins of clean examples $\boldsymbol{x}$ and corresponding adversarial examples $\boldsymbol{x}^*$ with their KL divergence. In each plot, the upper and the right histograms show the distribution of $M(\boldsymbol{x})$ and $M(\boldsymbol{x}^*)$, respectively. We colored each point by the KL divergence $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ to measure the smoothness. The red points have high KL divergence (poor smoothness) and the blue points have low KL divergence (better smoothness).

Now, we provide a detailed explanation of the plot. Each quadrant corresponds to:

- **Quadrant I:** $M(\boldsymbol{x}^*) > 0.$ → Adversarial robustness (1 - $\mathcal{R}_{rob}$).
- **Quadrant III:** $M(\boldsymbol{x}) < 0.$ → Natural classification error ($\mathcal{R}_{nat}$).
- **Quadrant IV:** $M(\boldsymbol{x}) > 0$ and $M(\boldsymbol{x}^*) < 0.$ → Boundary error ($\mathcal{R}_{bdy}$)

Note that there is no point in the second quadrant (**Quadrant II**) because adversarial attacks generally do not make incorrect examples ($M(\boldsymbol{x}) < 0$) correctly classified ($M(\boldsymbol{x}^*) > 0$). Here, $\mathcal{R}_{nat}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\arg\max_i f(\boldsymbol{x})_i \neq y\}$ is the natural classification error and $\mathcal{R}_{rob}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\exists \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } \arg\max_i f(\boldsymbol{x}')_i \neq y\}$ is the robust error. Then, $\mathcal{R}_{rob}(f)$ can be decomposed as follows:

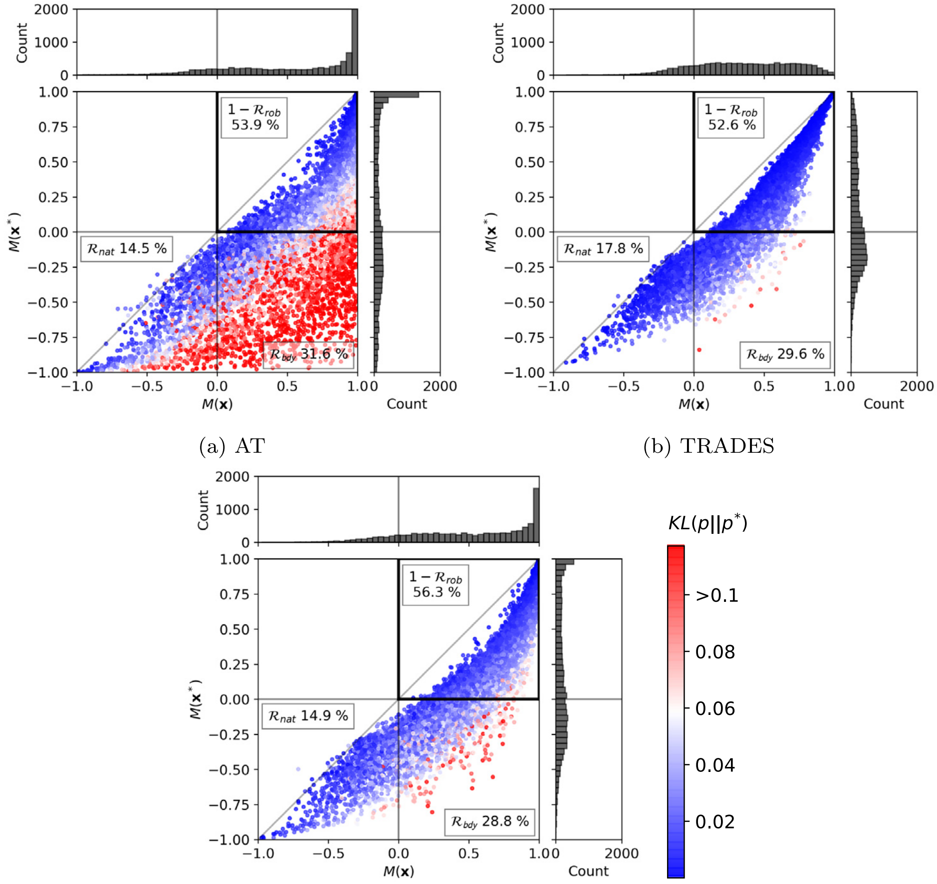
$$\mathcal{R}_{rob}(f) = \mathcal{R}_{nat}(f) + \mathcal{R}_{bdy}(f) \tag{8}$$

where $\mathcal{R}_{bdy}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\arg\max_i f(\boldsymbol{x})_i = y, \exists \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } \arg\max_i f(\boldsymbol{x})_i \neq \arg\max_i f(\boldsymbol{x}')_i\}$ is the boundary error in Zhang et al. (2019). Thus, the ultimate purpose of adversarial training is to move all the points to the first quadrant.

As shown in Fig. 8, the proposed method provide a balanced margin and smoothness with better robustness. Compared to TRADES, which has only a few samples with a high margin $M(\boldsymbol{x})$ and $M(\boldsymbol{x}^*)$, the proposed method shows better margin distributions near 1 for both clean and adversarial examples. This implies that our method successfully mitigates the negative effect of the regularization term on maximizing the margin as discussed in Section 4. As a result, our method achieves a lower natural classification error (14.9%) than that of TRADES (17.8%). Compared to AT, the proposed method has fewer red points, which implies that the proposed method provides smoothness. Due to the increased smoothness, the boundary error of the proposed method ($\mathcal{R}_{bdy} =$ 28.8%) is lower than that of AT ($\mathcal{R}_{bdy} = 31.6\%$).

5.3. Robustness

In this subsection, we verify the robustness of the proposed method. Here, we adopt three benchmark datasets, i.e, MNIST, CIFAR10, and Tiny ImageNet. They are widely used datasets to evaluate the performance of adversarially trained models. For more additional results on FMNIST and CIFAR100, please refer to Appendix C. To push further, in addition to AT and TRADES, we also consider MART (Wang et al., 2019), which aims to maximize the margin and recently achieved the best performance by focusing on misclassified examples.



(a) AT

(b) TRADES

(c) BAT

Fig. 8. (CIFAR10) Distribution of the margins $M(\boldsymbol{x})$ and $M(\boldsymbol{x}^*)$. Each point indicates each test example, and the color of each point indicates the KL divergence loss $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$. The darker red ones indicate a higher KL divergence loss.

Training setups. For MNIST, we train LeNet (LeCun, Bottou, Bengio, & Haffner, 1998) for 50 epochs with the Adam optimizer. Note that only MART is trained SGD with the initial learning rate 0.01 following (Wang et al., 2019), because MART converges to a constant function when the Adam optimizer is used. The initial learning rate is 0.001 and it is divided by 10 at 30 and 40 epochs. During training, we use PGD^{40} with $\epsilon = 0.3$ and 0.45, which are commonly used in prior work (Sitawarin et al., 2020; Wang et al., 2019). We use PGD^{40} with $\epsilon = 0.3$ and the step-size $\alpha = 0.02$ to generate adversarial examples in the training session. No preprocessing or input transformation is used.

For CIFAR10, we train WRN-28-10 with two different training schemes. The one is the step-wise decay setting in Section 5.1, which is generally used, and the other one is cyclic learning rate schedule (Smith, 2017), which is a recently proposed training scheme that enables the model to converge faster (Wong, Rice, & Kolter, 2020). In this subsection, we report the results with cyclic learning rate schedule, because its performance is slightly better than the step-wise decay setting. The results of the step-wise decay setting is in Appendix C. We use 0.3 as the maximum learning rate and a total of 30 epochs for training. During training, we use PGD^{10} with $\epsilon = 8/255$ and 16/255, which are commonly used in prior work (Madry et al., 2017; Wang et al., 2019). The step-size is set to $\alpha = 2/255$. Horizontal flip and random cropping with padding of 4 are used for data augmentation.

For Tiny ImageNet, we used PreActResNet18 (He, Zhang, Ren, & Sun, 2016). Following the settings of (Pang et al., 2020) and (Rice, Wong, & Kolter, 2020), we train a model 110 epochs with SGD optimizer and an initial learning rate is 0.1 and decayed with a factor of 0.1 at the 100th and 105th epoch. We use PGD^{10} with $\epsilon = 8/255$ and the step-size $\alpha = 2/255$ to generate adversarial examples in the training session as used in (Kim et al., 2021). Horizontal flip and random cropping with a padding of 4 are used for data augmentation.

Hyper-parameter selection. For MNIST and CIFAR10, we use the same settings in Section 5.1, because all training methods show stable performance with similar accuracy. However, for Tiny ImageNet, we find that the performance under $\beta > 5$ is sometimes better than that of $\beta = 5$. Thus, we perform grid search on $\beta = \{1, 5, 10, 20, 40\}$ and choose the best β that records the highest robustness against PGD^{50} with $\epsilon = 8/255$. Similarly, we found that $m = 3$ produces the best performance on Tiny ImageNet among $m = \{2, 3, 5, 10\}$. The detailed analysis on sensitivity is in Section 5.5.

Evaluation setups. We basically evaluate the robustness of all models with PGD^{50}. Furthermore, we also consider AutoAttack (Croce & Hein, 2020), which is a combination of three white-box attacks (Croce & Hein, 2019, 2020) and one black-box attack (Andriushchenko, Croce, Flammarion, & Hein, 2020). Note that AutoAttack is by far the most reliable attack to measure robustness. Unless otherwise specified, we use the same ϵ that was used in the training session during evaluation. We also conduct the evaluation with other attacks in Appendix C.

For all values, we report the average and the standard deviation of the performance over three runs with different random seeds. We use PyTorch (Paszke et al., 2019) and Torchattacks (Kim, 2020) for all experiments.

Table 1
Robustness accuracy (%) on MNIST. All models trained using PGD^{40} with $\epsilon = 0.3$ and 0.45, then evaluated by each attack with the same $\epsilon = 0.3$ and 0.45, respectively. We report the average and the standard deviation of the performance over three random seeds.

Method	Clean	PGD^{50}	AutoAttack
(Training $\epsilon = 0.3$)			
AT	98.79 ± 0.23	91.69 ± 1.25	88.64 ± 0.61
TRADES	**98.89 ± 0.01**	93.70 ± 0.01	**92.31 ± 0.21**
MART	98.78 ± 0.16	92.37 ± 1.21	89.62 ± 1.44
BAT	98.79 ± 0.06	**93.97 ± 0.16**	92.19 ± 0.13
(Training $\epsilon = 0.45$)			
AT	11.35 ± 0.00	11.35 ± 0.00	11.35 ± 0.00
TRADES	**99.42 ± 0.05**	11.34 ± 0.01	0.53 ± 0.26
MART	13.13 ± 2.54	7.80 ± 3.73	0.93 ± 1.32
BAT	97.72 ± 0.26	**88.20 ± 0.57**	**76.09 ± 1.65**

Table 2
Robustness accuracy (%) on CIFAR10. All models trained using PGD^{10} with $\epsilon = 8/255$ and 16/255, then evaluated by each attack with the same $\epsilon = 8/255$ and 16/255, respectively. We report the average and the standard deviation of the performance over three random seeds.

Method	Clean	PGD^{50}	AutoAttack
(Training $\epsilon = 8/255$)			
AT	**85.65 ± 0.33**	53.64 ± 0.03	50.87 ± 0.22
TRADES	82.22 ± 0.12	52.14 ± 0.08	48.90 ± 0.35
MART	77.51 ± 0.46	53.87 ± 0.08	48.25 ± 0.06
BAT	84.84 ± 0.28	**55.64 ± 0.37**	**52.41 ± 0.02**
(Training $\epsilon = 16/255$)			
AT	72.43 ± 0.01	29.01 ± 0.13	24.24 ± 0.51
TRADES	70.01 ± 0.44	24.52 ± 0.06	14.63 ± 0.22
MART	65.97 ± 0.54	**32.65 ± 0.40**	23.23 ± 0.14
BAT	**77.56 ± 0.01**	30.79 ± 0.35	**25.06 ± 0.37**

Table 3
Robustness accuracy (%) on Tiny ImageNet. All models trained using PGD^{10} with $\epsilon = 8/255$, then evaluated by each attack with the same $\epsilon = 8/255$. We report the average and the standard deviation of the performance over three random seeds.

Method	Clean	PGD^{50}	AutoAttack
AT	**46.68 ± 0.02**	15.26 ± 0.22	11.52 ± 0.11
TRADES	40.39 ± 0.05	20.48 ± 0.03	12.03 ± 0.13
MART	42.45 ± 0.12	19.43 ± 0.42	11.32 ± 0.51
BAT	42.47 ± 0.04	**21.52 ± 0.10**	**12.53 ± 0.26**

Results. As shown in Table 1, for MNIST with $\epsilon = 0.3$, all defenses show high robustness. However, for a large $\epsilon = 0.45$, all comparison methods converge to a constant function or fail to gain robustness. In other words, the existing methods have difficulty converging to the global optimal. For the cases of AT and MART, they have the term that maximizes the margin of adversarial examples so that it can have difficulty in convergence (Dong et al., 2021). In contrast, TRADES also fails to achieve stable robustness, because a larger perturbation brings stronger negative effect of $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ as we discussed in Section 3. However, the proposed method shows stable results even for $\epsilon = 0.45$. Considering that the difference between TRADES and the proposed method is the usage of bridging, this result tells us that the convergence becomes much easier by using the proposed bridged loss.

The proposed method also shows the best robustness on CIFAR10 (Table 2). Specifically, for $\epsilon = 16/255$, the proposed method achieves 77.56% of standard accuracy, which is 5% higher than AT. Compared to TRADES and MART, it is 6% and 12% higher, respectively. Simultaneously, it also achieves the best robustness 25.06% against AutoAttack. Note that the robustness of TRADES is only 14.63%, which shows the weakness of TRADES for a larger perturbation.

We can also observe the similar result on Tiny ImageNet. As shown in Table 3, the proposed method achieves the best robustness against PGD and AutoAttack. Specifically, the proposed method (21.52%) outperforms AT (15.26%) against PGD. In addition, the proposed method achieves a better clean and robust accuracy than TRADES and MART, which supports the stable performance of the proposed method.

5.4. Benchmarking recent techniques

In this subsection, we run more experiments on CIFAR10 with recent techniques that boost the robustness of the model to show the generality of our results. We consider two different methods, using additional data (Carmon et al., 2019) and adversarial weight perturbation (Wu et al., 2020). We report the average accuracy for three different random seeds, because the standard deviation is quite low (near zero) with these techniques.

Using additional data. Recently, it has been found that using additional data can greatly improve the standard accuracy and robustness (Alayrac et al., 2019; Carmon et al., 2019; Najafi, Maeda, Koyama, & Miyato, 2019; Zhai et al., 2019). Following Carmon et al. (2019), we train models with an additional 500K images mined from the 80 Million Tiny Images dataset (Torralba, Fergus, & Freeman, 2008) in addition to training images in CIFAR10. In this experiment, following the settings (Carmon et al., 2019), we train WRN-28-10 with the cosine learning rate annealing (Loshchilov & Hutter, 2016) without restarts for 200 epochs.

Table 4 shows the results of the experiment with additional data. For $\epsilon = 8/255$, the proposed method shows the best robustness against AutoAttack. Especially, for $\epsilon = 16/255$, the proposed method outperforms the other methods by a large margin. In particular, compared to TRADES, the proposed method shows an approximately 6% improvement in the standard accuracy, while the robustness is also greatly increased (3%).

Adversarial weight perturbation. To achieve better generalization on the test data, recent work has explored procedures to aim for a more flatter minima (Foret, Kleiner, Mobahi, & Neyshabur, 2020; Izmailov, Wilson, Podoprikhin, Vetrov, & Garipov, 2018). These techniques have improved performance of deep learning models, and therefore they are recently adopted to adversarial training (Gowal et al., 2020; Wu et al., 2020). Among them, Wu et al. (2020) boosts the robustness of various adversarial training methods by proposing adversarial weight perturbation (AWP) that flattens the loss landscape. Thus, following the settings in (Wu et al., 2020), we trained ResNet18 for 200 epochs with AWP. The initial learning rate of SGD is set to 0.1 and decayed with a factor of 0.1 at 100th and 150th epoch.

For the case when the maximum perturbation $\epsilon = 8/255$ during training, AT, TRADES, and the proposed method shows similar robustness under various test $\epsilon = 8/255, 12/255$ and 16/255. However, this phenomenon drastically changes for a larger maximum perturbation during the training phase. For $\epsilon = 12/255$ during training, the proposed method shows the highest robustness against various tests $\epsilon = 8/255, 12/255$ and 16/255. Interestingly, the proposed method also achieves the highest clean accuracy. Specifically, under training $\epsilon = 16/255$, the proposed method records the clean accuracy of 75.92%, which is approximately 6%p higher than AT and TRADES, while it also shows the highest robustness.

Interestingly, in terms of the relationship between ϵ used in training and evaluation, Tables 4 and 5 demonstrate that using $\epsilon = 16/255$ during training generally leads to higher robust accuracy against larger perturbations ($\epsilon = 16/255$), but it does not guarantee better performance against smaller perturbations ($\epsilon =$

Table 4
Robustness accuracy (%) with additional data on CIFAR10. All models trained using PGD^{10} with $\epsilon = 8/255$ and 16/255, then evaluated for $\epsilon = 8/255$, 12/255, and 16/255.

Method	Clean	PGD^{50}			AutoAttack		
		$\epsilon = 8/255$	12/255	16/255	8/255	12/255	16/255
(Training $\epsilon = 8/255$)							
AT-RST	**91.53**	59.89	37.21	19.52	58.41	33.87	15.11
TRADES-RST	89.73	61.87	42.41	23.69	59.45	37.97	20.50
MART-RST	89.71	62.11	41.11	22.75	57.97	34.96	16.67
BAT-RST	89.61	**62.38**	**42.74**	**24.33**	**59.54**	**38.07**	**20.86**
(Training $\epsilon = 16/255$)							
AT-RST	83.36	60.51	45.20	29.48	57.15	40.98	25.54
TRADES-RST	78.20	56.51	45.15	32.83	51.29	37.05	24.96
MART-RST	81.24	**61.58**	**47.90**	**34.14**	55.96	40.45	25.77
BAT-RST	**84.07**	61.42	46.70	33.78	**57.24**	**41.67**	**27.70**

Table 5
Robustness accuracy (%) with adversarial weight perturbation on CIFAR10. All models trained using PGD^{10} with $\epsilon = 8/255$ and 16/255, then evaluated for $\epsilon = 8/255$, 12/255, and 16/255.

Method	Clean	PGD^{50}			AutoAttack		
		$\epsilon = 8/255$	12/255	16/255	8/255	12/255	16/255
(Training $\epsilon = 8/255$)							
AT-AWP	**81.05**	54.79	37.94	21.50	49.60	31.16	15.64
TRADES-AWP	79.49	55.10	40.66	27.47	**51.05**	32.96	21.16
MART-AWP	77.96	55.31	40.75	26.50	47.79	32.11	17.67
BAT-AWP	79.40	**56.16**	**41.46**	**27.95**	50.42	**34.84**	**21.91**
(Training $\epsilon = 16/255$)							
AT-AWP	69.14	53.03	43.13	32.38	47.56	35.56	23.11
TRADES-AWP	68.77	47.26	37.11	27.61	42.02	28.96	19.13
MART-AWP	10.41	10.23	10.15	10.17	10.21	10.13	10.12
BAT-AWP	**75.92**	**53.96**	**43.60**	**33.04**	**48.97**	**35.63**	**24.13**

Table 6
(CIFAR10) Sensitivity of β. We report the average and the standard deviation of the performance over three random seeds.

Method	Clean	FGSM	PGD^{50}	AutoAttack
$\beta = 1$	**89.71 ± 0.09**	57.30 ± 0.21	52.11 ± 0.24	49.11 ± 0.02
$\beta = 2$	87.85 ± 0.16	58.55 ± 0.50	54.30 ± 0.26	51.30 ± 0.02
$\beta = 3$	86.36 ± 0.32	59.43 ± 0.29	54.70 ± 0.23	51.70 ± 0.03
$\beta = 4$	85.65 ± 0.27	59.84 ± 0.30	55.07 ± 0.26	52.07 ± 0.02
$\beta = 5$	84.84 ± 0.28	**60.18 ± 0.34**	**55.64 ± 0.37**	**52.41 ± 0.02**
$\beta = 6$	84.73 ± 0.17	60.01 ± 0.27	55.60 ± 0.38	52.40 ± 0.02
$\beta = 7$	83.20 ± 0.26	60.43 ± 0.36	55.55 ± 0.25	52.38 ± 0.03
$\beta = 8$	82.61 ± 0.66	60.54 ± 0.42	55.53 ± 0.32	52.37 ± 0.03

Table 7
(CIFAR10) Sensitivity of m. We report the average and the standard deviation of the performance over three random seeds.

Method	Clean	FGSM	PGD^{50}	AutoAttack
$m = 1$	82.22 ± 0.12	54.20 ± 0.22	52.14 ± 0.08	48.90 ± 0.35
$m = 2$	84.84 ± 0.28	**60.18 ± 0.34**	**55.64 ± 0.37**	**52.41 ± 0.02**
$m = 3$	86.00 ± 0.24	59.78 ± 0.23	54.58 ± 0.44	51.10 ± 0.01
$m = 4$	86.25 ± 0.40	58.24 ± 0.38	53.99 ± 0.21	50.60 ± 0.01
$m = 5$	**87.23 ± 0.33**	57.10 ± 0.27	51.12 ± 0.36	49.60 ± 0.02

8/255). In contrast, using $\epsilon = 8/255$ during training generally leads to higher robust accuracy for $\epsilon = 8/255$, but it shows lower robust accuracy against larger perturbations ($\epsilon = 16/255$). However, regardless of the specific ϵ value, BAT consistently shows stable performance across all combinations of attack and defense ϵ, making it an effective tool for gaining deeper understanding of adversarial robustness against unseen perturbations, which is a largely unexplored area (Shaeiri et al., 2020).

To push further, we also conducted an experiment with stochastic weight averaging (SWA) (Izmailov et al., 2018) following (Gowal et al., 2020), but it shows less improvement than AWP as described in Appendix C.6. However, we note that the proposed method also achieves the best robustness against AutoAttack.

5.5. Sensitivity analysis

Here, we evaluate the sensitivity on the hyper-parameters of the proposed method, β and m. Tables 6 and 7 shows the sensitivity of β and m, respectively. All experiments are performed on CIFAR10 with cyclic learning rate decay as same in Section 5.3. Here, we report the average and the standard deviation of the performance over three random seeds.

For the analysis on β, we fixed $m = 2$. In Table 6, as β increases the robustness increases until $\beta \leq 5$. Notably, this behavior is similar to the findings in TRADES, where values of $\beta = 5$ or $\beta = 6$ tend to yield the best robust accuracy (Pang et al., 2020; Zhang et al., 2019). However, beyond a certain threshold ($\beta \geq 6$), the robust accuracy does not show further improvement. We believe this is because very large values of β may lead to convergence instability, which should be further analyzed with loss landscapes (Liu et al., 2020) or optimization difficulty (Zhang et al., 2019) in future work.

To push further, we measure the margin and smoothness with varying β. For better visualization, we used the inverse of the expectation of the KL divergence (because the smoothness is better when $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ is lower). Thus, as shown in Fig. 9, we plot the inverse of the KL divergence (x-axis) and the margin (y-axis) for each model. In Fig. 9, we can gain further insights by considering the influence of β on the margin and smoothness. As β increases, BAT places more emphasis on improving smoothness. This balance between margin and smoothness contributes to the overall robustness achieved by BAT.

Table 7 shows the sensitivity of m. Here, we fix $\beta = 5$ from the observation in Table 6. Due to decreased negative effect, $m = 2$ shows a higher clean accuracy and robustness for diverse adversarial attacks than that of $m = 1$. Moreover, the best value of m is 2 in terms of adversarial robustness, which implies that

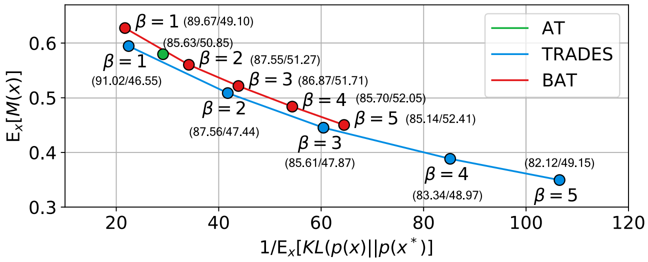


Fig. 9. Sensitivity of β in terms of margin and smoothness. x-axis denotes the inverse of the expectation of the KL divergence and y-axis denotes the expectation of the margin. The clean accuracy and robust accuracy against AutoAttack of each point in parentheses. Each expectation is calculated on the test set. For margin and smoothness, higher is better for both axes.

$m = 2$ is enough to mitigate the negative effect and achieve balance between margin and smoothness.

6. Conclusion

In this paper, we investigated the existing adversarial training methods from the perspective of margin and smoothness of the network. We found that AT and TRADES have different characteristics in terms of margin and smoothness due to their different regularizers. We mathematically proved that the regularization term designed for smoothness has a negative effect on training a larger margin. To this end, we proposed a new method that mitigates the negative effect by bridging the gap between clean and adversarial examples and achieved stable and better performance. It is important to note that AT and TRADES still serve as strong baselines and many of the state-of-the-art methods combine these techniques with newer approaches to achieve the highest performance. For example, the current state-of-the-art performance model also use TRADES with images generated by diffusion models (Wang et al., 2023). Regarding this trend, we believe that our findings can be easily extended and incorporated into future research. We believe that our investigation on margin and smoothness can provide a new perspective to better understand the adversarial robustness and to design a robust model.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. 2019R1A2C2002358; No. 2022R1F1A1074393; RS-2023-00244896), the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00984, Development of Artificial Intelligence Technology for Personalized Plug-and-Play Explanation and Verification of Explanation; No.RS-2023-002206284, Artificial intelligence for prediction of structure-based protein interaction reflecting physicochemical principles), and Hanyang University (HY-202300000000552).

Appendix A. Proofs of theoretical results

A.1. Proof of Proposition 1

Proof of Proposition 1 [Restated]. Let $\boldsymbol{p} = \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})$, $\boldsymbol{p}^* = \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*)$, $\nabla = \nabla_{\boldsymbol{\theta}}$. Assume the binary case, where $\boldsymbol{p} = [p, 1-p]$ and $\boldsymbol{p}^* = [p^*, 1-p^*]$. Then, $-\nabla \mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ has a gradient direction opposite to $\nabla M(\boldsymbol{x})$.

$$\nabla \mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*) = \frac{1}{2}\Big(\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}\Big)\cdot \nabla M(\boldsymbol{x}) + \boldsymbol{c}$$

Here, $\boldsymbol{c}$ is a linear combination of other gradient directions. Since $p > p^*$, $\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}$ is always positive. Thus, minimizing $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ hinders the increase in $M(\boldsymbol{x})$.

Proof. First, $\nabla \mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ can be formalized as follows:

$$\begin{aligned}
\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*) &= p\log p + (1-p)\log(1-p) \\
&\quad - p\log p^* - (1-p)\log(1-p^*) \\
\nabla \mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*) &= \frac{\partial \mathrm{KL}}{\partial p}\cdot\nabla p + \frac{\partial \mathrm{KL}}{\partial p^*}\cdot\nabla p^* \\
&= \Big(\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}\Big)\cdot\nabla p \\
&\quad + \Big(-\frac{p}{p^*} + \frac{1-p}{1-p^*}\Big)\cdot\nabla p^* \\
&= \Big(\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}\Big)\cdot\nabla p + \boldsymbol{c}
\end{aligned}$$

In addition, $\nabla M(\boldsymbol{x}) = 2\nabla p$, since $M(\boldsymbol{x}) = p - (1-p) = 2p - 1$. Thus,

$$\nabla \mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*) = \frac{1}{2}\Big(\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}\Big)\cdot \nabla M(\boldsymbol{x}) + \boldsymbol{c}. \quad \square$$

A.2. Proof of Proposition 2

Proof of Proposition 2 [Restated]. Let the correct label y, $t = \arg\max_{i\neq y} p_i^*$ and $\alpha_i = p_i/p_i^*$. Assume that $p_y > p_y^*$ and $p_t < p_t^*$. Then, $-\nabla \mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ is aligned with a gradient direction that penalizes p_i with the scale of $\log \alpha_i$, which minimizes the margin $M(\boldsymbol{x})$.

$$\nabla \mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*) = (\nabla \boldsymbol{p})^T \log \boldsymbol{\alpha} + \boldsymbol{c}$$

Here, $\boldsymbol{c}$ is a linear combination of other gradient directions. By the assumption, $\log \alpha_y > 0$ and $\log \alpha_t < 0$ so that $-(\nabla p_y)^T \log \alpha_y - (\nabla p_t)^T \log \alpha_t$ minimizes the margin $M(\boldsymbol{x})$.

Proof.

$$\begin{aligned}
&\nabla \mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*) \\
=&\nabla\left(\boldsymbol{p}^T\log\boldsymbol{p} - \boldsymbol{p}^T\log\boldsymbol{p}^*\right) \\
=&(\nabla\boldsymbol{p})^T\log\boldsymbol{p} + (\nabla\boldsymbol{p})^T\boldsymbol{1} - (\nabla\boldsymbol{p})^T\log\boldsymbol{p}^* - (\nabla\boldsymbol{p}^*)^T\frac{\boldsymbol{p}}{\boldsymbol{p}^*} \\
=&(\nabla\boldsymbol{p})^T\log\frac{\boldsymbol{p}}{\boldsymbol{p}^*} - (\nabla\boldsymbol{p}^*)^T\frac{\boldsymbol{p}}{\boldsymbol{p}^*} \\
=&(\nabla\boldsymbol{p})^T\log\boldsymbol{\alpha} - (\nabla\boldsymbol{p}^*)^T\boldsymbol{\alpha}.
\end{aligned} \tag{A.1}$$

which leads to the conclusion with $\boldsymbol{c} = -(\nabla\boldsymbol{p}^*)^T\boldsymbol{\alpha}$ $\square$

A.3. Proof of Theorem 1

To give a self-contained overview, we follow (Bartlett, Jordan, & McAuliffe, 2006). In the binary classification case, given a sample $\boldsymbol{x} \in \mathcal{X}$ and a label $y \in \{-1, 1\}$, a model can be denoted as

$f : \mathcal{X} \rightarrow \mathbb{R}$. We use sign($f(\boldsymbol{x})$) as a prediction value of y. Given a surrogate loss ϕ, the conditional ϕ-risk for $\eta \in [0, 1]$ can be denoted as $H(\eta) := \inf_{\alpha \in \mathbb{R}}(\eta\phi(\alpha) + (1 - \eta)\phi(-\alpha))$. Similarly, we can define $H^{-}(\eta) := \inf_{\alpha(2\eta-1)\leq 0}(\eta\phi(\alpha) + (1 - \eta)\phi(-\alpha))$. Now, we assume the surrogate loss ϕ is classification-calibrated, so that $H^{-}(\eta) > H(\eta)$ for any $\eta \neq 1/2$. Then, the ψ-transform of a loss function ϕ, which is the convexified version of $\hat{\psi}(\theta) = H^{-}(\frac{1+\theta}{2}) - H(\frac{1+\theta}{2})$, is continuous convex function on $\theta \in [-1, 1]$.

In the adversarial training framework, we train a model to reduce the robust error $\mathcal{R}_{rob}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\exists \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } f(\boldsymbol{x}')y \leq 0\}$ where $\mathbb{B}(\boldsymbol{x}, \epsilon)$ is a ball around an example x with a maximum perturbation ϵ. Here, $\mathbf{1}\{C\}$ denotes an indicator function which outputs 1 if the condition C is true and 0 otherwise. As Zhang et al. (2019) proposed, $\mathcal{R}_{rob}(f)$ can be decomposed as follows:

$$\mathcal{R}_{rob}(f) = \mathcal{R}_{nat}(f) + \mathcal{R}_{bdy}(f) \tag{A.2}$$

where the natural classification error $\mathcal{R}_{nat}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{f(\boldsymbol{x})y \leq 0\}$ and boundary error $\mathcal{R}_{bdy}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{f(\boldsymbol{x})y > 0, \exists \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } f(\boldsymbol{x})f(\boldsymbol{x}') \leq 0\}$. By definition, following inequality is satisfied:

$$\begin{aligned}\mathcal{R}_{bdy}(f) &= \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\boldsymbol{x} \in \mathbb{B}(\mathrm{DB}(f), \epsilon), f(\boldsymbol{x})y > 0\} \\ &\leq \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\boldsymbol{x} \in \mathbb{B}(\mathrm{DB}(f), \epsilon)\} \\ &= \mathbb{E} \max_{\boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x},\epsilon)} \mathbf{1}\{f(\boldsymbol{x}') \neq f(\boldsymbol{x})\} \\ &= \mathbb{E} \max_{\boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x},\epsilon)} \mathbf{1}\{\beta f(\boldsymbol{x}')f(\boldsymbol{x}) < 0\}.\end{aligned} \tag{A.3}$$

Let, $\mathcal{R}^{\star}_{nat} := \inf_f \mathcal{R}_{nat}(f)$ and $\mathcal{R}^{\star}_{\phi} := \inf_f \mathcal{R}_{\phi}(f)$ where $\mathcal{R}_{\phi}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\phi\{f(\boldsymbol{x})y \leq 0\}$ is a surrogate loss with a surrogate loss function ϕ. Then, under Assumption 1, following inequality is satisfied by (A.2) and (A.3).

$$\begin{aligned}\mathcal{R}_{rob}(f) - \mathcal{R}^{\star}_{nat} \leq& \psi^{-1}(\mathcal{R}_{\phi}(f) - \mathcal{R}^{\star}_{\phi}) \\ &+ \mathbb{E} \max_{\boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x},\epsilon)} \mathbf{1}\{\beta f(\boldsymbol{x}')f(\boldsymbol{x}) < 0\}\end{aligned} \tag{A.4}$$

Given example $\boldsymbol{x}$, adversarial example $\boldsymbol{x}^{*}$ and a continuous path $\gamma(\cdot)$ such that $\gamma(0) = \boldsymbol{x}$ and $\gamma(1) = \boldsymbol{x}^{*}$, following inequality holds:

$$\mathbf{1}\{\beta f(\boldsymbol{x}^{*})f(\boldsymbol{x}) < 0\} \leq \sum_{k=0}^{m-1} \mathbf{1}\{\beta f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})) \leq 0\} \tag{A.5}$$

where m is a hyper-parameter for dividing the path $\gamma(\cdot)$.

Proof of (A.5). First, it is clear when $f(\boldsymbol{x}^{*})f(\boldsymbol{x}) \geq 0$. Thus, we consider the case of $f(\boldsymbol{x}^{*})f(\boldsymbol{x}) < 0$. To prove the statement by contradiction, suppose that $1 = \mathbf{1}\{\beta f(\boldsymbol{x}^{*})f(\boldsymbol{x}) < 0\} > \sum_{k=0}^{m-1} \mathbf{1}\{\beta f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})) \leq 0\}$, or equivalently, $f(\gamma(\frac{k}{m}))$ $f(\gamma(\frac{k+1}{m})) > 0$ for all $k = 0, \dots, m-1$. In other words, we have the same sign values for every $(\frac{k}{m}, \frac{k+1}{m})$ pairs which leads a contradiction with the assumption that f has different sign values at the end points $\gamma(0) = \boldsymbol{x}$ and $\gamma(1) = \boldsymbol{x}^{*}$. □

Now, we establish a new upper bound on $\mathcal{R}_{rob}(f) - \mathcal{R}^{\star}_{nat}$.

Proof of Theorem 1 [Restated]. Given a sample $\boldsymbol{x}$ and a positive β, let $\gamma : [0, 1] \rightarrow \mathcal{X}$ be a continuous path from $\gamma(0) = \boldsymbol{x}$ to $\gamma(1) = \boldsymbol{x}^{*}$ where $\boldsymbol{x}^{*} = \arg\max_{\boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x},\epsilon)} \mathbf{1}\{\beta f(\boldsymbol{x}')f(\boldsymbol{x}) < 0\}$. Then, for any non-negative classification-calibrated loss function ϕ such that $\phi(0) \geq 1$, we have

$$\begin{aligned}\mathcal{R}_{rob}(f) - \mathcal{R}^{\star}_{nat} \leq& \psi^{-1}(\mathcal{R}_{\phi}(f) - \mathcal{R}^{\star}_{\phi}) \\ &+ \mathbb{E}_{(\boldsymbol{x},y)} \sum_{k=0}^{m-1} \phi(\beta f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})))\end{aligned}$$

where $\mathcal{R}^{\star}_{nat} := \inf_f \mathcal{R}_{nat}(f)$, $\mathcal{R}^{\star}_{\phi} := \inf_f \mathcal{R}_{\phi}(f)$ and ψ^{-1} is the inverse function of the ψ-transform of ϕ.

Proof. By (A.4), the first inequality holds. Similarly, the second inequality holds by (A.5), and the last inequality holds because we choose a classification-calibrated loss ϕ.

$$\begin{aligned}\mathcal{R}_{rob}(f) - \mathcal{R}^{\star}_{nat} &\leq \psi^{-1}(\mathcal{R}_{\phi}(f) - \mathcal{R}^{\star}_{\phi}) + \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\beta f(\boldsymbol{x}^{*})f(\boldsymbol{x}) < 0\} \\ &\leq \psi^{-1}(\mathcal{R}_{\phi}(f) - \mathcal{R}^{\star}_{\phi}) + \mathbb{E}_{(\boldsymbol{x},y)} \sum_{k=0}^{m-1} \mathbf{1}\{\beta f(\gamma(\frac{k}{m})) \\ &\quad \times f(\gamma(\frac{k+1}{m})) \leq 0\} \\ &\leq \psi^{-1}(\mathcal{R}_{\phi}(f) - \mathcal{R}^{\star}_{\phi}) + \mathbb{E}_{(\boldsymbol{x},y)} \sum_{k=0}^{m-1} \phi(\beta \\ &\quad \times f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m}))) \quad \square\end{aligned}$$

Following (Zhang et al., 2019), we use KL divergence loss (KL) as a classification-calibrated loss. To do this, we define $p(x) := \sigma(f(x))$ where σ is a sigmoid function. Then, a model output with softmax can be denoted as $\boldsymbol{p}(x) := [p(x), 1 - p(x)]$. In this setting, we can prove that the suggest loss is tighter than that of TRADES under a weak assumption on $\gamma(\cdot)$.

Assumption 1. *$[\boldsymbol{p}(\gamma(t))]_y$ is a decreasing function of $t \in [0, 1]$, where $[\boldsymbol{p}(\cdot)]_y$ indicates the probability corresponding to the correct label y.*

Theorem 2. *Under Assumption 1, the KL divergence loss has the following property:*

$$\sum_{k=0}^{m-1} KL(\boldsymbol{p}(\gamma(\frac{k}{m}))\|\boldsymbol{p}(\gamma(\frac{k+1}{m}))) \leq KL(\boldsymbol{p}(\gamma(0))\|\boldsymbol{p}(\gamma(1))).$$

Proof. Let $p_1(x)$, $p_2(x)$, and $p_3(x)$ denotes three different distribution with possible outcomes $x = \{-1, 1\}$ and $0 < p_1(x = 1) \leq p_2(x = 1) \leq p_3(x = 1) < 1$. Then,

$$\begin{aligned}&\mathrm{KL}(p_1||p_2) + \mathrm{KL}(p_2||p_3) - \mathrm{KL}(p_1||p_3) \\ &= \sum_{x\in\{0,1\}} p_1(x)\ln\frac{p_1(x)}{p_2(x)} + \sum_{x\in\{0,1\}} p_2(x)\ln\frac{p_2(x)}{p_3(x)} - \sum_{x\in\{0,1\}} p_1(x)\ln\frac{p_1(x)}{p_3(x)} \\ &= \sum_{x\in\{0,1\}} (p_2(x) - p_1(x))\ln p_2(x) + \sum_{x\in\{0,1\}} (p_1(x) - p_2(x))\ln p_3(x) \\ &= -\sum_{x\in\{0,1\}} (p_1(x) - p_2(x))(\ln p_2(x) - \ln p_3(x)) \\ &\leq 0\end{aligned}$$

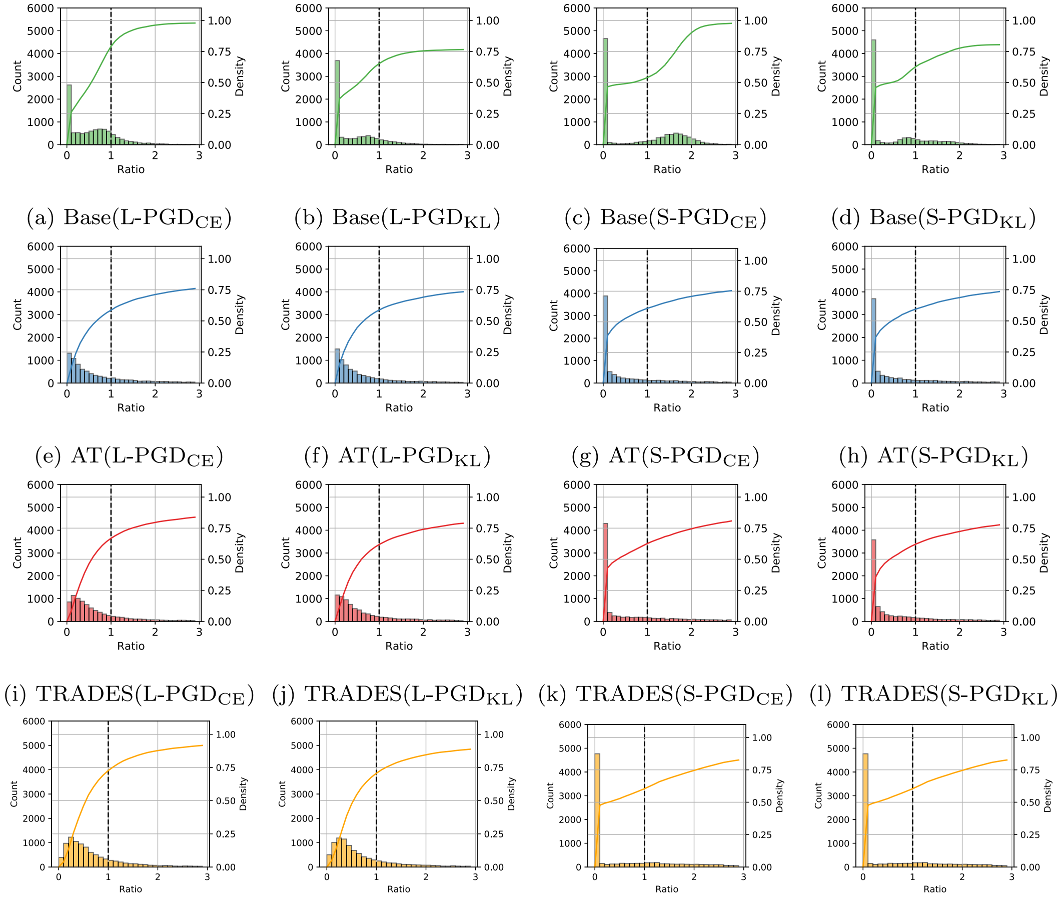
The last inequality holds because $p_1(x) - p_2(x)$ and $p_2(x) - p_3(x)$ have the same sign regardless of x. Likewise, for $0 < p_1(x = -1) \leq p_2(x = -1) \leq p_3(x = -1) < 1$, the statement also holds true. Thus, by mathematical induction, $\sum_{k=0}^{m-1} \mathrm{KL}(\boldsymbol{p}(\gamma(\frac{k}{m}))\| \boldsymbol{p}(\gamma(\frac{k+1}{m}))) \leq \mathrm{KL}(\boldsymbol{p}(\gamma(0))\|\boldsymbol{p}(\gamma(1)))$ under Assumption 1. □

For the multi-class problem, we can extend Theorem 2 by assuming $[\boldsymbol{p}(\gamma(u))]_i$ as a monotonic function for each individual component $i \in \mathcal{Y}$.

Appendix B. Ablation study on the path $\gamma(\cdot)$

The proposed method has the following hyper-parameters: the intermediate probability path γ, and the number of intermediate probabilities m, and the weight for the smoothness regularization term β. To analyze their effect of each hyper-parameter, we perform sensitivity analyses on CIFAR10 with $\epsilon = 8/255$.

In the main paper, we basically use the linear path as $\gamma(\cdot)$. However, we can also choose the step path; for example, if we



(a) Base(L-PGD$_{\text{CE}}$) (b) Base(L-PGD$_{\text{KL}}$) (c) Base(S-PGD$_{\text{CE}}$) (d) Base(S-PGD$_{\text{KL}}$)

(e) AT(L-PGD$_{\text{CE}}$) (f) AT(L-PGD$_{\text{KL}}$) (g) AT(S-PGD$_{\text{CE}}$) (h) AT(S-PGD$_{\text{KL}}$)

(i) TRADES(L-PGD$_{\text{CE}}$) (j) TRADES(L-PGD$_{\text{KL}}$) (k) TRADES(S-PGD$_{\text{CE}}$) (l) TRADES(S-PGD$_{\text{KL}}$)

(m) BAT(L-PGD$_{\text{CE}}$) (n) BAT(L-PGD$_{\text{KL}}$) (o) BAT(S-PGD$_{\text{CE}}$) (p) BAT(S-PGD$_{\text{KL}}$)

Fig. B.10. (CIFAR10) The ratio of the bridged KL divergence loss ($m = 2$) to the original KL divergence loss in (B.1). The bar chart shows the distribution of the ratio and the solid line denotes the cumulative distribution function.

use $m = 2$ and PGD^2 as an adversarial attack, given the original example $\boldsymbol{x}$, the first step adversarial example and the last (or second) step adversarial example $\boldsymbol{x}^*$ will be used as $\gamma(1/2)$ and $\gamma(1)$ instead of the linear path $\gamma(t) = (1-t)\boldsymbol{x} + t\boldsymbol{x}^*$.

In addition to the path, we can also decide how to generate adversarial example $\boldsymbol{x}^*$. We suggest two options; (1) PGD on the cross entropy loss $\max_{\boldsymbol{x}'\in\mathbb{B}(\boldsymbol{x},\epsilon)} \mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}'))$ (denoted as PGD$_{\text{CE}}$), (2) PGD on the KL divergence loss $\max_{\boldsymbol{x}'\in\mathbb{B}(\boldsymbol{x},\epsilon)} \mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})\|\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}'))$ (denoted as PGD$_{\text{KL}}$).

Thus, we consider four variations in total; (1) L-PGD$_{\text{CE}}$: the linear path and PGD$_{\text{CE}}$, (2) L-PGD$_{\text{KL}}$: the linear path and PGD$_{\text{KL}}$, (3) S-PGD$_{\text{CE}}$: the step path and PGD$_{\text{CE}}$, and (4) S-PGD$_{\text{KL}}$: the step path and PGD$_{\text{KL}}$. The method used in the main paper can be expressed as L-PGD$_{\text{CE}}$.

We first check each variation holds Theorem 2 for different m. To do this, we define a measure as follows:

$$\frac{\sum_{k=0}^{m-1} \mathrm{KL}(\boldsymbol{p}(\gamma(\frac{k}{m}))\|\boldsymbol{p}(\gamma(\frac{k+1}{m})))}{\mathrm{KL}(\boldsymbol{p}(\gamma(0))\|\boldsymbol{p}(\gamma(1)))} \tag{B.1}$$

For sufficient verification, we use the trained models by standard training without adversarial training (denoted as Base), AT, and TRADES. For each model and path $\gamma(\cdot)$, we use PGD^{10} to generate adversarial examples and plot the ratio in (B.1). As shown in Fig. B.10, for most cases, more than half of the samples satisfy Theorem 2. Especially, we observe 80% of samples have a ratio less than 1 for L-PGD$_{\text{CE}}$. However, in the case of S-PGD, we find that there are relatively many samples with a ratio of more than 1 for the standard trained model, e.g., Fig. B.10(c). This tells us L-PGD$_{\text{CE}}$ is more likely to hold Theorem 2.

Table B.8
(CIFAR10) Robustness of each path $\gamma(\cdot)$.

Method	Clean	FGSM	PGD^{50}	AutoAttack
L-PGD$_{\text{CE}}$	85.14	60.18	**56.01**	**52.41**
L-PGD$_{\text{KL}}$	**86.59**	59.71	53.69	51.37
S-PGD$_{\text{CE}}$	83.07	59.44	55.41	51.56
S-PGD$_{\text{KL}}$	84.53	**60.32**	54.25	51.89

The different behavior between standard training (Base) and other adversarial training methods (AT and TRADES) comes from that the standard trained model is easily attacked by an adversarial attack. Indeed, their probability vectors tend to be drastically changed by small perturbation (i.e., nearly zero robust accuracy) so that the ratio in (B.1) is near zero. In contrast, adversarial trained models tend to have smoother loss surfaces (Liu et al., 2020) than standard trained models so that the ratio in (B.1) monotonically increases.

We now compare their robustness with the same settings in Section 5.3. Here, we only performed each experiment once. All experiments are performed under $\epsilon = 8/255$. As shown in Table B.8, L-PGD$_{\text{CE}}$ shows the highest robustness with consideration of PGD and AutoAttack. We believe the reason why S-PGD does not achieve high performance as L-PGD is that S-PGD is more likely to violate Theorem 2 as illustrated in Fig. B.10. In the case of

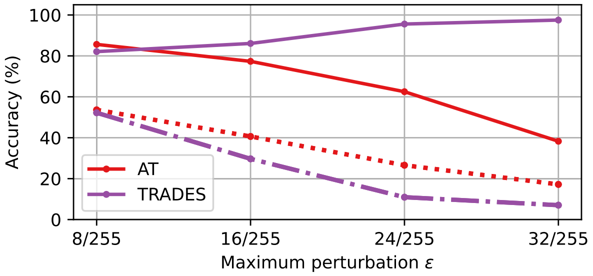


Fig. C.11. (CIFAR10) Stability of each method for a wide range of the maximum perturbation ϵ during training. Each model is trained on the given maximum perturbation ϵ and evaluated by using PGD^{50} with the same ϵ used during training. The dotted and solid lines indicate the clean accuracy and robust accuracy, respectively.

FGSM, S-PGD$_{\mathrm{KL}}$ shows better robustness, but we note that FGSM is a weaker version of PGD.

Computational costs. Notably, the path $\gamma(\cdot)$ also affect on the computational costs. For instance, when using S-PGD, BAT does not require any additional computation. This is because the path $\gamma(\cdot)$ is naturally calculated during the step-wise inner maximization. Therefore, the time consumption for BAT using S-PGD is same as other training methods, such as AT and TRADES. Indeed, we observed that BAT using S-PGD, AT, and TRADES all requires approximately 1.2 s per iteration. In contrast, when using L-PGD, which requires an additional forward computation to calculate intermediate probabilities, resulting the increased time consumption. However, compared to AT and TRADES, BAT using L-PGD only requires 1.3 s per iteration, demonstrating an approximately 8% increase in time.

Appendix C. Additional experiments

C.1. Extensive analysis on margin, smoothness, and gradient norm

Different optimization difficulty. In Fig. 3, we evaluate their clean and robust accuracy with varying maximum perturbations during the training on MNIST. Here, we conducted the same experiment on CIFAR10 and summarize the results in Fig. C.11. Similar to Fig. 3, AT has some drawbacks in convergence so that both clean accuracy and robust accuracy show extremely low values. In contrast, TRADES shows high clean accuracy but it fails to gain robustness for high ϵ. Thus, we can conclude that AT and TRADES also exhibit different behavior under the different dataset.

Different margin and smoothness. Here, to provide a rigorous verification of their different behavior in terms of margin and smoothness, we conduct additional analysis with different initialization and different training settings. Fig. C.12 shows that they show similar behavior observed in Fig. 2 even for different settings. Interestingly, Fig. C.12(b) exhibits extremely different behavior of AT and TRADES. We believe that this phenomenon might be caused by the convergence difficulty of adversarial training with respect to ϵ. In a recent study (Liu et al., 2020), it was discovered that larger values of ϵ can lead to more challenging convergence problems. In other words, both maximizing margin and improving smoothness become more difficult when $\epsilon = 16/255$ compared to $\epsilon = 8/255$. This leads to a more noticeable difference in the distribution of $M(\boldsymbol{x})$ and the KL divergence. This observation is also consistent with the behavior illustrated in Fig. 3, which highlights the distinct behavior of AT and TRADES under larger ϵ values (see Fig. C.13).

Effect on margin. In Fig. 6, we cut off the figure after 40 epochs, because it could clearly represent the effectiveness of our method, compared to TRADES and AT. Thus, we here extend Fig. 6 for whole training epochs in Fig. C.13. After 40 epochs, where the learning rate is decayed, for the effect of the gradient descent $\nabla_{\boldsymbol{\theta}}\ell$ on the margin maximization (measured by $-\nabla_{\boldsymbol{\theta}}M(\boldsymbol{x}) \cdot \nabla_{\boldsymbol{\theta}}\ell$), the proposed method shows the largest value after 40th epoch. Moreover, the margin $M(x)$ of the proposed method is also better than TRADES after 40th epoch. For the fact that the actual margin $M(x)$ of the proposed method is lower than AT, we believe that this can be explained by their reduced learning rate after 40th epoch. In summary, the result supports that the proposed method successfully mitigates the negative effect of $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^*)$ on maximizing the margin until the end of training.

Gradient norm. Following Liu et al. (2020), we visualized the normalized norm of gradient for the first 2000 batches in Fig. 7. In Fig. C.14, we plot the normalized norm of gradient until the end of training. Similar to Fig. 7, the proposed method shows the highest norm of gradient until the end of training.

C.2. Achieving both good margin and smoothness on MNIST

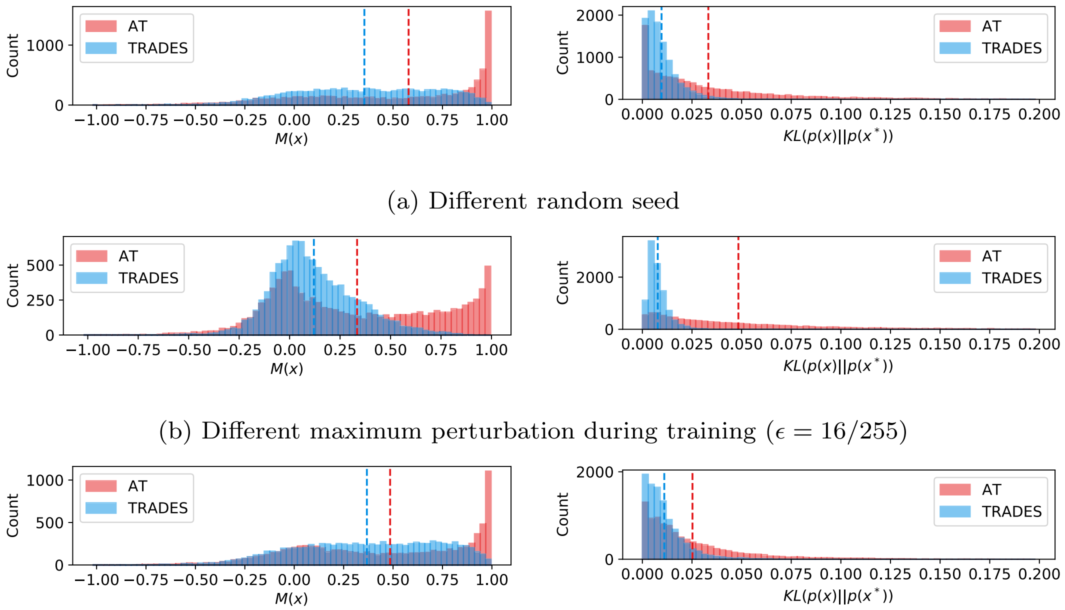
As Fig. 8, we check the quadrant plots for each method on MNIST. We note that for $\epsilon = 0.3$, all methods output near 98.8% of the standard accuracy so that it is hard to distinguish. Thus, we perform the experiment on $\epsilon = 0.35$ which is the maximum perturbation that AT does not converge to a constant function. All the other settings are remained the same. The result is demonstrated in Fig. C.15. Statistically, the average values of the margin of AT, TRADES and the proposed method are 0.948, 0.939 and 0.950, respectively. The average values of the smoothness of AT, TRADES and the proposed method are 0.674, 0.656, 0.416, respectively. Here again, TRADES and the proposed method shows a better smoothness than AT, while AT and the proposed method show a better margin than TRADES.

In addition, AT also shows the highest boundary error $\mathcal{R}_{bdy}$, 8.4%. The proposed method shows the highest robust accuracy 94.4% and the lowest boundary error 4.3%. Compared to TRADES, the proposed method shows better margin. Specifically, the ratio of clean examples with the margin $M(\boldsymbol{x})$ over 0.9 is 91.88% which is higher than that of TRADES (91.71%). Moreover, the proposed method also shows higher ratio of adversarial examples with the margin (79.71%) than that of TRADES (75.94%).

C.3. Step-wise learning rate decay on CIFAR10

For more reliable verification, we also perform an evaluation with step-wise learning rate decay. The learning rate is divided by 10 at epochs 40, 60, and 80. Furthermore, considering the recent work that uncovered the overfitting phenomenon in adversarial training (Rice et al., 2020), we select the best checkpoint by using PGD^{10} accuracy on the first batch of the test set. We summarize the performance of the final model and the best checkpoint model. We denote the best checkpoint model with early stopping as ES. As in Table C.9, almost all methods show improved performance against AutoAttack. TRADES with $\epsilon = 8/255$ is the only case which shows accuracy drop against AutoAttack. We presume that this is caused by using PGD accuracy to early stopping not AutoAttack.

TRADES shows better performance than AT without early stopping. This result is consistent with the results of the recent work (Rice et al., 2020). MART achieves the best robustness against PGD^{50}. However, against AutoAttack, MART shows a large decrease in robustness. Here, we note that MART seems to be overfitted to PGD as observed in Gowal et al. (2020). For $\epsilon = 8/255$, MART shows 53.2% accuracy against PGD with the best



(a) Different random seed

(b) Different maximum perturbation during training ($\epsilon = 16/255$)

(c) Different architecture (ResNet18)

Fig. C.12. (CIFAR10) Margin and smoothness of AT and TRADES for various settings (extended version of Fig. 2). AT always shows a higher margin than TRADES, while AT always shows a higher smoothness than TRADES.

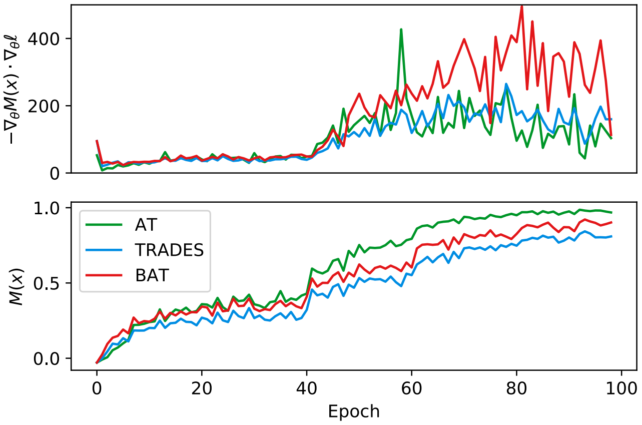


Fig. C.13. (CIFAR10) Analysis on the margin during training (extended version of Fig. 6).

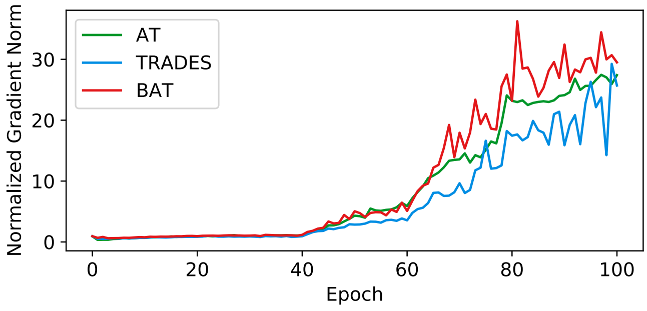


Fig. C.14. (CIFAR10) Normalized gradients of each loss term during training (extended version of Fig. 7).

checkpoint. Nevertheless, when we consider untargeted APGD and targeted APGD (Croce & Hein, 2020), the robustness decreases to 49.6% and 47.9%, respectively. This tendency becomes progressively worse as ϵ increases. The proposed model shows the best robust accuracy against AutoAttack as shown in Table C.9. Especially, for $\epsilon = 12/255$ and 16/255, the proposed method achieves the highest accuracy not only on the robustness but also on the standard accuracy.

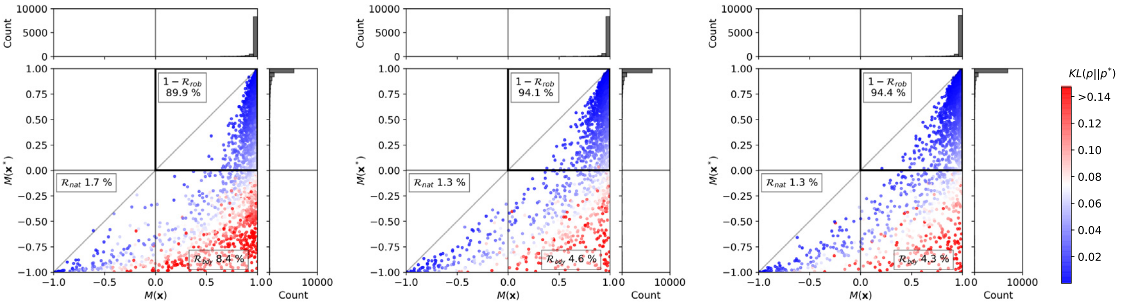
C.4. Comparison to the combined loss of AT and TRADES

In Section 5.1, we demonstrate that the proposed method enables the model to have balanced margin and smoothness (Fig. 8). Some can argue that a combine version of AT and TRADES may have the same effect as the proposed method. Thus, we here consider a combined loss function of AT and TRADES as follows:

$$\eta\{\mathrm{KL}(y\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))\} + (1-\eta)\cdot\{\mathrm{KL}(\boldsymbol{y}\|\boldsymbol{p}_\theta(\boldsymbol{x})) + \beta\mathrm{KL}(\boldsymbol{p}_\theta(\boldsymbol{x})\|\boldsymbol{p}_\theta(\boldsymbol{x}^*))\}. \tag{C.1}$$

where η is a hyper-parameter for adjusting the trade-off between the losses of AT and TRADES. We name this new adversarial training method ATRADES.

However, this combined version (ATRADES) is not efficient as the proposed method as shown in Table C.10. Here, we used the same setting in Section 5.3 on CIFAR10. Even we vary η in (C.1), the proposed method shows better performance than ATRADES. The proposed method achieves a higher clean accuracy and robust accuracy simultaneously compared to ATRADES with different η. Considering the standard deviation of the proposed method and ATRADES, their robust accuracy against AutoAttack are not overlapped, and this experimentally demonstrates that the proposed method brings statistically significant improvement in robustness. Indeed, with the theoretical analysis in Section 3, ATRADES cannot resolve the negative effect of the KL divergence



(a) AT (b) TRADES (c) BAT

Fig. C.15. Distribution of the margins $M(\boldsymbol{x})$ and $M(\boldsymbol{x}^*)$ on MNIST. Each point indicates each test example, and the color of each point indicates the KL divergence loss $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$. The darker red ones indicate a higher KL divergence loss.

Table C.9
Robustness accuracy (%) on CIFAR10. Each top line indicates the performance of the model at the end of training, and each bottom line (ES) indicates the performance of the best checkpoint by using early stopping.

Method	Clean	FGSM	PGD^{50}	AutoAttack
(Training $\epsilon = 8/255$)				
AT	86.85 ± 0.14	57.33 ± 0.23	45.33 ± 0.49	44.61 ± 0.45
(ES)	84.59 ± 0.01	59.19 ± 0.45	53.09 ± 0.13	50.39 ± 0.06
TRADES	86.37 ± 0.36	61.24 ± 0.30	51.81 ± 0.04	50.05 ± 0.03
(ES)	82.85 ± 0.71	57.73 ± 0.35	52.24 ± 0.59	49.25 ± 0.37
MART	84.79 ± 0.11	59.81 ± 0.73	49.91 ± 0.36	46.16 ± 0.08
(ES)	78.02 ± 0.99	57.19 ± 0.73	53.45 ± 0.14	48.19 ± 0.28
BAT	85.36 ± 0.01	57.53 ± 0.41	47.64 ± 0.20	46.28 ± 0.28
(ES)	84.02 ± 0.35	58.92 ± 0.03	55.13 ± 0.06	51.68 ± 0.31
(Training $\epsilon = 12/255$)				
AT	82.46 ± 0.17	47.73 ± 0.08	29.97 ± 0.04	27.98 ± 0.15
(ES)	78.29 ± 0.05	49.75 ± 0.41	39.77 ± 0.35	35.82 ± 0.17
TRADES	81.06 ± 0.59	50.59 ± 0.19	33.86 ± 1.44	25.21 ± 1.92
(ES)	78.17 ± 2.02	48.00 ± 2.03	37.27 ± 0.65	30.35 ± 3.30
MART	79.11 ± 0.23	51.23 ± 0.16	37.86 ± 0.85	31.74 ± 0.56
(ES)	72.04 ± 0.32	48.45 ± 1.03	41.82 ± 0.31	34.09 ± 0.87
BAT	81.53 ± 0.38	48.64 ± 0.23	33.42 ± 0.62	30.73 ± 0.42
(ES)	80.65 ± 0.04	49.47 ± 0.27	41.08 ± 0.40	36.03 ± 0.21
(Training $\epsilon = 16/255$)				
AT	79.10 ± 0.35	41.31 ± 0.71	19.05 ± 0.03	16.15 ± 0.11
(ES)	72.43 ± 0.01	42.48 ± 0.18	28.78 ± 0.16	24.23 ± 0.53
TRADES	76.69 ± 2.60	40.29 ± 1.94	23.83 ± 6.12	18.68 ± 4.65
(ES)	75.01 ± 0.46	41.30 ± 0.28	28.03 ± 0.06	21.86 ± 0.18
MART	76.04 ± 0.00	45.14 ± 0.18	26.53 ± 0.31	19.62 ± 0.52
(ES)	68.09 ± 0.45	41.83 ± 0.13	32.61 ± 0.40	23.36 ± 0.19
BAT	78.59 ± 0.43	42.12 ± 0.16	23.61 ± 0.43	19.95 ± 0.48
(ES)	76.06 ± 0.01	41.03 ± 0.43	30.91 ± 0.35	25.06 ± 0.34

Table C.10
Performance comparison between the proposed method (BAT) and the combined method (ATRADES). All models trained using PGD^{10} with $\epsilon = 8/255$, then evaluated by each attack with the same $\epsilon = 8/255$.

Method	Clean	PGD^{50}	AutoAttack
ATRADES($\eta =$ 0.25)	83.02 ± 0.21	55.65 ± 0.18	51.83 ± 0.06
ATRADES($\eta =$ 0.50)	83.58 ± 0.17	55.18 ± 0.07	51.24 ± 0.03
ATRADES($\eta =$ 0.75)	84.47 ± 0.20	55.50 ± 0.12	52.17 ± 0.04
BAT	**84.84 ± 0.28**	**55.95 ± 0.38**	**52.41 ± 0.02**

term and further have difficulty in converging because it uses the loss term of AT.

C.5. Considering other attacks

Here, we further consider various attacks: FGSM, CW and APGD (Croce & Hein, 2020) with the model trained in Section 5.3. Specifically, APGD mitigates the failures of PGD by using the adaptive step size and new loss function called difference of logits ratio (DLR). Through this step size-free optimization, APGD achieves a higher attack success rate than PGD. In addition, we also considered a targeted version of APGD (denoted as APGD-T), which uses the whole wrong labels $t \neq y$ as the target label.

Table C.11
Robustness accuracy (%) on CIFAR10 with various attacks. All models trained using PGD^{10} with $\epsilon = 8/255$, then evaluated by each attack with the same $\epsilon = 8/255$.

Method	FGSM	CW	PGD	APGD	APGD-T	AutoAttack
AT	**60.78**	**53.95**	53.64	**54.02**	50.89	50.87
TRADES	54.20	50.17	52.14	49.65	49.22	48.90
BAT	60.18	53.76	**55.64**	53.38	**53.15**	**52.41**

We summarize the result in Table C.11. While the proposed method shows the best robust accuracy against PGD, APGD-T, and AutoAttack, AT shows the best robust accuracy against FGSM, CW, and APGD (Croce & Hein, 2020). However, we emphasize that the gap between the robust accuracy of AT and the proposed method is very small (less than 0.6%p), compared to the robust accuracy against AutoAttack (more than 1.5%p). Moreover, the worst-case robust accuracy is always observed in AutoAttack (gray-colored) and the proposed method (52.41%) outperforms AT (50.87%) against AutoAttack.

Additionally, the strength of the proposed method can be observed in the comparison between APGD and APGD-T. In the case of AT, the robust accuracy against APGD is 54.02%. However, it dramatically decreases as we considered APGD-T. The robust accuracy against APGD-T is 50.82%, which is 3.2%p decrease compared to non-targeted version. In contrast, the proposed method shows the robust accuracy of 53.15%, which is only 0.2% dropped from the one against non-targeted APGD.

In summary, due to similar robustness against FGSM and CW and high robustness against targeted attack, the proposed method achieves the highest robustness against AutoAttack, which is the most powerful attack among the attacks in Table C.11.

C.6. Experiment with stochastic weight averaging

Stochastic weight averaging (SWA) (Izmailov et al., 2018) generally improves the accuracy in classical training and leads to better generalization. More importantly, Gowal et al. (2020) discovered that SWA also provides a consistent improvement in adversarial training. Thus, following the settings in (Gowal et al., 2020), we train ResNet18 using an exponential moving average of the model parameters with a decay rate $\tau = 0.995$ for 200 epochs. The initial learning rate of SGD is set to 0.1 and decayed with a factor of 0.1 at 100th and 150th epoch.

Table C.12
Robustness accuracy (%) with stochastic weight averaging on CIFAR10. All models trained using PGD^{10} with $\epsilon = 8/255$, then evaluated for $\epsilon = 8/255$, 12/255, and 16/255.

Method	Clean	PGD^{50}			AutoAttack		
		$\epsilon = 8/255$	12/255	16/255	8/255	12/255	16/255
AT-SWA	**82.98**	52.12	32.88	16.68	48.05	28.21	12.82
TRADES-SWA	82.30	51.93	34.75	20.42	48.61	30.35	15.93
MART-SWA	76.31	**53.98**	**40.40**	**31.30**	46.90	26.74	16.84
BAT-SWA	82.47	52.92	35.85	21.60	**48.80**	**31.45**	**17.09**

Table C.13
Robustness accuracy (%) on FMNIST. All models trained using PGD^{40} with $\epsilon = 0.1$ and 0.2, then evaluated for $\epsilon = 0.1, 0.15$, and 0.2.

Method	Clean	PGD^{50}			AutoAttack		
		$\epsilon = 0.1$	0.15	0.2	0.1	0.15	0.2
(Training $\epsilon = 0.1$)							
AT	85.76	76.23	35.70	6.74	74.15	24.69	2.66
TRADES	**85.89**	76.61	38.18	7.18	74.99	31.22	4.78
BAT	85.58	**77.07**	**42.23**	**10.86**	**75.18**	**34.84**	**7.04**
(Training $\epsilon = 0.2$)							
AT	80.04	73.08	69.87	66.71	70.73	66.43	60.86
TRADES	81.60	75.10	71.41	66.56	71.62	**68.03**	60.65
BAT	**81.70**	**75.68**	**72.70**	**69.57**	**71.81**	67.17	**61.25**

Table C.14
Robustness accuracy (%) on CIFAR100.

Method	Clean	FGSM	PGD^{50}	AutoAttack
AT	**55.92**	**30.13**	26.51	23.70
TRADES	54.56	29.82	26.48	23.28
BAT	53.67	29.96	**27.76**	**23.98**

As shown in Table C.12, the proposed method shows the best robustness. Specifically, even for large perturbations that never fed to the model during training phase, the proposed method outperforms AT and has better robustness than TRADES. In other words, the proposed method provides an improvement in the robustness even for unseen adversarial examples. Although BAT shows worse robust accuracy than MART against PGD50, we would like to highlight that BAT exhibits higher clean accuracy (82.47%) compared to MART (76.31%) and higher robust accuracy against AutoAttack (31.45%) than MART (26.74%). Based on this result, under the scheme of SWA, MART can be utilized than other methods when PGD is used as an adversarial attack. Notably, in a recent study, Gowal et al. (2020) uncovered that MART easily overfits to PGD adversarial examples, which can result in lower robust accuracy against AutoAttack, consistent with the observations in Table C.12. This phenomenon of MART remains largely unexplored and should be further analyzed in the future work.

While BAT may exhibit lower clean accuracy compared to AT in Table C.12, it demonstrates higher robust accuracy against AutoAttack. It is worth noting that the robust accuracy gap between AT and BAT is larger than the clean accuracy gap. Thus, we believe that BAT suggests potential future directions to overcome the tradeoff in adversarial training, which is a challenging aspect in adversarial training (Yang, Rashtchian et al., 2020; Zhang et al., 2019).

C.7. Experiment on various datasets

In the main paper, we mainly used MNIST and CIFAR10 for ease of comparison with prior studies (Croce et al., 2020; Madry et al., 2017). Here, to provide more reliable results, we also performed experiments on various datasets.

FMNIST. For FMNIST, we adopted the settings of MNIST in Section 5.3. we trained LeNet (LeCun et al., 1998) for 50 epochs with the Adam optimizer. The initial learning rate set to 0.001 and was divided by 10 at 30 and 40 epoch. following Liu, Khalil and Khreishah (2021), we basically used $\epsilon = 0.2$ as the maximum perturbation during training and we further considered $\epsilon = 0.1$. During training, we used PGD^{40} and the step-size $\alpha = 0.02$ to generate adversarial examples in the training session. No preprocessing or input transformation was used. For the robustness regularization hyper-parameter, we performed grid search on $\beta = \{5, 10\}$ and choose the best β that records the highest robustness against PGD^{50} with the same ϵ used during the training.

The result is summarized in Table C.13. The proposed method shows better performance than comparison methods for almost all cases. Specifically, the proposed method outperforms other methods against unseen adversarial examples. For training $\epsilon = 0.1$ and evaluation $\epsilon = 0.15$, the proposed method achieves the robust accuracy of 34.84% against AutoAttack, which is much higher than TRADES (31.22%) and AT (24.69%). We emphasize that the gap between the proposed method and AT is approximately 10% in this case. The proposed method also shows stable performance for training $\epsilon = 0.2$.

CIFAR100. Following (Gowal et al., 2020), we trained ResNet18 for 200 epochs with SGD, an initial learning rate of 0.1, momentum of 0.9, and weight decay of 5×10^{-4}. We used $\epsilon = 8/255$ and PGD^{10} to generate adversarial examples in the training session with the step-size of 2/255. We used step-wise learning rate decay divided by 10 at epochs 100 and 150. Horizontal flip and cropping are used for data augmentation. For the robustness regularization hyper-parameter, we performed grid search on $\beta = \{5, 10\}$ and choose the best β that records the highest robustness against PGD^{50} with $\epsilon = 8/255$.

As shown in Table C.14, the proposed method achieves highest robustness against PGD^{50} and AutoAttack. However, in contrast to the result on CIFAR10, the proposed method could not achieve a better standard accuracy. We expect that it is difficult to satisfy Assumption 1 for CIFAR100, because it has a higher output dimension.

References

Alayrac, J.-B., Uesato, J., Huang, P.-S., Fawzi, A., Stanforth, R., & Kohli, P. (2019). Are labels required for improving adversarial robustness? *Advances in Neural Information Processing Systems*, *32*, 12214–12223.

Andriushchenko, M., Croce, F., Flammarion, N., & Hein, M. (2020). Square attack: a query-efficient black-box adversarial attack via random search. In *European conference on computer vision* (pp. 484–501). Springer.

Anil, C., Lucas, J., & Grosse, R. (2019). Sorting out Lipschitz function approximation. In *International conference on machine learning* (pp. 291–301). PMLR.

Athalye, A., Carlini, N., & Wagner, D. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning* (pp. 274–283). PMLR.

Bartlett, P. L., Jordan, M. I., & McAuliffe, J. D. (2006). Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, *101*(473), 138–156.

Carlini, N., & Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *2017 Ieee symposium on security and privacy (Sp)* (pp. 39–57). IEEE.

Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J. C., & Liang, P. S. (2019). Unlabeled data improves adversarial robustness. In *Advances in neural information processing systems* (pp. 11192–11203).

Chen, J., Cheng, Y., Gan, Z., Gu, Q., & Liu, J. (2020). Efficient robust training via backward smoothing. arXiv preprint arXiv:2010.01278.

Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., & Usunier, N. (2017). Parseval networks: Improving robustness to adversarial examples. In *International conference on machine learning* (pp. 854–863). PMLR.

Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., et al. (2020). Robustbench: a standardized adversarial robustness benchmark. arXiv preprint arXiv:2010.09670.

Croce, F., & Hein, M. (2019). Minimally distorted adversarial examples with a fast adaptive boundary attack. arXiv preprint arXiv:1907.02044.

Croce, F., & Hein, M. (2020). Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning* (pp. 2206–2216). PMLR.

Ding, G. W., Sharma, Y., Lui, K. Y. C., & Huang, R. (2019). Mma training: Direct input space margin maximization through adversarial training. In *International conference on learning representations*.

Dong, Y., Xu, K., Yang, X., Pang, T., Deng, Z., Su, H., et al. (2021). Exploring memorization in adversarial training. arXiv preprint arXiv:2106.01606.

Elsayed, G., Krishnan, D., Mobahi, H., Regan, K., & Bengio, S. (2018). Large margin deep networks for classification. *Advances in Neural Information Processing Systems*, *31*, 842–852.

Fazlyab, M., Robey, A., Hassani, H., Morari, M., & Pappas, G. (2019). Efficient and accurate estimation of Lipschitz constants for deep neural networks. *Advances in Neural Information Processing Systems*, *32*, 11427–11438.

Foret, P., Kleiner, A., Mobahi, H., & Neyshabur, B. (2020). Sharpness-aware minimization for efficiently improving generalization. In *International conference on learning representations*.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.

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

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Identity mappings in deep residual networks. In *European conference on computer vision* (pp. 630–645). Springer.

Hein, M., & Andriushchenko, M. (2017). Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Proceedings of the 31st international conference on neural information processing systems* (pp. 2263–2273).

Izmailov, P., Wilson, A., Podoprikhin, D., Vetrov, D., & Garipov, T. (2018). Averaging weights leads to wider optima and better generalization. In *34th conference on uncertainty in artificial intelligence 2018, UAI 2018* (pp. 876–885).

Kannan, H., Kurakin, A., & Goodfellow, I. (2018). Adversarial logit pairing. arXiv preprint arXiv:1803.06373.

Kim, H. (2020). Torchattacks: A pytorch repository for adversarial attacks. arXiv preprint arXiv:2010.01950.

Kim, H., Lee, W., & Lee, J. (2021). Understanding catastrophic overfitting in single-step adversarial training. In *Proceedings of the AAAI conference on artificial intelligence, Vol. 35* (pp. 8119–8127).

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Lee, S., Lee, J., & Park, S. (2020). Lipschitz-certifiable training with a tight outer bound. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems, Vol. 33* (pp. 16891–16902). Curran Associates, Inc., URL: https://proceedings.neurips.cc/paper/2020/file/c46482dd5d39742f0bfd417b492d0e8e-Paper.pdf.

Lee, S., Lee, W., Park, J., & Lee, J. (2021). Towards better understanding of training certifiably robust models against adversarial examples. *Advances in Neural Information Processing Systems*, *34*.

Liu, F., Han, B., Liu, T., Gong, C., Niu, G., Zhou, M., et al. (2021). Probabilistic margins for instance reweighting in adversarial training. *Advances in Neural Information Processing Systems*, *34*.

Liu, G., Khalil, I., & Khreishah, A. (2021). Using single-step adversarial training to defend iterative adversarial examples. In *Proceedings of the eleventh ACM conference on data and application security and privacy* (pp. 17–27).

Liu, C., Salzmann, M., Lin, T., Tomioka, R., & Süsstrunk, S. (2020). On the loss landscape of adversarial training: Identifying challenges and how to overcome them. *Advances in Neural Information Processing Systems*, *33*.

Loshchilov, I., & Hutter, F. (2016). Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983.

von Luxburg, U., & Bousquet, O. (2004). Distance-based classification with Lipschitz functions. *Journal of Machine Learning Research*, *5*, 669–695.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083.

Najafi, A., Maeda, S.-i., Koyama, M., & Miyato, T. (2019). Robustness to adversarial perturbations in learning from incomplete data. In *Advances in neural information processing systems* (pp. 5541–5551).

Pang, T., Yang, X., Dong, Y., Su, H., & Zhu, J. (2020). Bag of tricks for adversarial training. arXiv preprint arXiv:2010.00467.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems* (pp. 8026–8037).

Rebuffi, S.-A., Gowal, S., Calian, D. A., Stimberg, F., Wiles, O., & Mann, T. (2021). Fixing data augmentation to improve adversarial robustness. arXiv preprint arXiv:2103.01946.

Rice, L., Wong, E., & Kolter, Z. (2020). Overfitting in adversarially robust deep learning. In *International conference on machine learning* (pp. 8093–8104). PMLR.

Ross, A. S., & Doshi-Velez, F. (2018). Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Thirty-second AAAI conference on artificial intelligence*.

Salman, H., Yang, G., Li, J., Zhang, P., Zhang, H., Razenshteyn, I., et al. (2019). Provably robust deep learning via adversarially trained smoothed classifiers. In *Proceedings of the 33rd international conference on neural information processing systems* (pp. 11292–11303).

Sanyal, A., Dokania, P. K., Kanade, V., & Torr, P. (2020). How benign is benign overfitting? In *International conference on learning representations*.

Shaeiri, A., Nobahari, R., & Rohban, M. H. (2020). Towards deep learning models resistant to large perturbations. arXiv preprint arXiv:2003.13370.

Sitawarin, C., Chakraborty, S., & Wagner, D. (2020). Improving adversarial robustness through progressive hardening. arXiv preprint arXiv:2003.09347.

Smith, L. N. (2017). Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)* (pp. 464–472). IEEE.

Sokolić, J., Giryes, R., Sapiro, G., & Rodrigues, M. R. (2017). Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, *65*(16), 4265–4280.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., et al. (2013). Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199.

Torralba, A., Fergus, R., & Freeman, W. T. (2008). 80 Million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(11), 1958–1970.

Tramer, F., Carlini, N., Brendel, W., & Madry, A. (2020). On adaptive attacks to adversarial example defenses. *Advances in Neural Information Processing Systems*, *33*.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., & McDaniel, P. (2017). Ensemble adversarial training: Attacks and defenses. arXiv preprint arXiv:1705.07204.

Wang, Z., Pang, T., Du, C., Lin, M., Liu, W., & Yan, S. (2023). Better diffusion models further improve adversarial training. arXiv preprint arXiv:2302.04638.

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., & Gu, Q. (2019). Improving adversarial robustness requires revisiting misclassified examples. In *International conference on learning representations*.

Wong, E., Rice, L., & Kolter, J. Z. (2020). Fast is better than free: Revisiting adversarial training. arXiv preprint arXiv:2001.03994.

Wu, D., Xia, S.-T., & Wang, Y. (2020). Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, *33*.

Xu, H., Caramanis, C., & Mannor, S. (2009). Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, *10*(7).

Yang, Y., Khanna, R., Yu, Y., Gholami, A., Keutzer, K., Gonzalez, J. E., et al. (2020). Boundary thickness and robustness in learning models. arXiv preprint arXiv:2007.05086.

Yang, Y.-Y., Rashtchian, C., Zhang, H., Salakhutdinov, R., & Chaudhuri, K. (2020). A closer look at accuracy vs. robustness. *Advances in Neural Information Processing Systems*.

Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. In E. R. H. Richard C. Wilson, & W. A. P. Smith (Eds.), *Proceedings of the British machine vision conference (BMVC)* (pp. 87.1–87.12). BMVA Press, http://dx.doi.org/10.5244/C.30.87.

Zhai, R., Cai, T., He, D., Dan, C., He, K., Hopcroft, J., et al. (2019). Adversarially robust generalization just requires more unlabeled data. arXiv preprint arXiv:1906.00555.

Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., & Jordan, M. (2019). Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning* (pp. 7472–7482). PMLR.