LAS-AT: Adversarial Training with Learnable Attack Strategy

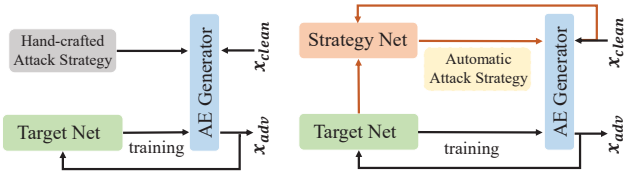
Xiaojun Jia[1,2,†*], Yong Zhang[3,*], Baoyuan Wu[4,5,‡], Ke Ma[6], Jue Wang[3], Xiaochun Cao[1,2,‡]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyberspace Security, University of Chinese Academy of Sciences, Beijing, China
[3] Tencent, AI Lab, Shenzhen, China
[4]School of Data Science, The Chinese University of Hong Kong, Shenzhen, China
[5] Secure Computing Lab of Big Data, Shenzhen Research Institute of Big Data, Shenzhen, China
[6] School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing, China
jiaxiaojun@iie.ac.cn; zhangyong201303@gmail.com; wubaoyuan@cuhk.edu.cn;
make@ucas.ac.cn; arphid@gmail.com; caoxiaochun@iie.ac.cn

Abstract

Adversarial training (AT) is always formulated as a minimax problem, of which the performance depends on the inner optimization that involves the generation of adversarial examples (AEs). Most previous methods adopt Projected Gradient Decent (PGD) with manually specifying attack parameters for AE generation. A combination of the attack parameters can be referred to as an attack strategy. Several works have revealed that using a fixed attack strategy to generate AEs during the whole training phase limits the model robustness and propose to exploit different attack strategies at different training stages to improve robustness. But those multi-stage hand-crafted attack strategies need much domain expertise, and the robustness improvement is limited. In this paper, we propose a novel framework for adversarial training by introducing the concept of "learnable attack strategy", dubbed LAS-AT, which learns to automatically produce attack strategies to improve the model robustness. Our framework is composed of a target network that uses AEs for training to improve robustness, and a strategy network that produces attack strategies to control the AE generation. Experimental evaluations on three benchmark databases demonstrate the superiority of the proposed method. The code is released at https://github.com/jiaxiaojunQAQ/LAS-AT.

1. Introduction

Although deep neural networks (DNNs) have achieved great success in academia and industry, they could be easily fooled by adversarial examples (AEs) [15,44] generated via adding indistinguishable perturbations to benign images. Recently, many studies [2, 3, 12, 13, 19, 25, 32, 47] focus on generating AEs. It has been proven that many real-world applications [14, 30] of DNNs are vulnerable to AEs, such as image classification [15, 22], object detection [26, 50], neural machine translation [23, 62], *etc*. The vulnerability of DNNs makes people pay attention to the safety of artificial intelligence and brings new challenges to the application of deep learning [17,18,54,55,61]. Adversarial training (AT) [33, 35, 43, 52] is considered as one of the most effective defense methods to improve adversarial robustness by injecting AEs into the training procedure through a minimax formulation. Under the minimax framework, the generation of AEs plays a key role in determining robustness.



(a) Conventional AT (b) LAS-AT

Figure 1. The difference between conventional AT and LAS-AT. (a) Conventional AT methods use a hand-crafted attack strategy to generate AEs. (b) The proposed LAS-AT uses a strategy network to automatically produce sample-dependent attack strategies.

Several recent works improve the standard AT method from different perspectives. Although existing methods [4, 9,10,16,27,39] have made significant progress in improving robustness, they rarely explore the impact of attack strategy on adversarial training. First, as shown in Fig. 1a, most existing methods leverage a hand-crafted attack strategy to generate AEs by manually specifying the attack parameters, *e.g.,* PGD attack with the maximal perturbation of 8, iteration of 10, and step size of 2. A hand-crafted attack strategy

*The first two authors contribute equally to this work. † Work done during an internship at Tencent AI Lab. ‡ Correspondence to: Baoyuan Wu (wubaoyuan@cuhk.edu.cn) and Xiaochun Cao (caoxiaochun@iie.ac.cn).

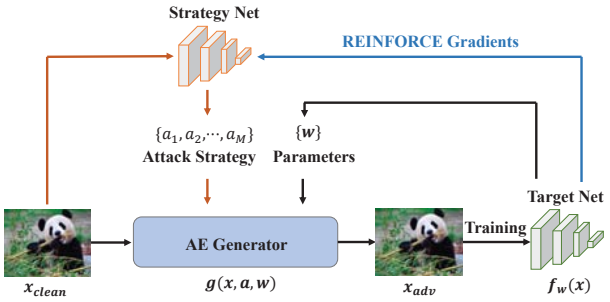


Figure 2. The framework of proposed LAS-AT. It consists of a target network and a strategy network. Given a clean image, the strategy network generates an attack strategy. The AE generator takes the strategy as well as the target network to generate an AE which is used to train the target network. Some non-differentiable operations (*e.g.* choosing the iteration times) related to attack break gradient flow from the target network to the strategy network. As an alternative approach, REINFORCE algorithm [51] is applied to optimize the strategy network and we utilize the so-called "REINFORCE gradient" to update the strategy network.

lacks flexibility and might limit the generalization performance. Second, most methods use only one attack strategy. Though some works [5, 48, 58] have realized that exploiting different attack strategies at different training stages could improve robustness, *i.e.,* using weak attacks at the early stages and strong attacks at the late stages, they use manually designed metrics to evaluate the difficulty of AEs and still use one strategy at each stage. However, they need much domain expertise, and the robustness improvement is limited. They use sample-agnostic attack strategies that are hand-crafted and independent of any information of specific samples. There exist statistical differences among samples, and attack strategy should be designed according to the information of the specific sample, *i.e.,* sample-dependent.

To alleviate these issues, we propose a novel adversarial training framework by introducing the concept of "learnable attack strategy", *dubbed LAS-AT*, which learns to automatically produce sample-dependent attack strategies for AE generation instead of using hand-crafted ones (see Fig. 1b). Our framework consists of two networks, *i.e.,* a target network and a strategy network. The former uses AEs for training to improve robustness, while the latter produces attack strategies to control the generation of AEs. The two networks play a game where the target network learns to minimize the training loss of AEs while the strategy network learns to generate strategies to maximize the training loss. Under such a gaming mechanism, at the early training stages, weak attacks can successfully attack the target network. As the robustness improves, the strategy network learns to produce strategies to generate stronger attacks. Unlike [5] [48], and [58] that use designed metrics and hand-crafted attack strategies, we use the strategy network to automatically produce an attack strategy according to the given sample. As the strategy network updates according to the robustness of the target model and the given sample, the strategy network figures out to produce different strategies accordingly at different stages, rather than setting up any manually designed metrics or strategies. We propose two loss terms to guide the learning of the strategy network. One evaluates the robustness of the target model updated with the AEs generated by the strategy. The other evaluates how well the updated target model performs on clean samples. Our main contributions are in three aspects: **1)** We propose a novel adversarial training framework by introducing the concept of "learnable attack strategy", which learns to automatically produce sample-dependent attack strategies to generate AEs. Our framework can be combined with other state-of-the-art methods as a plug-and-play component. **2)** We propose two loss terms to guide the learning of the strategy network, which involve explicitly evaluating the robustness of the target model and the accuracy of clean samples. **3)** We conduct experiments and analyses on three databases to demonstrate the effectiveness of the proposed method.

2. Related Work

Adversarial Attack Methods. As the vulnerability of deep learning models has been noticed [44], many works studied the model's robustness and proposed a series of adversarial attack methods. Fast Gradient Sign Method (FGSM) [15] was a classic adversarial attack method, which made use of the gradient of the model to generate AEs. Madry *et al.* [35] proposed a multi-step version of FGSM, called Projected Gradient Descent (PGD). To solve the problem of parameter selection in FGSM, Moosavi-Dezfooli *et al.* [36] proposed a simple but accurate method, called Deepfool, to attack deep neural networks. It generated AEs by using an iterative linearization of the classification model. Carlini-Wagner *et al.* [6] proposed several powerful attack methods that could be widely used to evaluate the robustness of deep learning models. Moreover, Croce *et al.* [8] proposed two improved methods (APGD-CE, APGD-DLR) of the PGD-attack. They did not need to choose a step size or alternate a loss function. And then they combined the proposed method with two complementary adversarial attack methods (FAB [7] and Square [1]) to evaluate the robustness, which was called AutoAttack (AA).

Adversarial Training Defense Methods. Adversarial training is an effective way to improve robustness by using AEs for training, such as [28, 37, 41, 45, 46, 49, 53]. The standard adversarial training (AT) is formulated as a minimax optimization problem in [35]. The objective function is defined as:

$$\min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\max_{\boldsymbol{\delta}\in\Omega} \mathcal{L}(f_{\mathbf{w}}(\mathbf{x}+\boldsymbol{\delta}), y)], \tag{1}$$

where $\mathcal{D}$ represents an underlying data distribution and Ω represents the perturbation set. $\mathbf{x}$ represents the example,

y represents the corresponding label, and $\boldsymbol{\delta}$ represents the indistinguishable perturbation. $f_{\mathbf{w}}(\cdot)$ represents the target network and $\mathcal{L}(f_{\mathbf{w}}(\mathbf{x}), y)$ represents the loss function of the target network. The inner maximization problem of standard AT can be regarded as the attack strategy that guides the creation of AEs, which is the core to improve the model robustness. A training strategy is designed accordingly, which significantly improves the network's robustness. Madry *et al*. proposed the prime AT framework, PGD-AT [35], to improve the robustness. And Rice *et al*. proposed a early stopping version [40] of PGD-AT, which gained a great improvement. Zhang *et al*. [57] explored a trade-off between standard accuracy and adversarial robustness and proposed a defense method (TRADES) that can trade standard accuracy off against adversarial robustness. Wu *et al.* [53] investigated the weight loss landscape and proposed an effective Adversarial Weight Perturbation (AWP) method to improve the robustness. Cui *et al*. [9] proposed to adapt the logits of one model trained on clean data to guide adversarial training (LBGAT). These AT methods adopted a fxed attack strategy to conduct AT. Some AT methods exploited different attack strategies at different training stages to improve robustness. In detail, Cai *et al*. [5] adopted curriculum adversarial training (CAT) to improve model robustness. Wang *et al*. [48] designed a criterion to measure the convergence quality and proposed dynamic adversarial training (DART) to improve the robustness of the target model. Zhang *et al.* [58] proposed to search for the least adversarial data for AT, which could be called friendly adversarial training (FAT).

3. The Proposed Approach

We propose a novel adversarial training framework by introducing the concept of "learnable attack strategy". We first introduce the pipeline of our framework in Sec. 3.1 and then present our novel formulation of adversarial training in Sec. 3.2 and our proposed loss terms in Sec. 3.3 followed by the proposed optimization algorithm in Sec. 3.4.

3.1. Pipeline of the Proposed Framework

The pipeline of our framework is shown in Fig. 2. Our model is composed of a target network and a strategy network. The former uses AEs for training to improve its robustness, whilst the latter generates attack strategies to create AEs to attack the target network. They are competitors.

Target Network. The target network is a convolutional network for image classification, denoted as $\hat{y} = f_{\mathbf{w}}(\mathbf{x})$ where $\hat{y}$ is the estimation of the label, $\mathbf{x}$ is an image, and $\mathbf{w}$ are the parameters of the network.

Strategy Network. The strategy network generates adversarial attack strategies to control the AE generation, which takes a sample as input and outputs a strategy. Since the strategy network updates gradually, it gives different strategies given the same sample as input according to the robustness of the target network at different training stages. The architecture of the strategy network is illustrated in the **supplementary material**. Given an image, the strategy network outputs an attack strategy, *i.e.,* the configuration of how to perform the adversarial attack. Let $\mathbf{a} = \{a_1, a_2, ..., a_M\} \in \mathcal{A}$ denote a strategy of which each element refers to an attack parameter. $\mathcal{A}$ denotes the value space of strategy. Parameter $a_m \in \{1, 2, ..., K_m\}$ has K_m options, which is encoded by a one-hot vector. The meaning of each option differs in different attack parameters. For example, PGD attack [35] has three attack parameters, *i.e.,* the attack step size α, the attack iteration I, and the maximal perturbation strength ε. Each parameter has K_m optional values to select, *e.g.,* the options for α could be $\{0.1, 0.2, 0.3, ...\}$ and the options for I could be $\{1, 10, 20, ...\}$. A combination of the selected values for these attack parameters is an attack strategy. The strategy is used to created AEs along with the target model. The strategy network captures the conditional distribution of a given $\mathbf{x}$ and $\boldsymbol{\theta}$, $p(\mathbf{a}|\mathbf{x}; \boldsymbol{\theta})$, where $\mathbf{x}$ is the input image and $\boldsymbol{\theta}$ denotes the strategy network parameters.

Adversarial Example Generator. Given a clean image, the process of the generation of AEs can be defined as:

$$\mathbf{x}_{adv} := \mathbf{x} + \boldsymbol{\delta} \leftarrow g(\mathbf{x}, \mathbf{a}, \mathbf{w}), \tag{2}$$

where $\mathbf{x}$ is a clean image, $\mathbf{x}_{adv}$ is its corresponding AEs, and $\boldsymbol{\delta}$ is the generated perturbation. $\mathbf{a}$ is an attack strategy. $\mathbf{w}$ represents the target network parameters, and $g(\cdot)$ is the PGD attack. The process is equivalent to solving the inner optimization problem of Eq. (1) given an attack strategy $\mathbf{a}$, *i.e.,* finding the optimal perturbation to maximize the loss.

3.2. Novel Formulation of Adversarial Training

By using Eq. (2) that represents the process of AE generation, the standard AT with a fixed attack strategy can be rewritten as:

$$\min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\, \mathcal{L}(f_{\mathbf{w}}(\mathbf{x}_{adv}), y), \tag{3}$$

where $\mathbf{x}_{adv} = g(\mathbf{x}, \mathbf{a}, \mathbf{w})$ and $\mathbf{a}$ is the hand-crafted attack strategy. $\mathcal{D}$ is the training set. $\mathcal{L}$ is the cross-entropy loss function which is used to measure the difference between the predicted label of the AE $\mathbf{x}_{adv}$ and the ground truth y.

Differently, instead of using a hand-crafted sample-agnostic strategy, we use a strategy network to produce automatically generated sample-dependent strategies, *i.e.,* $p(\mathbf{a}|\mathbf{x}; \boldsymbol{\theta})$. Our novel formulation for AT can be defined as:

$$\min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\max_{\boldsymbol{\theta}}\, \mathbb{E}_{\mathbf{a}\sim p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta})}\, \mathcal{L}(f_{\mathbf{w}}(\mathbf{x}_{adv}), y)]. \tag{4}$$

Compared to the standard AT, the most distinct difference lies in the generation of AEs, *i.e.,* $\mathbf{x}_{adv}$ (2). The standard AT uses a hand-crafted sample-agnostic strategy $\mathbf{a}$ to solve the inner optimization problem while we use a strategy network to produce the sample-dependent strategy by $p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta})$, *i.e.,* our strategy is learnable. Our AE generation involves the parameters $\boldsymbol{\theta}$ of the strategy network, which leads that our loss being a function of the parameters of both networks.

Comparing Eq. (3) and Eq. (4), our formulation is a minimax problem and the inner optimization involves the parameters of the strategy network. From Eq. (4), it can be observed that the two networks compete with each other in minimizing or maximizing the same objective. The target network learns to adjust its parameters to defend AEs generated by the attack strategies, while the strategy network learns to improve attack strategies according to the given samples to attack the target network. At the beginning of the training phase, the target network is vulnerable, which a weak attack can fool. Hence, the strategy network can easily generate effective attack strategies. The strategies could be diverse because both weak and strong attacks can succeed. As the training process goes on, the target network becomes more robust. The strategy network has to learn to generate attack strategies that create stronger AEs. Therefore, the gaming mechanism could boost the robustness of the target network gradually along with the improvement of the strategy network.

3.3. The Proposed Loss Terms

Loss of Evaluating Robustness. To guide the learning of the strategy network, we propose a new metric to evaluate attack strategy by using the robustness of the one-step updated version of the target model. Specifically, an attack strategy $\mathbf{a}$ is first used to create an AE $\mathbf{x}_{adv}$ which is then used to adjust the parameters of the target model $\mathbf{w}$ for one step through the first-order gradient descent. The attack strategy is criticized to be effective if the updated target model $\hat{\mathbf{w}}$ can correctly predict labels for AEs $\mathbf{x}_{adv}^{\hat{\mathbf{a}}}$ that generated by another attack strategy $\hat{\mathbf{a}}$, *e.g.,* PGD with the maximal perturbation strength of 8, iterative steps of 10 and step size of 2. The loss function of evaluating robustness can be defined as:

$$\mathcal{L}_2(\boldsymbol{\theta}) = -\mathcal{L}(f(\mathbf{x}_{adv}^{\hat{\mathbf{a}}}, \hat{\mathbf{w}}), y), \tag{5}$$

where $\hat{\mathbf{w}} = \mathbf{w} - \lambda \nabla_{\mathbf{w}} \mathcal{L}_1|_{\mathbf{x}_{adv}}$ is the parameters of the updated target network and λ is the step size. $\mathcal{L}_1$ refers to the loss in Eq. (4), *i.e.,* $\mathcal{L}_1(\mathbf{w}, \boldsymbol{\theta}) := \mathcal{L}(f(\mathbf{x}_{adv}, \mathbf{w}), y)$. $\mathbf{x}_{adv}$ is created by the attack strategy $\mathbf{a}$, which is to be evaluated. $\mathbf{x}_{adv}^{\hat{\mathbf{a}}} := g(\mathbf{x}, \hat{\mathbf{a}}, \hat{\mathbf{w}})$ is the AE created by another attack strategy $\hat{\mathbf{a}}$, which is used to evaluate the robustness of the updated model $\hat{\mathbf{w}}$. Please note that $\mathcal{L}_2$ is used to evaluate the attack strategy and $\mathbf{w}$ is treated as a variable here rather than parameters to optimize. Hence, the value of $\mathbf{w}$ is used in Eq. (5), but the gradient of $\mathcal{L}_2$ will not be back-propagated to update $\mathbf{w}$ through $\hat{\mathbf{w}}$. Eq. (5) indicates that a larger $\mathcal{L}_2$ means the updated target model is more robust, *i.e.,* a better attack strategy.

Loss of Predicting Clean Samples. A good attack strategy should not only improve the robustness of the target model but also maintain the performance of predicting clean samples, *i.e.,* clean accuracy. To further provide guidance for learning the strategy network, we also consider the performance of the one-step updated target model in predicting clean samples. The loss of evaluating the attack strategy can be defined as:

$$\mathcal{L}_3(\boldsymbol{\theta}) = -\mathcal{L}(f(\mathbf{x}, \hat{\mathbf{w}}), y), \tag{6}$$

where $\hat{\mathbf{w}}$ is the same as that in Eq. (5), *i.e.,* the parameters of the one-step updated target model. $\mathcal{L}_3$ is the function of $\boldsymbol{\theta}$ as the AE $\mathbf{x}_{adv}$ involves computing $\hat{\mathbf{w}}$ and $\mathbf{a}$ is the output of the strategy network. Eq. (6) indicates that a larger $\mathcal{L}_3$ means the updated target model has a lower loss in clean samples, *i.e.,* a better attack strategy.

Formal Formulation. Incorporating the two proposed loss terms, our formulation for AT can be defined as:

$$\min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \Big[\max_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{a}\sim p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta})} [\mathcal{L}_1(\mathbf{w}, \boldsymbol{\theta}) + \alpha\mathcal{L}_2(\boldsymbol{\theta}) + \beta\mathcal{L}_3(\boldsymbol{\theta})] \Big], \tag{7}$$

where $\mathcal{L}_1$ is a function of the parameters of both the target network and the strategy network while $\mathcal{L}_2$ and $\mathcal{L}_3$ involve the parameters of the strategy network. α and β are the trade-off hyper-parameters of the two loss terms.

3.4. Optimization

We propose an algorithm to alternatively optimize the parameters of the two networks. Given $\boldsymbol{\theta}$, the subproblem of optimizing the target network can be defined as,

$$\min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \mathbb{E}_{\mathbf{a}\sim p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta})} [\mathcal{L}_1(\mathbf{w}, \boldsymbol{\theta})]. \tag{8}$$

Given a clean image, the strategy network generates a strategy distribution $p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta})$, and we randomly sample a strategy from the conditional distribution. The sampled strategy is used to generate AEs. After collecting the AEs for a batch of samples, we can update the parameters of the target model through gradient descent, *i.e.,*

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta_1 \frac{1}{N} \sum_{n=1}^{N} \nabla_{\mathbf{w}} \mathcal{L}\left(f(\mathbf{x}_{adv}^n, \mathbf{w}^t), y_n\right), \tag{9}$$

where N is the number of samples in a mini batch and η_1 is the learning rate.

Given $\mathbf{w}$, the subproblem of optimizing the parameters of the strategy network can be written as,

$$\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta}), \tag{10}$$

where $J(\boldsymbol{\theta}) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\, \mathbb{E}_{\mathbf{a}\sim p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta})}\, [\mathcal{L}_1 + \alpha\mathcal{L}_2 + \beta\mathcal{L}_3]$. The biggest challenge of this optimization problem is that the process of AE generation (see Eq. (2)) is not differentiable, namely, the gradient can not be backpropagated to the attack strategy through the AEs. Moreover, there are some non-differentiable operations (*e.g.* choosing the iteration times) related to attack [24, 38], which sets an obstacle to backpropagate the gradient to the strategy network.

Following by the REINFORCE algorithm [51], we can compute the derivative of the objective function $J(\boldsymbol{\theta})$ with respect to the parameters $\boldsymbol{\theta}$ as:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \mathbb{E}_{\mathbf{a}\sim p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta})}\, [\mathcal{L}_0] \\ &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \int_{\mathbf{a}} \mathcal{L}_0 \cdot \nabla_{\boldsymbol{\theta}} p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta}) d\mathbf{a} \\ &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \int_{\mathbf{a}} \mathcal{L}_0 \cdot p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta}) d\mathbf{a} \\ &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \mathbb{E}_{\mathbf{a}\sim p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta})} [\mathcal{L}_0 \cdot \nabla_{\boldsymbol{\theta}} \log p(\mathbf{a}|\mathbf{x};\boldsymbol{\theta})],\end{aligned} \tag{11}$$

where $\mathcal{L}_0 = \mathcal{L}_1 + \alpha\mathcal{L}_2 + \beta\mathcal{L}_3$. Similar to solving Eq. (8), we sample attack strategy from the conditional distribution of strategy to generate AEs. The gradient with respect to the parameters can be approximately computed as:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \approx \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}_0(\mathbf{x}^n;\boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{a}^n|\mathbf{x}^n). \tag{12}$$

Then, the parameters of the strategy network can be updated through gradient ascent, *i.e.,*

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t + \eta_2 \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}^t), \tag{13}$$

where η_2 is the learning rate. And $\boldsymbol{\theta}$ and $\mathbf{w}$ are updated iteratively. We update $\mathbf{w}$ every k times of updating $\boldsymbol{\theta}$.

3.5. Convergence Analysis

Based on (9) and (13), we have the following convergence result of the proposed adversarial training algorithm.

Theorem 1. *Suppose that the objective function $\mathcal{L}_0 = \mathcal{L}_1 + \alpha\mathcal{L}_2 + \beta\mathcal{L}_3$ in (7) satisfied the gradient Lipschitz conditions w.r.t. $\boldsymbol{\theta}$ and $\mathbf{w}$, and $\mathcal{L}_0$ is μ-strongly concave in $\boldsymbol{\Theta}$, the feasible set of $\boldsymbol{\theta}$. If $\hat{\mathbf{x}}_{adv}(\mathbf{x},\mathbf{w})$ is a σ-approximate solution of the ℓ_∞ ball with radius ϵ constraint, the variance of the stochastic gradient is bounded by a constant $\sigma^2 > 0$, and we set the learning rate of $\mathbf{w}$ as*

$$\eta_1 = \min\left(\frac{1}{L_0}, \sqrt{\frac{\mathcal{L}_0(\mathbf{w}^0) - \min_{\mathbf{w}} \mathcal{L}_0(\mathbf{w})}{\sigma^2 T L_0}}\right), \tag{14}$$

Table 1. Test robustness (%) on the CIFAR-10 database using ResNet18. Number in bold indicates the best.

Method	PGD-AT [40]	k=1	k=10	k=20	k=40	k=60
Clean	82.56	**82.88**	82.38	82.00	82.3	82.10
PGD-10	53.15	53.71	53.89	53.53	**54.29**	53.85
Time(min)	261	1378	432	418	365	333

where $L_0 = L_{\mathbf{w}\boldsymbol{\theta}} L_{\boldsymbol{\theta}\mathbf{w}}/\mu + L_{\mathbf{ww}}$ is the Lipschitz constants of $\mathcal{L}_0$, it holds that

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[\|\nabla \mathcal{L}_0(\mathbf{w}^t)\|_2^2\right] \leq 4\sigma\sqrt{\frac{\Delta L_0}{T}} + \frac{5\delta L_{\mathbf{w}\boldsymbol{\theta}}^2}{\mu}, \tag{15}$$

where T is the maximum adversarial training epoch number and $\Delta = \mathcal{L}_0(\mathbf{w}^0) - \min_{\mathbf{w}} \mathcal{L}_0(\mathbf{w})$.

The detailed proof is presented in the **supplementary material**. By Theorem 1, if the inner maximization process can obtain a δ-approximation of $\mathbf{x}_{\text{adv}}^*$, the proposed method **LAS-AT** can archive a stationary point by sub-linear rate with the precision $5\delta L_{\mathbf{w}\boldsymbol{\theta}}^2/\mu$. Moreover, if $5\delta L_{\mathbf{w}\boldsymbol{\theta}}^2/\mu$ is sufficient small, our method can find the desired robust model $\mathbf{w}^T$ with a good approximation of $\mathbf{x}_{\text{adv}}^*$.

4. Experiments

To evaluate the proposed method, we conduct experiments on three databases, *i.e.,* CIFAR10 [29], CIFAR100 [29], and Tiny ImageNet [11]. The details of these databases are presented in the **supplementary material**.

4.1. Settings

Competitive Methods. To evaluate the proposed method effectiveness in improving the robustness of a target model, we combine it with several state-of-the-art adversarial training methods and illustrate its performance improvements. We choose not only the most popular methods such as the early stopping PGD-AT [40] and TRADES [57] as base models but also the recently proposed method AWP [53]. The combinations of our method and these models are referred to as LAS-PGD-AT, LAS-TRADES, and LAS-AWP, respectively. Note that we use the **same training settings** as the base models [40, 53, 57] to train our proposed models, including data splits and training losses. Then we compare the proposed LAS-PGD-AT, LAS-TRADES, and LAS-AWP with the following baselines: (1) PGD-AT [40], (2) TRADES [57], (3) SAT [42], (4) MART [49], (5) FAT [58], (6) GAIRAT [59], (7) AWP [53] and (8) LBGAT [9]. Moreover, we compare our method with CAT [5], DART [48] and FAT [58] They use different attack strategies at different training stages to conduct AT. Besides, we also compare our method with other state-of-the-art hyperparameter search methods [34, 60] to evaluate our method.

Table 2. Test robustness (%) on the CIFAR-10 database using WRN34-10. Number in bold indicates the best.

Method	Clean	PGD-10	PGD-20	PGD-50	C&W	AA
PGD-AT [40]	85.17	56.07	55.08	54.88	53.91	51.69
TRADES [57]	85.72	56.75	56.1	55.9	53.87	53.40
MART [49]	84.17	58.98	58.56	58.06	54.58	51.10
FAT [58]	**87.97**	50.31	49.86	48.79	48.65	47.48
GAIRAT [59]	86.30	60.64	59.54	58.74	45.57	40.30
AWP [53]	85.57	58.92	58.13	57.92	56.03	53.90
LBGAT [9]	88.22	56.25	54.66	54.3	54.29	52.23
LAS-AT(ours)	86.23	57.64	56.49	56.12	55.73	53.58
LAS-TRADES(ours)	85.24	58.01	57.07	56.8	55.45	54.15
LAS-AWP(ours)	87.74	**61.09**	**60.16**	**59.79**	**58.22**	**55.52**

Table 3. Test robustness (%) on the CIFAR-100 database using WRN34-10. Number in bold indicates the best.

Method	Clean	PGD-10	PGD-20	PGD-50	C&W	AA
PGD-AT [40]	60.89	32.19	31.69	31.45	30.1	27.86
TRADES [57]	58.61	29.20	28.66	28.56	27.05	25.94
SAT [42]	62.82	28.1	27.17	26.76	27.32	24.57
AWP [53]	60.38	34.13	33.86	33.65	31.12	28.86
LBGAT [9]	60.64	35.13	34.75	34.62	30.65	29.33
LAS-AT(ours)	61.80	33.45	32.77	32.54	31.12	29.03
LAS-TRADES(ours)	60.62	32.99	32.53	32.39	29.51	28.12
LAS-AWP(ours)	**64.89**	**37.11**	**36.36**	**36.13**	**33.92**	**30.77**

Evaluation. We choose several adversarial attack methods to attack the trained models, including PGD [35], C&W [6] and AA [8] which consists of APGD-CE [8], APGD-DLR [8], FAB [7] and Square [1]. Following the default setting of AT, the max perturbation strength ϵ is set to 8 for all attack methods under the L_∞. The clean accuracy and robust accuracy are used as the evaluation metrics.

Implementation Details. On CIFAR-10 and CIFAR-100, we use ResNet18 [20] or WideResNet34-10(WRN34-10) [56] as the target network. On Tiny ImageNet, we use PreActResNet18 [21] as the target model. For all experiments, we train the target network for defense baselines, following their original papers. For the training hyper-parameters of the target network of our method, we use the **same setting** as the base models [40, 53, 57]. The detailed settings are presented in the **supplementary material**. For the target network, we adopt SGD momentum optimizer with a learning rate of 0.1, weight decay of 5×10^{-4}. We use ResNet18 as the backbone of the strategy network. For the strategy network of our method, we adopt SGD momentum optimizer with a learning rate of 0.001. The trade-off hyper-parameters α and β are set to 2.0 and 4.0. The range of the maximal perturbation strength is set from 3 to 15, the range of the attack step is set from 1 to 6, and the range of the attack iteration is set from 3 to 15.

4.2. Hyper-parameter Selection

The hyper-parameter k controls the alternative update of $\mathbf{w}$ and $\boldsymbol{\theta}$. We update $\mathbf{w}$ every k times of updating $\boldsymbol{\theta}$. It not only affects model robustness but also affects model training efficiency. A hyper-parameter selection experiment with ResNet18 is conducted on CIFAR-10 to select the optimal hyper-parameter k. The results are shown in Table 1. The training time of the proposed LAS-PGD-AT decreases along with the increase of parameter k. The more time the strategy network requires for training, the smaller the k is. When k = 40, the proposed LAS-PGD-AT achieves the best adversarial robustness. Considering AT efficiency, we set k to 40. The selection of hyper-parameters α and β is presented in the **supplementary material**.

4.3. Comparisons with Other AT Methods

Our method is a plug-and-play component that can be combined with other AT methods to boost their robustness.

Comparisons on CIFAR-10 and CIFAR-100. The results on CIFAR-10 and CIFAR-100 are shown in Table 2 and Table 3. Analyses are as follows. First, the three proposed models outperform their base models under most attack scenarios. In a lot of cases, our method not only improves the robustness but also improves the clean accuracy of the base models though there is always a trade-off between accuracy and robustness. For example, on CIFAR-10 when

Table 4. Test robustness (%) on the Tiny Imagenet database using PreActResNet18. Number in bold indicates the best.

Method	Clean	PGD-50	C&W	AA
PGD-AT [40]	43.98	19.98	17.6	13.78
TRADES [57]	39.16	15.74	12.92	12.32
AWP [53]	41.48	22.51	19.02	17.34
LAS-AT(ours)	44.86	22.16	18.54	16.74
LAS-TRADES(ours)	41.38	18.36	14.5	14.08
LAS-AWP(ours)	**45.26**	**23.42**	**19.88**	**18.42**

Table 5. Test robustness (%) on the CIFAR-10 and CIFAR-100 database. Number in bold indicates the best.

Database	Target network	Method	Clean	AA
CIFAR-10	WRN70-16	Gowal *et al* [16]	85.29	57.20
		LAS-AWP(ours)	**85.66**	**57.61**
CIFAR-100	WRN34-20	LBGAT [9]	62.55	30.20
		LAS-AWP(ours)	**67.31**	**31.92**

Table 6. Test robustness (%) on the CIFAR-10 database using ResNet18. Number in bold indicates the best.

$\mathcal{L}_1$	$\mathcal{L}_2$	$\mathcal{L}_3$	clean	PGD-10	AA
✓			81.83	53.88	49.06
✓	✓		81.54	53.98	49.34
✓		✓	81.90	53.89	49.20
✓	✓	✓	**82.3**	**54.29**	**49.89**

Table 7. Test robustness (%) on the CIFAR-10 database using WRN34-10. Comparisons with Madry, CAT, DART and FAT. The results are reported in [58]. Number in bold indicates the best.

Method	Clean	FGSM	PGD-20	C&W
Madry-AT [35]	87.3	56.1	45.8	46.8
CAT [5]	77.43	57.17	46.06	42.28
DART [48]	85.03	63.53	48.70	47.27
FAT [58]	**87.97**	65.94	49.86	48.65
LAS-Madry-AT	84.95	**67.16**	**55.61**	**54.31**

using WRN34-10 as the target network, our method improves the clean accuracy of powerful AWP by about 2.2% and also improves the performance of AWP under PGD-10 attack and AA attack by about 2.1% and 1.62%, respectively. Moreover, the proposed LAS-AWP achieves the best robustness performance under all attack scenarios. We attribute the improvements to using automatically generated attack strategies instead of hand-crafted ones. Second, on CIFAR-100, the proposed LAS-AWP not only achieves the highest accuracy on clean images but also achieves the best robustness performance under all attack scenarios. In detail, our LAS-AWP outperforms the original AWP 4.5% and 1.9% on the clean accuracy and AA attack accuracy, respectively. Moreover, our LAS-AWP outperforms the powerful LBGAT under all attack scenarios.

Comparisons on Tiny ImageNet. Following [31], we use PreActResNet18 [21] as the target model for evaluation on Tiny ImageNet. The results are shown in Table 4. As Tiny ImageNet has more classes than CIFAR-10 and CIFAR-100, the defense of AEs is more challenging. Our method improves the clean and adversarial robustness accuracy of the three base models.

Comparisons with state-of-the-art robustness model. Auto Attack (AA) is a reliable and strong attack method to evaluate model robustness. It consists of three white-box attacks and a black-box attack. The details is introduced in Sec 2. Under their leaderboard results [1], on CIFAR-10, Gowal *et al*. [16] study the impact of hyper-parameters (such as model weight averaging and model size) on model robustness and adopt WideResNet70-16 (WRN-70-16) to conduct AT, which ranks the 1st under AA attack without additional real or synthetic data. We also adopt WRN-70-16 for our method. LAS-AWP can boost the model robustness and achieve higher robustness accuracy. On CIFAR-100, Cui *et al*. train WideResNet34-20 (WRN-34-20) for LBGAT and achieves state-of-the-art robustness without additional real or synthetic data. We also adopt WRN-34-20 for our method. LAS-AWP can also achieve higher robustness accuracy. The result is shown Table 5.

4.4. Ablation Study

In our formulation in Eq. (7), besides the loss $\mathcal{L}_1$, we propose two additional loss terms to guide the learning of the strategy network, *i.e.,* the loss of evaluating robustness $\mathcal{L}_2$ and the loss of predicting clean samples $\mathcal{L}_3$. To validate the effectiveness of each element in the objective function, we conduct ablation experiments with ResNet18 on CIFAR-10. We train four LAS-PGD-AT models by using $\mathcal{L}_1$, $\mathcal{L}_1\&\mathcal{L}_2$, $\mathcal{L}_1\&\mathcal{L}_3$, and $\mathcal{L}_1\&\mathcal{L}_2\&\mathcal{L}_3$, respectively. The trained models are attacked by a set of adversarial attack methods. The results are shown in Table 6. The classification accuracy is the evaluation metric. *Clean* represents using clean images for testing while other attack methods use AEs for testing.

Analyses are summarized as follows. First, when incorporating the loss $\mathcal{L}_2$ only, the performance of robustness under all attacks improves while the clean accuracy slightly drops. When incorporating the loss $\mathcal{L}_3$ only, the clean accuracy improves, but the performance of robustness under partial attacks slightly drops. The results show that $\mathcal{L}_2$ contributes more to improve the robustness and $\mathcal{L}_3$ contributes more to improve the clean accuracy. Second, using all losses achieves the best performance in robustness as well as the clean accuracy, which indicates that the two losses are compatible and combining them could remedy the side effect of independent use.

4.5. Performance Analysis

Comparisons with hand-crafted attack strategy methods. To investigate the effectiveness of automatically generated attack strategies generated by our method, we compare

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

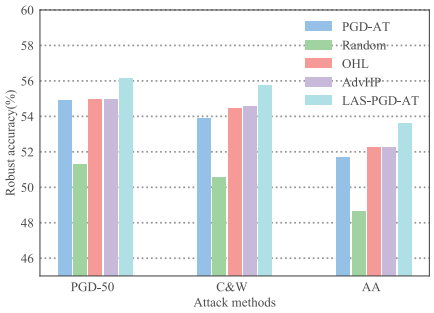


Figure 3. Comparisons with the hyper-parameter search methods using WRN34-10 on the CIFAR-10 database. x-axis represents the attack methods. y-axis represents the robust accuracy.

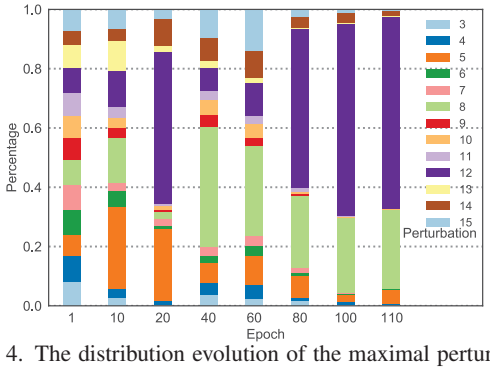


Figure 4. The distribution evolution of the maximal perturbation strength in LAS-PGD-AT during training.

our LAS-AT with some AT methods (CAT [5], DART [48] and FAT [58]) which adopt dynamic hand-crafted attack strategies for training. For a fair comparison, we keep the training and evaluation setting as same as those used in FAT [58]. The details are presented in the **supplementary material**. The result is shown in Table 7. Our method outperforms competing methods under all attacks. It indicates that compared with previous hand-crafted attack strategies, the proposed automatically generated attack strategies can achieve the greater robustness improvement.

Comparisons with hyper-parameter search methods. We compare the proposed method with other hyper-parameter search methods that include a classical hyper-parameter search method (random search) and two automatic hyper-parameter search methods (OHL [34] and AdvHP [60]). For a fair comparison, the same hyper-parameters and search range that are used in our method (see Sec 4.1) are adopted for them. The detail settings are presented in the **supplementary material**. The result is shown in Fig. 3. It can be observed that our method achieves the best robustness performance under all attack scenarios. The automatically generated attack strategies generated by our method are more suitable for AT.

Adversarial Training from Easy to Difficult. To investigate how LAS-AT works, we analyze the distribution of the strategy network's attack strategies at different training stages. Experiments using ResNet18 with LAS-PGD-AT are performed on the CIFAR-10 database. The range of the maximal perturbation strength is set from 3 to 15. The distribution evolution of the maximal perturbation strength during adversarial training is illustrated in Fig. 4.

At the beginning of AT, the distribution covers all the optional values of the maximal perturbation strength. Each value has a chance to be selected, which ensures the diversity of AEs. As the training process goes on, the percentage of small perturbation strengths decreases. At the late stages, the distribution of the maximal perturbation strength is occupied by several large values. This phenomenon indicates that the strategy network gradually increases the percentage of large perturbation strengths to generate strong AEs because the robustness of the target network is gradually boosted by training with the generated AEs. Therefore, it can be observed that under the gaming mechanism, our method starts training with diverse AEs when the target network is vulnerable, and then learns with more strong AEs at the late stages when the robustness of the target network improves. CAT [5], DART [48] and FAT [58] adopt hand-crafted strategies to use weak AEs at early stages and then use strong AEs at late stages. Unlike them, under our framework, the strategy network automatically generates strategies that determine the difficulty of AEs, according to the robustness of the target network at different stages.

5. Conclusion and Discussion

We propose a novel adversarial training framework by introducing the concept of "learnable attack strategy", which is composed of two competitors, *i.e.,* a target network and a strategy network. Under the gaming mechanism, the strategy network learns to produce dynamic sample-dependent attack strategies according to the robustness of the target model for adversarial example generation, instead of using hand-crafted attack strategies. To guide the learning of the strategy network, we also propose two loss terms that involve evaluating the robustness of the target network and predicting clean samples. Extensive experimental evaluations are performed on three benchmark databases to demonstrate the superiority of the proposed method.

Acknowledgement

Supported by the National Key R&D Program of China under Grant 2019YFB1406500, National Natural Science Foundation of China (No.62025604, 62076213, 62006217). Open Project Program of State Key Laboratory of Virtual Reality Technology and Systems, Beihang University (No.VRLAB2021C06). The university development fund of the Chinese University of Hong Kong, Shenzhen under grant No. 01001810, and Tencent AI Lab Rhino-Bird Focused Research Program under grant No.JR202123.

References

[1] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. *arXiv preprint arXiv:1912.00049*, 2019. 2, 6

[2] Jiawang Bai, Bin Chen, Yiming Li, Dongxian Wu, Weiwei Guo, Shu-tao Xia, and En-hui Yang. Targeted attack for deep hashing based retrieval. In *European Conference on Computer Vision*, pages 618–634. Springer, 2020. 1

[3] Jiawang Bai, Baoyuan Wu, Yong Zhang, Yiming Li, Zhifeng Li, and Shu-Tao Xia. Targeted attack against deep neural networks via flipping limited weight bits. *arXiv preprint arXiv:2102.10496*, 2021. 1

[4] Yang Bai, Yuyuan Zeng, Yong Jiang, Shu-Tao Xia, Xingjun Ma, and Yisen Wang. Improving adversarial robustness via channel-wise activation suppressing. *arXiv preprint arXiv:2103.08307*, 2021. 1

[5] Qi-Zhi Cai, Min Du, Chang Liu, and Dawn Song. Curriculum adversarial training. *arXiv preprint arXiv:1805.04807*, 2018. 2, 3, 5, 7, 8

[6] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017. 2, 6

[7] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. *arXiv preprint arXiv:1907.02044*, 2019. 2, 6

[8] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. *arXiv preprint arXiv:2003.01690*, 2020. 2, 6

[9] Jiequan Cui, Shu Liu, Liwei Wang, and Jiaya Jia. Learnable boundary guided adversarial training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15721–15730, 2021. 1, 3, 5, 6, 7

[10] Sihui Dai, Saeed Mahloujifar, and Prateek Mittal. Parameterizing activation functions for adversarial robustness. *arXiv preprint arXiv:2110.05626*, 2021. 1

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[12] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018. 1

[13] Yanbo Fan, Baoyuan Wu, Tuanhui Li, Yong Zhang, Mingyang Li, Zhifeng Li, and Yujiu Yang. Sparse adversarial attack via perturbation factorization. In *European conference on computer vision*, pages 35–50. Springer, 2020. 1

[14] Weiwei Feng, Baoyuan Wu, Tianzhu Zhang, Yong Zhang, and Yongdong Zhang. Meta-attack: Class-agnostic and model-agnostic physical adversarial attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7787–7796, 2021. 1

[15] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1, 2

[16] Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020. 1, 7

[17] Jindong Gu, Volker Tresp, and Han Hu. Capsule network is not more robust than convolutional network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14309–14317, 2021. 1

[18] Jindong Gu, Baoyuan Wu, and Volker Tresp. Effective and efficient vote attack on capsule networks. *arXiv preprint arXiv:2102.10055*, 2021. 1

[19] Jindong Gu, Hengshuang Zhao, Volker Tresp, and Philip Torr. Adversarial examples on segmentation models can be easy to transfer. *arXiv preprint arXiv:2111.11368*, 2021. 1

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016. 6, 7

[22] Hokuto Hirano, Akinori Minagi, and Kazuhiro Takemoto. Universal adversarial attacks on deep neural networks for medical image classification. *BMC medical imaging*, 21(1):1–13, 2021. 1

[23] Shujian Huang, Jun Xie, Xinyu Dai, CHEN Jiajun, et al. A reinforced generation of adversarial examples for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3486–3497, 2020. 1

[24] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *arXiv preprint arXiv:1905.02175*, 2019. 5

[25] Xiaojun Jia, Xingxing Wei, Xiaochun Cao, and Xiaoguang Han. Adv-watermark: A novel watermark perturbation for adversarial examples. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1579–1587, 2020. 1

[26] Xiaojun Jia, Huanqian Yan, Yonglin Wu, Xingxing Wei, Xiaochun Cao, and Yong Zhang. An effective and robust detector for logo detection. *arXiv preprint arXiv:2108.00422*, 2021. 1

[27] Xiaojun Jia, Yong Zhang, Baoyuan Wu, Jue Wang, and Xiaochun Cao. Boosting fast adversarial training with learnable adversarial initialization. *arXiv preprint arXiv:2110.05007*, 2021. 1

[28] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018. 2

[29] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[30] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016. 1

[31] Saehyung Lee, Hyungyu Lee, and Sungroh Yoon. Adversarial vertex mixup: Toward better adversarially robust generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 272–281, 2020. 7

[32] Yandong Li, Lijun Li, Liqiang Wang, Tong Zhang, and Boqing Gong. Nattack: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. In *International Conference on Machine Learning*, pages 3866–3876. PMLR, 2019. 1

[33] Yiming Li, Baoyuan Wu, Yan Feng, Yanbo Fan, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Semi-supervised robust training with generalized perturbed neighborhood. *Pattern Recognition*, 124:108472, 2022. 1

[34] Chen Lin, Minghao Guo, Chuming Li, Xin Yuan, Wei Wu, Junjie Yan, Dahua Lin, and Wanli Ouyang. Online hyper-parameter learning for auto-augmentation strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6579–6588, 2019. 5, 8

[35] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 1, 2, 3, 6, 7

[36] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016. 2

[37] Tianyu Pang, Xiao Yang, Yinpeng Dong, Kun Xu, Jun Zhu, and Hang Su. Boosting adversarial training with hypersphere embedding. *arXiv preprint arXiv:2002.08619*, 2020. 2

[38] Xi Peng, Zhiqiang Tang, Fei Yang, Rogerio S Feris, and Dimitris Metaxas. Jointly optimize data augmentation and network training: Adversarial data augmentation in human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2226–2234, 2018. 5

[39] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan A Calian, Florian Stimberg, Olivia Wiles, and Timothy Mann. Fixing data augmentation to improve adversarial robustness. *arXiv preprint arXiv:2103.01946*, 2021. 1

[40] Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pages 8093–8104. PMLR, 2020. 3, 5, 6, 7

[41] Kevin Roth, Yannic Kilcher, and Thomas Hofmann. Adversarial training is a form of data-dependent operator norm regularization. *arXiv preprint arXiv:1906.01527*, 2019. 2

[42] Chawin Sitawarin, Supriyo Chakraborty, and David Wagner. Sat: Improving adversarial training via curriculum-based loss smoothing. In *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, pages 25–36, 2021. 5, 6

[43] Chuanbiao Song, Yanbo Fan, Yichen Yang, Baoyuan Wu, Yiming Li, Zhifeng Li, and Kun He. Regional adversarial training for better robust generalization. *arXiv preprint arXiv:2109.00678*, 2021. 1

[44] 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. 1, 2

[45] Haotao Wang, Tianlong Chen, Shupeng Gui, Ting-Kuei Hu, Ji Liu, and Zhangyang Wang. Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free. *arXiv preprint arXiv:2010.11828*, 2020. 2

[46] Qizhou Wang, Feng Liu, Bo Han, Tongliang Liu, Chen Gong, Gang Niu, Mingyuan Zhou, and Masashi Sugiyama. Probabilistic margins for instance reweighting in adversarial training. *arXiv preprint arXiv:2106.07904*, 2021. 2

[47] Xiaosen Wang and Kun He. Enhancing the transferability of adversarial attacks through variance tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1924–1933, 2021. 1

[48] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *ICML*, volume 1, page 2, 2019. 2, 3, 5, 7, 8

[49] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*, 2019. 2, 5, 6

[50] Xingxing Wei, Siyuan Liang, Ning Chen, and Xiaochun Cao. Transferable adversarial attacks for image and video object detection. *arXiv preprint arXiv:1811.12641*, 2018. 1

[51] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992. 2, 5

[52] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020. 1

[53] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33, 2020. 2, 3, 5, 6, 7

[54] Jia Xu, Yiming Li, Yong Jiang, and Shu-Tao Xia. Adversarial defense via local flatness regularization. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 2196–2200. IEEE, 2020. 1

[55] Bangjie Yin, Wenxuan Wang, Taiping Yao, Junfeng Guo, Zelun Kong, Shouhong Ding, Jilin Li, and Cong Liu. Advmakeup: A new imperceptible and transferable attack on face recognition. *arXiv preprint arXiv:2105.03162*, 2021. 1

[56] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 6

[57] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019. 3, 5, 6, 7

[58] 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*, pages 11278–11287. PMLR, 2020. 2, 3, 5, 6, 7, 8

[59] Jingfeng Zhang, Jianing Zhu, Gang Niu, Bo Han, Masashi Sugiyama, and Mohan Kankanhalli. Geometry-aware instance-reweighted adversarial training. *arXiv preprint arXiv:2010.01736*, 2020. 5, 6

[60] Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial autoaugment. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 5, 8

[61] Yuli Zheng, Zhenyu Wu, Ye Yuan, Tianlong Chen, and Zhangyang Wang. Pcal: A privacy-preserving intelligent credit risk modeling framework based on adversarial learning. *arXiv preprint arXiv:2010.02529*, 2020. 1

[62] Wei Zou, Shujian Huang, Jun Xie, Xinyu Dai, and Jiajun Chen. A reinforced generation of adversarial examples for neural machine translation. *arXiv preprint arXiv:1911.03677*, 2019. 1