SELF-ADAPTIVE PERTURBATION RADII FOR ADVERSARIAL TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial training has been shown to be the most popular and effective technique to protect models from imperceptible adversarial samples. Despite its success, it also accompanies the significant performance degeneration to clean data. To achieve a good performance on both clean and adversarial samples, the main effort is searching for an adaptive perturbation radius for each training sample, which essentially suffers from a conflict between exact searching and computational overhead. To address this conflict, in this paper, firstly we show the superiority of adaptive perturbation radii intuitively and theoretically regarding the accuracy and robustness respectively. Then we propose our novel self-adaptive adjustment framework for perturbation radii without tedious searching. We also discuss this framework on both deep neural networks (DNNs) and kernel support vector machines (SVMs). Finally, extensive experimental results show that our framework can improve not only natural generalization performance but also adversarial robustness. It is also competitive with existing searching strategies in terms of running time.

1 INTRODUCTION

The security of machine learning models has long been questioned since most models are vulnerable to perturbations (Papernot et al., 2016). Extremely tiny perturbations may be imperceptible to human beings but yet cause poor performance of models such as deep neural networks (DNNs) (Goodfellow et al., 2014; Madry et al., 2017; Papernot et al., 2017), support vector machines (SVMs) (Xiao et al., 2012; Biggio et al., 2012; 2014) and logistic regression (LR) (Papernot et al., 2016). Examples attacked by such perturbations are generally called as adversarial examples.

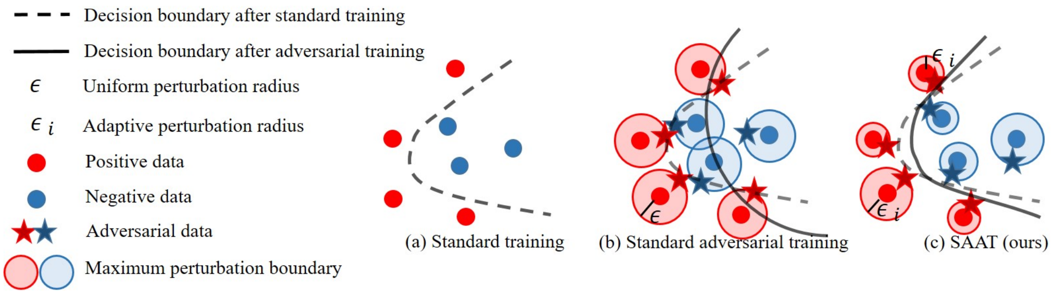
To learn robust models, adversarial training has now become one of the most effective and widely-used methods, especially on DNNs and SVMs (Zhou et al., 2012; Kurakin et al., 2017; Miyato et al., 2018; Wang et al., 2019; Shafahi et al., 2019; Wu et al., 2021). However, the success of adversarial training comes at a cost (Tsipras et al., 2018; Zhang et al., 2019). Specifically, as stated in (Tsipras et al., 2018), robustness may be at odds with accuracy, which means models after adversarial training may fail to generalize well on unperturbed examples. It is generally believed that this phenomenon is due to the fixed strength of attack throughout the training process, which ignores the fact that every example may have different intrinsic robustness (Cheng et al., 2020; Zhang et al., 2020).

Naturally, the main effort to mitigate this issue is to find the suitable perturbation radius ϵ_i for each training sample with explicit or implicit searching strategies. For explicit searching strategies, IAAT (Balaji et al., 2019) uses the brute-force search to find the suitable perturbation radii. MMA (Ding et al., 2018) aims to find the optimal ϵ_i^* via the bisection search. For implicit searching strategy, Zhang et al. propose an early-stopped PGD strategy called FAT, which adjusts ϵ implicitly in essence. Although FAT skillfully skips the step of searching ϵ_i^*, it is sensitive to hyperparameters such as steps of PGD attack τ and the uniform perturbation radius ϵ. Thus, in this paper, we mainly focus on explicit searching strategies. We also give a brief review of the above algorithms in Table 1. From this table, we can see that these searching strategies essentially have an inherent conflict between exact searching and time complexity.

To solve this conflict, in this paper, we propose a novel self-adaptive adjustment framework (SAAT) for perturbation radii. It achieves a better trade-off between natural generalization performance and adversarial robustness without much computational overhead. Firstly, for the adaptive perturbation

Table 1: Comparisons of different adversarial training algorithms which aim at achieving better generalization performance on DNNs and SVMs. (Complexity here refers to the time complexity. n is the training size, T is the number of epochs, K and τ are the numbers of steps for PGD attack, where $\tau \leq K$, c_1 and c_2 denote searching steps for ϵ_i.)

	Algorithm	**Problem**	**Finding suitable ϵ_i**		**Complexity**
			Solving inner problem	**Searching strategy**	
DNNs	Standard (Madry et al., 2017)	Minimax	K-PGD attack	–	$O(nKT)$
	IAAT (Balaji et al., 2019)	Minimax	K-PGD attack	Brute-force search	$O(c_1nKT)$
	MMA (Ding et al., 2018)	Minimax	K-PGD attack	Bisection search	$O(c_2nKT)$
	FAT (Zhang et al., 2020)	Minimax	K-τ-PGD attack	Early-stopped PGD attack	$O(n\tau T)$
	SAAT-kernel (Ours)	Minimization	Closed-form solution	Closed-form solution	$O(nT)$
	SAAT-minimax (Ours)	Minimax	K-PGD attack	Closed-form + fine search	$O(nKT)$
SVMs	adv-SVM (Wu et al., 2021)	Minimization	Closed-form solution	–	$O(nT)$
	SAAT-SVM (Ours)	Minimization	Closed-form solution	Closed-form solution	$O(nT)$


Decision boundary after standard training
Decision boundary after adversarial training
ϵ Uniform perturbation radius
ϵ_i Adaptive perturbation radius
Positive data
Negative data
Adversarial data
Maximum perturbation boundary
ϵ
ϵ_i
ϵ_i

(a) Standard training (b) Standard adversarial training (c) SAAT (ours)

Figure 1: Conceptual illustration of standard adversarial training and our self-adaptive adversarial training (i.e., SAAT).

radius, we intuitively show its superiority in generalization and theoretically illustrate its strength in robustness. Then we design a new learning objective that can construct a self-adaptive perturbation radius for each sample inspired by self-paced learning (SPL) (Jiang et al., 2015). We discuss SAAT on not only DNNs but also kernel SVMs. Correspondingly, we propose two types of optimization algorithms. One is built on the original minimax formulation of adversarial training. It determines the optimal perturbation radii based on the observation that the inner maximization is piecewise approximately linear to perturbation radii. Then we use a fine search to calibrate the values. The other is built on kernel perspective for SVMs and DNNs which transforms the original minimax objective function into an equivalent minimization one. Extensive experimental results show that our framework enjoys better natural generalization performance and higher adversarial robustness compared with other adversarial training algorithms. It is also competitive with existing searching strategies in terms of training time. We summarize the main contributions as follows:

- Theoretically, we prove that adaptive perturbation radii contribute to a lower expected adversarial risk than fixed and uniform perturbation radii, which implies higher robustness against adversarial examples.
- Our self-adaptive adversarial training algorithms can skillfully assign the optimal perturbation radius for each data, which avoids the step of exact searching and achieves a better trade-off between adversarial robustness and natural accuracy.
- The self-adaptive adversarial training strategy that we propose from the kernel perspective is applicable to both SVMs and DNNs. It efficiently optimizes a minimization problem instead of the conventional minimax one since we transform the inner maximization into a simplified and equivalent form.

2 PRELIMINARIES

2.1 NOTATIONS

We focus on C-class classification problems, then the dataset can be defined as $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ is the input data, and $y_i \in \{1, \cdots, C\}$ is the label. We will use $\mathbb{I}\{a\}$, the 0-1 loss, to represent an indicator function, which returns 1 if a is true and 0 otherwise.We will use $l(\cdot)$ to indicate the surrogate loss function of 0-1 loss. The set $\mathcal{B}(\delta, \epsilon) = \{\delta : ||\delta||_p \leq \epsilon\}$ means that the sample is constrained by an l_p-normed [1] perturbation δ with the perturbation radius ϵ. We denote the maximum adversarial loss as $\hat{l}(x, y, f, \epsilon) = \max_{\delta \in \mathcal{B}(\delta,\epsilon)} l(f(x + \delta), y)$, where $f \in \mathcal{F}$ and $\mathcal{F} : \mathcal{X} \to \mathbb{R}$ is one neural network class with depth-D and width-H:

$$\mathcal{F} = \{x \to W_D\rho(W_{D-1}\rho(\cdots W_1x \cdots)), ||W_i||_F \leq M_i, i \in [D]\}. \tag{1}$$

Here $\rho(\cdot)$ is an activation function with L_ρ-Lipschitz and W_i is a $H_i \times H_{i-1}$ matrix. Then, we have $H = \max\{H_0, \cdots, H_D\}$, $H_D = 1$ and $H_0 = d$. Thus the class of the maximum adversarial loss can be formulated as $\hat{l}_{\mathcal{F}} = \{\hat{l}_f : f \in \mathcal{F}\}$.

2.2 STANDARD ADVERSARIAL TRAINING

The standard adversarial training considers a minimax problem as follows:

$$\min_w \frac{1}{n} \sum_{i=1}^{n} \max_{\delta_i \in \mathcal{B}(\delta_i,\epsilon)} l(y_i, f_w(x_i + \delta_i)), \tag{2}$$

where w is the model parameter, $x_i + \delta_i$ is the adversarial example of x_i. The inner maximization problem actually follows the principle of adversarial attack and aims to construct the most aggressive adversarial examples (Madry et al., 2017), while the outer minimization is to find model parameters to minimize the loss caused by the adversarial examples. It is notable that a fixed and uniform perturbation radius ϵ is exerted for all training samples here.

3 SELF-ADAPTIVE ADVERSARIAL TRAINING

In this section, we first show the superiority of adaptive perturbation radii intuitively and theoretically. Inspired by that, we formulate a novel framework for self-adaptive adversarial training. Then we propose SAAT-minimax to solve the objective.

3.1 SUPERIORITY OF ADAPTIVE PERTURBATION RADII

Although adversarial training with adaptive perturbation radii has been widely studied empirically, its theoretical advantages are seldom explored. To fill this vacancy, in the following section, we first intuitively show its superiority on natural generalization and then theoretically illustrate its strength on adversarial robustness.

Adaptive Perturbation Radii Contribute to Better Generalization Performance.
Intuitively, as shown in Fig. 1b, for standard adversarial training, the perturbation radii are kept the same for all training samples. However, for samples near the decision boundary, enforcing large perturbation radii will lead to the cross-over mixture of samples in different classes. In this case, it leads to a distorted and undesirable decision boundary and unavoidably destroy the accuracy on unperturbed examples. Thus, we come to the idea of adversarial training with adaptive perturbation radii. As shown in Fig. 1c, the perturbation radii are set according to the specific location of the samples. It effectively avoids the severe distortion of the decision boundary and will not hurt the natural generalization much.

Adaptive Perturbation Radii Contribute to Lower Adversarial Risk.
In this part, we theoretically prove that adaptive perturbation radii can lead to a tighter upper bound of adversarial risk than fixed ones in the case of binary classification, which implies higher robustness against adversarial examples.

Firstly, we provide the definition of the expected adversarial risk $\mathcal{R}_{rob}$ as follows:

Definition 1. (Expected Adversarial Risk) Following Zhang et al. (2019); Schmidt et al. (2018); Bubeck et al. (2019), to characterize the robustness of a binary classifier f: $\mathbb{R} \to \{0, 1\}$, the expected adversarial risk can be defined as

$$\mathcal{R}_{rob}(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\mathbb{I}\{\exists\delta \in \mathcal{B}(\delta, \epsilon) : yf(x + \delta) \leq 0\} \tag{3}$$

---

[1] In this paper, we consider the l_p-norm ball of $p \geq 1$ such that the region is convex.

Based on Figure 1 above, we do not want to further increase ϵ_i if the adversarial example already can be misclassified by the classifier. This leads to the definition of the theoretically optimal adaptive perturbation radius ϵ_i^* as follows:

Definition 2. (Optimal Adaptive Perturbation Radius) Theoretically, the optimal adaptive perturbation radius ϵ_i^* for each sample can be defined as

$$\epsilon_i^* = \begin{cases} \epsilon_{\max}, & \text{if } \forall \delta_i \in \mathcal{B}(\delta_i, \epsilon_{\max}),\ y_i f(x_i + \delta_i) > 0, \\ \arg\min_{\epsilon_i \leq \epsilon_{\max}} \epsilon_i,\ s.t.\ \exists \delta_i \in \mathcal{B}(\delta_i, \epsilon_i),\ y_i f(x_i + \delta_i) \leq 0, & \text{otherwise.} \end{cases} \tag{4}$$

where $\epsilon_{\max}$ is the maximum perturbation radius, ϵ_i is the perturbation radius assigned to x_i.

Remark 3. A few examples of the optimal adaptive perturbation radius can be seen in Figure 1c. For the samples that are misclassified after adversarial attack, ϵ_i^* is the minimum radii that achieve this goal. For the samples that can be robustly classified even with $\epsilon_{\max}$, ϵ_i^* equals to $\epsilon_{\max}$.

Before giving our main theorem (i.e., Theorem 5), we provide Assumption 4 as follows.

Assumption 4. For the binary classification surrogate loss function $l(\cdot)$, we assume it can be written as $l(f(x), y) = \phi(yf(x))$, where ϕ is a non-increasing function and is L_ϕ-Lipschitz.

Examples of satisfied loss functions include hinge loss, logistic loss(Xiang, 2011), exponential loss (Wyner, 2003) and many others. Based on Assumption 4, the upper bound to the expected adversarial risk can be gotten as follows, the detailed proof is in Appendix A.

Theorem 5. *When Assumption 4 holds, for any $\omega \in (0, 1)$ and any $\hat{l}_f \in \hat{l}_{\mathcal{F}}$, with probability at least $1 - \omega$, the following holds:*

$$R_{rob}(f) \leq \frac{1}{n} \sum_{i=1}^{n} \hat{l}\,(x_i, y_i, f, \epsilon_i^*) + 3B\sqrt{\frac{\log 2/\omega}{2n}} + \frac{24B}{\sqrt{n}} L_\phi L_\rho^{D-1} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(X_p + \epsilon_{\max})Q.$$

where $X_p = \max\{||x_i||_p\}_{i=1}^n$, $Q = \frac{24B}{\sqrt{n}} L_\phi L_\rho^{D-1} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(X_p + \epsilon_{\max})\sqrt{\log \prod_{i=1}^D \frac{\pi^{H_i H_{i-1}/2}}{\Gamma(\frac{H_i H_{i-1}}{2}+1)} M_i^{H_i H_{i-1}}} \prod_{i=1}^D M_i$ *and* Γ *means the gamma function.*

Then we give Theorem 6 to show that the maximum loss function $\hat{l}(x_i, y_i, f, \epsilon)$ increases with regard to ϵ. The detailed proof is provided in Appendix B.

Theorem 6. *The maximum loss function $\hat{l}(x_i, y_i, f, \epsilon)$ is an increasing function with regard to the perturbation radius ϵ.*

Remark 7. Combing Theorem 5 with Theorem 6, it is evident that replacing $\epsilon_{\max}$ with ϵ_i^* will contribute to a tighter upper bound for the expected adversarial risk $\mathcal{R}_{rob}$. It indicates that adaptive perturbation radius in training stage is a better choice than fixed and uniform radius that can lead to higher adversarial robustness.

3.2 FRAMEWORK OF SELF-ADAPTIVE ADVERSARIAL TRAINING

Although several methods have been proposed to search for a suitable perturbation radius for each training sample, there exists a conflict between exact searching and computational load, as mentioned in Section 1 and Table 1. To achieve fast self-adaptive adversarial training, we creatively introduce a self-adaptive regularizer of perturbation radii (i.e., $-\lambda \frac{1}{n} \sum_{i=1}^n \epsilon_i$) into formulation (2), and give our new formulation of self-adaptive adversarial training (SAAT) as follows:

$$\begin{aligned} &\min_{w,\epsilon} \frac{1}{n} \sum_{i=1}^{n} \left\{ \max_{\|\delta_i\|_p \leq \epsilon_i} l(y_i, f_w(x_i + \delta_i)) - \lambda \epsilon_i \right\}, \\ &s.t.\ \epsilon_i \in [0, \epsilon_{\max}],\quad i = 1, \ldots, n. \end{aligned} \tag{5}$$

where ϵ_i is the customized perturbation radius of x_i achieved by the self-adaptive item and λ is the regularization parameter. Thus ϵ_i can update dynamically as the maximum adversarial loss of x_i changes.

Remark 8. Note that a similar term is used in self-paced learning (SPL) (Jiang et al., 2015). The core idea of SPL is to learn a model by gradually including samples from easy to complex according to their losses since SPL decides whether the samples can be selected into training via a self-paced regularization. Inspired by SPL, we aim to assign a specific perturbation radius ϵ_i to each sample according to its loss. Formally, we design a self-adaptive regularizer imposed on ϵ_i and add it to the original formulation of adversarial training.

3.3 SAAT-MINIMAX

In this part, we aim to optimize the SAAT framework (5). Specifically, we propose a two-stage search strategy to find the approximate optimal perturbation radii. The first stage is built on the closed-form solution via the piecewise approximate linearity of $\hat{l}(x_i, y_i, f_w, \epsilon_i)$ wrt. the perturbation radii ϵ_i. The second stage is a fine search to calibrate the results of the first stage. In the following, we will discuss the two-stage search strategy in detail.

Firstly, we observe that $\hat{l}(x_i, y_i, f_w, \epsilon_i)$ is piecewise approximately linear with regard to ϵ_i for each sample as shown in Fig. 2 and propose Assumption 9. This assumption is verified in Appendix F.3.

Assumption 9. $\hat{l}(x_i, y, f_w, \epsilon_i)$ is piecewise linear with regard to ϵ_i as follows:

$$\hat{l}(x_i, y_i, f_w, \epsilon_i) = \max(0, k\epsilon_i + b) \quad (6)$$

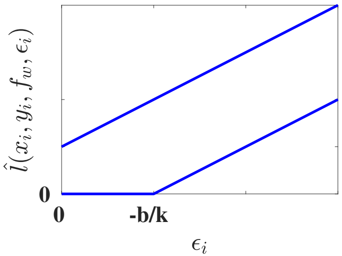


Figure 2: The sketch map of $\hat{l}_f(z_i, \epsilon_i)$ wrt. ϵ_i.

where $k > 0$ is the slope of $\hat{l}$ with regard to ϵ_i and b denotes y-intercept.

Then, with the aid of Assumption 9, we come to Theorem 10, which provides the approximate optimal perturbation radii ϵ_i^* for optimizing objective function (5). Its proof is presented in Appendix C. The detailed setting of k_i and b_i can be seen in section 5.1.3.

Theorem 10. *For the minimization problem* $\min_{\epsilon_i \in [0, \epsilon_{max}]} \hat{l}(x_i, y_i, f_w, \epsilon_i) - \lambda\epsilon_i$, *if* f_w *is given, and Assumption 9 holds, we have the optimal* ϵ_i^* *as follows:*

$$\epsilon_i^* = \begin{cases} 0, & \textit{if} \;\; b_i \geq 0 \textit{ and } k_i \geq \lambda; \\ -\frac{b_i}{k_i}, & \textit{if} \;\; b_i < 0 \textit{ and } k_i \geq \lambda; \\ \epsilon_{max}, & \textit{otherwise}. \end{cases} \quad (7)$$

The second stage is a fine search to calibrate the results of Theorem 10. Since Assumption 9 may not hold exactly, we use a simple search strategy with a fixed step size to find more accurate values of ϵ_i^*. Specifically, if the PGD attack fails to find an adversarial image $x_i + \delta_i$ that can be misclassified, it implies ϵ_i^* is too small. Thus, we set $\epsilon_i^* = \epsilon_i^* + \eta$. Otherwise, we set $\epsilon_i^* = \epsilon_i^* - \eta$, where η is a pre-specified fixed step size.

Finally, we combine the above two-stage search strategy with the standard adversarial training procedure and give the pseudo-code of our SAAT-minimax in Algorithm 1.

4 SELF-ADAPTIVE ADVERSARIAL TRAINING FROM KERNEL PERSPECTIVE

As we all know, traditional adversarial training aims to optimize a minimax problem. It typically uses a gradient-based iterative solver such as multi-step PGD to approximately solve the inner problem, which often leads to high computational overhead. To solve this problem, we propose a new self-adaptive adversarial training strategy from kernel perspective[2]. Specifically, it efficiently transforms the minimax problem (5) into an equivalent minimization one. Then we discuss the detailed self-adaptive adversarial training algorithms via the kernel perspective for both DNNs and SVMs.

---

[2] The kernel perspective means that our function f is in the reproducing kernel Hilbert space (RKHS) (Iii, 2004).

Algorithm 1 SAAT-minimax with l_∞-norm constrained perturbations

```
Input: $\mathcal{D}$ : training set; $T$ : number of epochs; $\epsilon_{max}$ : maximum perturbation radius; $\gamma$ : learning rate; $K$ : PGD steps; $\alpha$ : PGD step size; $B$ : batch size.
Output: $w$.
for epoch= $1, \cdots, T$ do
    Choose a batch of training samples $\{(x_i, y_i)\}_{i=1}^B \sim \mathcal{D}$.
    Obtain $\epsilon_i^*$ via Theorem 10.
    $\epsilon_i^* = \max(\min(\epsilon_i^*, \epsilon_{max}), 0)$.
    for $k = 1, \cdots, K$ do
        $\delta_i = \delta_i + \alpha \cdot \mathrm{sign}(\nabla_{\delta_i} l(y_i, f_w(x_i + \delta_i)))$.
        $\delta_i = \max(\min(\delta_i, \epsilon_i^*), -\epsilon_i^*)$.
        Calibrate $\epsilon_i^*$ via the fine search strategy.
    end for
    $w = w - \gamma \nabla_w l(y_i, f_w(x_i + \delta_i))$.
end for
```

4.1 PRIMARY RESULTS FROM KERNEL PERSPECTIVE

The transformation of the minimax problem (5) contains two steps: firstly we map the perturbations from linear to kernel spaces, then we can solve the unconstrained equivalent form of the inner minimization.

We first discuss the kernelization of the perturbations δ. For an adversarial example $x + \delta$ in the linear space, it is known that if we map it into the kernel space, the kernelized example $\phi(x + \delta)$ will be unpredictable, here $\phi(\cdot)$ is the feature mapping function. Fortunately, Theorem 14 in (Xu et al., 2009) provides a tight connection between perturbations in the linear and kernel space, i.e., the perturbation range of $\phi(x) + \delta_\phi$ tightly covers that of $\phi(x + \delta)$, where δ_ϕ is the perturbation in the kernel space and $\|\delta_\phi\|_2 \leq \sqrt{2f(0) - 2f(\epsilon)}$. Since we can use a l_2-norm ball to wrap a l_p-norm ball, e.g., $\{\|\delta\|_\infty \leq \epsilon\} \subseteq \left\{\|\delta\|_2 \leq \sqrt{2}\epsilon\right\}$, this theorem is applicable to other norms as well.

Based on it, our formulation of self-adaptive adversarial training (5) can be rewritten as the following form in the RKHS $\mathcal{H}$:

$$\min_{f \in \mathcal{H}, \epsilon'} \frac{1}{n} \sum_{i=1}^{n} \left\{ \max_{\|\delta_\phi^i\|_2 \leq \epsilon_i'} l\left(y_i, \langle f, \phi(x_i) + \delta_\phi^i \rangle_{\mathcal{H}}\right) - \lambda \epsilon_i' \right\},$$
$$s.t.\ \epsilon_i' \in [0, \epsilon_{\max}'],\ \ i = 1, \ldots, n. \tag{8}$$

where $\epsilon_i' = \sqrt{2f(0) - 2f(\epsilon_i)}$, $\epsilon_{\max}' = \sqrt{2f(0) - 2f(\epsilon_{\max})}$.

Then we can obtain the simplified and equivalent form of the inner maximization of Eq. (8) via Theorem 11. The detailed proof can be found in Appendix D.

Theorem 11. *If f is a function in an RKHS $\mathcal{H}$, the inner maximization problem $\max_{\|\delta_\phi^i\|_2 \leq \epsilon'} l(y_i, \langle f, \phi(x_i) + \delta_\phi^i \rangle_{\mathcal{H}})$ in (8) is equivalent to the regularized loss function $l\left(y_i, f(x_i) + \epsilon' \|f\|_{\mathcal{H}}\right)$, where $\|\cdot\|_{\mathcal{H}}$ stands for the norm in the RKHS.*

According to this theorem, our goal turns to optimize the following minimization problem:

$$\min_{f \in \mathcal{H}, \epsilon'} \frac{1}{n} \sum_{i=1}^{n} \left\{ l\left(y_i, f(x_i) + \epsilon_i' \|f\|_{\mathcal{H}}\right) - \lambda \epsilon_i' \right\}. \tag{9}$$
$$s.t.\ \epsilon_i' \in [0, \epsilon_{\max}'],\ \ i = 1, \ldots, n.$$

For the new problem (9), it is obvious that Theorem 10 can be easily applied here to get the optimal perturbation radius $\epsilon_i'^*$ as well, since we denote $l\left(y_i, f(x_i) + \epsilon_i' \|f\|_{\mathcal{H}}\right)$ as $\hat{l}(x_i, y_i, f, \epsilon_i')$.

In this case, we give the optimization framework of SAAT from the kernel perspective in Algorithm 2, which clearly shows the alternative updating for $\{\epsilon_i'^*\}_{i=1}^n$ and function f. In the following subsection, we will discuss its applications on DNNs and kernel SVMs in detail.

Algorithm 2 SAAT on Kernel Perspective

```
Input: $\epsilon'_{max}$, $\lambda_0$, $\mu$.
Output: $\{\epsilon_i'^*\}_{i=1}^n$.
Initialize $\lambda = \lambda_0$.
while not converged do
    Update $\{\epsilon_i'^*\}_{i=1}^n$ via Theorem 10 with fixed $f$.
    Update $f$ with fixed $\{\epsilon_i'^*\}_{i=1}^n$ on DNNs or kernel SVMs.
    $\lambda \leftarrow \mu\lambda$.
end while
```

4.2 SPECIFIC ALGORITHMS ON DNNS AND SVMS

4.2.1 SAAT-KERNEL ON DNNS

Since the RKHS norm $\|f\|_{\mathcal{H}}$ cannot be computed on DNNs, we use the lower bound of $\|f\|_{\mathcal{H}}$ proposed in (Bietti et al., 2019) to approximate its value:

$$\|f\|_{\mathcal{H}} \geq \|f\|_{\delta}^2 := \sup_{\|\delta\|_2 \leq 1} f(x+\delta) - f(x). \tag{10}$$

In this way, since the optimal solution for the perturbation radii ϵ_i' has already been attained, we can easily optimize learning objective (9) via optimization algorithms such as SGD (Bottou, 2010) and ADAM (Kingma & Ba, 2014). The procedures to alternatively optimize $\{\epsilon_i'^*\}_{i=1}^n$ and the model function f is shown in Algorithm 2.

4.2.2 SAAT-SVM ON KERNEL SVMS

Similar with SAAT on DNNs, SAAT on kernel SVMs can be formulated as the following problem:

$$\begin{aligned} \min_{f\in\mathcal{H},\epsilon'} \ & \frac{\|f\|_{\mathcal{H}}^2}{2} + \frac{C}{n}\sum_{i=1}^{n}\left\{l(y_i, f(x_i) + \epsilon_i'\|f\|_{\mathcal{H}}) - \lambda\epsilon_i'\right\}. \\ & s.t.\ \epsilon_i' \in [0, \epsilon'_{\max}],\ \ i = 1, \dots, n. \end{aligned} \tag{11}$$

where $\frac{1}{2}\|f\|_{\mathcal{H}}^2$ is the added norm similar to the SVM formulation in (Dai et al., 2014). As the doubly stochastic gradient descent (DSG) algorithm (Dai et al., 2014) has been proved to be a powerful technique for scalable kernel learning, here we use it optimize Eq. (11). The detailed optimization procedure is provided in Appendix E.

5 EXPERIMENTS

In this section, we compare SAAT with different adversarial training algorithms on MNIST (Lecun & Bottou, 1998), CIFAR10 (Krizhevsky & Hinton, 2009) and CIFAR100 (Krizhevsky & Hinton, 2009) under l_2/l_∞-norm constrained perturbations. Due to the page limit, we only show partial results of l_∞ norm in the following, other results are presented in Appendix F.1 and F.2. Experiments on kernel SVMs and the verification of Assumption 9 are also presented in Appendix F.4 and F.3.

5.1 EXPERIMENTAL SETUP

5.1.1 COMPARED ALGORITHMS:

- **Natural**: Natural model training on DNNs which minimizes the cross entropy loss.
- **Standard** (Madry et al., 2017): The standard adversarial training method which uses the K-step PGD as an attacker.
- **IAAT** (Balaji et al., 2019): Instance adaptive adversarial training which uses brute-force search to assign instance-specific perturbation radius ϵ_i to each sample.
- **MMA** (Ding et al., 2018): Max-margin adversarial training which directly maximizes the distances from inputs to the decision boundary via binary search for the optimal perturbation radii.
- **FAT** (Zhang et al., 2020): A friendly adversarial training strategy which generates friendly adversarial data by stopping the adversarial data searching algorithms early.

- **TRADES** (Zhang et al., 2019): This method aims to achieve a trade-off between robustness and accuracy via decomposing the robust error as the sum of natural error and boundary error.
- **SAAT-kernel**: Our self-adaptive adversarial training algorithm on DNNs from the kernel perspective. We apply both the hinge loss and the cross entropy loss in the experiments, i.e., SAAT-kernel$_h$ and SAAT-kernel$_c$.
- **SAAT-minimax**: Our self-adaptive adversarial training algorithm on the minimax problem for DNNs. We apply both the hinge loss and the cross entropy loss in the experiments, i.e., SAAT-minimax$_h$ and SAAT-minimax$_c$.

Table 2: Test accuracy (%) of various defense methods trained on MNIST with l_∞-norm constrained perturbations on DNNs. (The results of Natural on clean data are just baselines for reference.)

Method	Clean	FGSM	10-PGD	CW	AutoAttack
Natural	*98.83±0.29*	14.84±0.66	0.00±0.00	0.00±0.00	0.00±0.00
Standard	97.82±0.35	95.00±0.78	89.55±0.89	87.27±0.44	86.14±0.81
IAAT	98.57±0.39	92.54±0.67	85.17±1.04	83.64±0.79	79.13±0.64
MMA	98.85±0.63	92.87±1.07	81.87±0.88	75.33±1.19	70.32±0.94
FAT	97.08±0.59	94.11±0.94	80.47±0.86	78.92±0.77	66.53±0.65
TRADES	98.78±0.47	97.07±0.52	90.44±0.66	86.74±0.67	88.09±0.46
SAAT-kernel$_h$	98.47±0.33	78.92±0.72	67.76±0.85	62.45±0.46	52.79±0.46
SAAT-kernel$_c$	98.61±0.29	80.55±0.68	62.86±0.55	63.39±0.74	53.64±0.67
SAAT-minimax$_h$	98.44±0.67	**97.42±0.82**	**94.80±0.76**	**94.42±0.59**	**93.35±0.46**
SAAT-minimax$_c$	**98.88±0.27**	96.29±0.54	91.97±0.69	92.27±0.74	90.03±0.57

Table 3: Test accuracy (%) of various defense methods trained on CIFAR10 with l_∞-norm constrained perturbations on DNNs. (The results of Natural on clean data are just baselines for reference.)

Method	Clean	FGSM	10-PGD	CW	AutoAttack
Natural	*92.26±0.45*	5.79±0.36	0.00±0.00	0.00±0.00	0.00±0.00
Standard	75.65±0.37	62.73±1.09	48.51±0.64	40.37±0.84	42.44±0.63
IAAT	74.56±0.52	49.88±0.82	38.02±0.69	36.79±0.74	30.66±0.79
MMA	79.15±0.67	62.83±0.74	45.73±0.82	44.34±0.96	34.28±0.68
FAT	86.80±0.45	61.49±0.62	46.17±0.77	45.14±0.83	33.34±0.87
TRADES	86.74±1.27	63.42±0.64	49.76±0.65	45.39±0.76	49.55±0.72
SAAT-kernel$_h$	83.89±0.37	24.97±0.79	18.54±0.82	16.33±0.87	11.95±0.58
SAAT-kernel$_c$	83.06±0.73	27.84±0.59	20.77±0.47	18.75±0.64	14.33±0.71
SAAT-minimax$_h$	85.29±0.68	61.52±0.85	**53.86±1.37**	47.00±0.96	49.84±0.33
SAAT-minimax$_c$	**86.98±0.52**	**63.73**±0.72	51.70±0.66	**49.37±0.84**	**50.68±0.54**

Table 4: Test accuracy (%) of various defense methods trained on CIFAR100 with l_∞-norm constrained perturbations on DNNs. (The results of Natural on clean data are just baselines for reference.)

Method	Clean	FGSM	10-PGD	CW	AutoAttack
Natural	*77.79±0.47*	1.16±0.21	0.00±0.00	0.00±0.00	0.00±0.00
Standard	58.13±0.36	35.99±0.44	27.20±0.72	23.64±0.77	22.63±0.94
IAAT	59.47±0.37	27.23±0.65	18.58±0.59	17.79±0.34	13.73±0.88
MMA	49.03±0.77	27.79±0.74	20.50±0.67	18.96±0.51	15.39±0.74
FAT	61.28±0.88	25.03±0.72	19.73±0.56	19.92±0.81	13.09±0.69
TRADES	50.71±0.67	28.99±0.84	21.59±0.65	17.41±0.57	16.47±0.77
SAAT-kernel$_h$	66.88±0.73	13.64±0.81	8.77±0.61	7.65±0.84	7.98±0.72
SAAT-kernel$_c$	68.56±0.53	12.71±0.66	6.79±0.75	6.87±0.52	5.34±0.69
SAAT-minimax$_h$	**70.72±0.47**	**47.18±0.33**	**39.68±0.51**	**34.70±0.36**	**32.77±0.57**
SAAT-minimax$_c$	68.11±0.57	43.80±0.44	35.33±0.39	31.69±0.67	30.83±0.62

5.1.2 ATTACK SETTINGS:

Four popular attack methods are used in the experiments: FGSM (Goodfellow et al., 2014), 10-PGD (PGD with 10 steps) (Madry et al., 2017), CW (Carlini & Wagner, 2017) and AutoAttack (**?**). All the attacks can be performed with both l_2 and l_∞ versions. In the l_∞ version, for FGSM and 10-PGD, the perturbation radius is set as $\epsilon_{test} = 0.3$ for MNIST and $\epsilon_{test} = 8/255$ for CIFAR10 and CIFAR100, the step size for 10-PGD is $\epsilon_{test}/4$, which is a standard setting for adversarial attack (Madry et al., 2017; Ding et al., 2018).

5.1.3 IMPLEMENTATION DETAILS:

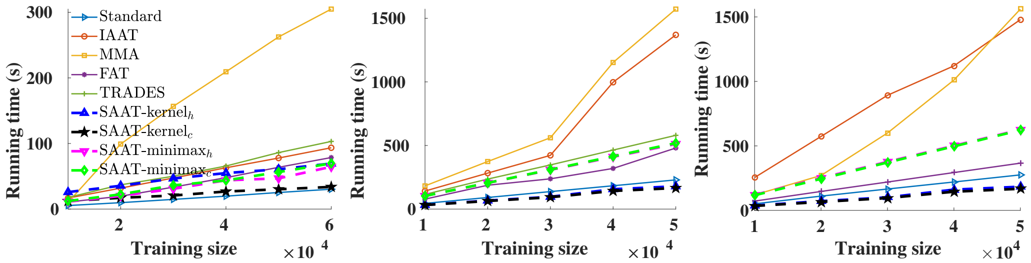
Under l_∞-norm constrained perturbations, we set $\epsilon_{\max} = 0.3$ for MNIST, $\epsilon_{\max} = 8/255$ for CIFAR10 and Tiny Imagenet, and the step size is set as $\epsilon_{\max}/4$. For all algorithms, we set the batch size as 100 with 10 epochs. We use 5-fold cross validation to choose the optimal learning rate $\gamma \in 2^{[-3,3]}$.

We use the PreAct ResNet18 architecture for CIFAR10 and CIFAR100 and use two convolutional networks with 16 and 32 convolutional filters followed by a fully connected layer of 100 units for MNIST, which are the same model structures provided by Wong et al. (2020). For all the compared algorithms, we use the cross entropy loss function. For SAAT-minimax and SAAT-kernel, b_i is gotten by $\hat{l}(x_i, y_i, f_w, \epsilon_{\max}) - k_i\epsilon_{\max}$, we set $k_i = 2$, $\eta = 0.05$, linearly increase regularization parameter λ from 1 to 3 on MNIST, set $k_i = 0.15$, $\eta = 0.3/255$, linearly increase λ from 0 to 0.5 on CIFAR10, and set $k_i = 0.3$, $\eta = 0.3/255$, linearly increase λ from 0 to 0.6 on CIFAR100.

5.2 EXPERIMENTAL RESULTS AND ANALYSES

Robustness against various attacks. We first explore robustness of adversarial training algorithms against different attacks in Tables 2, 3, 4. It can be seen clearly that our SAAT-minimax not only improves natural generalization performance, but also enjoys stronger defensive ability against various adversarial examples. Moreover, it indicates that hinge loss contributes to higher adversarial robustness than cross entropy. Although SAAT-kernel is not as robustness as adversarial training algorithms on the minimax problem, it largely improves accuracy on clean data. As for the compared algorithms, although they improve the generalization performance on clean data to some extent, they sacrifice much robustness on strong attacks, especially CW and AutoAttack.

Running time with different sizes of training samples. Fig. 3 shows the running time of various adversarial training algorithms when training samples of different sizes. We can find that SAAT-kernel is much more efficient due to its one-layer objective function. For other algorithms on the minimax problem, the time-consuming factor lies on the K-step PGD attack. Among adversarial training algorithms with adaptive ϵ_i, SAAT-minimax is superior to others since it avoids brute-force search for the optimal ϵ_i^*. We also note that the time of SAAT-minimax costs a little longer than that of FAT since FAT applies the early-stopped PGD strategy. But the extra time can be ignored compared with the superiority we have in robustness and generalization.



(a) MNIST (b) CIFAR10 (c) CIFAR100

Figure 3: Running time of adversarial training algorithms against different sizes of training samples.

6 CONCLUSION

To achieve a better trade-off between robustness and accuracy without much computation overhead, in this paper, we propose an adversarial training framework with self-adaptive perturbation radii named SAAT. This framework can also get the closed-form solution of the optimal perturbation radii and avoids tedious searching compared with existing works, which is applicable to both DNNs and kernel SVMs. Comprehensive experimental results verify that our algorithms not only improve adversarial robustness and natural generalization, but also can be competitive with other adversarial training algorithms in terms of running time.

REFERENCES

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

Alberto Bietti, Grégoire Mialon, Dexiong Chen, and Julien Mairal. A kernel perspective for regularizing deep neural networks. In *International Conference on Machine Learning*, pp. 664–674. PMLR, 2019.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *International Conference on Machine Learning*, pp. 1467–1474, 2012.

Battista Biggio, Igino Corona, Blaine Nelson, Benjamin IP Rubinstein, Davide Maiorca, Giorgio Fumera, Giorgio Giacinto, and Fabio Roli. Security evaluation of support vector machines in adversarial environments. In *Support Vector Machines Applications*, pp. 105–153. Springer, 2014.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

Sébastien Bubeck, Yin Tat Lee, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. In *International Conference on Machine Learning*, pp. 831–840. PMLR, 2019.

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

Minhao Cheng, Qi Lei, Pin-Yu Chen, Inderjit Dhillon, and Cho-Jui Hsieh. Cat: Customized adversarial training for improved robustness. *arXiv preprint arXiv:2002.06789*, 2020.

Bo Dai, Bo Xie, Niao He, Yingyu Liang, Anant Raj, Maria-Florina F Balcan, and Le Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pp. 3041–3049, 2014.

Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. *arXiv preprint arXiv:1812.02637*, 2018.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *International Conference on Learning Representations*, 2014.

Hal Daumé Iii. From zero to reproducing kernel hilbert spaces in twelve pages or less. http://legacydirs.umiacs.umd.edu/ hal/docs/daume04rkhs.pdf, 2004.

Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 2694–2700, 2015.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

Alexey Kurakin, Ian Goodfellow, Samy Bengio, and David Wagner. Adversarial machine learning at scale. *International Conference on Learning Representations*, 2017.

Y Lecun and L Bottou. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: isoperimetry and processes*, volume 23. Springer Science & Business Media, 1991.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *International Conference on Learning Representations*, 2017.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1979–1993, 2018.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

Nicolas Papernot, Patrick Mcdaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. *Computer and Communications Security*, pp. 506–519, 2017.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pp. 1177–1184, 2008.

L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and Aleksander Mdry. Adversarially robust generalization requires more data. *Neural Information Processing Systems*, 2018.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2018.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.

Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *International Conference on Machine Learning*, 2019.

Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

Huimin Wu, Zhengmian Hu, and Bin Gu. Fast and scalable adversarial training of kernel svm via doubly stochastic gradients. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12): 10329–10337, May 2021.

Abraham J Wyner. On boosting and the exponential loss. In *International Workshop on Artificial Intelligence and Statistics*, pp. 323–329. PMLR, 2003.

Dao-Hong Xiang. Logistic classification with varying gaussians. *Computers & Mathematics with Applications*, 61(2):397–407, 2011.

Han Xiao, Huang Xiao, and Claudia Eckert. Adversarial label flips attack on support vector machines. *European Conference on Artificial Intelligence*, pp. 870–875, 2012.

Jiancong Xiao, Yanbo Fan, Ruoyu Sun, and Zhi-Quan Luo. Adversarial rademacher complexity of deep neural networks. 2021.

Cihang Xie, Mingxing Tan, Boqing Gong, Alan Yuille, and Quoc V Le. Smooth adversarial training. *arXiv preprint arXiv:2006.14536*, 2020.

Huan Xu, Constantine Caramanis, and Shie Mannor. Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, 10:1485–1510, 2009.

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

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

Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Bowei Xi. Adversarial support vector machine learning. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pp. 1059–1067, 2012.

A PROOF OF THEOREM 5

For our upper bound of the adversarial risk, *i.e.*, Theorem 5, we firstly give a brief proof process.

Step 1: For each sample (x, y), we define the optimal adaptive perturbation radius ϵ_i^* for each sample as:

$$\epsilon_i^* = \begin{cases} \epsilon_{\max}, & \text{if } \forall \delta_i \in \mathcal{B}(\delta_i, \epsilon_{\max}),\ y_i f(x_i + \delta_i) > 0, \\ \arg\min\limits_{\epsilon_i \leq \epsilon_{\max}} \epsilon_i,\ s.t.\ \exists \delta_i \in \mathcal{B}(\delta_i, \epsilon_i),\ y_i f(x_i + \delta_i) \leq 0, & \text{otherwise.} \end{cases} \tag{12}$$

which is the minimum perturbation radius that guarantees at least one adversarial sample misleads the model. Based on the optimal adaptive perturbation radius ϵ_i^*, we further use the Rademacher complexity, *i.e.*, Definition 10, to bound the adversarial risk R_{rob}, *i.e.*, Lemma 14:

$$R_{rob} \leq \frac{1}{n} \sum_{i=1}^{n} \hat{l}(x_i, y_i, f, \epsilon_i^*) + 2B\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*) + 3B\sqrt{\frac{\log 2/\omega}{2n}}, \tag{13}$$

where $\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*)$ is the empirical Rademacher complexity of the adversarial loss class $\hat{l}_{\mathcal{F}}^* = \{\hat{l}(x, y, f, \epsilon_i^*) : f \in \mathcal{F}\}$ with respect to the sample set $x, y \in \mathcal{D}$. Based on the above bound, the following two steps separately discuss the two terms: $\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*)$ and $\hat{l}(x_i, y_i, f, \epsilon_i^*)$.

Step 2: For the empirical Rademacher complexity $\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*)$, we use the covering number, *i.e.*, Definition 15, to get its bound, *i.e.*, Lemma 20:

$$\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*) \leq \frac{12}{\sqrt{n}} L_\phi L_\rho^{D-1} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(X_p + \epsilon_{max}) \sqrt{\log \prod_{i=1}^{D} \frac{\pi^{H_i H_{i-1}/2}}{\Gamma(\frac{H_i H_{i-1}}{2} + 1)} M_i^{H_i H_{i-1}}} \prod_{i=1}^{D} M_i, \tag{14}$$

where $X_p = \max\{||x_i||_p\}_{i=1}^n$ and Γ means the gamma function.

Combing Step 2 with Step 1, we can then get Theorem 5.

The following is the completed proof process.

Step 1: Rademacher complexity (Mohri et al., 2018) is one of the classic measures of generalization error. Here, we present its formal definition.

Definition 12. (Rademacher Complexity) (Mohri et al., 2018) For any function class $\mathcal{H} : \mathcal{Z} \to \mathbb{R}$, given a sample set $T = \{z_1, \cdots, z_n\}, z \in \mathcal{Z}$, the empirical Rademacher complexity of $\mathcal{H}$ with respect to T is defined as:

$$\mathfrak{R}_T(\mathcal{H}) = \frac{1}{n} \mathbb{E}_\sigma [\sup_{h \in \mathcal{H}} \sum_{i=1}^{n} \sigma_i h(z_i)], \tag{15}$$

where $\sigma_1, \cdots, \sigma_n$ are i.i.d. Rademacher random variables with $\mathbb{P}(\sigma_i = 1) = \mathbb{P}(\sigma_i = -1) = \frac{1}{2}$.

We then have the following Theorem 13 which connects the empirical and expected risks via Rademacher complexity.

Theorem 13. *(Mohri et al., 2018) Let $\mathcal{H}$ be a function class mapping from $\mathcal{Z}$ to $[0, B]$ and $T = \{z_1, \cdots, z_n\}$, $z \in \mathcal{Z}$ be an i.i.d. sample set drawn from the distribution P. Then, for any $\omega \in (0, 1)$, with probability at least $1 - \omega$, the following holds for all $h \in \mathcal{H}$:*

$$\mathbb{E}_{z \sim P}[h(z)] \leq \frac{1}{n} \sum_{z_i \in T} h(z_i) + 2B\mathfrak{R}_T(\mathcal{H}) + 3B\sqrt{\frac{\log 2/\omega}{2n}}. \tag{16}$$

Based on the above Theorem 13, we bound the adversarial risk R_{rob} in the following Lemma 14.

Lemma 14. *When Assumption 4 holds, given the sample set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, for any $\omega \in (0, 1)$ and any $f \in \mathcal{F}$, with probability at least $1 - \omega$, the following holds:*

$$R_{rob} \leq \frac{1}{n} \sum_{i=1}^{n} \hat{l}(x_i, y_i, f, \epsilon_i^*) + 2B\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*) + 3B\sqrt{\frac{\log 2/\omega}{2n}}, \tag{17}$$

where $\hat{l}_{\mathcal{F}}^* = \{\hat{l}(x, y, f, z, \epsilon_i^*) : f \in \mathcal{F}\}$.

Proof. For each sample (x_i, y_i), we introduce the optimal perturbation radius ϵ_i^*:

$$\epsilon_i^* = \begin{cases} \epsilon_{\max}, & \text{if } \forall \delta_i \in \mathcal{B}(\delta_i, \epsilon_{\max}),\ y_i f(x_i + \delta_i) > 0, \\ \arg\min\limits_{\epsilon_i \leq \epsilon_{\max}} \epsilon_i,\ s.t.\ \exists \delta_i \in \mathcal{B}(\delta_i, \epsilon_i),\ y_i f(x_i + \delta_i) \leq 0, & \text{otherwise.} \end{cases} \tag{18}$$

Then, we have:

$$\begin{aligned} & \mathbb{I}\{\exists \delta \in \mathcal{B}(\delta_i, \epsilon_{max}) : y_i f(x_i + \delta_i) \leq 0\} \\ =& \mathbb{I}\{\exists \delta \in \mathcal{B}(\delta_i, \epsilon_i^*) : y_i f(x_i + \delta_i) \leq 0\} \\ \leq & \max_{\delta_i \in \mathcal{B}(\delta_i, \epsilon_i^*)} l_f(x_i + \delta_i, y_i) \end{aligned} \tag{19}$$

where the last inequality is because of Assumption 4. Finally, take expectation for the above inequality, we obtain:

$$\begin{aligned} R_{rob} &\leq \mathbb{E}_{(x,y)\sim\mathcal{D}} \max_{\delta \in \mathcal{B}(\delta, \epsilon^*)} l_f(x + \delta, y) \\ &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\ \hat{l}_f(x, y, \epsilon^*) \end{aligned} \tag{20}$$

Considering $l_f \in [0, B]$ and Theorem 13, we conclude that for any $\omega \in (0, 1)$ and any $f \in \mathcal{F}$, with probability at least $1 - \omega$, the following holds:

$$\begin{aligned} R_{rob} &\leq \mathbb{E}_{(x,y)\sim\mathcal{D}}\ \hat{l}_f(x, y, \epsilon^*) \\ &\leq \frac{1}{n}\sum_{i=1}^{n} \hat{l}(x_i, y_i, f, \epsilon_i^*) + 2B\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*) + 3B\sqrt{\frac{\log 2/\omega}{2n}}. \end{aligned} \tag{21}$$

□

Based on the above bound, the following step discusses the term $\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*)$.

Step 2: For the empirical Rademacher complexity $\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*)$, we introduce the covering number (Wainwright, 2019) to get its bound.

Definition 15. (μ-Covering Number) (Wainwright, 2019) Let $\mu > 0$ and $(\mathcal{H}, l(\cdot,\cdot))$ be a metric space, where $l(h, h'), h, h' \in \mathcal{H}$ is a (pseudo)-metric. $\widetilde{\mathcal{H}} \subset \mathcal{H}$ is the μ-cover of $\mathcal{H}$, if for any $h \in \mathcal{H}$, there exists $\widetilde{h} \in \widetilde{\mathcal{H}}$ *s.t.* $l(\widetilde{h}, h) \leq \mu$. Define the smallest cardinality $|\widetilde{\mathcal{H}}|$ as the μ-covering number of $\mathcal{H}$, and denote the μ-covering number as $\mathcal{N}(\mathcal{H}, l(\cdot,\cdot), \mu)$.

Then, we introduce Theorem 16, which provides the upper bound of the empirical Rademacher complexity with covering number.

Theorem 16. *(Wainwright, 2019) Given the sample dataset* $T = \{z_1, \cdots, z_n\}, z \in \mathcal{Z}$, *the pseudo-norm of the function class* $\mathcal{H} : \mathcal{Z} \to \mathbb{R}$ *is defined as:* $\forall h \in \mathcal{H}, ||h||_T = \left(\frac{1}{n}\sum_{i=1}^{n} |h(z_i)|^2\right)^{\frac{1}{2}}$, *and the pseudo-metric of* $\mathcal{H}$ *is* $\forall h, \widetilde{h} \in \mathcal{H}, ||h - \widetilde{h}||_T = \left(\frac{1}{n}\sum_{i=1}^{n} |h(z_i) - \widetilde{h}(z_i)|^2\right)^{\frac{1}{2}}$. *Then, the empirical Rademacher complexity* $\mathfrak{R}_T(\mathcal{H})$ *satisfies*

$$\mathfrak{R}_T(\mathcal{H}) \leq \frac{12}{\sqrt{n}} \int_0^{\max_{h\in\mathcal{H}} ||h||_T} \sqrt{\log \mathcal{N}(\mathcal{H}, ||\cdot||_T, \mu)}\, d\mu, \tag{22}$$

where $\mathcal{N}(\mathcal{H}, ||\cdot||_T, \mu)$ *means the* μ*-covering number of* $\mathcal{H}$*.*

In the following, we introduce some necessary theorems first (i.e., Theorem 17-19), then provide the bound of the empirical Rademacher complexity $\mathcal{R}_T(\hat{l}_{\mathcal{F}}^*)$, *i.e.*, Lemma 20.

Theorem 17. *(Ledoux & Talagrand, 1991) Let* $\phi : \mathbb{R} \to \mathbb{R}$ *be a* L_ϕ*-Lipschitz function, then*

$$\mathcal{R}_T(\phi \circ \mathcal{H}) \leq L_\phi \mathcal{R}_T(\mathcal{H}) \tag{23}$$

where $\mathcal{R}_T(\mathcal{H})$ *is the empirical Rademacher complexity of the function class* $\mathcal{H} : \mathcal{Z} \to \mathbb{R}$ *with respect to the sample set* $\mathcal{D} = \{(x_i, y_i), \cdots, (x_n, y_n)\}$ *and* $\phi \circ \mathcal{H} = \{\phi(h(x_i, y_i)) : h \in \mathcal{H}\}$.

Theorem 18. *(Xiao et al., 2021) When* $\delta \in \mathcal{B}(\delta, \epsilon)$ *and* $x \in \mathcal{R}^d$*, we have:*

$$||x + \delta||_2 \leq \max\{1, d^{\frac{1}{2} - \frac{1}{p}}\}(||x||_p + \epsilon). \tag{24}$$

Theorem 19. *(Xiao et al., 2021) Let W be a $m \times n$ matrix and x be a n-dimension vector, we have*

$$||W \cdot x||_2 \le ||W||_F ||x||_2. \tag{25}$$

Here, we provide the bound of the empirical Rademacher complexity $\mathcal{R}_T(\hat{l}^*_{\mathcal{F}})$.

Lemma 20. *When Assumption 4 holds, given the sample dataset $T = \{z_1, \cdots, z_n\}, z \in \mathcal{Z}$, we have:*

$$\mathcal{R}_T(\hat{l}^*_{\mathcal{F}}) \le \frac{12}{\sqrt{n}} L_\phi L_\rho^{D-1} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(X_p + \epsilon_{max}) \sqrt{\log \prod_{i=1}^{D} \frac{\pi^{H_i H_{i-1}/2}}{\Gamma(\frac{H_i H_{i-1}}{2} + 1)} M_i^{H_i H_{i-1}} \prod_{i=1}^{D} M_i}, \tag{26}$$

where $X_p = \max\{||x_i||_p\}_{i=1}^n$ and Γ means the gamma function.

Proof. First of all, according to Assumption 4, we rewrite $\hat{l}^*_f$ as:

$$\begin{aligned} \hat{l}^*_f(x, y, \epsilon) &= \max_{\delta \in \mathcal{B}(\delta, \epsilon^*)} l_f(x + \delta, y) \\ &= \phi(\inf_{\delta \in \mathcal{B}(\delta, \epsilon^*)} y f(x + \delta)) := \phi(\tilde{l}^*_f(x, y)), \end{aligned} \tag{27}$$

where $\phi(\cdot)$ is a non-increasing function and is L_ϕ-Lipschitz. Then, according to Theorem 17, we have:

$$\mathcal{R}_T(\hat{l}^*_{\mathcal{F}}) = \mathcal{R}_T(\phi(\tilde{l}^*_{\mathcal{F}}(x, y))) \le L_\phi \mathcal{R}_T(\tilde{l}^*_{\mathcal{F}}) \tag{28}$$

where $\tilde{l}^*_{\mathcal{F}} = \{\tilde{l}^*_f : f \in \mathcal{F}\}$. In this case, we first find the upper bound of $\mathcal{R}_T(\tilde{l}^*_{\mathcal{F}})$. Immediately, according to Theorem 16, we have:

$$\mathfrak{R}_T(\tilde{l}^*_{\mathcal{F}}) \le \frac{12}{\sqrt{n}} \int_0^{\max_{\tilde{l}^*_f \in \tilde{l}^*_{\mathcal{F}}} ||\tilde{l}^*_f||_T} \sqrt{\log \mathcal{N}(\tilde{l}^*_{\mathcal{F}}, ||\cdot||_T, \mu)} d\mu. \tag{29}$$

Based on this, we will separately discuss the two terms: $\max_{\tilde{l}^*_f \in \tilde{l}^*_{\mathcal{F}}} ||\tilde{l}^*_f||_T$ and $\mathcal{N}(\tilde{l}^*_{\mathcal{F}}, ||\cdot||_T, \mu)$.

For the term $\max_{\tilde{l}^*_f \in \tilde{l}^*_{\mathcal{F}}} ||\tilde{l}^*_f||_T$, we first let $x^* = x + \delta^*, \delta^* = arginf_{\delta \in \mathcal{B}(\delta, \epsilon^*)} y f(x + \delta)$ and x^i be the output of x^* pass through the first to the $(i-1)$-th layer of the neural network, then we have:

$$\begin{aligned} |\tilde{l}^*_f(x, y)| &= |\inf_{\delta \in \mathcal{B}(\delta, \epsilon^*)} y f(x + \delta)| \\ &= |W_D \rho(W_{D-1} x^{D-1})| \\ &\le ||W_D||_F \cdot ||\rho(W_{D-1} x^{D-1})||_2 \\ &\le M_D ||\rho(W_{D-1} x^{D-1}) - \rho(0)||_2 \\ &\le L_\rho M_D ||W_{D-1} x^{D-1}||_2 \\ &\le \cdots \\ &\le L_\rho^{D-1} \prod_{i=1}^{D} M_i \cdot ||x^*||_2 \\ &\le L_\rho^{D-1} \prod_{i=1}^{D} M_i \cdot \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(||x||_p + \epsilon^*) \end{aligned} \tag{30}$$

where the first inequality is due to Theorem 19, the second inequality is because of $||W_i||_F \le M_i, i \in [D]$, the third inequality is because $\rho(\cdot)$ is L_ρ-Lipschitz and the last inequality is due to Theorem 18. Therefore, we have

$$\max_{\tilde{l}^*_f \in \tilde{l}^*_{\mathcal{F}}} ||\tilde{l}^*_f||_T = \max_{\tilde{l}^*_f \in \tilde{l}^*_{\mathcal{F}}} \big(\frac{1}{n} \sum_{i=1}^{n} |\tilde{l}^*_f(x_i, y_i)|^2\big)^{\frac{1}{2}} \le L_\rho^{D-1} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(X_p + \epsilon_{max}) \prod_{i=1}^{D} M_i, \tag{31}$$

where $X_p = \max\{||x_i||_p\}_{i=1}^{n}$.

For the term $\mathcal{N}(\tilde{l}^*_{\mathcal{F}}, ||\cdot||_T, \mu)$, according to Definition 15, we have:

$$\mathcal{N}(\tilde{l}^*_{\mathcal{F}}, ||\cdot||_T, \mu) \leq |\tilde{l}^*_{\mathcal{F}}| = |\mathcal{F}| = \prod_{i=1}^{D} |W_i|, \tag{32}$$

where the last equality is because of the definition of $\mathcal{F}$, *i.e.*, Eq. (1). Further more, for a matrix W with the shape of $h_1 \times h_2$, we have:

$$||W||_F = \sqrt{\sum_{a=1}^{h_1}\sum_{b=1}^{h_2} W_{ab}^2} \leq M \overset{(a)}{\Longrightarrow} |W| = \frac{\pi^{h_1 h_2/2}}{\Gamma(\frac{h_1 h_2}{2}+1)} M^{h_1 h_2}, \tag{33}$$

where the arrow (a) follows from the volume formula of $h_1 h_2$-dimensional sphere and Γ means the gamma function. Then, we can obtain:

$$\mathcal{N}(\tilde{l}^*_{\mathcal{F}}, ||\cdot||_T, \mu) \leq \prod_{i=1}^{D} |W_i| = \prod_{i=1}^{D} \frac{\pi^{H_i H_{i-1}/2}}{\Gamma(\frac{H_i H_{i-1}}{2}+1)} M_i^{H_i H_{i-1}}. \tag{34}$$

Finally, combined with Eqs. (31) and (34), we have

$$\begin{aligned}\mathfrak{R}_T(\tilde{l}^*_{\mathcal{F}}) &\leq \frac{12}{\sqrt{n}} \int_0^{\max_{\tilde{l}^*_f \in \tilde{l}^*_{\mathcal{F}}} ||\tilde{l}^*_f||_T} \sqrt{\log \mathcal{N}(\tilde{l}^*_{\mathcal{F}}, ||\cdot||_T, \mu)}\, d\mu \qquad (35)\\ &\leq \frac{12}{\sqrt{n}} L_\rho^{D-1} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(X_p + \epsilon_{max}) \sqrt{\log \prod_{i=1}^{D} \frac{\pi^{H_i H_{i-1}/2}}{\Gamma(\frac{H_i H_{i-1}}{2}+1)} M_i^{H_i H_{i-1}} \prod_{i=1}^{D} M_i}.\end{aligned}$$

Then, according to the relationship between $\mathfrak{R}_T(\tilde{l}^*_{\mathcal{F}})$ and $\mathfrak{R}_T(\hat{l}^*_{\mathcal{F}})$, *i.e.*, Eq. (28), we conclude:

$$\begin{aligned}&\mathcal{R}_T(\hat{l}^*_{\mathcal{F}}) \leq L_\phi \mathcal{R}_T(\tilde{l}^*_{\mathcal{F}}) \qquad (36)\\ \leq &\frac{12}{\sqrt{n}} L_\phi L_\rho^{D-1} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(X_p + \epsilon_{max}) \sqrt{\log \prod_{i=1}^{D} \frac{\pi^{H_i H_{i-1}/2}}{\Gamma(\frac{H_i H_{i-1}}{2}+1)} M_i^{H_i H_{i-1}} \prod_{i=1}^{D} M_i}.\end{aligned}$$

□

Then combing Eq.(36) with Eq. (21), we can get

$$\begin{aligned}R_{rob} \leq &\frac{1}{n}\sum_{i=1}^{n} \hat{l}\,(x_i, y_i, f, \epsilon_i^*) + 3B\sqrt{\frac{\log 2/\omega}{2n}} \qquad (37)\\ &+ \frac{24B}{\sqrt{n}} L_\phi L_\rho^{D-1} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(X_p + \epsilon_{\max}) \sqrt{\log \prod_{i=1}^{D} \frac{\pi^{H_i H_{i-1}/2}}{\Gamma(\frac{H_i H_{i-1}}{2}+1)} M_i^{H_i H_{i-1}} \prod_{i=1}^{D} M_i}.\end{aligned}$$

Thus we obtain Theorem 5.

B PROOF OF THEOREM 6

Recall that we denote $\hat{l}(x_i, y_i, f, \epsilon)$ as $\max_{\|\delta_i\|_2 \leq \epsilon} l(y_i, f(x_i + \delta_i))$.

We set $\epsilon_1 < \epsilon_2$, L_1, L_2 are the sets of all possible values of $l(y_i, f(x_i + \delta_i))$, where $L_1 = \{l(y_i, f(x_i + \delta_i)) \mid \|\delta_i\|_p \leq \epsilon_1\}$, $L_2 = \{l(y_i, f(x_i + \delta_i)) \mid \|\delta_i\|_p \leq \epsilon_2\}$.

It is obvious that the constraint range of L_1 is smaller L_2, thus we can naturally get that $L_1 \subseteq L_2$. In this case, we can come to the conclusion that $\max_{\|\delta_i\|_p \leq \epsilon_1} l(y_i, f(x_i + \delta_i))$ is no larger than $\max_{\|\delta_i\|_p \leq \epsilon_2} l(y_i, f(x_i + \delta_i))$.

Thus we complete the proof.

C PROOF OF THEOREM 10

If Assumption 9 holds, for the minimization problem

$$\min_{\epsilon_i \in [0, \epsilon_{max}]} \hat{l}(x_i, y_i, f_w, \epsilon_i) - \lambda \epsilon_i, \tag{38}$$

1. If $b_i \geq 0$, we have $\hat{l}(x_i, y_i, f_w, \epsilon_i) = k_i \epsilon_i + b_i$,
   then problem (38) becomes $\min_{\epsilon_i \in [0, \epsilon_{max}]} (k_i - \lambda)\epsilon_i + b_i$, thus the optimal ϵ_i^* is
   $$\epsilon_i^* = \begin{cases} 0, & \text{if } k_i \geq \lambda; \\ \epsilon_{max}, & \text{otherwise.} \end{cases} \tag{39}$$
2. If $b_i < 0$, we have $\hat{l}(x_i, y_i, f_w, \epsilon_i) = \max(0, k_i \epsilon_i + b_i)$,
   - When $\epsilon_i \in \left[0, -\frac{b_i}{k_i}\right]$, problem (38) becomes $\min_{\epsilon_i \in [0, \epsilon_{max}]} -\lambda \epsilon_i$, thus the optimal ϵ_i^* is
     $$\epsilon_i^* = -\frac{b_i}{k_i}. \tag{40}$$
   - When $\epsilon_i \in \left[-\frac{b_i}{k_i}, \epsilon_{\max}\right]$, problem (38) becomes $\min_{\epsilon_i \in [-\frac{b_i}{k_i}, \epsilon_{max}]} (k_i - \lambda)\epsilon_i + b_i$, thus the optimal ϵ_i^* is
     $$\epsilon_i^* = \begin{cases} -\dfrac{b_i}{k_i}, & \text{if } k_i \geq \lambda; \\ \epsilon_{max}, & \text{otherwise.} \end{cases} \tag{41}$$

Combining the conclusions together, we can get the theorem.

D PROOF OF THEOREM 11

Let $\mathcal{T} = \{\delta_\phi^i \mid \|\delta_\phi^i\|_2 \leq \epsilon'\}$. We define $v = l(y_i, f(x_i) + \epsilon' \|f\|_{\mathcal{H}})$. To prove the theorem, we first prove $v \leq \max l(y_i, \langle f,\ \phi(x_i) + \delta_\phi^i \rangle_{\mathcal{H}})$, then prove $\max l(y_i, \langle f,\ \phi(x_i) + \delta_\phi^i \rangle_{\mathcal{H}}) \leq v$. In the following, we give the details to prove these two sub-conclusions.

Step 1: We first prove $v \leq \max l(y_i, \langle f,\ \phi(x_i) + \delta_\phi^i \rangle_{\mathcal{H}})$.

Since $\mathcal{T} = \{\delta_\phi^i \mid \|\delta_\phi^i\|_2 \leq \epsilon'\}$, we define a subset of $\mathcal{T}$ as $\mathcal{T}' = \{y_i \epsilon' \frac{f}{\|f\|_{\mathcal{H}}}\}$. Hence,

$$\begin{aligned} & \max_{\delta_\phi^i \in \mathcal{T}'} l(y_i, \langle f,\ \phi(x_i) + \delta_\phi^i \rangle_{\mathcal{H}}) \\ = & \max_{\delta_\phi^i \in \mathcal{T}'} l(y_i, \langle f, \phi(x_i) \rangle_{\mathcal{H}} + \langle f, \delta_\phi^i \rangle_{\mathcal{H}}) \\ = & l(y_i, f(x_i) + \epsilon' \, \|f\|_{\mathcal{H}}) \end{aligned}$$

Since $\mathcal{T}' \subseteq \mathcal{T}$, the first sub-conclusion can be proved.

Step 2: Next we prove $\max l(y_i, \langle f,\ \phi(x_i) + \delta_\phi^i \rangle_{\mathcal{H}}) \leq v$.

$$\begin{aligned} & \max_{\delta_\phi^i \in \mathcal{T}} l(y_i, \langle f,\ \phi(x_i) + \delta_\phi^i \rangle_{\mathcal{H}}) \\ = & \max_{\delta_\phi^i \in \mathcal{T}} l(y_i, \langle f, \phi(x_i) \rangle_{\mathcal{H}} + \langle f, \delta_\phi^i \rangle_{\mathcal{H}}) \\ \leq & \max_{\delta_\phi^i \in \mathcal{T}} l(y_i, f(x_i) + \|f\|_{\mathcal{H}} \cdot \|\delta_\phi^i\|_{\mathcal{H}}) \\ \leq & l(y_i, f(x_i) + \epsilon' \|f\|_{\mathcal{H}}) \end{aligned}$$

The first inequality is due to the Cauchy-Schwarz inequality. The second inequality holds since $\|\delta_\phi^i\|_2 \leq \epsilon'$. Hence the second sub-conclusion holds.

Step 3: Combining these two steps, we have (42):

$$\max_{\|\delta_\phi^i\|_2 \leq \epsilon'} l(y_i, \langle f,\ \phi(x_i) + \delta_\phi^i \rangle_{\mathcal{H}}) = l(y_i, f(x_i) + \epsilon' \|f\|_{\mathcal{H}}). \tag{42}$$

E OPTIMIZATION PROCEDURE OF SAAT-SVM

Self-adaptive adversarial training on kernel SVMs can be formulated as follows:

$$\begin{aligned} &\min_{f\in\mathcal{H},\epsilon'} \frac{\|f\|_{\mathcal{H}}^2}{2} + \frac{C}{n}\sum_{i=1}^{n} \left\{ l(y_i, f(x_i) + \epsilon_i'\|f\|_{\mathcal{H}}) - \lambda\epsilon_i' \right\}. \\ &s.t.\ \epsilon_i' \in [0, \epsilon_{\max}'],\ \ i = 1, \ldots, n. \end{aligned} \tag{43}$$

In the following, we use the doubly stochastic gradient (DSG) algorithm (Dai et al., 2014), which skillfully combines the stochastic gradient algorithm with random feature approximation algorithm, to update the solution f when $\{\epsilon_i'^*\}_{i=1}^n$ is fixed. The loss function here is the commonly used hinge loss and we express it as $[\cdot]_+$. Thus, the objective function (43) can be rewritten as

$$\begin{aligned} &\min_{f\in\mathcal{H}} R(f) && (44) \\ =&\min_{f\in\mathcal{H}} \frac{1}{2}\|f\|_{\mathcal{H}}^2 + \frac{C}{n}\sum_{i=1}^{n}\left[1 - y_i f(x_i) + \epsilon_i'^*\|f\|_{\mathcal{H}}\right]_+ . \end{aligned}$$

Here we only discuss the case when the hinge loss is greater than 0.

Stochastic Gradient Descent. The gradient of the objective (44) can be written as:

$$\nabla R(f) = f(\cdot) + C\left[-yk(x_i, \cdot) + \epsilon_i'^* \frac{f(\cdot)}{\|f\|_{\mathcal{H}}}\right]. \tag{45}$$

It is noted that $\nabla R(f)$ is the derivative wrt. f.

Random Feature Approximation. Since high computational complexity is still needed for kernel functions, we use the random feature approximation method (Rahimi & Recht, 2008) to approximate the stationary kernels such as RBF, Laplacian and Cauchy kernels by explicitly computing random features $\phi_\omega = \frac{1}{\sqrt{m}}[\phi_{\omega_1}(x), \phi_{\omega_2}(x), \cdots, \phi_{\omega_m}(x)]$, i.e., $k(x, x') \approx \phi_\omega(x)\phi_\omega^T(x')$, where m is the number of random features, $\phi_{\omega_i}(x)$ denotes $[\cos(\omega_i^T x), \sin(\omega_i^T x)]^T$ and ω is drawn from the measure $p(\omega)$. (The detailed measure $p(\omega)$ of the kernels is shown in Table 1 of Dai et al. (2014).) Thus, Eq. (45) can be approximated as follows:

$$\nabla\hat{R}(f) = f(\cdot) + C\left[-y\phi_\omega(x)\phi_\omega(\cdot) + \epsilon_i'^* \frac{f(\cdot)}{\|f\|_{\mathcal{H}}}\right]. \tag{46}$$

Update Rules. Thus we can get the update rule for SAAT-SVM according to the principle of the SGD method.

$$f_{t+1}(\cdot) = f_t(\cdot) - \gamma_t\nabla\hat{R}(f) = \sum_{i=1}^{t} a_t^i\zeta_i(\cdot). \tag{47}$$

where γ_t is the stepsize in the t-th iteration, the initial value $f_1(\cdot) = 0$ and $\zeta_i(\cdot) = -Cy_i\phi_{\omega_i}(x_i)\phi_{\omega_i}(\cdot)$. The value of a_t^i can be easily inferred as $-\gamma_i\prod_{j=i+1}^{t}\left(1 - \gamma_j\left(1 - \frac{\epsilon_i'^* C}{\|f_j\|_2}\right)\right)$.

Following the update rule (47), the training and prediction algorithms of SAAT-SVM are shown in Algorithms 3 and 4.

A crucial step of DSG in Algorithm 3 and 4 is sampling ω_i with seed i. As the seeds are aligned for the training and prediction processes in the same iteration, we only need to save the seeds instead of the whole random features, which is memory friendly.

F EXPERIMENTS

F.1 EXPERIMENTS UNDER l_2-NORM CONSTRAINED PERTURBATIONS

In this section, We present the robustness of different adversarial training algorithms under l_2-norm constrained perturbations on DNNs in Tables 5, 6 and 7. We set $\epsilon_{\max} = 2$ for MNIST, $\epsilon_{\max} = 0.5$ for CIFAR10 and CIFAR100, and the step size is set as $\epsilon_{\max}/4$. As for the attack methods, for FGSM and 10-PGD, the perturbation radius is set as $\epsilon_{test} = 2$ for MNIST and $\epsilon_{test} = 0.5$ for CIFAR10 and CIFAR100, the step size for 10-PGD is $\epsilon_{test}/4$. Obviously, our algorithms can achieve the balance between robustness and accuracy under the l_2-norm constrained perturbations.

Algorithm 3 $\{\alpha_i\}_{i=1}^t = \textbf{Train}(\mathbb{P}(x, y))$

```
Input: $\mathcal{D},\ p(\omega),\ C$.
for $i = 1, \cdots, t$ do
    Sample $(x_i, y_i) \sim \mathcal{D}$.
    Sample $\omega_i \sim p(\omega)$ with seed $i$.
    $f(x_i) = \textbf{Predict}(x_i, \{\alpha_j\}_{j=1}^{i-1})$.
    Define $\gamma_i = \frac{\theta}{i}$ (diminishing stepsize).
    $\alpha_i = \gamma_i C y_i \phi_{\omega_i}(x_i)$.
    $\alpha_j = (1 - \gamma_i(1 + \frac{\epsilon_i'^* C}{\|f_j\|_{\mathcal{H}}}))\alpha_j$ for $j = 1, \cdots, i-1$
end for
```

Algorithm 4 $f(x) = \textbf{Predict}(x, \{\alpha_i\}_{i=1}^t)$

```
Input: $p(\omega), \phi_\omega(x)$.
Set $f(x) = 0$.
for $i = 1, \cdots, t$ do
    Sample $\omega_i \sim p(\omega)$ with seed $i$;
    $f(x) = f(x) + \alpha_i \phi_{\omega_i}(x)$.
end for
```

F.2 SENSITIVITY ANALYSIS AGAINST THE PGD ATTACK

In this part, we explore the defensive ability of minimax-based adversarial training algorithms against l_∞-norm PGD attack in terms of the perturbation radius ϵ_{test} (Fig. 4) and the attack steps K (Fig. 5).

Sensitivity against different perturbation radii ϵ_{test}. We compare adversarial training algorithms against 10-PGD attacks with different ϵ_{test}, i.e., $\epsilon_{test} \in [0.2,\ 0.4]$ for MNIST, $\epsilon_{test} \in [4/255,\ 8/255]$ for CIFAR10 and Tiny Imagenet. Fig. 4 shows their performance on adversarial examples with different strength. We can see clearly that, with the increase of ϵ_{test}, since the generated adversarial examples are more aggressive, the accuracy of the seven adversarial training algorithms decreases in different degrees. But it is obvious that the compared algorithms are more sensitive to large perturbation radii, while our algorithms especially SAAT-minimax with hinge loss can keep stable relatively.

Sensitivity against different PGD steps. We also explore the sensitivity of adversarial training algorithms against K-step PGD attack. Here ϵ_{test} is fixed as 0.3 for MNIST, 8/255 for CIFAR10 and 100. It is obvious that almost all algorithms can keep stable under different steps of PGD. Compared Fig. 4 with Fig. 5, we can see that adversarial examples constructed by enlarging perturbation radius ϵ_{test} of PGD attack is much more aggressive than enlarging steps K.

Table 5: Test accuracy (%) of various defense methods trained on MNIST with l_2-norm constrained perturbations on DNNs. (The results of Natural on clean data are just baselines for reference.)

Method	Clean	FGSM	10-PGD	CW	AutoAttack
Natural	*98.83±0.23*	14.84±0.37	0.03±0.01	0.00±0.00	0.00±0.00
Standard	97.60±0.46	89.61±0.42	73.25±0.36	72.09±0.53	70.88±0.79
IAAT	98.44±0.37	86.78±0.44	67.88±0.74	68.54±0.67	66.59±0.64
MMA	98.97±0.53	88.29±0.46	68.55±0.72	67.86±0.69	67.74±0.83
FAT	98.74±0.49	89.39±0.62	70.66±0.57	69.15±0.82	71.52±0.97
TRADES	99.00±0.47	88.48±0.54	73.87±0.67	72.44±0.47	70.87±0.68
SAAT-kernel$_h$	96.76±0.41	82.56±0.65	62.26±0.74	62.46±0.52	59.46±0.74
SAAT-kernel$_c$	98.54±0.33	80.63±0.59	65.76±0.72	64.95±0.67	60.77±0.88
SAAT-minimax$_h$	**99.10±0.34**	89.47±0.49	71.47±0.53	70.68±0.59	71.56±0.44
SAAT-minimax$_c$	98.77±0.38	**89.99±0.67**	**74.17±0.66**	**73.54±0.52**	**72.39±0.59**

Table 6: Test accuracy (%) of various defense methods on CIFAR10 with l_2-norm constrained perturbations on DNNs. (The results of Natural on clean data are just baselines for reference.)

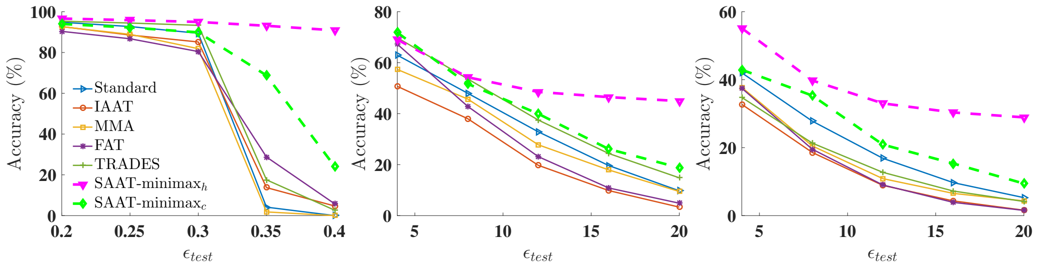
Method	Clean	FGSM	10-PGD	CW	AutoAttack
Natural	*92.26±0.44*	12.36±0.36	1.79±0.21	0.00±0.00	0.00±0.00
Standard	76.65±0.54	55.29±0.37	52.60±0.72	47.61±0.42	48.66±0.84
IAAT	81.58±0.66	55.48±0.49	51.23±0.72	44.37±0.61	43.56±0.59
MMA	83.97±0.55	63.01±0.64	49.77±0.72	46.55±0.59	47.32±0.48
FAT	89.06±0.69	75.62±0.36	73.51±0.85	64.03±0.77	67.33±0.74
TRADES	87.66±0.52	71 83±0.47	69.66±0.75	62.33±0.57	63.49±0.87
SAAT-kernel$_h$	89.74±0.33	40.35±0.82	35.72±0.64	23.38±0.61	22.77±0.72
SAAT-kernel$_c$	86.52±0.35	41.75±0.72	34.88±0.54	21.07±0.73	23.35±0.66
SAAT-minimax$_h$	91.24±0.55	**80.10±0.67**	**79.70±0.71**	**65.37±0.51**	**70.33±0.64**
SAAT-minimax$_c$	**91.56±0.57**	76.20±0.66	72.07±0.49	65.20±0.64	68.77±0.39

Table 7: Test accuracy (%) of various defense methods trained on CIFAR100 with l_2-norm constrained perturbations on DNNs. (The results of Natural on clean data are just baselines for reference.)

Method	Clean	FGSM	10-PGD	CW	AutoAttack
Natural	*77.79±0.46*	6.19±0.37	0.13±0.07	0.00±0.00	0.00±0.00
Standard	64.97±0.69	34.70±0.84	32.47±0.47	31.54±0.62	30.04±0.88
IAAT	59.56±0.77	30.72±0.61	25.77±0.87	23.32±0.75	20.76±0.66
MMA	61.70±0.59	29.56±0.64	18.63±0.72	19.33±0.84	16.32±0.75
FAT	65.51±0.33	**46.14±0.61**	35.16±0.73	35.41±0.67	33.19±0.49
TRADES	54.50±0.55	36.51±0.74	35.50±0.51	33.20±0.79	32.14±0.63
SAAT-kernel$_h$	68.95±0.76	17.36±0.54	10.23±0.88	8.52±0.74	9.36±0.57
SAAT-kernel$_c$	**72.53±0.75**	20.66±0.35	13.57±0.81	10.33±0.94	9.84±0.77
SAAT-minimax$_h$	67.58±0.67	38.07±0.54	**37.57±0.56**	35.32±0.49	**34.65±0.64**
SAAT-minimax$_c$	70.73±0.47	40.49±0.47	35.55±0.62	**36.16±0.71**	33.65±0.66

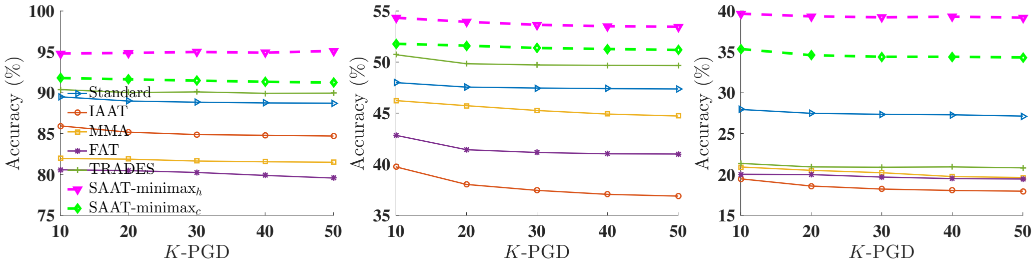
F.3 VERIFICATION OF ASSUMPTION 6

We verify that the maximum loss function $\hat{l}(x_i, y_i, f_w, \epsilon_i)$, i.e., $\max_{\|\delta_i\|_p \leq \epsilon_i} l(y_i, f_w(x_i + \delta_i))$, is piecewise approximately linear with regard to perturbation radii ϵ_i whether in the form of cross entropy or hinge loss in Figs. 6 and 7. Here we randomly choose 50 samples from three datasets respectively. For perturbation under l_2-norm, we perform 10-PGD attack with ϵ_i =0, 1, 2, 3 on MNIST, ϵ_i =0, 0.5, 1, 1.5 on CIFAR10 and CIFAR100. For perturbation under l_∞-norm, we perform 10-PGD attack with ϵ_i =0, 0.1, 0.2, 0.3 on MNIST, ϵ_i =0, 4/255, 8/255, 12/255 on CIFAR10 and Tiny Imagenet. We can see clearly that Figs. 6 and 7 are approximately consistent with the sketch map we present in Fig. 2 in the main body of this paper, which reveals that our assumption of the piecewise approximate linearity is reasonable.



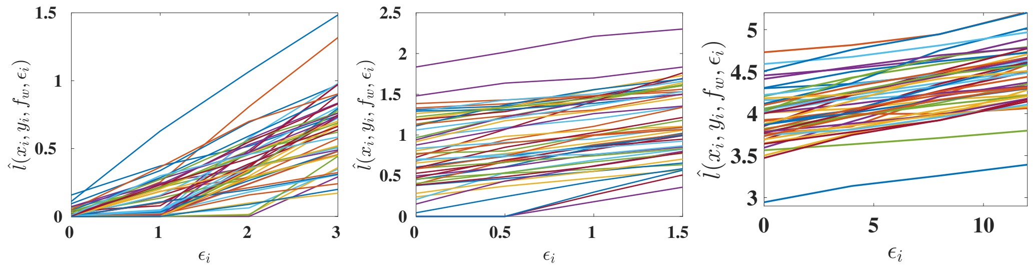
(a) MNIST (b) CIFAR10 (c) CIFAR100

Figure 4: Comparisons on PGD test accuracy with different perturbation radii ϵ_{test} on three datasets.

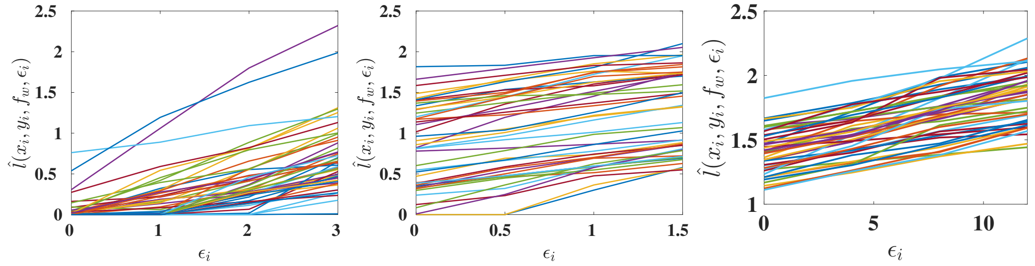


(a) MNIST (b) CIFAR10 (c) CIFAR100

Figure 5: Comparisons on PGD test accuracy with different steps K on three datasets.



(a) MNIST (cross entropy) (b) CIFAR10 (cross entropy) (c) CIFAR100 (cross entropy)



(d) MNIST (hinge) (e) CIFAR10 (hinge) (f) CIFAR100 (hinge)

Figure 6: Maximum loss value $\hat{l}(x_i, y_i, f_w, \epsilon_i)$ with different perturbation radii ϵ_i on three datasets under l_2-norm constrained perturbations.

F.4 EXPERIMENTS ON KERNEL SVMS

F.4.1 COMPARED ALGORITHMS

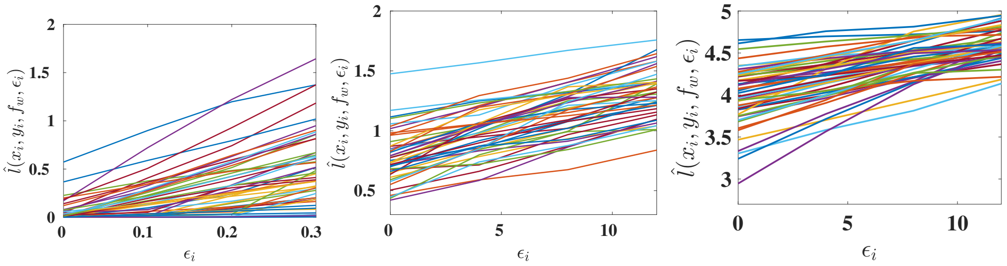
- **DSG** (Dai et al., 2014): Natural model training algorithm on kernel SVMs with doubly stochastic gradient descent.
- **adv-SVM** (Wu et al., 2021): Scalable adversarial training algorithm on kernel SVMs with doubly stochastic gradient descent.
- **SAAT-SVM**: Our self-adaptive adversarial training algorithm on kernel SVMs.

F.4.2 IMPLEMENTATION DETAILS

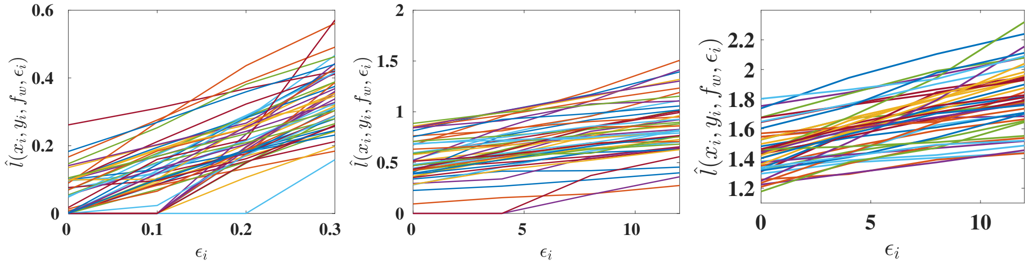
The kernel function that we use for algorithms on kernel SVMs is RBF kernel. The experiments are conducted on datasets MNIST8m[3] and CIFAR10. For SAAT-SVM, we linearly increase λ from 1 to 5 on MNIST8m, from 3 to 8 on CIFAR10.

---

[3] MNIST8m is available at `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets`.



(a) MNIST (cross entropy) (b) CIFAR10 (cross entropy) (c) CIFAR100 (cross entropy)



(d) MNIST (hinge) (e) CIFAR10 (hinge) (f) Tiny Imagenet (hinge)

Figure 7: Maximum loss value $\hat{l}(x_i, y_i, f_w, \epsilon_i)$ with different perturbation radii ϵ_i under l_∞-norm constrained perturbations.

F.4.3 RESULTS ON KERNEL SVMS

Since we focus on binary classification on kernel SVMs, we select two similar classes from MNIST8m and CIFAR10 respectively, i.e., MNIST8m 6 vs. 8 and CIFAR10 automobile vs. truck. MNIST8m 6 vs. 8 has 200,000 samples with 784 features, CIFAR10 automobile vs. truck has 10,000 samples with 3,072 features.

The robustness of the algorithms against different attacks is shown in Table 5. It is obvious that our SAAT-SVM enjoys superior performance on clean data which is even better than natural training method DSG and keeps robustness against adversarial examples at the same time.

Table 8: Test accuracy (%) of various defense methods on kernel SVMs. (The results of DSG on clean data are just baselines for reference.)

Method	Clean	FGSM	10-PGD	CW	ZOO
MNIST8m 6vs. 8					
DSG	*99.72±0.05*	98.33±0.27	97.52±0.46	91.35±0.55	91.90±0.49
adv-SVM	99.35±0.26	98.89±0.39	**98.63±0.33**	94.55±0.54	94.37±0.68
SAAT-SVM	**99.64±0.13**	**99.07±0.19**	98.32±0.35	**95.46±0.52**	**94.83±0.64**
CIFAR10 automobile vs. truck					
DSG	*75.65±0.33*	65.13±0.44	63.21±0.31	48.05±0.64	51.82±0.77
adv-SVM	75.25±0.38	73.84±0.55	65.82±0.63	50.55±0.84	53.64 ±0.76
SAAT-SVM	**76.70±0.27**	**74.55±0.43**	**67.34±0.49**	**57.95±0.69**	**54.10±0.81**