SWITCHING ONE-VERSUS-THE-REST LOSS TO INCREASE LOGIT MARGINS FOR ADVERSARIAL ROBUSTNESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial training is a promising method to improve the robustness against adversarial attacks. To enhance its performance, recent methods impose high weights on the cross-entropy loss for important data points near the decision boundary. However, these importance-aware methods are vulnerable to sophisticated attacks, e.g., Auto-Attack. In this paper, we experimentally investigate the cause of their vulnerability via margins between logits for the true label and the other labels because they should be large enough to prevent the largest logit from being flipped by the attacks. Our experiments reveal that the histogram of the logit margins of naïve adversarial training has two peaks. Thus, the levels of difficulty in increasing logit margins are roughly divided into two: difficult samples (small logit margins) and easy samples (large logit margins). On the other hand, only one peak near zero appears in the histogram of importance-aware methods, i.e., they reduce the logit margins of easy samples. To increase logit margins of difficult samples without reducing those of easy samples, we propose switching one-versus-the-rest loss (SOVR), which switches from cross-entropy to one-versus-the-rest loss (OVR) for difficult samples. We derive trajectories of logit margins for a simple problem and prove that OVR increases logit margins two times larger than the weighted cross-entropy loss. Thus, SOVR increases logit margins of difficult samples, unlike existing methods. We experimentally show that SOVR achieves better robustness against Auto-Attack than importance-aware methods.

1 INTRODUCTION

For multi-class classification problems, deep neural networks have become the de facto standard method in this decade. They classify a data point into the label that has the largest logit, which is input of a softmax function. However, the largest logit is easily flipped and deep neural networks can misclassify slightly perturbed data points, which are called adversarial examples (Szegedy et al., 2013). Various methods have been presented to search the adversarial examples, and Auto-Attack (Croce & Hein, 2020) is one of the most successful methods at finding the worst-case attacks. For trustworthy deep learning applications, classifiers should be robust against the worst-case attacks. To improve the robustness, many defense methods have also been presented (Kurakin et al., 2016; Madry et al., 2018; Wang et al., 2020b; Cohen et al., 2019). Among them, adversarial training is a promising method, which empirically achieves good robustness (Carmon et al., 2019; Kurakin et al., 2016; Madry et al., 2018). However, adversarial training is more difficult than standard training, e.g., it requires higher sample complexity (Schmidt et al., 2018; Wang et al., 2020a) and model capacity (Zhang et al., 2021b).

To alleviate the difficulties, several methods focus on the difference in importance of data points (Wang et al., 2020a; Liu et al., 2021; Zhang et al., 2021b). These studies hypothesize that data points closer to a decision boundary are more important for adversarial training (Wang et al., 2020a; Zhang et al., 2021b; Liu et al., 2021). To focus on such data points, GAIRAT (Zhang et al., 2021b) and MAIL (Liu et al., 2021) use weighted softmax cross-entropy loss, which controls weights on the losses on the basis of the closeness to the boundary. As the measure of the closeness, GAIRAT uses the least number of steps at which the iterative attacks make models misclassify the data point. On the other hand, MAIL uses the measure based on the softmax outputs. However, these importance-aware

methods fail to improve the robustness against Auto-Attack. Thus, it is still unclear *how to treat the difference in training data points in adversarial training for good robustness*.

In this paper, we experimentally investigate the cause of their vulnerability via margins between logits for the true label and the other labels because they should be large enough to prevent the largest logit from being flipped by the attacks. Our experiments show that the histogram of the logit margins of naïve adversarial training has two peaks, i.e., small and large logit margins. This indicates that the levels of difficulty in increasing the logit margins are roughly divided into two: *difficult samples* and *easy samples*. On the other hand, logit margins of importance-aware methods concentrate near zero, and thus, importance-aware methods reduce the logit margins of easy samples. This implies that the weighted cross-entropy used in importance-aware methods is not very effective in increasing logit margins. To increase the logit margins of difficult samples, we propose switching one-versus-the-rest loss (SOVR), which switches between cross-entropy and one-versus-the-rest loss (OVR) for easy and difficult samples, instead of weighting cross-entropy. We prove that OVR is always greater than or equal to cross-entropy on any logits. Furthermore, we theoretically derive the trajectories of logit margin losses in minimizing OVR and cross-entropy by using gradient flow on a simple problem and reveal that OVR increases logit margins two times larger than weighted cross-entropy losses. Experiments demonstrate that SOVR increases logit margins more than the naïve adversarial training and outperforms GAIRAT (Zhang et al., 2021b), MAIL (Liu et al., 2021), MART (Wang et al., 2020a), MMA (Ding et al., 2020), and EWAT (Kim et al., 2021) in terms of robustness against Auto-Attack. In addition, we find that our method improves the performance of other recent methods (Wu et al., 2020; Wang & Wang, 2022) that reduce generalization gap of adversarial training.

2 PRELIMINARIES

2.1 ADVERSARIAL TRAINING

Given N data points $\boldsymbol{x}_n \in \mathbb{R}^d$ and class labels $y_n \in \{1, \dots, K\}$, adversarial training (Madry et al., 2018) attempts to solve the following minimax problem with respect to the model parameter $\boldsymbol{\theta} \in \mathbb{R}^m$:

$$\min_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{AT}}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{n=1}^{N} \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}'_n, \boldsymbol{\theta}), y_n), \tag{1}$$

$$\boldsymbol{x}'_n = \boldsymbol{x}_n + \boldsymbol{\delta}_n = \boldsymbol{x}_n + \arg\max_{||\boldsymbol{\delta}_n||_p \leq \varepsilon} \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}_n + \boldsymbol{\delta}_n, \boldsymbol{\theta}), y_n), \tag{2}$$

where $\boldsymbol{z}(\boldsymbol{x}, \boldsymbol{\theta}) = [z_1(\boldsymbol{x}, \boldsymbol{\theta}), \dots, z_K(\boldsymbol{x}, \boldsymbol{\theta})]^T$ and $z_k(\boldsymbol{x}, \boldsymbol{\theta})$ is the k-th logit of the model, which is input of softmax: $f_k(\boldsymbol{x}, \boldsymbol{\theta}) = e^{z_k(\boldsymbol{x})} / \sum_i e^{z_i(\boldsymbol{x})}$. ℓ_{CE} is a cross-entropy function, and $||\cdot||_p$ and ε are L_p norm and the magnitude of perturbation $\boldsymbol{\delta}_n \in \mathbb{R}^d$, respectively. The inner maximization problem is solved by projected gradient descent (PGD) (Kurakin et al., 2016; Madry et al., 2018), which updates the adversarial examples as

$$\boldsymbol{\delta}_t = \Pi_\varepsilon \left(\boldsymbol{\delta}_{t-1} + \eta \mathrm{sign} \left(\nabla_{\boldsymbol{\delta}_{t-1}} \ell_{\mathrm{CE}} \left(\boldsymbol{z}(\boldsymbol{x} + \boldsymbol{\delta}_{t-1}, \boldsymbol{\theta}), y \right) \right) \right), \tag{3}$$

for $\mathcal{K}$ steps where η is a step size. Π_ε is a projection operation into the feasible region $\{\boldsymbol{\delta} \mid \boldsymbol{\delta} \in \mathbb{R}^d, ||\boldsymbol{\delta}||_p \leq \varepsilon\}$. Note that we focus on $p = \infty$ since it is a common setting. For trustworthy deep learning, we should improve the true robustness: the robustness against the worst-case attacks in the feasible region. Thus, the evaluation of robustness should use crafted attacks, e.g., Auto-Attack (Croce & Hein, 2020), since PGD often fails to find the adversarial examples misclassified by models.

2.2 IMPORTANCE-AWARE ADVERSARIAL TRAINING

GAIRAT (geometry aware instance reweighted adversarial training) (Zhang et al., 2021b) and MAIL (margin-aware instance reweighting learning) (Liu et al., 2021) regard data points closer to the decision boundary of the model $\boldsymbol{f}$ as important samples and assign higher weights to the loss for them:

$$\mathcal{L}_{\mathrm{weight}}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} \bar{w}_n \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}'_n, \boldsymbol{\theta}), y_n), \tag{4}$$

where $\bar{w}_n \geq 0$ is a weight normalized as $\bar{w}_n = \frac{w_n}{\sum_l w_l}$ and $\sum_n \bar{w}_n = 1$. GAIRAT determines the weights through the $w_n = \frac{1 + \tanh(\lambda + 5(1 - 2\kappa_n / \mathcal{K}))}{2}$ where κ_n is the least steps at which PGD succeeds at attacking models, and λ is a hyperparameter. On the other hand, MAIL uses $w_n = \mathrm{sigmoid}(-\gamma(PM_n - \beta))$ where $PM_n = f_{y_n}(\boldsymbol{x}'_n, \boldsymbol{\theta}) - \max_{k \neq y_n} f_k(\boldsymbol{x}'_n, \boldsymbol{\theta})$. β and γ are hyperparameters. MART (misclassification aware adversarial training) (Wang et al., 2020a) uses a similar approach.

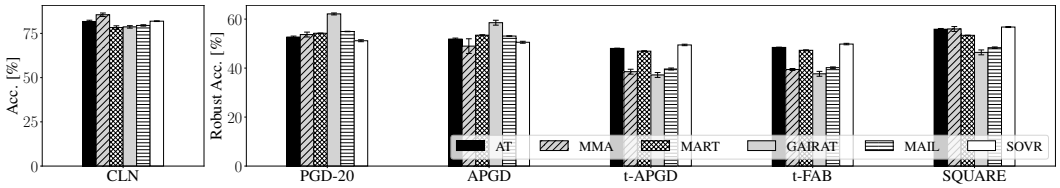


Figure 1: Robustness against PGD and components of Auto-Attack on CIFAR10 (Krizhevsky & Hinton, 2009) with PreActResNet18 (RN18). SOVR is our proposed method.

It regards misclassified samples as important samples and controls the difference between the loss on unimportant and important samples. MMA (max-margin adversarial training) (Ding et al., 2020) also adaptively changes the loss function and ε for each data point, and thus, MMA also has a similar effect to the above methods. We collectively call the above methods *importance-aware methods*.

2.3 Vulnerability of importance-aware methods to Auto-Attack

Hitaj et al. (2021); Croce & Hein (2020); Kim et al. (2021) have reported that the robust accuracies of GAIRAT, MART, and MMA are lower than naïve adversarial training when using logit scaling attacks or Auto-Attack (Croce & Hein, 2020). Since Auto-Attack searches adversarial examples by using various attacks, it achieves a larger success attack rate than using one attack, e.g., PGD. To clarify the vulnerabilities, we individually evaluate the robustness against the components of Auto-Attack and PGD ($\mathcal{K}=20$) on CIFAR10 (Fig. 1). The training setup is the same as in Section 6, and we add the results of our method (SOVR) as a reference. This figure shows that almost all importance-aware methods can improve the robustness against PGD and APGD compared with naïve adversarial training (AT (Madry et al., 2018)). However, they do not improve the true robustness; i.e., their robust accuracies against the worst-case attack are lower than that of AT. Since the reasons of this vulnerability have not been discussed well, we investigate them in the next section.

3 Evaluation of robustness via logit margin loss

We investigate the causes of the vulnerabilities of importance-aware methods by comparing histograms of logit margin losses. First, we explain that logit margin losses determine the robustness. Next, we experimentally reveal that logit margin losses of importance-aware methods concentrate on zero; i.e., their logit margins are smaller than AT. We use training data for empirical evaluation in this section because the goal of this section is to investigate the effect of importance-aware methods, which modify the loss function based on training data points. Experimental setups are provided in Appendix E.6.

3.1 Potentially misclassified data detected by logit margin loss

To investigate the robustness of models near each data point, we apply *logit margin loss* (Ding et al., 2020) to the models trained by importance-aware methods. Logit margin loss is

$$\ell_{\mathrm{LM}}(\boldsymbol{z}(\boldsymbol{x}_n,\boldsymbol{\theta}),y) = z_{k^*}(\boldsymbol{x}') - z_{y_n}(\boldsymbol{x}') = \max_{k\neq y} z_k(\boldsymbol{x}') - z_y(\boldsymbol{x}'), \tag{5}$$

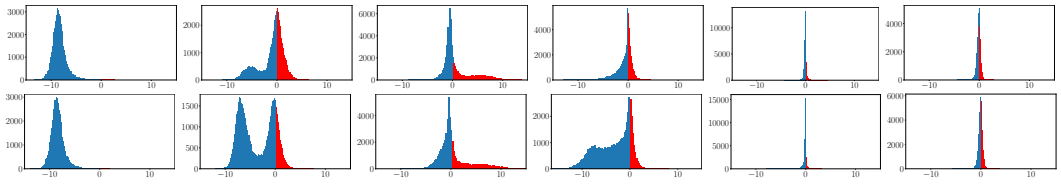
where $k^* = \mathrm{argmax}_{k\neq y} z_k(\boldsymbol{x}')$. Since the classifier infers the label of $\boldsymbol{x}$ as $\hat{y} = \arg\max_k z_k(\boldsymbol{x})$, it correctly classifies $\boldsymbol{x}'$ if $\ell_{\mathrm{LM}} \leq 0$. Thus, the logit margin loss on a difficult sample in adversarial training takes a value near zero. We refer to the absolute value of a logit margin loss $|\ell_{\mathrm{LM}}|$ as *logit margin*. In contrast to PM_n of MAIL, ℓ_{LM} is not bounded since $z_k(\boldsymbol{x})$ can take an arbitrary value in $\mathbb{R}$.

To explain the effect of logit margins, we assume that the Lipschitz constant of the k-th logit function is L_k as $|z_k(\boldsymbol{x}_1) - z_k(\boldsymbol{x}_2)| \leq L_k||\boldsymbol{x}_1 - \boldsymbol{x}_2||_\infty$. In this case, we have the following inequality:

$$\max_k z_k(\boldsymbol{x}') - z_y(\boldsymbol{x}') \leq \max_k [z_k(\boldsymbol{x}) - z_y(\boldsymbol{x}) + (L_k + L_y)\varepsilon] \leq z_{k*}(\boldsymbol{x}) - z_y(\boldsymbol{x}) + (L_{\hat{k}} + L_y)\varepsilon, \tag{6}$$

where $\hat{k} = \arg\max_k L_k$. From the above, we define the potentially misclassified sample:

Definition 3.1. If a data point $\boldsymbol{x}$ satisfies $z_{k^*}(\boldsymbol{x}) - z_y(\boldsymbol{x}) > -(L_{\hat{k}} + L_y)\varepsilon$, we call it a *potentially misclassified sample*.



(a) ST (b) AT (c) MMA (d) MART (e) GAIRAT (f) MAIL

Figure 2: Histogram of ℓ_{LM} for training data of CIFAR10 with RN18 at the best (top) and the last (bottom) epoch. Best epoch is the epoch when models achieved the best robust accuracy against PGD by early stopping. ST denotes standard training, i.e., training on clean data. For standard training, we use ℓ_{LM} on clean data $\boldsymbol{x}$, while we plot that on adversarial examples $\boldsymbol{x}'$ for the other methods. Blue bins are the correctly classified data points, and red bins are misclassified samples.

By using the above definition, we can derive the following:

Proposition 3.2. *If data points are not potentially misclassified samples, models are guaranteed to have the certified robustness on them as* $y = \arg\max_k z_k(\boldsymbol{x}+\boldsymbol{\delta})$ *for any* $\boldsymbol{\delta}$ *satisfying* $||\boldsymbol{\delta}||_\infty \leq \varepsilon$.

All proofs are provided in Appendix A. We can estimate the true robustness of each method by counting the number of potentially misclassified samples. Definition 3.1 and Proposition 3.2 indicate that large logit margins $|\ell_{\mathrm{LM}}|$ or small Lipschitz constants L_k are necessary for the robustness. Thus, the logit margin loss can be the metric of robustness, and we evaluate it in Section 3.2. In Section 6.2.1, we provide the estimated number of potentially misclassified samples for each method.

3.2 Histograms of logit margin loss

Since logit margin losses determine the number of potentially misclassified samples, we show the histogram of them for each method on CIFAR10 at the last epoch in Fig. 2. Comparing AT (Fig. 2(b)) with standard training (ST, Fig. 2(a)), AT has two peaks in the histogram. This indicates that the levels of difficulty in increasing the margins in AT are roughly divided into two: *difficult samples* (right peak) and *easy samples* (left peak). Difficult samples correspond to the data close to the boundary; i.e., important samples. Next, comparing AT (Fig. 2(b)) with importance-aware methods (Figs. 2(c)-(f)), their logit margin losses ℓ_{LM} concentrate on zero, and their peaks are sharper than that of AT. This indicates that importance-aware methods fail to increase the logit margins $|\ell_{\mathrm{LM}}|$ for not only difficult samples but also easy samples because the weights for easy samples are relatively small. Thus, it is necessary to increase the small logit margins for difficult samples without reducing those of easy samples. Appendix F.1 provides results under various settings, which show similar tendencies.

4 Proposed method

In Section 3.2, we observe that (a) training samples are roughly divided into two types via logit margins; difficult samples and easy samples, and (b) importance-aware methods reduce the logit margins on easy samples since they excessively focus on difficult samples. From these observations, our method is based on two ideas: (i) we switch from cross-entropy to an alternative loss for difficult samples by the criterion of the logit margin loss, and (ii) this alternative loss increases the logit margins of difficult samples more than weighted cross-entropy.

4.1 One-versus-the-rest loss (OVR)

The logit margin $|z_{k^*}(\boldsymbol{x}) - z_y(\boldsymbol{x})|$ should be large while keeping Lipschitz constants of logit functions small values. To this end, we need a loss function to penalize small logit margins. The logit margin loss can be an intuitive candidate as such a loss function. However, the logit margin loss only considers the pair of the largest logit z_{k^*} and the logit for the true label z_y, and this is not sufficient for the robustness because k^* and $\hat{k}$ in Eq. (6) are not necessarily the same. Moreover, the logit margin loss does not have the desirable property for multi-class classification: infinite sample consistent (ISC) (Zhang, 2004; Bartlett et al., 2003; Lin, 2002). To consider the logits for all classes and satisfy

ISC, our proposed method uses the one-versus-the-rest loss (OVR):

$$\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}),y)=\phi(z_y(\boldsymbol{x}))+\textstyle\sum_{k\neq y}\phi(-z_k(\boldsymbol{x})). \tag{7}$$

When ϕ is a differentiable non-negative convex function and satisfies $\phi(z)<\phi(-z)$ for $z>0$, OVR satisfies ISC (Zhang, 2004). As such a function, we set $\phi(z)=\log(1+e^{-z})$ and use the following loss:

$$\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}),y)=\log(1+e^{-z_y(\boldsymbol{x})})+\textstyle\sum_{k\neq y}\log(1+e^{z_k(\boldsymbol{x})})=-z_y(\boldsymbol{x})+\sum_k\log(1+e^{z_k(\boldsymbol{x})}). \tag{8}$$

We provide the detailed reason for this selection of $\phi(z)$ in Appendix D.

4.2 Behavior of logit margin losses by OVR and cross-entropy

To show the effectiveness of OVR in increasing logit margins, we theoretically discuss the difference between OVR and cross-entropy. First, OVR has the following property compared with cross-entropy:

Theorem 4.1. *If we use OVR (Eq. (8)) and softmax as $f_k(\boldsymbol{x})=e^{z_k(\boldsymbol{x})}/\sum_i e^{z_i(\boldsymbol{x})}$, we have*

$$0\leq\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}),y)\leq\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x}),y),\ \ \forall(\boldsymbol{x},y). \tag{9}$$

When $z_y(\boldsymbol{x})\to+\infty$ and $z_k(\boldsymbol{x})\to-\infty$ for $k\neq y$, we have $\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x}),y)\to0$ and $\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}),y)\to0$.

Thus, OVR is always larger than or equal to cross-entropy, and OVR and cross-entropy approach asymptotically to zero when $|\ell_{\mathrm{LM}}|$ grows to infinity. In fact, we observed that $\ell_{\mathrm{OVR}}(\boldsymbol{z},y)$ is about four times greater than $\ell_{\mathrm{CE}}(\boldsymbol{z},y)$ for random logits $\boldsymbol{z}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$ and randomly selected y from $\{1,\dots,10\}$. Thus, we expect OVR to penalize the small logit margin more strongly than cross-entropy.

Besides this general result, we further investigate the effect of OVR in logit margin losses by using a simple problem. To analyze the behavior of logit margins, we formulate the following problem:

$$\min_{\boldsymbol{z}} w\ell_*(\boldsymbol{z},y), \tag{10}$$

where ℓ_* is set to ℓ_{OVR} or ℓ_{CE}. $\boldsymbol{z}\in\mathbb{R}^K$ is a logit vector for a data point $\boldsymbol{x}$, and we assume that we can directly move it in this problem. $w\in\mathbb{R}$ is a weight of the loss, which appears in Eq. (4). To analyze the dynamics of training on Eq. (10), we use the following assumption.

Assumption 4.2. A logit vector $\boldsymbol{z}$ follows the following gradient flow to solve Eq. (10):

$$\tfrac{d\boldsymbol{z}}{dt}=-\nabla_{\boldsymbol{z}}w\ell_*(\boldsymbol{z},y), \tag{11}$$

where t is a time step of training. We assume that $\boldsymbol{z}$ is initialized to zeros $\boldsymbol{z}=\boldsymbol{0}$ at $t=0$.

Equation (11) is a continuous approximation of gradient descent $\boldsymbol{z}^{\tau+1}=\boldsymbol{z}^{\tau}-\eta\nabla_{\boldsymbol{z}}w\ell_*$ and matches it in the limit as $\eta\to0$. It is a commonly used method to analyze the dynamics of training (Kunin et al., 2021; Elkabetz & Cohen, 2021).

Under Assumption 4.2, we have the following lemmas about the logits in the training of Eq. (10):

Lemma 4.3. *If we use OVR $\ell_{\mathrm{OVR}}(\boldsymbol{z},y)$ in Eq. (10), the k-th logit $z_k(t)$ at time t is*

$$z_k(t)=\begin{cases}wt+1-W(e^{wt+1}) & k=y,\\ -wt-1-W(e^{wt+1}) & \text{otherwise},\end{cases} \tag{12}$$

where W is Lambert W function, which is a function satisfying $x=W(xe^x)$ (Corless et al., 1996).

Lemma 4.4. *If we use cross-entropy $\ell_{\mathrm{CE}}(\boldsymbol{z},y)$ in Eq. (10), the k-th logit $z_k(t)$ at time t is*

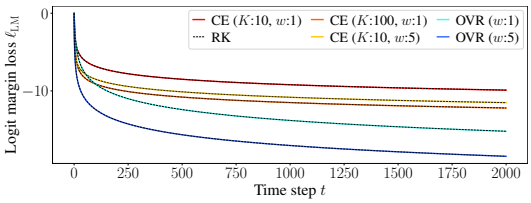
$$z_k(t)=\begin{cases}wt+\frac{1}{K}-\frac{K-1}{K}W(\frac{1}{K-1}e^{\frac{K}{K-1}wt+\frac{1}{K-1}}) & k=y,\\ -\frac{1}{K-1}wt-\frac{1}{K(K-1)}+\frac{1}{K}W(\frac{1}{K-1}e^{\frac{K}{K-1}wt+\frac{1}{K-1}}) & \text{otherwise}.\end{cases} \tag{13}$$

These lemmas give the trajectories of logit vectors in minimization of OVR and cross-entropy, respectively. By using the above lemmas, we derive the trajectory of the logit margin loss:

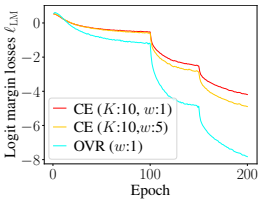
Theorem 4.5. *Logit margin losses for the logit vector $\boldsymbol{z}^{OVR}$ in the minimization of weighted OVR and logit vector $\boldsymbol{z}^{CE}$ in the minimization of weighted cross-entropy at time t are*

$$\ell_{\mathrm{LM}}(\boldsymbol{z}^{OVR}(t))=-2w_1t-2+2W(e^{w_1t+1}), \tag{14}$$

$$\ell_{\mathrm{LM}}(\boldsymbol{z}^{CE}(t))=-\tfrac{K}{K-1}w_2t-\tfrac{1}{K-1}+W(\tfrac{1}{K-1}e^{\frac{K}{K-1}w_2t+\frac{1}{K-1}}), \tag{15}$$



(a) Minimization of Eq. (10)



(b) CIFAR 10 (RN18)

Figure 3: Trajectories of $\ell_{\rm LM}$. CE denotes cross-entropy. In (a), RK denotes 4-th order Runge–Kutta method with the step size of 0.1. (b) is trajectory in adversarial training on CIFAR10, and its setup is provided in Appendix F.3

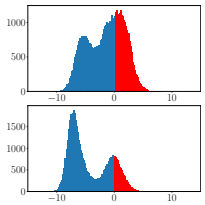


Figure 4: Histogram of $\ell_{\rm LM}$ of SOVR at the best (top) and the last (bottom) epoch.

where $w_1 \in \mathbb{R}$ and $w_2 \in \mathbb{R}$ are weights for OVR and cross-entropy, respectively. For large t, they are approximated by

$$\ell_{\rm LM}(\boldsymbol{z}^{OVR}(t)) \approx -\log(w_1 t + 1)^2, \tag{16}$$

$$\ell_{\rm LM}(\boldsymbol{z}^{CE}(t)) \approx -\log(K w_2 t + 1 - (K-1)\log(K-1)), \tag{17}$$

and we have $\lim_{t\to\infty} \frac{\ell_{\rm LM}(\boldsymbol{z}^{OVR})(t)}{\ell_{\rm LM}(\boldsymbol{z}^{CE})(t)} = 2$ for any fixed w_1, w_2, and K.

This theorem shows the difference in trajectories of logit margin losses between OVR and cross-entropy under Assumption 4.2. Regardless of the values of weights w_1 and w_2, cross-entropy does not increase the logit margins as large as OVR for sufficiently large t. Thus, OVR increases the small logit margins more effectively than GAIRAT and MAIL, which use weighted cross-entropy (Eq. (4)).

Figure 3(a) plots the trajectory of logit margin losses $\ell_{\rm LM}$ in the minimization of Eq. (10). This figure shows the solutions in Theorem 4.5 (solid lines): we use Eqs. (14) and (15) unless overflow occurs due to exponential functions and use Eqs. (16) and (17) when it occurs. It also plots numerical solutions of Eq. (11) by using gradients and the Runge–Kutta method as a reference (dashed lines). In Fig. 3(a), Eqs. (14)-(17) exactly match the numerical solutions of RK, and thus, logit margins follow Theorem 4.5. In addition, Fig. 3(a) shows that OVR decreases logit margin losses more than cross-entropy against t regardless of K and w. Thus, using OVR is more suitable for increasing the logit margin on difficult data than previous weighting approaches like GAIRAT and MAIL. Figure 3(b) plots trajectories of $\ell_{\rm LM}$ in adversarial training (Eq. (1)) on CIFAR10. It shows that the logit margin $|\ell_{\rm LM}|$ of OVR is about twice as large as that of cross-entropy at the last epoch ($\frac{\ell_{\rm LM}(\boldsymbol{z}^{OVR})}{\ell_{\rm LM}(\boldsymbol{z}^{CE})} = 1.87$ for $w = 1$), like the case of Theorem 4.5. Thus, problem Eq. (10) is simple but precise enough to explain the difference in the logit margins between OVR and cross-entropy on a real dataset. In fact, $\frac{\ell_{\rm LM}(\boldsymbol{z}^{OVR})}{\ell_{\rm LM}(\boldsymbol{z}^{CE})}$ at the last epoch is in [1.5, 2] on other datasets, including CIFAR100 ($K = 100$) (Appendix F.3).

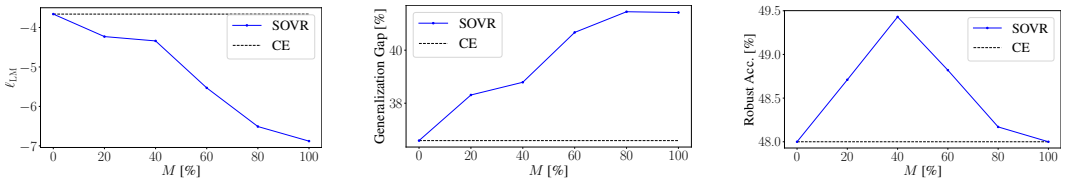
As above, we prove that OVR is more effective for increasing logit margins of difficult samples than weighting cross-entropy. In the next section, we compose the objective functions switching between OVR and cross-entropy for difficult and easy samples.

4.3 Proposed objective function: SOVR

Our proposed objective function is

$$\mathcal{L}_{\rm SOVR}(\boldsymbol{\theta}) = \frac{1}{N}\left[\sum_{(\boldsymbol{x},y)\in\mathbb{S}} \ell_{\rm CE}(\boldsymbol{z}(\boldsymbol{x}',\boldsymbol{\theta}),y) + \lambda \sum_{(\boldsymbol{x},y)\in\mathbb{L}} \ell_{\rm OVR}(\boldsymbol{z}(\boldsymbol{x}',\boldsymbol{\theta}),y)\right], \tag{18}$$

where $\mathbb{S}$ is a set where logit margin losses ℓ_{LM} are smaller than those in the set $\mathbb{L}$, and we have $|\mathbb{S}|+|\mathbb{L}| = N$. These sets correspond to easy and difficult samples in Fig. 2(b). In our method, we regard the top M % data points in minibatch of SGD as the samples in $\mathbb{L}$. λ is a hyperparameter to balance the loss, and $\boldsymbol{x}'$ is an adversarial example generated by Eq. (2). The proposed algorithm is shown in Appendix B. Since we do not additionally generate the adversarial examples for $\ell_{\rm OVR}$, the overhead of our method is negligible: $O(b \log \frac{M}{100} b)$ where b is minibatch size. In the same way to Section 3.2, we evaluate the histograms of logit margin losses for SOVR on CIFAR10 in Fig. 4. It shows that SOVR succeeds at increasing the left peak compared with AT (Fig. 2(b)). This is because OVR strongly penalizes difficult samples in the right peak and moves them into the left peak.



(a) $\ell_{\mathrm{LM}}(\boldsymbol{x}')$ (b) Generalization Gap (c) Robust Acc.

Figure 5: The effect of rate of applying OVR M. λ is set to 0.4. $M = 0$ corresponds to the result of AT with cross-entropy. Generalization gap is a gap between training robust accuracy and test robust accuracy against PGD ($\mathcal{K}$=20) at the last epoch. Robust Acc. is robust accuracy against Auto-Attack.

Though OVR increases logit margins as explained in Section 4.2, we found that OVR ($M = 100$) is inferior to SOVR because OVR for easy samples can cause overfitting. Figure 5 plots the effect of M in terms of ℓ_{LM} at the last epoch, generalization gap at the last epoch, and robust accuracy against Auto-Attack on CIFAR10. It shows that ℓ_{LM} monotonically decreases, i.e., robustness improves, when increasing M. However, the generalization gap increases at the same time. Robust accuracy takes the largest value at $M = 40$. Thus, it is necessary to switch losses to focus on difficult (important) samples, like existing importance-aware methods. We provide the evaluation of effects of λ in Appendix F.4, which shows the similar tendencies.

4.4 Extension for other defense methods

Since SOVR only modifies the objective function, it can be used with the optimization algorithms for robustness, e.g., adversarial weight perturbation (AWP) (Wu et al., 2020) or self-ensemble adversarial training (SEAT) (Wang & Wang, 2022), which improve generalization performance in adversarial training. However, SOVR is difficult to use with TRADES (Zhang et al., 2019) because TRADES also modifies the objective function. To combine our method with TRADES, we propose TSOVR, which uses SOVR instead of cross-entropy for clean data:

$$\mathcal{L}_{\mathrm{TSOVR}}(\boldsymbol{\theta}) = \frac{1}{N}\left[\mathcal{L}_{\mathrm{SOVR}} + \sum_{n=1}^{N} \beta_T \max_{||\boldsymbol{\delta}_n||_p \leq \varepsilon} \mathrm{KL}(\boldsymbol{f}(\boldsymbol{x}_n, \boldsymbol{\theta}), \boldsymbol{f}(\boldsymbol{x}_n + \boldsymbol{\delta}_n, \boldsymbol{\theta}))\right]. \tag{19}$$

$$\mathcal{L}_{\mathrm{SOVR}} = \left[\sum_{(\boldsymbol{x},y)\in\mathbb{S}} \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}, \boldsymbol{\theta}), y) + \lambda \sum_{(\boldsymbol{x},y)\in\mathbb{L}} \ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x}, \boldsymbol{\theta}), y)\right] \tag{20}$$

where λ and β_T are hyperparameters. We evaluate the combinations of SOVR with TRADES, AWP (Wu et al., 2020), and SEAT (Wang & Wang, 2022) in the experiments.

5 Related work

The difference in importance of data points in adversarial training has been investigated in several studies (Wang et al., 2020a; Zhang et al., 2020; Sanyal et al., 2021; Dong et al., 2022). Zhang et al. (2020) investigated the effect of difficult samples on natural generalization performance; i.e., generalization performance on clean data, in adversarial training. They presented friendly adversarial training (FAT) that improves the robustness without compromising the natural generalization performance. Unlike FAT, SOVR focuses on the robust performance. Sanyal et al. (2021) and Dong et al. (2022) have investigated the effect of memorization in adversarial training: memorizing difficult samples hurts the generalization performance of adversarial training. Whereas they focused on reducing generalization gap by regularization, our method reduces robust error on test data by reducing training robust error; i.e., mitigating underfitting more than overfitting. Cisse et al. (2017); Tsuzuku et al. (2018); Zhang et al. (2021a) used logit margins and Lipschitz constants to present certified defense methods. We present a similar result only to justify our evaluation using logit margin loss in Section 3. Though Padhy et al. (2020) used OVR for OOD detection, they did not discuss the effect of OVR in logit margins.

Studies of (Hitaj et al., 2021; Croce & Hein, 2020; Kim et al., 2021) have reported the vulnerabilities of some importance-aware methods to logit scaling attacks or Auto-Attack but few studies discuss the causes. Kim et al. (2021) have pointed out that the cause is high entropy in GAIRAT and MART and presented EWAT (entropy-weighted adversarial training), which imposes a higher weight on the higher entropy. However, the logit margin is more related to robustness than entropy as discussed in Section 3. Furthermore, weighted cross-entropy in EWAT is less effective than OVR as shown Theorem 4.5.

Table 1: Robust accuracy against Auto-Attack and clean accuracy on test datasets. Robust accuracies of SOVR is statistically significantly different.

Robust Accuracy against Auto-Attack (L_∞, $\varepsilon = 8/255$)							
	AT	MART	MMA	GAIRAT	MAIL	EWAT	SOVR
CIFAR10 (RN18)	48.0±0.2	46.9±0.3	37.2±0.9	37.7± 1	39.6±0.4	48.2±0.7	$\mathbf{49.4 \pm 0.3}$
CIFAR10 (WRN)	51.9±0.5	50.44±0.09	43.1±1	41.8±0.6	43.3±0.1	51.6±0.3	$\mathbf{53.1 \pm 0.2}$
SVHN (RN18)	45.6±0.4	46.9±0.3	41.0±1	37.6±0.6	41.2±0.3	47.6±0.4	$\mathbf{48.5 \pm 0.4}$
CIFAR100 (RN18)	23.7±0.3	23.9±0.1	18.4±0.2	19.8±0.5	16.7±0.3	23.52±0.06	$\mathbf{24.3 \pm 0.2}$
Clean Accuracy							
CIFAR10 (RN18)	81.6±0.5	78.3±1	$\mathbf{85.5 \pm 0.7}$	78.7± 0.7	79.5±0.4	82.8±0.4	81.9±0.2
CIFAR10 (WRN)	85.6±0.1	81.5±1	$\mathbf{87.8 \pm 1}$	83.0±0.7	82.2±0.4	86.0±0.5	85.0±0.2
SVHN (RN18)	89.8±0.6	86.9±0.6	$\mathbf{93.9 \pm 0.4}$	89.9±0.4	89.4±0.4	90.2±0.6	90.0±1
CIFAR100 (RN18)	53.0±0.7	49.2±0.1	$\mathbf{60.6 \pm 0.6}$	52.0±0.5	46.5±0.5	54.2±1	52.1±0.8

6 EXPERIMENTS

6.1 SETUP

We conducted the experiments for evaluating SOVR. We first compare SOVR with Madry's AT (Madry et al., 2018), MMA (Ding et al., 2020), MART (Wang et al., 2020a), GAIRAT (Zhang et al., 2021b), MAIL (Liu et al., 2021), and EWAT (Kim et al., 2021) on three datasets: CIFAR10, SVHN, and CIFAR100 (Krizhevsky & Hinton, 2009; Netzer et al., 2011). Next, we evaluate the combination of SOVR with TRADES (Zhang et al., 2019), AWP (Wu et al., 2020), and SEAT (Wang & Wang, 2022). Our experimental codes are based on source codes provided by (Wu et al., 2020; Wang et al., 2020a; Ding et al., 2020). We used PreActResNet-18 (RN18) (He et al., 2016) for all datasets and WideResNet-34-10 (WRN) (Zagoruyko & Komodakis, 2016) for CIFAR10. We used PGD ($\mathcal{K} = 10$, $\eta = 2/255$, $\varepsilon = 8/255$) in training. We used early stopping by evaluating test robust accuracies against PGD with $\mathcal{K} = 10$. For AWP and SEAT, we use the original public codes (Wu et al., 2020; Wang & Wang, 2022).[1] For AT+AWP and SOVR+AWP, we use the training setup that is used for TRADES+AWP in Wu et al. (2020) by changing losses because we found that it achieves a better result. We trained models three times and show the average and standard deviation of test accuracies. For hyperparameters in SOVR, we set (M, λ) to $(40, 0.4)$ for CIFAR10 (RN18), $(30, 0.4)$ for CIFAR10 (WRN) and $(50, 0.5)$ for CIFAR100, and $(20, 0.2)$ for SVHN. We set (M, λ) to $(20, 0.8)$ for TSOVR and $(100, 1.2)$ for TSOVR+AWP. We use Auto-Attack to evaluate the robust accuracy on test data. In Appendix E and F, we provide the details of setups and additional results, e.g., evaluation using other various attacks. To determine the statistical significant difference, we use t-test with p-value of 0.05.

6.2 RESULTS

We list the robust accuracy against Auto-Attack on all datasets in Tab. 1. In this table, SOVR outperforms the importance-aware methods and AT in terms of the robustness against Auto-Attack. This is because SOVR increases the logit margins $|\ell_{\mathrm{LM}}|$ by using OVR. In fact, Fig. 4 shows that SOVR increases the logit margins $|\ell_{\mathrm{LM}}|$ for difficult samples. MART improved robustness on SVHN and CIFAR100. This might be because MART does not just impose weights on the loss. However, its improvement is less than SOVR. EWAT also achieves higher robust accuracies than AT on several datasets in Tab. 1. However, EWAT is not as robust as SOVR because EWAT employs weighted cross-entropy, which is less effective at increasing logit margins than OVR (Theorem 4.5). In Tab. 1, SOVR slightly sacrifices clean accuracies under some settings. We provide the histograms of logit margin losses on all datasets in Appendix F.1, which also show SOVR increases margins.

6.2.1 EMPIRICAL EVALUATION OF POTENTIALLY MISCLASSIFIED SAMPLES

As discussed in Section 3.1, we can evaluate the robustness near each data point via logit margins and Lipschitz constants of the class-wise logit functions. In this section, we estimate the number of potentially misclassified samples for each method. Since Lipschitz constants for deep neural networks

[1] We could not reproduce the results reported in Wu et al. (2020); Wang & Wang (2022) even though we did not modify their codes. This might be because we report the averaged values for reproducibility.

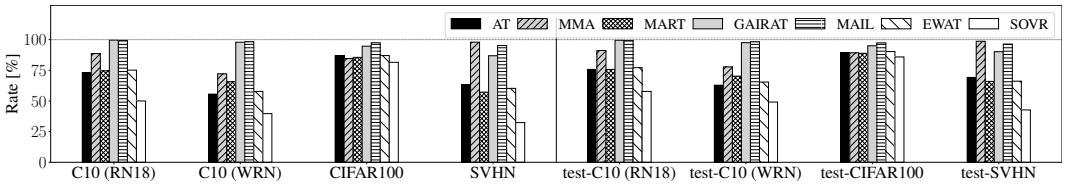


Figure 6: Rate of data satisfying Eq. (21). C10 and test- represent CIFAR10 and test data, respectively.

Table 2: Robust accuracy against Auto-Attack (L_∞, $\varepsilon = 8/255$) on test dataset of CIFAR10. SEAT uses WideResNet32-10 following Wang & Wang (2022)

	AWP (WRN)		TRADES (WRN)				SEAT (WRN32-10)	
	AT+AWP	SOVR+AWP	TRADES	TSOVR	TRADES+AWP	TSOVR+AWP	SEAT	+SOVR
Robust Acc.	55.0±0.2	56.03±0.07	52.9±0.3	53.3±0.1	55.6±0.4	56.4±0.2	54.9±0.1	55.5±0.3
Clean Acc.	87.8±0.1	86.9±0.4	84.2±0.7	83.1±0.2	84.9±0.5	84.9±0.4	86.8±0.4	86.5±0.1

are difficult to compute due to the complexity, we compute the gradient norm of the logit function instead of Lipschitz constants. This is because the gradient norm satisfies $\sup_{\boldsymbol{x}} ||\nabla_{\boldsymbol{x}} z_k(\boldsymbol{x})||_1 = L_k$ for L_k such as $|z_k(\boldsymbol{x}_1) - z_k(\boldsymbol{x}_2)| \leq L_k ||\boldsymbol{x}_1 - \boldsymbol{x}_2||_\infty$ (Jordan & Dimakis, 2020), and we have

Proposition 6.1. *If a data point $\boldsymbol{x}$ satisfies*

$$z_{k^*}(\boldsymbol{x}) - z_y(\boldsymbol{x}) > -(\max_k ||\nabla_{\boldsymbol{x}} z_k(\boldsymbol{x})||_1 + ||\nabla_{\boldsymbol{x}} z_y(\boldsymbol{x})||_1)\varepsilon, \tag{21}$$

it is a potentially misclassified sample.

Thus, we can empirically estimate the number of potentially misclassified samples on each method by the gradient norms. Figure 6 plots the rate of data points that satisfy Eq. (21), which are potentially misclassified samples. Comparing Fig. 6 with Tab. 1, when the methods have the large rate on test data, they have low robust accuracies against Auto-Attack. This indicates that this rate is a reasonable metric for estimating robustness though it uses gradient norms instead of Lipschitz constants. Whereas most importance-aware methods have higher rates than AT due to small logit margins, SOVR has lower rate. This is because SOVR increases logit margins without increasing gradient norms by using OVR, which is more effective in increasing logit margins than weighted cross-entropy (Section 4.2). In Fig. 6, EWAT does not necessarily increase the rate because EWAT uses weighted cross-entropy. We discuss the reason the rate of AT gets close to SOVR on CIFAR100 in Appendix F.10.

6.2.2 Extension for other methods

To improve the robust accuracy on test data, our method mostly focuses on improving the robustness around training data points rather than regularization. Even so, SOVR can be used with recent regularization methods (Wu et al., 2020; Wang & Wang, 2022), which sometimes sacrifice the training accuracy to reduce the generalization gap. We evaluated the combination of OVR and TRADES (TSOVR), SOVR and AWP, SOVR and SEAT. Table 2 lists the robust accuracy of the combinations against Auto-Attack and shows that SOVR improved the performance of other recent methods. Thus, SOVR and these methods complementarily improve the performance. Among AT, SOVR, TRADES, and TSOVR, SOVR achieved the best trade-off: SOVR achieved similar robust accuracies to TRADES, while it achieved better clean accuracy (Appendix F.6 provides the detailed discussion about trade-off.). We compare them on other datasets in Appendix F.9. TSOVR+AWP achieved the best robust accuracy, which is statistically significantly different from TRADES+AWP. Note that whereas AWP and SEAT require the time and space complexity depending on model sizes, respectively, the overhead of SOVR depends on only batch-size, and thus, it is easy to use with other methods.

7 Conclusion

We investigated the reason importance-aware methods fail to improve the robustness against Auto-Attack. Our empirical results showed the reason to be that they reduce logit margins of easy samples besides those of difficult samples. From the observation, we proposed SOVR, which switches from cross-entropy to OVR by the criterion of the logit margin loss. We theoretically showed OVR increases logit margins more than cross-entropy for a simple problem and experimentally showed that SOVR increases the margins and improves the robustness.

REFERENCES

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proc. ICML*, pp. 274–283, 2018.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. 2003.

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

Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. In *Proc. NeurIPS*, pp. 11190–11201, 2019.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proc. ICML*, pp. 854–863, 2017.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *Proc. ICML*, pp. 1310–1320, 2019.

Robert M Corless, Gaston H Gonnet, David EG Hare, David J Jeffrey, and Donald E Knuth. On the lambertw function. *Advances in Computational mathematics*, 5(1):329–359, 1996.

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

Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. In *Proc. ICLR*, 2020.

Yinpeng Dong, Ke Xu, Xiao Yang, Tianyu Pang, Zhijie Deng, Hang Su, and Jun Zhu. Exploring memorization in adversarial training. In *Proc. ICLR*, 2022.

Omer Elkabetz and Nadav Cohen. Continuous vs. discrete optimization of deep neural networks. *Proc. NeurIPS*, 34:4947–4960, 2021.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, pp. 770–778, 2016.

Dorjan Hitaj, Giulio Pagnotta, Iacopo Masi, and Luigi V Mancini. Evaluating the robustness of geometry-aware instance-reweighted adversarial training. *arXiv preprint arXiv:2103.01914*, 2021.

Abdolhossein Hoorfar and Mehdi Hassani. Approximation of the lambert w function and hyperpower function. *Research report collection*, 10(2), 2007.

Matt Jordan and Alexandros G Dimakis. Exactly computing the local lipschitz constant of relu networks. *Proc. NeurIPS*, 33:7344–7353, 2020.

Minseon Kim, Jihoon Tack, Jinwoo Shin, and Sung Ju Hwang. Entropy weighted adversarial training. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, 2009.

Daniel Kunin, Javier Sagastuy-Brena, Surya Ganguli, Daniel LK Yamins, and Hidenori Tanaka. Neural mechanics: Symmetry and broken conservation laws in deep learning dynamics. In *Proc. ICLR*, 2021.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Yi Lin. Support vector machines and the bayes rule in classification. *Data Mining and Knowledge Discovery*, 6(3):259–275, 2002.

Feng Liu, Bo Han, Tongliang Liu, Chen Gong, Gang Niu, Mingyuan Zhou, Masashi Sugiyama, et al. Probabilistic margins for instance reweighting in adversarial training. *Proc. NeurIPS*, 34, 2021.

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

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Shreyas Padhy, Zachary Nado, Jie Ren, Jeremiah Liu, Jasper Snoek, and Balaji Lakshminarayanan. Revisiting one-vs-all classifiers for predictive uncertainty and out-of-distribution detection in neural networks. In *ICML Workshop on Uncertainty and Robustness in Deep Learning*, 2020.

Amartya Sanyal, Puneet K. Dokania, Varun Kanade, and Philip Torr. How benign is benign overfitting? In *Proc. ICLR*, 2021.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. *Proc. NeurIPS*, 31, 2018.

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

Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. In *Proc. NeurIPS*, pp. 6542–6551, 2018.

Jonathan Uesato, Brendan O'Donoghue, Pushmeet Kohli, and Aaron van den Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *Proc. ICML*, volume 80, pp. 5025–5034. PMLR, 2018.

Hongjun Wang and Yisen Wang. Self-ensemble adversarial training for improved robustness. In *Proc. ICLR*, 2022.

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

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

Dongxian Wu, Shu tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *Proc. NeurIPS*, 2020. URL `https://github.com/csdongxian/AWP`.

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

Bohang Zhang, Tianle Cai, Zhou Lu, Di He, and Liwei Wang. Towards certifying l-infinity robustness using neural networks with l-inf-dist neurons. In *Proc. ICML*, volume 139, pp. 12368–12379, 2021a.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *Proc. ICML*, volume 97, pp. 7472–7482. PMLR, 2019.

Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. 119:11278–11287, 2020.

Jingfeng Zhang, Jianing Zhu, Gang Niu, Bo Han, Masashi Sugiyama, and Mohan Kankanhalli. Geometry-aware instance-reweighted adversarial training. In *Proc. ICLR*, 2021b.

Tong Zhang. Statistical analysis of some multi-category large margin classification methods. *Journal of Machine Learning Research*, 5(Oct):1225–1251, 2004.

A PROOFS

A.1 THE PROOF OF PROPOSITION 3.2

Proof. From the definition of Lipschitz constants, we have

$$|z_k(\boldsymbol{x}+\boldsymbol{\delta}) - z_k(\boldsymbol{x})| \leq L_k||\boldsymbol{x}+\boldsymbol{\delta}-\boldsymbol{x}||_\infty = L_k\varepsilon, \tag{22}$$

Thus, we have $z_k(\boldsymbol{x}+\boldsymbol{\delta}) \leq z_k(\boldsymbol{x}) + L_k\varepsilon$ if $z_k(\boldsymbol{x}+\boldsymbol{\delta}) \geq z_k(\boldsymbol{x})$ and $z_k(\boldsymbol{x}+\boldsymbol{\delta}) \geq z_k(\boldsymbol{x}) - L_k\varepsilon$ if $z_k(\boldsymbol{x}+\boldsymbol{\delta}) \leq z_k(\boldsymbol{x})$. Therefore, the following inequalities hold for the not potentially misclassified samples:

$$\begin{aligned}\max_{k\neq y} z_k(\boldsymbol{x}+\boldsymbol{\delta}) - z_y(\boldsymbol{x}+\boldsymbol{\delta}) &\leq z_{k'}(\boldsymbol{x}) + L_{k'}\varepsilon - (z_y(\boldsymbol{x}) - L_y\varepsilon) \\ &\leq z_{k^*}(\boldsymbol{x}) + L_{\hat{k}}\varepsilon - z_y(\boldsymbol{x}) + L_y\varepsilon \\ &\leq z_{k^*}(\boldsymbol{x}) - z_y(\boldsymbol{x}) + (L_{\hat{k}} + L_y)\varepsilon, \end{aligned} \tag{23}$$

where $k' = \arg\max_{k\neq y} z_k(\boldsymbol{x}+\boldsymbol{\delta})$, $k^* = \arg\max_{k\neq y} z_k(\boldsymbol{x})$, and $\hat{k} = \arg\max_k L_k$. From $z_{k^*}(\boldsymbol{x}) - z_y(\boldsymbol{x}) \leq -(L_{\hat{k}}+L_y)\varepsilon$ for not potentially misclassified samples and Eq. (23), we have $\max_{k\neq y} z_k(\boldsymbol{x}+\boldsymbol{\delta}) - z_y(\boldsymbol{x}+\boldsymbol{\delta}) \leq 0, \ \ \forall\boldsymbol{\delta} \in \{||\boldsymbol{\delta}||_\infty \leq \varepsilon\}$. Thus, models are guaranteed to classify adversarial examples of these data points accurately. □

A.2 THE PROOF OF THEOREM 4.1

Proof. By using logit functions, ℓ_{CE} can be written as

$$\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) = -z_y(\boldsymbol{x}) + \log\textstyle\sum_k e^{z_k(\boldsymbol{x})}. \tag{24}$$

when a model uses a softmax function. Compared with OVR $\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) = -z_y(\boldsymbol{x}) + \sum_k \log(1+e^{z_k(\boldsymbol{x})})$, the difference is only the second term. Thus, $\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) - \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y)$ can be written as

$$\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) - \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) = \textstyle\sum_k \log(1+e^{z_k}) - \log\sum_k e^{z_k}, \tag{25}$$

$$= \textstyle\log\prod_k(1+e^{z_k}) - \log\sum_k e^{z_k}. \tag{26}$$

Since a logarithm is a strictly increasing function, we have

$$\textstyle\prod_k(1+e^{z_k}) - \sum_k e^{z_k} \geq 0 \Rightarrow \log\prod_k(1+e^{z_k}) - \log\sum_k e^{z_k} \geq 0. \tag{27}$$

Since $e^{z_k} \geq 0$ for any $z_k \in \mathbb{R}$, we have

$$\textstyle\prod_k(1+e^{z_k}) - \sum_k e^{z_k} = 1 + \sum_k e^{z_k} + R(e^{z_k}) - \sum_k e^{z_k} = 1 + R(e^{z_k}) \geq 0 \tag{28}$$

where $R(e^{z_k})$ is the second or higher order terms of e^{z_k}, and it takes a positive value because $e^{z_k} \geq 0$. Thus, the left hand side of Eq. (27) holds, and we have $\log\prod_k(1+e^{z_k}) - \log\sum_k e^{z_k} \geq 0$. Therefore, we have $\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) - \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) \geq 0$: i.e., $0 \leq \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) \leq \ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y)$ since $\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) \geq 0$. Next, when $z_k(\boldsymbol{x}) \to -\infty$ for $k \neq y$, we have $e^{z_k} \to 0$ and

$$\lim_{\substack{z_y\to+\infty,\\ z_k\to-\infty \text{ for } k\neq y}} \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) = \lim_{\substack{z_y\to+\infty,\\ z_k\to-\infty \text{ for } k\neq y}} -z_y(\boldsymbol{x}) + \log\sum_k e^{z_k(\boldsymbol{x})} = -z_y + \log(e^{z_y}) = 0. \tag{29}$$

On the other hand, when $z_y(\boldsymbol{x}) \to +\infty$ and $z_k(\boldsymbol{x}) \to -\infty$ for $k \neq y$, we have $\log(1+e^{z_y}) \to z_y$ and $\log(1+e^{z_k}) \to 0$. Thus, we have

$$\lim_{\substack{z_y\to+\infty,\\ z_k\to-\infty \text{ for } k\neq y}} \ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}), y) = \lim_{\substack{z_y\to+\infty,\\ z_k\to-\infty \text{ for } k\neq y}} -z_y(\boldsymbol{x}) + \sum_k \log(1+e^{z_k(\boldsymbol{x})}) = -z_y + z_y = 0, \tag{30}$$

which completes the proof. □

A.3 THE PROOF OF LEMMA 4.3

Proof. From the assumption, we consider the following ordinary differential equation (ODE):

$$\frac{dz_k}{dt} = -\frac{\partial w \ell_{OVR}(\boldsymbol{z}, y)}{\partial z_k}. \tag{31}$$

The initial condition is $\boldsymbol{z}(0) = \mathbf{0}$. For the correct label y, the gradient of OVR is given by

$$\frac{\partial \ell_{OVR}(\boldsymbol{z}, y)}{\partial z_y} = -1 + \frac{e^{z_y}}{e^{z_y} + 1}. \tag{32}$$

Thus, ODE becomes

$$\frac{dz_y}{dt} = w(1 - \frac{e^{z_y}}{e^{z_y} + 1}) \tag{33}$$

$$(1 + e^{z_y})dz_y = wdt \tag{34}$$

$$z_y + e^{z_y} = wt + c \tag{35}$$

$$e^{z_y + e^{z_y}} = e^{wt+c}, \tag{36}$$

where c is a constant, which is determined by the initial condition. We apply the Lambert W function (Corless et al., 1996) for both sides and use $\log W(x) = \log x - W(x)$ for $x > 0$ as

$$e^{z_y} = W(e^{wt+c}), \tag{37}$$

$$z_y = wt + c - W(e^{wt+c}). \tag{38}$$

From the assumption, we have $z_y(0) = 0$, and thus, c satisfies the following equality:

$$c - W(e^c) = 0. \tag{39}$$

From $W(xe^x) = x$, we have $c = 1$ and the logit of the correct label is given by

$$z_y = wt + 1 - W(e^{wt+1}). \tag{40}$$

Next, we consider the logit of another label z_k for $k \neq y$. Since the gradient for the logit of incorrect label is $\frac{\partial \ell_{OVR}(\boldsymbol{z}, y)}{\partial z_k} = \frac{e^{z_k}}{e^{z_k}+1}$, we have

$$\frac{dz_k}{dt} = -\frac{we^{z_k}}{1 + e^{z_k}} \tag{41}$$

$$(e^{-z_k} + 1)dz_k = -wdt. \tag{42}$$

It is solved in the same way as z_y, and we have

$$z_k = -wt - 1 + W(e^{wt+1}), \tag{43}$$

for $k \neq y$, which completes the proof. □

A.4 THE PROOF OF LEMMA 4.4

Proof. We first solve the ODE for the logit of the correct label z_y. The gradient of cross-entropy is

$$\frac{\partial \ell_{CE}(\boldsymbol{z}, y)}{\partial z_k} = -\delta_{ky} + \frac{e^{z_k}}{\sum_m e^{z_m}}. \tag{44}$$

From Eq. (44), we have the following ODE:

$$\frac{dz_y}{dt} = w\frac{\sum_{m \neq y} e^{z_m}}{\sum_{m \neq y} e^{z_m} + e^{z_y}}. \tag{45}$$

Since $\boldsymbol{z} = \mathbf{0}$ at $t = 0$, we have $z_i = z_j$ for $\forall i, j \neq y$. In addition, we have $\sum_i \frac{dz_i(t)}{dt} = \sum_i \frac{w\partial \ell_{CE}(\boldsymbol{z}, y)}{\partial z_i} = 0$ for $\forall t$. Thus, logits satisfy the following equality:

$$z_y = -(K - 1)z_k, \tag{46}$$

for $k \neq y$. From Eq. (46), Eq. (45) becomes

$$\frac{dz_y}{dt} = w\frac{(K-1)e^{-\frac{1}{K-1}z_y}}{(K-1)e^{-\frac{1}{K-1}z_y} + e^{z_y}} \tag{47}$$

$$(K-1+e^{\frac{K}{K-1}z_y})dz_y = w(K-1)dt \tag{48}$$

$$\frac{K}{K-1}z_y + \frac{1}{K-1}e^{\frac{K}{K-1}z_y} = \frac{K}{K-1}wt + c \tag{49}$$

$$\frac{1}{K-1}e^{\frac{K}{K-1}z_y}e^{\frac{1}{K-1}e^{\frac{K}{K-1}z_y}} = \frac{1}{K-1}e^{\frac{K}{K-1}wt+c} \tag{50}$$

$$\frac{1}{K-1}e^{\frac{K}{K-1}z_y} = W(\frac{1}{K-1}e^{\frac{K}{K-1}wt+c}) \tag{51}$$

$$\frac{K}{K-1}z_y = \log\left\{(K-1)W(\frac{1}{K-1}e^{\frac{K}{K-1}wt+c})\right\} \tag{52}$$

$$z_y = wt + \frac{K-1}{K}c - \frac{K-1}{K}W(\frac{1}{K-1}e^{\frac{K}{K-1}wt+c}) \tag{53}$$

where c is a constant, which is determined by the initial condition. From Assumption, we have

$$z_y(0) = \frac{K-1}{K}c - \frac{K-1}{K}W(\frac{1}{K-1}e^{c}) = 0 \tag{54}$$

$$W(\frac{1}{K-1}e^{c}) = c. \tag{55}$$

Thus, we have $c = \frac{1}{K-1}$ since $W(xe^x) = x$. From Eqs. (46) and (53), we have

$$z_y(t) = wt + \frac{1}{K} - \frac{K-1}{K}W(\frac{1}{K-1}e^{\frac{K}{K-1}wt+\frac{1}{K-1}}) \tag{56}$$

$$z_k(t) = -\frac{1}{K-1}wt - \frac{1}{K(K-1)} + \frac{1}{K}W(\frac{1}{K-1}e^{\frac{K}{K-1}wt+\frac{1}{K-1}}) \text{ for } k \neq y \tag{57}$$

which completes the proof. □

A.5 THE PROOF OF THEOREM 4.5

From Lemmas 4.3 and 4.4, we have

$$\ell_{\mathrm{LM}}(\boldsymbol{z}^{OVR}(t)) = -2w_1t - 2 + 2W(e^{w_1t+c}), \tag{58}$$

$$\ell_{\mathrm{LM}}(\boldsymbol{z}^{CE}(t)) = -\frac{K}{K-1}w_2t - \frac{1}{K-1} + W(\frac{1}{K-1}e^{\frac{K}{K-1}w_2t+\frac{1}{K-1}}). \tag{59}$$

Since $W(x) = \log(x) - \log(\log(x)) + O(1)$ for large x (Hoorfar & Hassani, 2007), we have

$$\ell_{\mathrm{LM}}(\boldsymbol{z}^{OVR}(t)) \approx -2w_1t - 2 + 2(w_1t + 1 - \log(w_1t+1)) \tag{60}$$

$$= -\log(w_1t+1)^2 \tag{61}$$

$$\ell_{\mathrm{LM}}(\boldsymbol{z}^{CE}(t)) \approx -\frac{K}{K-1}w_2t - \frac{1}{K-1} \tag{62}$$

$$-\log(K-1) + \frac{K}{K-1}w_2t + \frac{1}{K-1} \tag{63}$$

$$-\log(-\log(K-1) + \frac{K}{K-1}w_2t + \frac{1}{K-1}) \tag{64}$$

$$= -\log(Kw_2t + 1 - (K-1)\log(K-1)). \tag{65}$$

Algorithm 1 Switching one-versus-the-rest by the criterion of a logit margin loss

1: Select the minibatch $\mathbb{B}$
2: **for** $\boldsymbol{x}_n \in \mathbb{B}$ **do**
3: Generate adversarial examples $\boldsymbol{x}'_n = \arg\max_{||\boldsymbol{x}'_n - \boldsymbol{x}_n||_\infty \leq \varepsilon} \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}'_n, \boldsymbol{\theta}), y_n)$ by PGD
4: $\ell_{\mathrm{LM}}(\boldsymbol{z}(\boldsymbol{x}'_n, \boldsymbol{\theta}), y_n) = \max_{k \neq y_n} z_k(\boldsymbol{x}'_n) - z_{y_n}(\boldsymbol{x}'_n)$
5: **end for**
6: Select top $\frac{M}{100}|\mathbb{B}|$ samples of $(\boldsymbol{x}'_n, y_n)$ in terms of $\ell_{\mathrm{LM}}(\boldsymbol{z}(\boldsymbol{x}'_n, \boldsymbol{\theta}), y_n)$ and add them to $\mathbb{L}$
7: $\mathcal{L}_{\mathrm{SOVR}}(\boldsymbol{\theta}) = \frac{1}{|\mathbb{B}|}\left[\sum_{(\boldsymbol{x},y)\in\mathbb{B}\setminus\mathbb{L}} \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}', \boldsymbol{\theta}), y) + \lambda \sum_{(\boldsymbol{x},y)\in\mathbb{L}} \ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x}', \boldsymbol{\theta}), y)\right]$
8: Update the parameter $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{SOVR}}(\boldsymbol{\theta})$

From the above, we have

$$\lim_{t\to\infty} \frac{\ell_{\mathrm{LM}}(\boldsymbol{z}^{OVR})(t)}{\ell_{\mathrm{LM}}(\boldsymbol{z}^{CE})(t)} = \lim_{t\to\infty} \frac{\log(w_1 t + 1)^2 + O(1)}{\log(K w_2 t + 1 - (K-1)\log(K-1)) + O(1)}, \tag{66}$$

$$= \lim_{t\to\infty} \frac{2\log t + 2\log(w_1 + t^{-1}) + O(1)}{\log t + \log(K w_2 + t^{-1}(1 - (K-1)\log(K-1))) + O(1)}, \tag{67}$$

$$= \lim_{t\to\infty} \frac{2 + \frac{2}{\log t}\log(w_1 + t^{-1}) + \frac{O(1)}{\log t}}{1 + \frac{\log(K w_2 + t^{-1}(1-(K-1)\log(K-1)))}{\log t} + \frac{O(1)}{\log t}}, \tag{68}$$

$$= 2, \tag{69}$$

which completes the proof.

A.6 The proof of Proposition 6.1

Proof. Since we have $\sup_{\boldsymbol{x}} ||\nabla_{\boldsymbol{x}} z_k(\boldsymbol{x})||_q = L_k$ for an L_k-Lipschitz function such as $|z_k(\boldsymbol{x}_1) - z_k(\boldsymbol{x}_2)| \leq L_k ||\boldsymbol{x}_1 - \boldsymbol{x}_2||_p$ where $1/q + 1/p = 1$ (Jordan & Dimakis, 2020), the following inequality holds if $\boldsymbol{z}_{k^*}(\boldsymbol{x}) - \boldsymbol{z}_y(\boldsymbol{x}) > -(\max_k ||\nabla_{\boldsymbol{x}} z_k(\boldsymbol{x})||_1 + ||\nabla_{\boldsymbol{x}} z_y(\boldsymbol{x})||_1)\varepsilon$ and $p = \infty$:

$$\begin{aligned}\boldsymbol{z}_{k^*}(\boldsymbol{x}) - \boldsymbol{z}_y(\boldsymbol{x}) &> -(\max_k ||\nabla_{\boldsymbol{x}} z_k(\boldsymbol{x})||_1 + ||\nabla_{\boldsymbol{x}} z_y(\boldsymbol{x})||_1)\varepsilon \\ &\geq -(\max_k \sup_x ||\nabla_{\boldsymbol{x}} z_k(\boldsymbol{x})||_1 + \sup_x ||\nabla_{\boldsymbol{x}} z_y(\boldsymbol{x})||_1)\varepsilon \geq -(L_{\hat{k}} + L_y)\varepsilon,\end{aligned} \tag{70}$$

because $||\nabla_{\boldsymbol{x}} z_k(\boldsymbol{x})||_1 \leq L_k$ for $p = \infty$. Thus, we have $z_{k^*}(\boldsymbol{x}) - z_y(\boldsymbol{x}) > -(L_{\hat{k}} + L_y)\varepsilon$ on this condition, which completes the proof. □

B Algorithm

The proposed algorithm is shown in Algorithm 1. We first generate $\boldsymbol{x}'$ in Line 3 and compute the ℓ_{LM} for them in Line 4. In Line 6, we select the top M % samples in minibatch and add them to $\mathbb{L}$. Finally, we compute the objective $\mathcal{L}_{\mathrm{SOVR}}$ and its gradient to update $\boldsymbol{\theta}$. Since we do not additionally generate the adversarial examples for ℓ_{OVR}, the overhead of our method is $O(|\mathbb{B}| \log \frac{M}{100}|\mathbb{B}|)$, which is the computation cost of the heap sort for selecting $\mathbb{L}$ in Line 6. It is negligible in the whole computation since deep models have huge parameter-size compared with batch-size $|\mathbb{B}|$.

C Additional explanation about previous methods

C.1 MART (Wang et al., 2020a)

MART (Wang et al., 2020a) uses a similar approach to importance-aware methods. It regards misclassified samples as important samples and minimizes

$$\ell_{\mathrm{MART}}(\boldsymbol{x}', y, \boldsymbol{\theta}) = \mathrm{BCE}(\boldsymbol{f}(\boldsymbol{x}', \boldsymbol{\theta}), y) + \lambda \mathrm{KL}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}), \boldsymbol{f}(\boldsymbol{x}', \boldsymbol{\theta})) \cdot (1 - f_y(\boldsymbol{x}, \boldsymbol{\theta})), \tag{71}$$

where $\mathrm{BCE}(\boldsymbol{f}(\boldsymbol{x}', \boldsymbol{\theta}), y) = -\log(f_y(\boldsymbol{x}', \boldsymbol{\theta})) - \log(1 - \max_{k\neq y} f_k(\boldsymbol{x}', \boldsymbol{\theta}))$ and KL is Kullback-Leibler divergence. MART controls the difference between the loss on unimportant and important samples via $1 - f_y(\boldsymbol{x}_n, \boldsymbol{\theta})$: MART tends to ignore the second term when the model is confident in the true label.

C.2 MMA (DING ET AL., 2020)

MMA (Ding et al., 2020) attempts to maximize the distance[2] between data points and the decision boundary for robustness. MMA regards $\min_{\boldsymbol{\delta}} ||\boldsymbol{\delta}_n||_\infty$ subject to $\{\ell_{\mathrm{LM}}(\boldsymbol{z}(\boldsymbol{x}+\boldsymbol{\delta},\boldsymbol{\theta}),y)\geq 0\}$ as the distance. By using this distance, MMA minimizes the following loss:

$$\mathcal{L}(\boldsymbol{\theta}) = \tfrac{1}{3}\sum_{n=1}^{N}\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}_n,\boldsymbol{\theta}),y_n) + \tfrac{2}{3}\mathcal{L}_{\mathrm{MMA}}(\boldsymbol{\theta}) \tag{72}$$

$$\mathcal{L}_{\mathrm{MMA}}(\boldsymbol{\theta}) = \sum_{(\boldsymbol{x},y)\in\mathcal{S}^+\cap\mathcal{H}}\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}+\boldsymbol{\delta}_{\mathrm{MMA}},\boldsymbol{\theta}),y) + \sum_{(x,y)\in\mathcal{S}^-}\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}),y) \tag{73}$$

$$\boldsymbol{\delta}_{\mathrm{MMA}} = \arg\min_{\ell_{\mathrm{SLM}}(\boldsymbol{z}(\boldsymbol{x}+\boldsymbol{\delta},\boldsymbol{\theta}),y)\geq 0}||\boldsymbol{\delta}||_\infty \tag{74}$$

$$\ell_{\mathrm{SLM}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}),y) = \log\textstyle\sum_{k\neq y}e^{z_k(\boldsymbol{x})} - z_y(x) \tag{75}$$

where $\mathcal{S}^+$ is a set of correctly classified data points, and $\mathcal{S}^-$ is a set of misclassified samples. $\mathcal{H}$ is a set of data points that have a smaller distance than threshold $d_{\max}$ as $\mathcal{H} = \{(x_n,y_n)|\min_{\boldsymbol{\delta}_n}||\boldsymbol{\delta}_n||_\infty \leq d_{\max}\}$. Since MMA uses $\boldsymbol{\delta}$ whose magnitude $||\boldsymbol{\delta}||_\infty$ depends on data points as Eq. (74), we consider that it has similar effects to the importance-aware methods. In MMA, Ding et al. (2020) use $\ell_{\mathrm{SLM}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}),y)$ as an approximated differentiable logit margin loss by changing $\max$ into differentiable function $\log\sum_{k\neq y}e^{z_k(\boldsymbol{x})}$. Comparing OVR with $\ell_{\mathrm{SLM}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}),y)$, we have the following:

$$\ell_{\mathrm{SLM}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}),y) \leq \ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}),y) \leq \ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x},\boldsymbol{\theta}),y). \tag{76}$$

This is because we have $\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}),y) - \ell_{\mathrm{SLM}}(\boldsymbol{z}(\boldsymbol{x}),y) = \log\sum_k e^{z_k(\boldsymbol{x})} - \log\sum_{k\neq y}e^{z_k(\boldsymbol{x})} = \log\left(1+e^{z_y(\boldsymbol{x})}/\sum_{k\neq y}e^{z_k(\boldsymbol{x})}\right)\geq 0$ and $\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}),y)\leq\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x}),y)$ from Theorem 4.1. Thus, we expect that OVR more strongly penalizes the small logit margins than $\ell_{\mathrm{SLM}}(\boldsymbol{x},y)$. Note that training algorithms of MMA is also different from those of the other importance-aware methods (Ding et al., 2020).

C.3 EWAT (KIM ET AL., 2021)

EWAT uses a weighted cross-entropy like GAIRAT and MAIL, but it is added to cross-entropy as

$$\mathcal{L}_{\mathrm{weight}}(\boldsymbol{\theta}) = \tfrac{1}{N}\sum_{n=1}^{N}(1+\bar{w}_n)\,\ell_{\mathrm{CE}}(\boldsymbol{z}(\boldsymbol{x}'_n,\boldsymbol{\theta}),y_n), \tag{77}$$

where $\bar{w}_n \geq 0$ is a weight divided by batch-mean of the weight w_n as $\bar{w}_n = \frac{|\mathbb{B}|w_n}{\sum_{l=1}^{|\mathbb{B}|}w_l}$. EWAT determines the weights by using entropy as

$$w_n = -\textstyle\sum_{k=1}^{K}f_k(\boldsymbol{x}'_n,\boldsymbol{\theta})\log(f_k(\boldsymbol{x}'_n,\boldsymbol{\theta})) \tag{78}$$

where $f_k(\boldsymbol{x}_n,\boldsymbol{\theta})$ is the k-th softmax output, and thus, it can be regarded as the probability for the k-th class label. EWAT is based on the observation that importance-aware methods tend to have high entropy, and it causes their vulnerability. Our theoretical results about logit margins and experiments seem to indicate that a logit margin loss is a more reasonable criterion to evaluate the robustness and improve the robustness by using it than entropy. Furthermore, Theorem 4.5 shows that the weighted cross-entropy is less effective than OVR at increasing logit margins.

D SELECTION OF ϕ IN OVA

D.1 INFINITE SAMPLE CONSISTENCY

Infinite-sample consistency (ISC, also known as *classification calibrated* or *Fisher consistent*) is a desirable property for multi-class classification problems (Zhang, 2004; Bartlett et al., 2003; Lin, 2002). We first introduce ISC in this section. Let $\boldsymbol{f}(\boldsymbol{x})$ be a model and c be the classifier $c(\boldsymbol{x}) = \arg\max_k f_k(\boldsymbol{x})$. The classification error ℓ_* is

$$\ell_*(c(\cdot)) := \mathbb{E}_{\boldsymbol{x}}\textstyle\sum_{k=1,k\neq c(\boldsymbol{x})}^{K}a_k p(y=k|\boldsymbol{x}) \tag{79}$$

[2]For clarity, we use the term "margin" only for the distance between logits of the true labels and of the label that has the largest logit except for the true label, not for the distance between data points and the decision boundary.

where a_k is a weight for the k-th label and is usually set to one. The optimal classification rule called a Bayes rule is given by

$$c_*(\cdot) = \max_{k\in\{1,\dots,K\}} a_k p(y=k|\boldsymbol{x}). \tag{80}$$

Since Eq. (79) is difficult to minimize directly, we use a surrogate loss function ℓ. In classification problems, we obtain the model $\hat{f}(\cdot)$ by the minimization of the empirical risk using ℓ as

$$\hat{f}(\cdot) = \arg\max_{\boldsymbol{f}(\cdot)} \frac{1}{n}\sum_{i=1}^{N} \ell(\boldsymbol{f}(\boldsymbol{x}_i), y_i). \tag{81}$$

On the other hand, the true risk using ℓ is written by

$$\mathbb{E}_{\boldsymbol{x},y}\ell(\boldsymbol{f}, y) = \mathbb{E}_{\boldsymbol{x}} W(\boldsymbol{p}(\cdot|\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x})) \tag{82}$$

$$W(\boldsymbol{q}, \boldsymbol{f}) := \sum_{k=1}^{K} q_k \ell(\boldsymbol{f}, k) \tag{83}$$

where $\boldsymbol{p}(\cdot|\boldsymbol{x}) = [p(1|\boldsymbol{x}), \dots, p(K|\boldsymbol{x})]$ and $\boldsymbol{q}$ is a vector in the set Λ_K:

$$\Lambda_K := \left\{ \boldsymbol{q} \in \mathbb{R}^K : \sum_{k=1}^{K} q_k = 1, q_k \geq 0 \right\}. \tag{84}$$

$W(\boldsymbol{q}, \boldsymbol{f})$ is the point-wise true loss of model $\boldsymbol{f}$ with the conditional probability $\boldsymbol{q}$. By using the above, ISC is defined as the following definition:

Definition D.1. (Zhang, 2004) We say that the formulation is infinite-sample consistent (ISC) on a set $\Omega \subseteq \mathbb{R}^K$ with respect to Eq. (79) if the following condition holds:

- For each k, $\ell(\cdot, k) : \Omega \to \mathbb{R}$ is bounded below and continuous
- $\forall \boldsymbol{q} \in \Lambda_K$ and $k \in \{1, \dots, K\}$ such that $a_k q_k < \sup_i a_i q_i$, we have

$$W^*(\boldsymbol{q}) := \inf_{\boldsymbol{f}\in\Omega} W(\boldsymbol{q}, \boldsymbol{f}) < \inf \{W(\boldsymbol{q}, \boldsymbol{f}) | \boldsymbol{f} \in \Omega, f_k = \sup_i f_i\} \tag{85}$$

This definition indicates that the optimal solution of $W(\boldsymbol{q}, \cdot)$ leads to a Bayes rule with respect to classification error Zhang (2004): the minimizer of Eq. (82) becomes the minimizer of classification error ℓ_* (Eq. (79)). Thus, surrogate loss functions ℓ, e.g., cross-entropy or OVR, should satisfy ISC to minimize the classification error.

D.2 Evaluation of ϕ

It is known that ISC is satisfied when ϕ in Eq. (7) is a differentiable non-negative convex function and satisfies $\phi(z) < \phi(-z)$ for $z > 0$. Among common nonlinear functions used in deep neural networks, e^{-z} and $\log(1 + e^{-z})$ satisfy this condition. We first evaluated e^{-z} and observed that e^{-z} causes numerical unstability. On the other hand, $\log(1 + e^{-z})$ tends to be stable in computation. This is because $\log(1 + e^{-z})$ asymptotically closes to $\max(-z, 0)$. In addition, let the conditional probability $p(y|\boldsymbol{x})$ for the class k given $\boldsymbol{x}$ be

$$p(k|\boldsymbol{x}) = \frac{1}{1 + e^{-z_k(\boldsymbol{x})}} \tag{86}$$

when we choose $\phi(z) = \log(1 + e^{-z})$. We have

$$\ell_{\mathrm{OVR}}(\boldsymbol{z}(\boldsymbol{x}, \boldsymbol{\theta}), y) = \log(1+e^{-z_y(\boldsymbol{x})}) + \sum_{k\neq y} \log(1+e^{z_k(\boldsymbol{x})}) \tag{87}$$

$$= -\log p(y|\boldsymbol{x}) + \textstyle\sum_{k\neq y} -\log(1 - p(k|\boldsymbol{x})) \tag{88}$$

and we can regard models as K independent binary classifier (Padhy et al., 2020).

E Experimental Setups

We conducted the experiments for evaluating our proposed method. We first compared our method with baseline methods; Madry's AT (Madry et al., 2018), MMA (Ding et al., 2020), MART (Wang et al., 2020a), GAIRAT (Zhang et al., 2021b), MAIL (Liu et al., 2021), and EWAT (Kim et al., 2021) on three datasets; CIFAR10, SVHN, and CIFAR100 (Krizhevsky & Hinton, 2009; Netzer

et al., 2011). Next, we evaluated the combination of our method with TRADES (Zhang et al., 2019), AWP (Wu et al., 2020), and SEAT (Wang & Wang, 2022). Our experimental codes are based on source codes provided by Wu et al. (2020); Wang et al. (2020a); Ding et al. (2020). We used PreActResNet-18 (RN18) (He et al., 2016) and WideResNet-34-10 (WRN) (Zagoruyko & Komodakis, 2016) following Wu et al. (2020). The L_∞ norm of the perturbation was set to $\varepsilon = 8/255$, and all elements of $x_i + \delta_i$ were clipped so that they were in [0,1]. We used early stopping by evaluating test robust accuracies against 20-step PGD. To evaluate TRADES, AWP, and SEAT, we used the original public code (Wu et al., 2020; Wang & Wang, 2022). We trained models three times and show the average and standard deviation of test accuracies. We used Auto-Attack to evaluate the robust accuracy on test data. We used one GPU among NVIDIA ®V100 and NVIDIA®A100 for each training in experiments. We trained models three times and show the average and standard deviation of test accuracies. For MART, we used *mart_loss* in the original code (Wang et al., 2020a)[3] as the loss function. λ of MART was set to 6.0. For GAIRAT and MAIL, we also used the loss functions in the original codes (Zhang et al., 2021b; Liu et al., 2021),[4][5] and thus, hyperparameters of their loss functions were based on them. λ of GAIRAT was set to ∞ until the 50-th epoch and then set to 3.0. (γ, β) of MAIL was set to $(10, 0.5)$. For all settings, the size of minibatch was set to 128. The detailed setup for each dataset was as follows.

E.1 MMA

We trained models by using MMA based on the original code (Ding et al., 2020)[6]. Thus, the learning rate schedules and hyperparameters of PGD for MMA were different from those for other methods because the training algorithm of MMA is different from the other methods. The step size of PGD in MMA was set to $\frac{2.5\varepsilon}{10}$ in training by following Ding et al. (2020). For AN-PGD in MMA, the maximum perturbation length was 1.05 times the hinge threshold $\varepsilon_{\max} = 1.05 d_{\max}$, and $d_{\max}$ was set to 0.1255. The learning rate of SGD was set to 0.3 at the 0-th parameter update, 0.09 at the 20000-th parameter update, 0.03 at the 30000-th parameter update, and 0.009 at the 40000-th parameter update.

E.2 CIFAR10

For PreActResNet18, the learning rate of SGD was divided by 10 at the 100-th and 150-th epoch except for EWAT, and the initial learning rate was set to 0.05 for SOVR and 0.1 for others. We tested the initial learning rate of 0.05 for the other methods and found that the setting of 0.1 achieved better robust accuracies against Auto-Attack than the setting of 0.05 when using ResNet18. For EWAT, we divided the learning rate of SGD at the 100-th and 105-th epoch following Kim et al. (2021) after we found that the division at the 100-th and 150-th epoch was worse than the division at the 100-th and 105-th epoch. When using WideResNet34-10, we set the initial learning rate to 0.1 and divided by 10 at the 100-th and 150-th epoch. We used momentum of 0.9 and weight decay of 0.0005 and early stopping by evaluating test accuracies. We standardized datasets by using mean = [0.4914, 0.4822, 0.4465] and std = [0.2471, 0.2435, 0.2616] as the pre-process. (M, λ) was tuned by grid search over $M \in [20, \ldots, 80, 100]$ and $\lambda \in [0.2, \ldots, 0.8, 1.0]$ for RN18, and tuned by coarse tuning for WRN due to high computation cost.

E.3 CIFAR100

We used PreActResNet18 for CIFAR100. The learning rate of SGD was divided by 10 at the 100-th and 150-th epoch except for EWAT, and the initial learning rate was set to 0.1. Note that we found that the above setting is better than the initial learning rate of 0.05 for all methods. For EWAT, we divided the learning rate of SGD at the 100-th and 105-th epoch following Kim et al. (2021) after we found that the division at the 100-th and 150-th epoch was worse than the division at the 100-th and 105-th epoch. We randomly initialized the perturbation and updated it for 10 steps with a step size of 2/255 for PGD. We used momentum of 0.9 and weight decay of 0.0005 and early stopping by evaluating test accuracies. We standardized datasets by

[3] https://github.com/YisenWang/MART

[4] https://github.com/zjfheart/Geometry-aware-Instance-reweighted-Adversarial-Training

[5] https://github.com/QizhouWang/MAIL

[6] https://github.com/BorealisAI/mma_training

using mean = [0.5070751592371323, 0.48654887331495095, 0.4409178433670343], and std = [0.2673342858792401, 0.2564384629170883, 0.27615047132568404] as the pre-process. (M, λ) was set to (0.5, 0.5) based on the coarse hyperparameter tuning.

E.4 SVHN

We used PreActResNet18 for SVHN. The learning rate of SGD was divided by 10 at the 100-th and 150-th, and the initial learning rate was set to 0.05 for SOVR and 0.01 for others. We tested the initial learning rate of 0.05 for the other methods and found that the setting of 0.01 achieved better robust accuracies against Auto-Attack than the setting of 0.05. For EWAT, the learning rate of SGD was divided by 10 at the 100-th and 105-th epoch after we found that this setting was better than the division at 100-th and 105-th epoch. The hyperparameters for PGD were based on (Wu et al., 2020): We randomly initialized the perturbation and updated it for 10 steps with a step size of 1/255. For the preprocessing, we standardized data by using the mean of [0.5, 0.5, 0.5], and standard deviations of [0.5, 0.5, 0.5]. (M, λ) is set to (0.2,0.2) on the basis of the coarse hyperparameter tuning.

E.5 TRADES, AWP, AND SEAT

For experimental settings of TRADES and AWP, we followed Wu et al. (2020) and only changed the training loss into SOVR in the training procedure and in the algorithm for computing the perturbation of AWP for SOVR+AWP, TSOVR, and TSOVR+AWP. We used the original codes of AWP (Wu et al., 2020)[7]. For AWP and AWP+SOVR, we found that models trained under the setup for TRADES+AWP in original codes, where the dataset is not standardized and AWP is applied after 10 epochs, achieves better robust accuracy than those trained under the setup for cross-entropy+TRADES in original codes. Thus, we used the code for TRADES+AWP by changing the loss functions. β_T of TRADES and TSOVR were set to 6, and γ of AWP is set to 0.01 for AWP and AWP+SOVR. γ of AWP was set to 0.005 for TRADES+AWP and TSOVR+AWP. AWP is applied after the 10-th epoch. We used WideResNet34-10 following Wu et al. (2020). We used SGD with momentum of 0.9 and weight decay of 0.0005 for 200 epochs. The learning rate was set to 0.1 and was divided by 10 at the 100-th and 150-th epoch. For experimental settings of SEAT, we followed Wang & Wang (2022) and only changed the loss into SOVR in the original code (Wang & Wang, 2022).[8] We did not evaluate SEAT with CutMix in our experiments, but we fairly compare SEAT+SOVR with SEAT under the same condition. We used SGD with momentum of 0.9 and weight decay of 7×10^{-4} for 120 epochs. The initial learning rate was set to 0.1 till the 40-th epoch and then linearly reduced to 0.01 and 0.001 at the 60-th epoch and 120-th epoch, respectively. We used WideResNet32-10 following Wang & Wang (2022) for SEAT. (M, λ) is tuned by grid search over $M \in [20, \ldots, 80, 100]$ and $\lambda \in [0.2, \ldots, 0.8, 1.0]$ for SOVR+AWP, and (M, λ) is tuned by grid search over $M \in [20, \ldots, 80, 100]$ and $\lambda \in [0.2, \ldots, 1.0, 1.2]$ for TSOVR. (M, λ) is set to (0.5,0.5) for SOVR+SEAT after coarse hyperparameter tuning.

E.6 EXPERIMENTAL SETUPS IN SECTION 3

For the experiments in Section 3, we used the models obtained under the above settings, which are the same as models used in Section 6. To obtain the histograms of logit margins, we used the models and computed logit margin loss on adversarial examples of training data set for each data point at 200 epochs. Thus, the number of data points of CIFAR10 and CIFAR100 is 50,000, and that of SVHN is 73,257. We also provide the results of the models obtained by the early stopping in Fig. **??**.

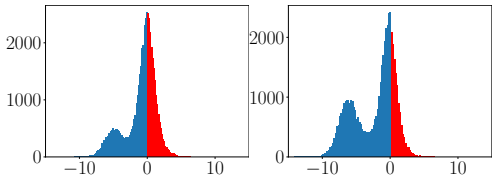
F ADDITIONAL RESULTS

F.1 HISTOGRAMS OF LOGIT MARGIN LOSSES

We show the additional histograms of logit margin losses in this section. First, Fig. 7 plots the result of EWAT on training samples of CIFAR10 at the last epoch. Compared with SOVR, EWAT does not increase logit margins for difficult samples (right peak). Figures 8-11 plot the histograms when

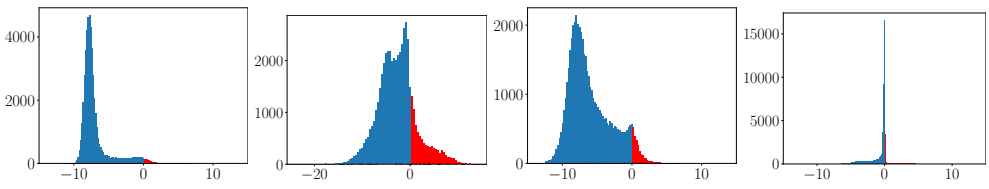
[7] https://github.com/csdongxian/AWP

[8] https://github.com/whj363636/Self-Ensemble-Adversarial-Training

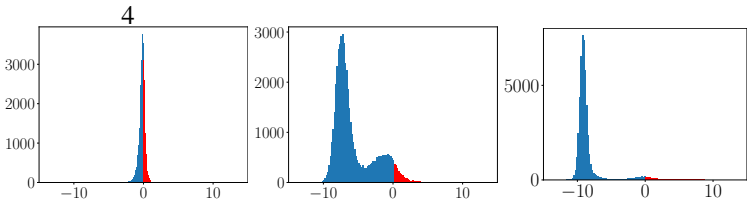


(a) Best epoch (b) Last epoch

Figure 7: Histogram of logit margin losses of EWAT for training data on CIFAR10.



(a) AT (b) MMA (c) MART (d) GAIRAT



(e) MAIL (f) EWAT (g) SOVR

Figure 8: Histogram of logit margin losses for training samples of CIFAR10 with WideResNet34-10 at the last epoch. We plot those on adversarial data $\boldsymbol{x}'$ for the other methods. Blue bins are the data points that models correctly classify.

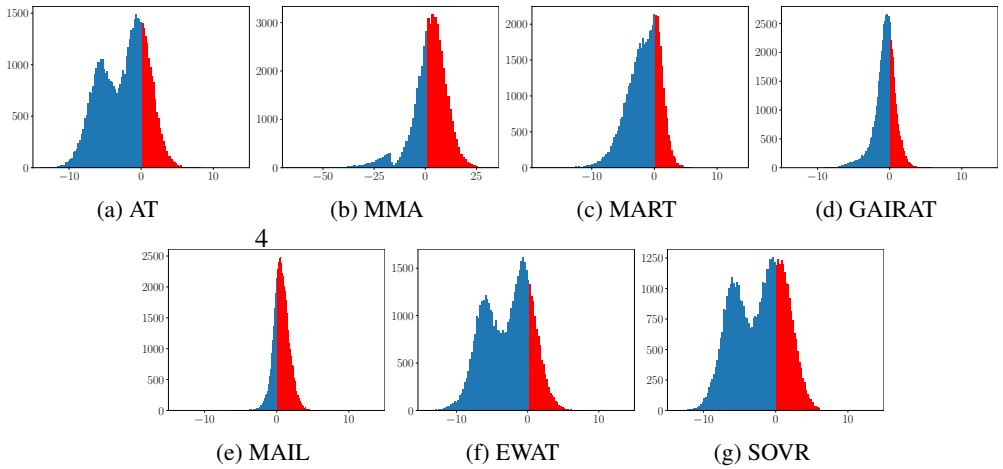
using WideResNet and other datasets. SOVR tends to increase the left peak under all conditions, and thus, it decreases logit margin losses ℓ_{LM}, and thus, it increases the logit margins $|\ell_{\mathrm{LM}}|$. Figure 10 shows that AT does not have two peaks on SVHN. To investigate histograms on SVHN in detail, we additionally evaluate logit margin losses at the 100-th epoch in Fig. 11. This figure shows that the histogram on SVHN has two peaks at the 100-th epoch, but they became one peak at the 200-th epoch (Fig. 10). This might cause the optimal (M, λ) for SOVR to be smaller than that for other datasets. Figure 12 plots the histograms of TRADES and shows that TRADES has two peaks but they are close to each other. This might be because the objective functions for adversarial examples and parameters are different. Table 3 lists the average of logit margin losses. Since the distributions of logit margin losses are long-tailed as shown in histograms, the difference in average values of logit margin losses among methods is small. Even so, SOVR tends to have the lowest logit margin losses under almost all settings.

F.2 Evaluation of Gradient Norms

Even though logit margins of importance-aware methods are very small, they are robust against PGD and some attacks (Fig. 1). To reveal the cause of this robustness, we additionally evaluate th gradient norms for loss and logit functions (Fig. 13). In this figure, the gradient norms of cross-

Table 3: Average logit margin losses for training dataset $\boldsymbol{x}'$ at the last epoch.

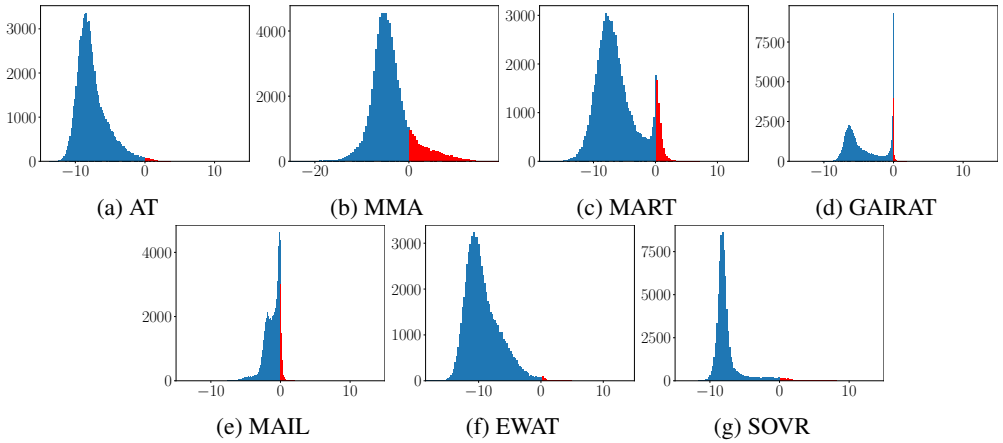
Dataset	AT	MART	MMA	GAIRAT	MAIL	EWAT	SOVR
CIFAR10 (RN18)	-3.66±0.02	-3.46±0.02	0.558±0.1	-0.258± 0.02	-0.0293± 0.02	-2.251±0.007	-4.34± 0.02
CIFAR10 (WRN)	-6.96±0.01	-5.65±0.7	-2.73±0.6	-0.717± 0.03	-0.203± 0.01	-5.69±0.04	-8.56± 0.04
CIFAR100	-2.41±0.02	-1.76±0.01	2.07±0.06	-0.68±0.02	0.684±0.1	-1.36±0.02	-2.44±0.1
SVHN	-7.55±0.01	-8.08±2	-4.31±0.04	-3.91±0.01	-1.17±0.03	-9.32±0.04	-7.59±0.1



(a) AT (b) MMA (c) MART (d) GAIRAT

(e) MAIL (f) EWAT (g) SOVR

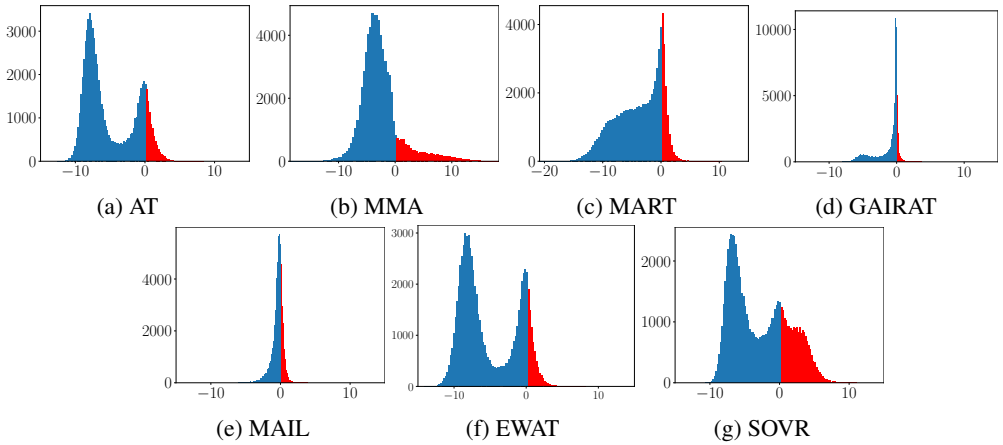
Figure 9: Histogram of logit margin losses for training samples of CIFAR100 at the last epoch. We plot those on adversarial data $\boldsymbol{x}'$ for the other methods. Blue bins are the data points that models correctly classify.



(a) AT (b) MMA (c) MART (d) GAIRAT

(e) MAIL (f) EWAT (g) SOVR

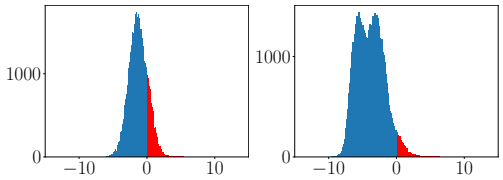
Figure 10: Histogram of logit margin losses for training samples of SVHN at the last epoch. We plot those on adversarial data $\boldsymbol{x}'$ for the other methods. Blue bins are the data points that models correctly classify.



(a) AT (b) MMA (c) MART (d) GAIRAT

(e) MAIL (f) EWAT (g) SOVR

Figure 11: Histogram of logit margin losses for training samples of SVHN at the 100-th epoch. We plot those on adversarial data $\boldsymbol{x}'$ for the other methods. Blue bins are the data points that models correctly classify.



(a) Best epoch (b) Last epoch

Figure 12: Histogram of logit margin losses on CIFAR10 with WRN for TRADES.

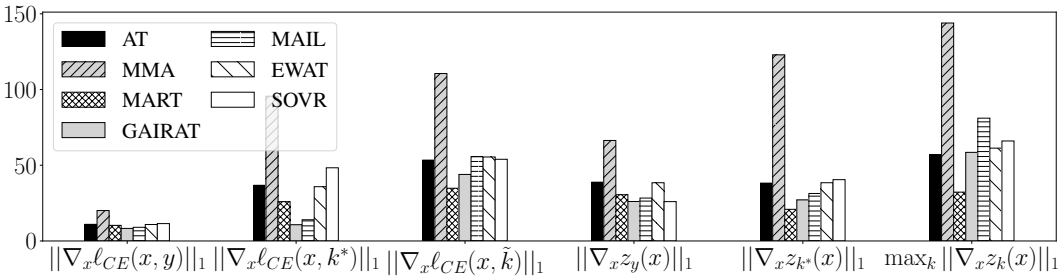
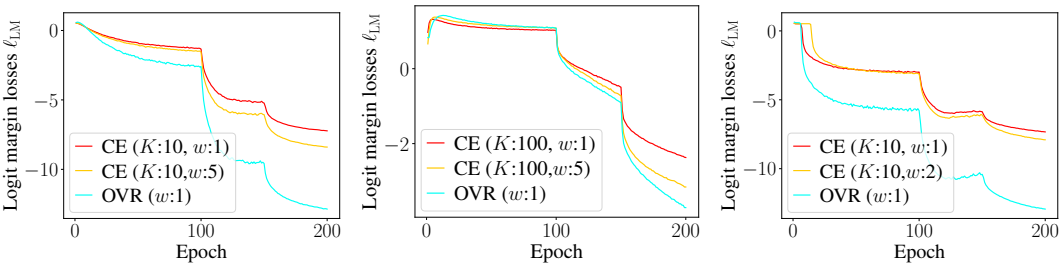


Figure 13: Average of gradient norms with respect to data points. $\tilde{k}$ is randomly selected labels, and $k^* = \arg\max_{k \neq y} z_k(\boldsymbol{x})$.

entropy $||\nabla_x \ell_{\mathrm{CE}}(\boldsymbol{x}, y)||_1$ are relatively small values in all methods. This indicates that adversarial training essentially attempts to suppress the gradient norms for the cross-entropy. MMA has the largest gradient norms, and this is the reason MMA is not robust against Auto-Attack except for SQUARE (Fig. 1), which does not use gradient. GAIRAT and MAIL have the smallest and second smallest $||\nabla_x \ell_{\mathrm{CE}}(\boldsymbol{x}, y)||_1$, and this is the reason they are robust against PGD despite the small logit margins (Fig. 2). On the other hand, $\max_k ||\nabla_x z_k(\boldsymbol{x})||_1$ of importance-aware methods is larger than $||\nabla_x \ell_{\mathrm{CE}}(\boldsymbol{x}, y)||_1$ of them and that of AT. As a result, they can have larger rate of potentially misclassified samples (Fig. 6). Gradient norms of cross-entropy for the label that has the largest logit except for the true label $||\nabla_x \ell_{\mathrm{CE}}(\boldsymbol{x}, k^*)||_1$ are smaller than those for the randomly selected labels. This implies $k^* \neq \hat{k}$, and we need to use the loss that depends on the logits for all classes rather than logit margin loss, which only cares z_k^* and z_y. Gradient norms of EWAT and SOVR are not significantly different from those of AT. Thus, SOVR can reduce the rate of potentially misclassified samples by large logit margins and not large gradient norm.

F.3 Trajectories of logit margin losses in adversarial training with OVR and cross-entropy

In this section, we evaluate the trajectories of logit margin losses in adversarial training on real data. Experimental setup is the same with that of Section 6 except for the learning rate on CIFAR10, and

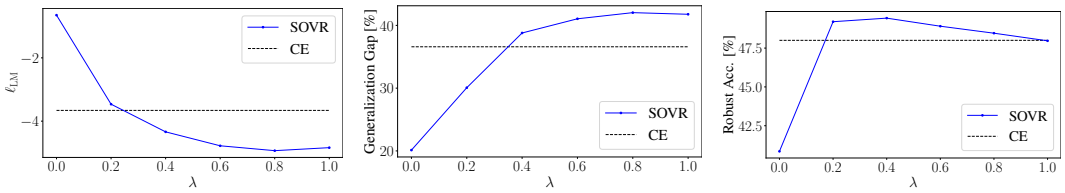


(a) CIFAR 10 (WRN) (b) CIFAR 100 (c) SVHN

Figure 14: Trajectories of logit margin losses ℓ_{LM} in adversarial training using cross-entropy and OVR. RN18 is used on CIFAR100 and SVHN.

CIFAR10 (RN18)	CIFAR10 (WRN)	CIFAR100	SVHN
1.87	1.78	1.56	1.76

Table 4: $\ell_{\rm LM}(\boldsymbol{z}^{OVR})/\ell_{\rm LM}(\boldsymbol{z}^{CE})$ with $w = 1.0$ at the last epoch



(a) $\ell_{\rm LM}(\boldsymbol{x}')$ Vs λ (b) Generalization Gap Vs λ (c) Robust Acc. Vs λ

Figure 15: The effect of rate of applying OVR λ. M is set to 40. Generalization gap is a gap between training robust accuracy and test robust accuracy against PGD ($\mathcal{K}$=20) at the last epoch. Robust Acc. is robust accuracy against Auto-Attack. Dashed gray line corresponds to the results of AT using cross-entropy loss.

thus, we minimize the OVR and cross-entropy averaged over the dataset, unlike Eq. (10). While the learning rates are set to 0.1 for cross-entropy and 0.05 for SOVR in Section 6 on CIFAR10, learning rate is set to 0.05 on CIFAR10 for both cross-entropy and OVR to fairly compare their logit margin losses in this experiment. Since we could not obtain results on SVHN with the weight of $w = 5$ due to unstability, we used $w = 2$ on SVHN. Fig. 14 plots the logit margin losses averaged over the dataset against epochs in adversarial training with OVR and cross-entropy on CIFAR10, CIFAR100, and SVHN. In Fig. 14, OVR decreases the logit margin losses more than cross-entropy on all dataset. We also evaluate $\ell_{\rm LM}(\boldsymbol{z}^{OVR})/\ell_{\rm LM}(\boldsymbol{z}^{CE})$ at the last epoch, which is expected to be about two from Theorem 4.5. Table 4 list $\ell_{\rm LM}(\boldsymbol{z}^{OVR})/\ell_{\rm LM}(\boldsymbol{z}^{CE})$ at the last epoch, and it is about two, and thus, logit margin losses in adversarial training follow Theorem 4.5 well even though we assume a simple problem that only considers one data point and assumes that logits are directly moved by the gradient for Theorem 4.5. Since the number of classes K of CIFAR100 is 100 and larger than other datasets, the logit margins of cross-entropy is larger than OVR at the beginning of training. This result corresponds to the case of CE ($K = 100$) in Fig. 3(a), and this phenomenon is also able to be explained by the simple problem Eq. (10). To the best of our knowledge, this is the first study that explicitly reveals the logit margin of minimization of cross-entropy depends on the number of classes. Though the logit margin loss of OVR in Theorem 4.5 does not depend on K, the logit margins of OVR on CIFAR100 is smaller than those on CIFAR10 and SVHN. This is because CIFAR100 is a more difficult dataset than CIFAR10 and SVHN: robust accuracies of CIFAR100 is about 25 % whereas those of CIFAR10 and SVHN are about 50 % in Tab. 1.

F.4 Effects of hyperparameters λ

SOVR has hyperparameters (M, λ). In this section, we evaluate the effects of λ. Figure 15 plots ℓ_{LM} on CIFAR10 with RN18, generalization gap, and robust accuracy against Auto-Attack. We set M to 40. Note that $\lambda = 0$ corresponds to that models are trained on only a set of $\mathbb{S}$, i.e., AT only using the 60 percent of the data points in minibatch when $M = 40$. First, $\ell_{LM}(\boldsymbol{x}')$ is monotonically decreasing due to increases in λ (Fig. 15(a)). However, robust accuracies against Auto-Attack are not monotonically increasing against λ (Fig. 15(c)). This is because generalization gap increases (Fig. 15(b)). Thus, too high weights on difficult samples causes overfitting. SOVR is always superior or comparable to AT in terms of robustness against Auto-Attack under all tested values of ($0 < M \leq 100$, $0 < \lambda \leq 1$).

F.5 Individually test of Auto-Attack

For importance-aware methods, we evaluate the robust accuracies against all components of Auto-Attack in Section 2.3. In this section, we additionally evaluate EWAT by individually using Auto-Attack and discuss the results of SOVR. Figure 16 plots the results, and SOVR is the most robust

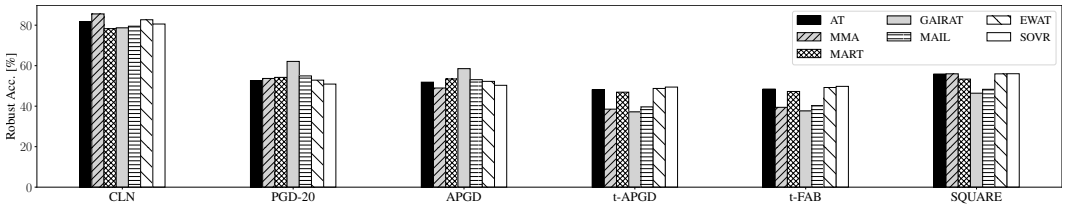


Figure 16: The robustness against PGD-20 and Auto-Attack on the test set of CIFAR10. We decompose the robust accuracy for Auto-Attack into robust accuracy in each phase.

against t-APG and t-FABand. In addition, it is more robust against SQUARE than AT and EWAT. Although the robust accuracy of SOVR against PGD-20 is lower than those of AT and EWAT, SOVR outperforms other methods in terms of the robustness against the worst-case attacks, which is the goal of this study.

F.6 Trade-off between clean accuracy and robust accuracy

Figure 17 plots the trade-off between clean accuracy and robust accuracy against Auto-Attack when using WideResNet34-10 (* uses WideResNet32-10) on CIFAR10. These results are the same as those reported in Tabs 1 and 2. Diagonal lines are lines satisfying $\mathrm{CleanAcc.} + \mathrm{RobustAcc.} = \mathrm{Const}$ through SOVR or SOVR+AWP. SOVR achieved good trade-off: Robust Acc. + Clean Acc. of SOVR and SOVR+AWP are the best value under each condition. If we require the most robust models with sacrificing clean accuracies, TSOVR and TSOVR+AWP achieved the best robust accuracy against Auto-Attack. Therefor, SOVR and TSOVR are better objective function than TRADES and cross-entropy in terms of trade-off and robustness. Note that AWP and SEAT require large overheads compared with SOVR.

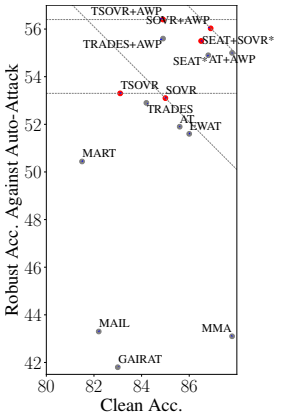


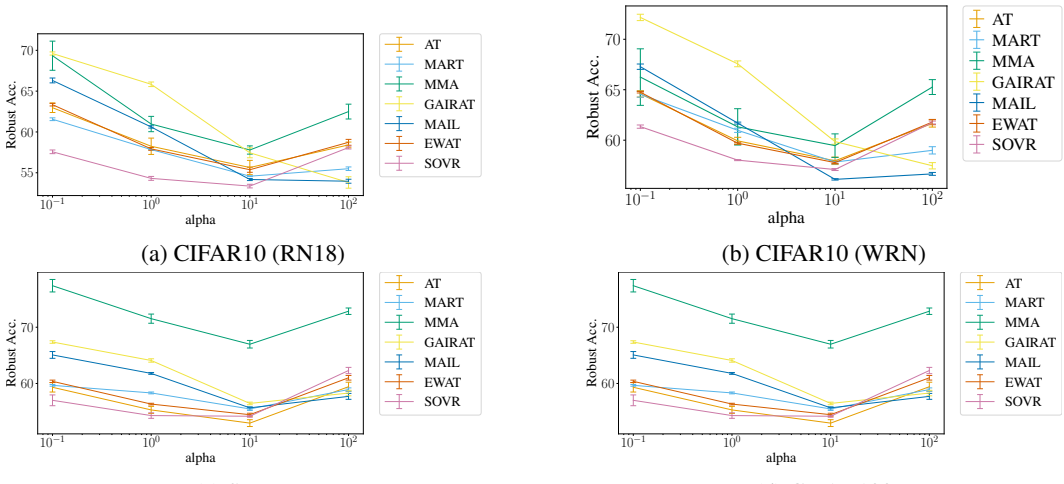
Figure 17: Trade-off between clean accuracy and robust accuracy.

F.7 Evaluation using various attacks

We list robust accuracies against various attacks; FGSM (Goodfellow et al., 2014), 100-step PGD (Madry et al., 2018), 100-step PGD with CW loss (Madry et al., 2018; Carlini & Wagner, 2017), SPSA (Uesato et al., 2018) in Tab. 5. Hyperparameters of SPSA are as follows: the number of steps is set to 100, the perturbation size is set to 0.001, learning rate is set to 0.01, and the number of samples for each gradient estimation is set to 256. In this table, we repeat the clean accuracies and robust accuracies against Auto-Attack from the table in the main paper. In addition, we list the worst robust accuracies, which are the least robust accuracy among attacks in the table for each method. In this table, importance-aware methods tend to fail to improve the robustness against SPSA. Since SPSA does not directly use gradients, this result indicates that importance-aware methods improve the robustness by obfuscating gradients (Athalye et al., 2018). Against some attacks, MMA achieves the highest robust accuracy on several datasets. However, our goal is improving the true robustness, i.e., robust accuracies against the worst-case attacks in $\boldsymbol{\delta} \in \{||\boldsymbol{\delta}||_\infty \leq 8/255\}$. MMA does not improve the robustness against the worst-case attacks (the columns of Worst). We can see that Auto-Attack always achieves the least robust accuracies, and SOVR improves them: Robust accuracies against Auto-Attack of SOVR are 5.9-12.2 percent points greater than those of MMA.

Table 5: Robust Accuracy against various attacks (L_∞, $\varepsilon = 8/255$). CLN denotes accuracy on clean data, and AA denotes Auto-Attack. Worst represents the least robust accuracy among attacks in the table for each method.

	Method	CLN	FGSM	PGD	CW	SPSA	AA	Worst
C10 (RN18)	AT	81.6±0.5	57.6±0.1	52.5±0.4	50.0±0.4	56.8±0.2	48.0±0.2	48.0±0.2
	MART	78.3±1	58.0±0.3	54.0±0.1	48.7±0.2	54.2±0.1	46.9±0.3	46.9±0.3
	MMA	**85.5 ± 0.7**	**65.5 ± 2**	51.6±0.2	51.0±0.6	56.3±1	37.2±0.9	37.2±0.9
	GAIRAT	78.7±0.7	63.1±0.7	**62.0 ± 0.4**	40.01±1	47.4±1	37.7±1	37.7±1
	MAIL	79.5±0.4	57.8±0.1	54.97±0.08	42.1±0.2	49.1±0.4	39.6±0.4	39.6±0.4
	EWAT	82.8±0.4	57.7±0.4	52.3±0.4	50.4±0.7	56.8±0.2	48.2±0.7	48.2±0.7
	SOVR	81.9±0.2	57.0±0.2	50.9±0.5	**51.5 ± 0.2**	**57.7 ± 0.2**	**49.4 ± 0.3**	**49.4 ± 0.3**
C10 (WRN)	AT	85.6±0.5	60.9±0.4	55.1±0.4	54.0±0.6	60.8±0.5	51.9±0.5	51.9±0.5
	MART	81.5±1	61.3±0.6	57.2±0.2	52.1±0.3	57.8±0.6	50.44±0.09	50.44±0.09
	MMA	**87.8 ± 1**	**68.6 ± 1**	55.7±1	**55.4 ± 0.7**	59.6±2	43.1±0.6	43.1±0.6
	GAIRAT	83.0±0.7	64.1±0.5	**62.9 ± 0.4**	44.4±0.7	52.1±0.5	41.8±0.6	41.8±0.6
	MAIL	82.2±0.4	59.3±0.5	56.0±0.5	45.7±0.2	53.0±0.2	43.3±0.1	43.3±0.1
	EWAT	86.0±0.5	60.6±0.4	54.5±0.1	53.8±0.3	60.7±0.4	51.6±0.3	51.6±0.3
	SOVR	85.0±1	60.8±0.1	54.5±0.2	55.2±0.2	**61.6 ± 0.1**	**53.1 ± 0.2**	**53.1 ± 0.2**
C100	AT	89.8±0.6	30.1±0.4	27.7±0.2	25.6±0.3	29.3±0.3	23.7±0.3	23.7±0.3
	MART	86.9±0.6	**31.0 ± 0.2**	**29.36 ± 0.06**	25.4±0.3	28.5±0.1	23.9±0.3	23.9±0.3
	MMA	**93.9 ± 0.4**	25.7±0.3	19.4±0.2	20.5±0.1	24.3±0.3	18.4±0.2	18.4±0.2
	GAIRAT	89.9±0.4	27.9±0.3	26.0±0.2	21.9±0.4	25.9±0.1	19.8±0.5	19.8±0.5
	MAIL	89.4±0.4	24.81±0.08	23.29±0.06	18.3±0.5	21.9±0.5	16.7±0.3	16.7±0.3
	EWAT	90.2±0.6	30.07±0.08	27.4±0.3	25.3±0.2	29.3±0.1	23.52±0.06	23.52±0.06
	SOVR	90.0±1	30.2±0.2	27.4±0.2	**26.1 ± 0.1**	**29.9 ± 0.1**	**24.3 ± 0.2**	**24.3 ± 0.2**
SVHN	AT	53.0±0.7	61.1±0.5	50.6±0.4	47.7±0.8	55.7±0.9	45.6±0.4	45.6±0.4
	MART	49.2±0.1	64.4±0.5	56.5±0.2	49.0±0.4	56.69±0.08	46.9±0.3	46.9±0.3
	MMA	**60.6 ± 0.6**	**79.6 ± 0.8**	**65.0 ± 1**	**59.1 ± 2**	**63.2 ± 2**	41.0±0.3	41.0±0.3
	GAIRAT	52.0±0.5	65.8±0.4	60.4±0.6	40.6±0.7	48.8±0.6	37.6±0.6	37.6±0.6
	MAIL	46.5±0.5	64.6±0.4	58.2±0.3	44.1±0.7	52.3±0.5	41.2±0.3	41.2±0.3
	EWAT	54.2±1	61.8±0.5	51.7±0.2	50.2±0.4	57.4±0.4	47.6±0.4	47.6±0.4
	SOVR	52.1±0.8	65.3±2	50.7±0.2	52.5±0.2	60.7±0.4	**48.5 ± 0.4**	**48.5 ± 0.4**



(a) CIFAR10 (RN18)

(b) CIFAR10 (WRN)

(c) SVHN

(d) CIFAR100

Figure 18: Robust Accuracy against logit scaling attack.

Table 6: Robust accuracy against Auto-Attack and clean accuracy for AT, SOVR, TRADES, and TSOVR.

Robust Accuracy against Auto-Attack (L_∞, $\varepsilon = 8/255$)				
	AT	SOVR	TRADES	TSOVR
CIFAR10 (RN18)	48.0±0.2	49.4± 0.3	48.8±0.3	$\mathbf{49.8 \pm 0.1}$
CIFAR10 (WRN)	51.9±0.5	53.1±0.2	52.9 ± 0.3	$\mathbf{53.3 \pm 0.1}$
SVHN (RN18)	45.6±0.4	48.5±0.4	49.4± 0.3	$\mathbf{49.87 \pm 0.02}$
CIFAR100 (RN18)	23.7±0.3	24.3±0.2	23.2±0.1	$\mathbf{24.61 \pm 0.07}$
Clean Accuracy				
CIFAR10 (RN18)	81.6±0.5	81.9±0.2	$\mathbf{82.5 \pm 1}$	81.4±0.2
CIFAR10 (WRN)	$\mathbf{85.6 \pm 0.1}$	85.0±0.2	84.2±0.7	83.1±0.2
SVHN (RN18)	89.8±0.6	$\mathbf{90.0 \pm 1}$	88.7±1	88.5±0.5
CIFAR100 (RN18)	53.0±0.7	52.1±0.8	$\mathbf{53.8 \pm 0.3}$	51.6±0.2

F.8 Evaluation using logit scaling attack

In this section, we evaluate the robustness against the logit scaling attack (Hitaj et al., 2021). Hitaj et al. (2021) reveals that GAIRAT tends to be vulnerable to logit scaling attacks. The logit scaling attack multiplies logits by α before applying softmax when generating PGD attacks. We set $\alpha = [0.1, 1.0, 10, 100]$. Fig. 18 plots robust accuracy against α. This figure shows that the robust accuracies of GAIRAT and MAIL tend to decrease when increasing α. Though the robust accuracy of SOVR is the lowest on CIFAR 10 (RN18), it is higher than the robust accuracy against Auto-Attack. Thus, the logit scaling attack is not the worst-case attack. Since the robust accuracy of SOVR does not necessarily decrease when increasing α, the results seem to be caused by high robustness against PGD (Tab. 5) of other methods rather than vulnerability to logit scaling of SOVR. Previous methods tend to be designed to increase the robustness against PGD since Auto-Attack is a relatively recent attack. On the other hand, SOVR is designed to increase the robustness against the worst-case attack, which is Auto-Attack for now.

F.9 Comparison with TRADES

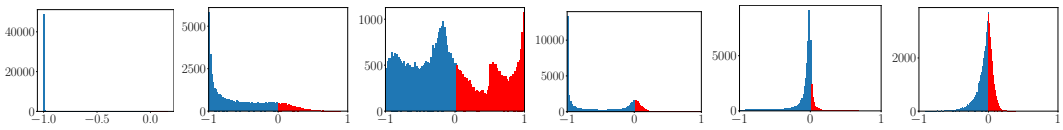
Section 6 gives the results of TRADES and TSOVR on CIFAR10 with WRN. This section compares our methods with TRADES by using other datasets and RN18. For TSOVR, (M, λ) is set to (80, 0.8) for CIFAR10 (RN18), set to (50, 0.5) for CIFAR100, and set to (20, 0.8) for SVHN. Tab. 6 lists the robust accuracies against Auto-Attack and clean accuracies. We can see that TSOVR achieves the best robust accuracy against Auto-Attack.

F.10 Dependence of the number of classes

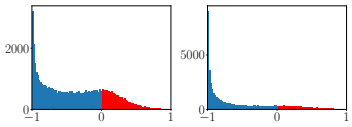
Figure 6 shows that the rate of AT gets close to SOVR on CIFAR100. This is because the number of classes of CIFAR100 ($K = 100$) is ten times larger than other datasets ($K = 10$), and logit margins of cross-entropy depend on the number of classes K (Eq. (17)). Thus, this result is a piece of evidence that Theorem 4.5 explains the difference of logit margins between OVR and cross-entropy. In certain finite time step t (not the limit), Eqs. (16) and (17) show that the difference between OVR and cross-entropy depends on the number of classes K. Even so, Fig. 14(b) shows that the increase rate of logit margins of OVR is larger than that of cross-entropy against epochs. To achieve better performance, we can tune the hyper-parameter λ, which corresponds to w_1 in Eq. (16) of Theorem 4.5. When using λ= 0.6, SOVR achieves better robustness than $\lambda = 0.5$ on CIFAR100 (Tab. 7). Cross-entropy also has the weight w_2 in Eq. (17), and it is automatically tuned in GAIRAT, MAIL, and EWAT. However, this tuning does not achieve comparable performance to SOVR.

Table 7: Robust accuracy against Auto-Attack on CIFAR100 tuning λ.

Robust Accuracy against Auto-Attack (L_∞, $\varepsilon = 8/255$)			
	AT	SOVR (λ=0.5)	SOVR (λ=0.6)
CIFAR100 (RN18)	23.7±0.3	24.3±0.2	$\mathbf{24.6 \pm 0.1}$
Clean Accuracy			
CIFAR100 (RN18)	$\mathbf{53.0 \pm 0.7}$	52.1±0.8	51.9 ± 0.6



(a) ST (b) AT (c) MMA (d) MART (e) GAIRAT (f) MAIL



(g) EWAT (h) SOVR

Figure 19: Histogram of probabilistic margin losses for training data of CIFAR10 with PreActResNet18 at the last epoch. ST denotes standard training, i.e., training on clean data. For standard training, we use PM on clean data $\boldsymbol{x}$, while we plot that on adversarial examples $\boldsymbol{x}'$ for the other methods. Blue bins correspond to the correctly classified data points, and red bins are misclassified samples.

G HISTOGRAM OF PROBABILISTIC MARGIN LOSSES

While our study focuses on the logit margin loss, MAIL (Liu et al., 2021) uses the probabilistic margin,

$$PM_n = f_{y_n}(\boldsymbol{x}'_n, \boldsymbol{\theta}) - \max_{k \neq y_n} f_k(\boldsymbol{x}'_n, \boldsymbol{\theta}), \tag{89}$$

to evaluate the difficulty of data point. In the same way as Fig. 2, Fig. 19 plots the histograms of probabilistic margins on CIFAR10 with PreActResNet18. Since softmax output is bounded in $[0, 1]$, PM is bounded in [-1,1]. As a result, most correctly classified data points concentrate near -1. In addition, softmax uses exponential functions, distributions of PM are similar to the exponential distributions. Due to these effects, histograms of PM make it more difficult to discover the fact that there are two types of data points (easy samples and difficult samples). Since softmax preserves the order of logit, and a classifier infers the label by using the largest logit, the analysis by using PM can under-estimate the distribution of difficult samples. Thus, logit margin losses are more suitable to empirically analyze trained models. Since softmax preserves the order of logit, the probabilistic margin can be used to determine $\mathbb{L}$ and $\mathbb{S}$ in SOVR.