Practical Evaluation of Adversarial Robustness via Adaptive Auto Attack

Ye Liu[1], Yaya Cheng[1], Lianli Gao[1], Xianglong Liu[2], Qilong Zhang[1], Jingkuan Song[1]*
[1] Center for Future Media and School of Computer Science and Engineering
University of Electronic Science and Technology of China, China
[2] Beihang University, China
liuye66a@gmail.com, yaya.cheng@hotmail.com, lianli.gao@uestc.edu.cn
xlliu@buaa.edu.cn, qilong.zhang@std.uestc.edu.cn, jingkuan.song@gmail.com

Abstract

Defense models against adversarial attacks have grown significantly, but the lack of practical evaluation methods has hindered progress. Evaluation can be defined as looking for defense models' lower bound of robustness given a budget number of iterations and a test dataset. A practical evaluation method should be convenient (i.e., parameter-free), efficient (i.e., fewer iterations) and reliable (i.e., approaching the lower bound of robustness). Towards this target, we propose a parameter-free Adaptive Auto Attack (A^3) evaluation method which addresses the efficiency and reliability in a test-time-training fashion. Specifically, by observing that adversarial examples to a specific defense model follow some regularities in their starting points, we design an Adaptive Direction Initialization strategy to speed up the evaluation. Furthermore, to approach the lower bound of robustness under the budget number of iterations, we propose an online statistics-based discarding strategy that automatically identifies and abandons hard-to-attack images. Extensive experiments on nearly 50 widely-used defense models demonstrate the effectiveness of our A^3. By consuming much fewer iterations than existing methods, i.e., $1/10$ *on average (10× speed up), we achieve lower robust accuracy in all cases. Notably, we won* ***first place*** *out of 1681 teams in CVPR 2021 White-box Adversarial Attacks on Defense Models competitions with this method. Code is available at: https://github.com/liuye6666/adaptive_auto_attack*

1. Introduction

Despite the breakthroughs for a wide range of fields, deep neural networks (DNNs) [21, 23, 42, 52, 57] have been shown high vulnerabilities to adversarial examples. For instance, inputs added with human-imperceptible perturbations can deceive DNNs to output unreasonable predictions [4, 11, 14–16, 30, 33, 50, 60–62]. To tackle this issue, various adversarial defense methods [9, 17, 18, 49] have been proposed to resist against malicious perturbations. Unfortunately, these defense methods could be broken by more advanced attack methods [7, 12, 44, 45], making it difficult to identify the state-of-the-art. Therefore, we urgently need a practical evaluation method to judge the adversarial robustness of different defense strategies.

Robustness evaluation can be defined as looking for defense models' lower bound of robustness given a budget number of iterations and a test dataset [7]. White-box adversarial attacks on defense models are crucial for testing adversarial robustness. Among these methods, widely-used random sampling has been proven effective in generating diverse **starting points** for attacks in a large-scale study [7, 20, 31, 43]. In general, there are two kinds of random sampling strategies. From the perspective of input space, given an original image $\boldsymbol{x}$ of label y and a random perturbation $\boldsymbol{\zeta}$ sampled from uniform distributions, the starting point is $\boldsymbol{x_{st}} = \boldsymbol{x} + \boldsymbol{\zeta}$, *e.g.*, Projected Gradient Descent (PGD) [31]. From the perspective of output space, given a classifier f and a randomly sampled direction of diversification $\boldsymbol{w_d}$, evaluators generate starting points by maximizing the change of output, *i.e.*, $\boldsymbol{w_d^\intercal} f(\boldsymbol{x})$. Intuitively, random sampling strategy is sub-optimal since it is model-agnostic.

Comprehensive statistics on random sampling are conducted to verify whether it is sub-optimal. In other words, we want to study whether adversarial examples, *i.e.*, images that successfully fooling the victim's models, to a specific defense model follow some regularities in their starting points. Statistic results on input space show that adversarial examples' starting points are actually random. This is reasonable because starting points are highly dependent on their corresponding input images, and input images are randomly distributed in high-dimensional space. Different from input space, statistics results in output space show

*Corresponding author

that the direction of diversification $\boldsymbol{w_d}$ to a specific defense model follows some regularities. Specifically, $\boldsymbol{w_d}$ is not uniformly distributed but with a model-specific bias in the positive/negative direction. Therefore, random sampling in the output space cannot obtain a good starting point, which may slow down the evaluation. To speed up the evaluation, we propose an Adaptive Direction Initialization (ADI) strategy in this paper. ADI firstly adopts an observer to record the direction of diversification of adversarial examples at the first restart. Then, based on these directions, ADI introduces a novel way to generate better staring points than random sampling for the following restarts.

In addition to using ADI to accelerate robustness evaluation, we design another strategy named online statistics-based discarding for improving the reliability of existing methods. Currently, the naïve iterative strategy that treats all images evenly and allocates them the same iterations is widely applied to robustness evaluation [6, 7, 20, 31, 32, 43, 51]. However, this strategy is unreasonable because it pays unnecessary efforts to perturb hard-to-attack images. Intuitively, given the budget number of iterations, the more examples we successfully attack, the closer robustness to the lower bound we obtain. Therefore, the number of iterations assigned to hard-to-attack images is a lower priority. Based on our observation that loss values can roughly distinguish the difficulty of attacks, we propose an online statistics-based discarding strategy that automatically identifies and abandons hard-to-attack images. Specifically, we stop perturbing images with considerable difficulties at the beginning of every restart. For remaining images, the same number of iterations are allocated to them. Obviously, online statistics-based discarding strategy makes full use of the number of iterations and increases the chance of perturbing images to adversarial examples. We can further approach the lower bound of robustness based on this reliable strategy. Essentially, speeding up the evaluation is also closely related to improving the reliability because the saved iterations can be used to attack easy-to-attack examples, resulting in lower robust accuracy. By incorporating the above two strategies, a practical evaluation method Adaptive Auto Attack (A^3) is proposed.

To sum up, our main contributions are three-fold: 1) Based on comprehensive statistics, we propose an adaptive direction initialization (ADI) strategy which generates better starting points than random sampling to speed up the robustness evaluation. 2) We propose an online statistics-based discarding strategy that automatically identifies and abandons hard-to-attack images to approach further the lower bound of robustness under the budget number of iterations. 3) Extensive experiments demonstrate the effectiveness and reliability of the method. Particularly, we apply A^3 to nearly 50 widely-used defense models, without parameters adjustment, our method achieves a lower robust accuracy and a faster evaluation.

2. Related Works

Many white-box attacks against defense models have been proposed for robustness evaluation. Fast Adaptive Boundary Attack (FAB) [6] aims at finding the minimal perturbation necessary to change the class of a given input. Projected Gradient Descent (PGD) [31] further improves the evaluation performance by assigning random perturbations as the starting point at every restart. Based on PGD, Gowal *et al*. [20] proposes a MultiTargeted attack (MT) that picks a new target class at each restart. Tashiro [43] provides a more effective initialization strategy to generate diverse starting points. Unfortunately, most of these promising methods overestimate the robustness [2, 12, 44, 45]. Potential reasons for this are improper hyper-parameters tuning and gradient masking [7]. Therefore, we urgently need a practical evaluation method that is convenient (*i.e*., parameter-free), efficient (*i.e*., less iterations), and reliable (*i.e*., approaching the lower bound of robustness).

To address this issue, Croce *et al*. [7] proposes Auto Attack (AA) by integrating four attacks methods. Large-scale studies have shown that AA achieves lower robust test accuracy than existing methods. However, AA is inefficient since it requires a massive number of iterations for robustness evaluation. Instead of adopting an ensemble strategy, Yu *et al*. [51] uses latent features and introduces a unified ℓ_∞-norm white-box attack algorithm LAFEAT to evaluate robustness more reliably. However, the time and space complexity of LAFEAT is unacceptable, as it requires the training of new modules for each defense model. Generally speaking, it is impracticable since training sets are often inaccessible in practical applications.

3. Methodology

3.1. Preliminaries

In this section, we give the background knowledge of adversarial attacks. Given a C-class classifier $f : \boldsymbol{x} \in [0,1]^D \rightarrow \mathbb{R}^C$, where $\boldsymbol{x}$ is original image with label y, model prediction is calculated by:

$$h(\boldsymbol{x}) = \underset{c=1,\ldots,C}{\operatorname{argmax}} f_c(\boldsymbol{x}), \tag{1}$$

where $f_c(\boldsymbol{x})$ refers to the output logits of $\boldsymbol{x}$ on the c-th class. In this paper, we mainly focus on untargeted attacks. The goal of untargeted adversarial attacks is fooling f to missclassify a human-imperceptible adversarial example $\boldsymbol{x_{adv}} = \boldsymbol{x} + \boldsymbol{\delta}$, *i.e*., $h(\boldsymbol{x_{adv}}) \neq y$. By adopting ℓ_∞ distance to evaluate the imperceptibility of $\boldsymbol{\delta}$, *i.e*., $\|\boldsymbol{\delta}\|_\infty \leq \epsilon$, the constrained optimization problem is defined as:

$$\arg\max \mathcal{L}\left(f\left(\boldsymbol{x_{adv}}\right), y\right) \quad s.t. \left\|\boldsymbol{x_{adv}} - \boldsymbol{x}\right\|_\infty \leq \epsilon. \tag{2}$$

In the scenario of white-box adversarial attacks, attackers can access all information of the victims' model. In this case, one of the most popular methods is PGD [31]. Specifically, PGD calculates the gradient at iteration t:

$$\boldsymbol{g^t} = \nabla_{\boldsymbol{x}^t_{\boldsymbol{adv}}} \mathcal{L}\left(f\left(\boldsymbol{x}^{\boldsymbol{t}}_{\boldsymbol{adv}}\right), y\right), \tag{3}$$

where $\boldsymbol{x}^{\boldsymbol{t}}_{\boldsymbol{adv}}$ is the adversarial example at iteration t and the starting point $\boldsymbol{x_{st}}$ is generated as:

$$\boldsymbol{x_{st}} = \boldsymbol{x} + \boldsymbol{\zeta}, \tag{4}$$

where $\boldsymbol{\zeta}$ is a random perturbation sampled from a uniform distribution $U\left(-\epsilon, \epsilon\right)^D$. Then, PGD generates an adversarial example by performing the iterative update:

$$\boldsymbol{x}^{\boldsymbol{t+1}}_{\boldsymbol{adv}} = P_{\boldsymbol{x},\epsilon}\left(\boldsymbol{x}^{\boldsymbol{t}}_{\boldsymbol{adv}} + \eta^t \cdot sign(\boldsymbol{g^t})\right), \tag{5}$$

where η^t is the step size at iteration t and $\boldsymbol{x}^0_{\boldsymbol{adv}} = \boldsymbol{x_{st}}$, function $P_{\boldsymbol{x},\epsilon}\left(\cdot\right)$ clips the input to the ϵ-ball of $\boldsymbol{x}$. To accurately evaluate the robustness of defense models, PGD usually adopts multiple restarts. At each restart, starting points are randomly sampled from the perturbation space.

To further improve the diversity of starting points, Tashiro *et al*. [43] propose ODI to find starting points by maximizing the change in the output space. To be specific, given a random direction of diversification $\boldsymbol{w_d}$ sampled from uniform distributions $U\left(-1, 1\right)^C$, ODI firstly calculates a normalized perturbation vector as follows:

$$\boldsymbol{v}\left(\boldsymbol{x}, f, \boldsymbol{w_d}\right) = \frac{\nabla_{\boldsymbol{x}} \boldsymbol{w}_{\boldsymbol{d}}^{\intercal} f\left(\boldsymbol{x}\right)}{\left\|\nabla_{\boldsymbol{x}} \boldsymbol{w}_{\boldsymbol{d}}^{\intercal} f\left(\boldsymbol{x}\right)\right\|}, \tag{6}$$

and then generates starting points by maximizing the output change via the following iterative update:

$$\begin{aligned} \boldsymbol{x}^{\boldsymbol{t+1}}_{\boldsymbol{adv}} &= \boldsymbol{x}^{\boldsymbol{t}}_{\boldsymbol{adv}} + \eta_{odi} \cdot \operatorname{sign}\left(\boldsymbol{v}\left(\boldsymbol{x}^t_{\boldsymbol{adv}}, f, \boldsymbol{w_d}\right)\right), \\ \boldsymbol{x}^{\boldsymbol{t+1}}_{\boldsymbol{adv}} &= P_{\boldsymbol{x},\epsilon}\left(\boldsymbol{x}^{\boldsymbol{t+1}}_{\boldsymbol{adv}}\right), \end{aligned} \tag{7}$$

where η_{odi} is the step size for ODI, which is usually set to ϵ, and in this paper we keep the same setting. $\boldsymbol{x}^0_{\boldsymbol{adv}}$ is calculated by Eq. (4). After N_{odi} iterations, *i.e*., number of iteration for initialization, ODI obtains the starting point $\boldsymbol{x_{st}}$:

$$\boldsymbol{x_{st}} = \boldsymbol{x}^{\boldsymbol{N_{odi}}}_{\boldsymbol{adv}}. \tag{8}$$

As same as PGD, ODI performs multiple restarts to achieve lower robust accuracy.

3.2. Motivations

A practical evaluation method should be convenient (*i.e*., parameter-free[1]), efficient (*i.e*., fewer iterations), and reliable (*i.e*., approaching the lower bound of robustness). Although using a large number of iterations, most of the existing methods usually overestimate the robustness. There are two potential reasons for this: a) Despite the effectiveness of generating diverse starting points for attacks, random sampling is sub-optimal since it is model-agnostic. Exploiting random sampling to generate starting points will slow down the robustness evaluation, and b) The widely adopted naïve iterative strategy, *i.e*., assigning the same number of iteration to all test examples, is unreasonable. Intuitively, naïve iterative strategy pays unnecessary efforts to perturb hard-to-attack images.

[1]Following [7], 'parameter-free' indicates that we do not need the fine-tuning of parameters for every new defense.

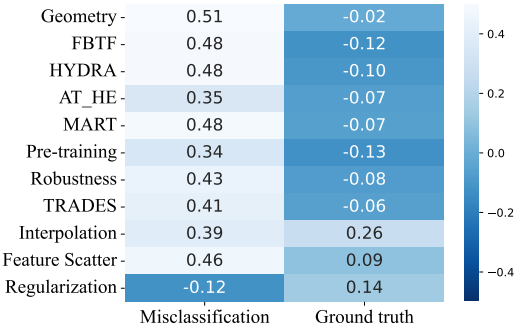

Figure 1. Quantitative statistical results of the diversified direction $\boldsymbol{w_d}$ of adversarial examples on 11 models. The diversified direction $\boldsymbol{w_d}$ for all models disobey uniform distribution.

To verify the above two points, we perform a comprehensive statistical analysis of some white-box adversarial attacks methods against defense models.

Random sampling is sub-optimal. Considering Eq. (8) and Eq. (4), random sampling is widely used to generate diverse starting points for attacks in the input space (*e.g*., PGD), and output space (*e.g*., ODI). It is worth noting that there is no need to do statistics on random sampling in the input space, since the starting points are highly dependent on the corresponding input images and the input images are randomly distributed in the high-dimensional space of the input space. Therefore, we mainly focus on the output space.

As shown in Eq. (7), the noise in the output space is determined by $\boldsymbol{w_d}$. Therefore, to verify the unreasonableness of random sampling in the output space, we analyze $\boldsymbol{w_d}$ of adversarial examples (*i.e*., examples that were successfully attacked) and explore what kind of $\boldsymbol{w_d}$ is conducive. Specifically, we adopt ODI parameterized by number of restarts $R = 50$, $N_{odi} = 7$ and $\eta_{odi} = \epsilon$. The number of iterations for attacks at each restarts N_{atk} is set to 30, and the step size at t-th iteration is formulated as follows:

$$\eta^t = \frac{1}{2}\epsilon \cdot \left(1 + cos\left(\frac{t \bmod N_{atk}}{N_{atk}}\pi\right)\right). \tag{9}$$

Then we use ODI to attack 11 defense models, including Geometry [59], FBTF [47], HYDRA [39],

AT_HE [35], MART [46], Pre-training [22], Robustness [13], TRADES [56], Interpolation [55], Feature Scatter [54], Regularization [25]. Among adversarial examples against different models, we summarize statistic results of $\boldsymbol{w_d}$ in Fig. 1. The 1st and 2nd columns give mean values of $\boldsymbol{w_d}$ at the $\hat{y}$-th (the misclassification label), *i.e.*, $\overline{\boldsymbol{w}_{\boldsymbol{d}}^{\hat{y}}}$, and the y-th (the ground truth), *i.e.*, $\overline{\boldsymbol{w}_{\boldsymbol{d}}^{y}}$.

From Fig. 1, we have the following observations. Firstly, for adversarial examples, their direction of diversification $\boldsymbol{w_d}$ disobeys uniform distribution. Secondly, there is a model-specific positive/negative bias in the direction of diversification. Mainly, there are three kinds of biases: **a**) $\overline{\boldsymbol{w}_{\boldsymbol{d}}^{y}} < 0$, $\overline{\boldsymbol{w}_{\boldsymbol{d}}^{\hat{y}}} > 0$, **b**) $\overline{\boldsymbol{w}_{\boldsymbol{d}}^{y}} > 0$, $\overline{\boldsymbol{w}_{\boldsymbol{d}}^{\hat{y}}} > 0$, and **c**) $\overline{\boldsymbol{w}_{\boldsymbol{d}}^{y}} > 0$, $\overline{\boldsymbol{w}_{\boldsymbol{d}}^{\hat{y}}} < 0$. Considering Eq. (7), for **a**, $f_y\left(\boldsymbol{x_{st}}\right)$ and $f_{\hat{y}}\left(\boldsymbol{x_{st}}\right)$ will decrease and increase respectively, which satisfies the goal of adversarial attacks. For **b**, both of $f_y\left(\boldsymbol{x_{st}}\right)$ and $f_{\hat{y}}\left(\boldsymbol{x_{st}}\right)$ will increase. For **c**, $f_y\left(\boldsymbol{x_{st}}\right)$ will increase and $f_{\hat{y}}\left(\boldsymbol{x_{st}}\right)$ will decrease. Obviously, cases **b** and **c** are counter-intuitive, a potential reason is that these defense models adopt gradient masks [34]. Based on these observations, random sampling is sub-optimal since it is model-agnostic. Adopting it to generate starting points hinders the algorithms from approaching the lower bound of robustness rapidly. For more detailed statistical results of $\boldsymbol{w_d}$, please refer to the Appendix. B.

Limitations of Naïve iterative strategy. Most of the existing methods adopt the naïve iterative strategy, *i.e.*, treats all images evenly. However, based on two intuitions: (1) Images vary in difficulty to perturb them to adversarial examples, and (2) The higher the difficulty, the more iterations are needed to perturb them. Naïve iterative strategy is impractical because it pays unnecessary efforts to perturb hard-to-attack images. In order to successfully attack more images and get closer to the lower bound of robustness with the budget number of iterations, the number of iterations assigned to hard-to-attack images is a lower priority. Therefore, we need a method that can roughly distinguish hard-to-attack images and easy-to-attack images to allocate the budget number of iterations reasonably.

Intuitively, loss function values can roughly reflect the difficulty of perturbing an image to an adversarial example. Multiple loss functions $\mathcal{L}(\cdot)$ can be used for attacks, including cross-entropy loss and margin loss defined as $max_{c\neq y} f_c(\boldsymbol{x}) - f_y(\boldsymbol{x})$. In this paper, we use the margin loss and define hard-to-attack images as images that cannot be successfully attacked even after 2000 iterations. The other images that are successfully attacked, we define as easy-to-attack images.

To verify that loss values can distinguish between hard-to-attack and easy-to-attack images, given 2000 iterations for attacks, we first use ODI to attack all images of 5 models (including, AWP [48], FAT [58], Proxy [38], OAAT [1] and RLPE [41]), and then identify and mark the easy-to-attack images of each model. Finally, we attack the 5 models again with ODI to record the percentile of loss values in descending order for easy-to-attack images in the process of attacks. We visualized the statistical results in Fig. 2.

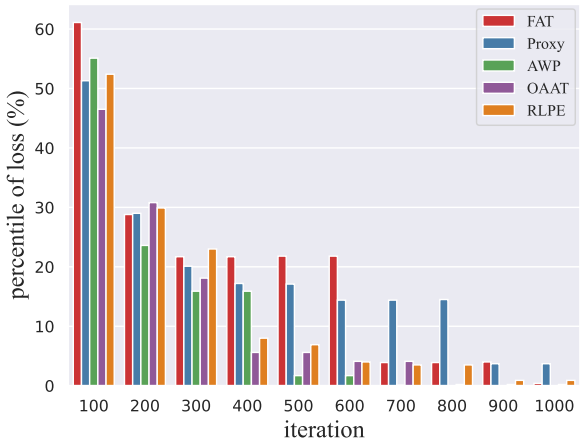

Figure 2. Quantitative statistical results of loss percentile of easy-to-attack images. As the number of iterations increases, the loss percentile of easy-to-attack images continually decrease.

From this figure, we have the following observations. As the number of iterations increases, the loss percentiles of easy-to-attack images continually decrease. Besides, the loss percentiles of easy-to-attack images always take a higher position than that of most hard-to-attack images. Take easy-to-attack image as an example, when the number of iterations reaches 100, the percentile of loss ranks in the top 60%, as the number of iterations increases, it decreases to the top 5% or even the top 0.1%. In other words, loss values of easy-to-attack and hard-to-attack images are not randomly distributed, and the loss values of hard-to-attack images are more likely to be small. We can distinguish between these images according to loss value. Based on these observations, to make full use of the budget number of iterations, we can automatically abandon hard-to-attack images with an increasing proportion according to loss values during the process of attacks.

3.3. Adaptive Direction Initialization

Inspired by the above analysis of random sampling in Sec. 3.2, we propose a method named Adaptive Direction Initialization (ADI) to generate better directions than random sampling to initialize the attacks. Specifically, ADI has two steps: *useful directions observer* and *adaptive directions generation*.

For *useful directions observer* step, ADI first adopts random sampling to generate the direction of diversification, *i.e.*, $\boldsymbol{w_d} = U\left(-1, 1\right)^C$. Then ADI uses starting points obtained by Eq. (8) to initialize PGD attacks and obtains adversarial examples crafted by PGD. We denote W as a set

containing $\boldsymbol{w_d}$ of all adversarial examples.

Motivated by Sec. 3.2, for *adaptive directions generation*, ADI adopts the sign of the summed $\boldsymbol{w_d}$ in W as prior knowledge to generate the adaptive direction $\boldsymbol{w_a}$:

$$\boldsymbol{\kappa}_c(W) = sign\left(\sum_{\boldsymbol{w_d}\in W} \boldsymbol{w_d^c}\right), \tag{10}$$

where $\boldsymbol{\kappa}_c(W)$ is the prior knowledge of generating $\boldsymbol{w_a^c}$, *i.e.*, the c-th dimension of $\boldsymbol{w_a}$. With the help of $\boldsymbol{\kappa}(\cdot)$, ADI generates the y-th components of $\boldsymbol{w_a}$ as:

$$\boldsymbol{w_a^y} = \begin{cases} \boldsymbol{w_a^y} \sim U\left(-0.5, 0.1\right), & \boldsymbol{\kappa}_y(\cdot) < 0, \\ \boldsymbol{w_a^y} \sim U\left(-0.1, 0.5\right), & \boldsymbol{\kappa}_y(\cdot) > 0. \end{cases} \tag{11}$$

To improve the effectiveness of $\boldsymbol{w_a}$, ADI randomly selects a label $\mathfrak{y}$ to follow the sign of symbol of $\boldsymbol{\kappa}_{\mathfrak{y}}(\cdot)$:

$$\boldsymbol{w_a^{\mathfrak{y}}} = \begin{cases} -0.8, & \boldsymbol{\kappa}_{\mathfrak{y}}(\cdot) < 0, \\ 0.8, & \boldsymbol{\kappa}_{\mathfrak{y}}(\cdot) > 0. \end{cases} \tag{12}$$

Notably, our method is insensitive to $\boldsymbol{w_a^{\mathfrak{y}}}$ experimentally. For simplicity, we set $\boldsymbol{w_a^{\mathfrak{y}}} = \pm 0.8$. For the remaining dimensions of adaptive direction $\boldsymbol{w_a}$, ADI calculates them as:

$$\boldsymbol{w_a^i} \sim U\left(-1, 1\right)^{C-2},\ i \in \mathcal{T}, \tag{13}$$

where the set $\mathcal{T} = \{1, ..., C\}\backslash\{y, \mathfrak{y}\}$ contains all classes other than y and $\mathfrak{y}$. Compared with random sampling adopted by ODI, the adaptive direction generated by ADI is guided by prior knowledge, *i.e.*, the direction of diversification of adversarial examples. Furthermore, ADI randomly generates the remaining $C-2$ dimensions of the adaptive direction to improve the diversity of starting points.

3.4. Online Statistics-based Discarding Strategy

In light of approaching the lower bound of robustness with the budget number of iterations, we propose a novel iterative strategy named Online Statistics-based Discarding (OSD). According to the observation in Sec. 3.2. OSD adopts loss values to distinguish between hard-to-attack and easy-to-attack images. OSD first sorts test images in descending order by the corresponding loss values at the beginning of every restart, and then discards hard-to-attack images, *i.e.*, stopping perturbing images with small loss values. Particularly, given an initial discarding rate ϕ and an discarding increment ι, the discarding rate at the r-th restart is formulated as follows:

$$\varsigma^r = \phi + r \times \iota. \tag{14}$$

For the remaining images, OSD assigns the same number of iterations to them. Intuitively, to further increase attack success rate, OSD allocates more iterations to remaining images at restart r than previous restart. Concretely, given an initial number of iteration γ for attacks and an iteration increment ν, the number of iterations for attacks at the r-th restart is computed as follows:

$$N_{atk}^r = \gamma + r \times \nu. \tag{15}$$

Compared with naïve iterative strategy, OSD makes full use of the budget number of iterations by automatically identifying and abandoning hard-to-attack images. In addition, by allocating various number of iterations for attacks at different restarts, OSD helps to further approach the lower bound of adversarial robustness.

Algorithm 1: Adaptive Auto Attack (A^3)

Inputs: norm bound ϵ, the number of iteration for initialization N, step sizes η, the number of iteration for attacks at the r-th restart N_{atk}^r, attack iteration step sizes η_{atk}, number of restarts R, test dataset $\mathcal{I}$

Outputs: Adversarial example $\boldsymbol{x_{adv}^{t+1}}$

for $r = 0 \rightarrow R$ **do**
  Update the test dataset $\mathcal{I}$ by OSD
  for $\boldsymbol{x}$ *in* $\mathcal{I}$ **do**
    Sample $\boldsymbol{\zeta}$ from $U\left(-\epsilon, \epsilon\right)^D$
    $\boldsymbol{x_{st}} = \boldsymbol{x} + \zeta$
    if $r = 0$ **then**
      Sample $\boldsymbol{w_d}$ from $U\left(-1, 1\right)^C$
    else
      $\boldsymbol{w_d} \leftarrow \boldsymbol{w_a}$
    `/* ADI */`
    for $n = 0$ *to* N **do**
      Compute $\boldsymbol{x_{adv}^{n+1}}$ by Eq. (7)
    for $t = 0$ *to* N_{atk}^r **do**
      Compute $\boldsymbol{x_{adv}^{t+1}}$ by Eq. (5)
    if *r=0* **then**
      Compute $\boldsymbol{w_a}$ by Eqs. (11) to (13).
    if $\boldsymbol{x_{adv}^{t+1}}$ *is an adversarial example* **then**
      return $\boldsymbol{x_{adv}^{t+1}}$

3.5. Adaptive Auto Attack

We integrate above two strategies to form a practical evaluation method Adaptive Auto Attack (A^3). Firstly, A^3 is convenient since we do not need the fine-tuning of parameters for every new defense models. Secondly, A^3 is efficient. For the gradient-based optimization to craft adversarial examples, unlike random sampling, our method A^3 generates adaptive directions for each model and provides better starting points to speed up the evaluation. Thirdly, A^3 is reliable. By discarding hard-to-attack images online and adjusting iterations for attacks adaptively, our method A^3 makes full use of a budget number of iterations and further approaches the lower bound of adversarial robustness.

Compared with the mainstream method AA, our parameter-free A^3 is a more efficient and reliable protocol for robustness evaluation. The algorithm of Adaptive Auto Attack is summarized in Algorithm 1.

4. Experiments

We conduct comprehensive experiments to evaluate the practicability of our method. Specifically, five baselines are included: PGD [31], ODI [43], MT-PGD [20], I-FGSM [29] and AA [7]. Nearly 50 ℓ_∞-defense models with 8 different architectures are chosen from recent conferences. To be specific, the evaluation is performed on 35 and 12 defense models trained on CIFAR-10 and CIFAR-100 [27] datasets, respectively. Notably, for fair comparisons, the step size is calculated by Eq. (9) for all attack methods in the experiments. We use margin loss for all attack methods.

Following AA, we adopt robust accuracy (**acc**) to reflect evaluation reliability. A robustness evaluation method is considered reliable if it can better downgrade the model's classification accuracy. In our experiments, we assume computational complexity of all methods in each iteration is similar. So the evaluation efficiency of each method can be reflected by the total number of iterations. For simplicity, we test the forward propagation ("$\rightarrow$") and backward propagation ("$\leftarrow$") to indicate the evaluation efficiency of different methods.

4.1. Comparisons with State-of-the-Art Attacks

To comprehensively validate the efficiency and reliability of our method, we compare AA, PGD, ODI with our A^3 on nearly 50 defense models. The evaluation results are shown in Tab. 1.

Setup. Following the setup of ODI and PGD, 100 iterations are allocated for each image (4 restarts, 25 iterations for attacks at each restart), $N_{odi} = 2$. For AA, the standard version[2] is adopted. For our A^3, initial number of iterations for attacking γ is set to 25, and the iteration increment ν is 5. The number of iteration for initialization $N = 7$. When the number of iterations reaches 50, we keep it unchanged to save the budget number of iterations. Initial discarding rate $\phi = 0$ and discarding increment $\iota = 0.1$, when the discarding rate reaches to 0.9, we gradually increase it to 0.97 at an interval of $\iota = 0.035$.

Results. As can be seen in Tab. 1, A^3 uses the same parameters for all defense models and achieves lower robust accuracy than AA in all cases, downgrading **acc** by 0.1% on average. Besides, our method achieves a faster evaluation, on average, $10.4\times$ speed up for forward propagation, and $5.4\times$ for backward propagation. In general, the nature of parameter-free, reliability, and efficiency enables A^3 to be a practical method for robustness evaluation. For the results on other datasets, network architectures and metrics (*i.e.*, ℓ_2-norm), please refer to the Appendix. C.

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

4.2. Ablation Study

Efficacy of adaptive direction initialization. To evaluate the effectiveness of ADI, we design another variant called Reverse Adaptive Direction Initialization (R-ADI) which adopts the reverse direction of the adaptive direction. For all methods, we allocate 150 iterations for each image (5 restarts and 30 iterations at each restart.), and N=10.

The comparison results among R-ADI, ODI, and ADI are reported in Tab. 2. As can be seen, compared to ODI, our ADI achieves better attack performance in all cases. The result indicates the initial direction does affect the performance. Compared with uniformly generating initial direction, our ADI can generate model-specific initial direction and help to obtain better performance. In general, R-ADI achieves the worst performance in all cases. One possible reason is that R-ADI chooses a bad initial direction, which hinders the performance.

Efficacy of online statistics-based discarding strategy. To verify the effect of OSD, we compare the robust accuracy curves of the defense models under the attack of ADI, ADI+OSD (A^3), and AA. For our methods ADI and ADI+OSD, the setup is the same as in Sec. 4.1.

Note that ADI+OSD denotes the Online Statistics-based Discarding strategy is applied on ADI. The result is shown in the Fig. 3. As can be observed, it costs more iterations for AA to achieve the same robust accuracy of ADI and ADI+OSD in all cases. Meanwhile, to achieve higher attack performance, ADI and AA require a large number of additional iterations, but ADI+OSD requires fewer iterations. The curves reveal the efficiency of our ADI+OSD, especially for reliable attacks.

Efficacy on other robustness evaluation methods. In this section, we study the effect of integrating ADI and OSD into different robustness evaluation methods. For all methods, we allocate 500 iterations for each image (5 restarts and 100 iterations for attacks at each restart), for ODI method, N_{odi}=10. For MT-PGD, the number of multiple targets is 3. For our A^3, $N = 10$. The other experimental settings are the same as in Sec. 4.1.

Tab. 3 shows the robust accuracy (%) of defense models under the attack of different attack methods [20, 29, 31, 43] integrated with our modules (ADI and OSD). It shows that ADI and OSD can be integrated into multiple attack methods and effectively improve their performance.

4.3. Result of Competition

With our proposed A^3, we participated in the CVPR 2021 White-box Adversarial Attacks on Defense Models competition launched by Alibaba Group and Tsinghua University. To evaluate performance fairly, all codes were

CIFAR-10 Defense Method	Model	Clean	Nominal	PGD	ODI	AA			A^3			Δ
		acc	acc	acc	acc	acc	→	←	acc	→	←	acc
ULAT [19]†	WRN-70-16	91.10	65.87	66.75	66.06	65.88	51.20	12.90	**65.78** ↓ **0.10**	**4.49**(**11.40**×)	**2.20**(**5.86**×)	↓ 0.09
Fixing Data [36]	WRN-70-16	88.54	64.20	65.10	64.46	64.25	50.82	12.59	**64.19** ↓ **0.06**	**4.41**(**11.52**×)	**2.17**(**5.81**×)	↓ 0.01
ULAT [19]†	WRN-28-10	89.48	62.76	63.63	63.01	62.80	49.62	12.30	**62.70** ↓ **0.10**	**4.28**(**11.58**×)	**2.10**(**5.85**×)	↓ 0.05
Fixing Data [36]	WRN-28-10	87.33	60.73	61.64	61.09	60.75	47.98	11.91	**60.66** ↓ **0.09**	**4.14**(**11.59**×)	**2.04**(**5.83**×)	↓ 0.07
RLPE [41]†	WRN-34-15	86.53	60.41	61.25	60.69	60.41	47.53	11.82	**60.31** ↓ **0.10**	**4.12**(**11.52**×)	**2.02**(**5.84**×)	↓ 0.10
AWP [48]†	WRN-28-10	88.25	60.04	60.55	60.23	60.04	47.20	11.70	**59.98** ↓ **0.06**	**4.09**(**11.54**×)	**2.01**(**5.82**×)	↓ 0.06
RLPE [41]†	WRN-28-10	89.46	59.66	60.78	59.88	59.66	47.09	11.72	**59.51** ↓ **0.15**	**4.10**(**11.49**×)	**2.00**(**5.85**×)	↓ 0.15
Geometry [59]†‡	WRN-28-10	89.36	59.64	60.17	59.59	59.64	47.10	11.67	**59.53** ↓ **0.11**	**4.10**(**11.49**×)	**2.00**(**5.85**×)	↓ 0.11
RST [5]†	WRN-28-10	89.69	62.50	60.64	59.44	59.53	47.10	11.70	**59.42** ↓ **0.11**	**4.10**(**11.49**×)	**2.01**(**5.82**×)	↓ 3.08
Proxy [38]†	WRN-34-10	85.85	59.09	60.51	59.94	59.09	46.70	11.60	**58.99** ↓ **0.10**	**4.04**(**11.56**×)	**1.98**(**5.86**×)	↓ 0.10
OAAT [1]	WRN-34-10	85.32	58.04	58.84	58.25	58.04	45.64	11.34	**57.98** ↓ **0.06**	**3.99**(**11.43**×)	**1.96**(**5.76**×)	↓ 0.06
HYDRA [39]†	WRN-28-10	88.98	59.98	58.27	57.60	57.14	45.20	11.20	**57.06** ↓ **0.08**	**3.91**(**11.56**×)	**1.92**(**5.83**×)	↓ 2.92
ULAT [19]	WRN-70-16	85.29	57.20	57.90	57.48	57.20	45.20	11.20	**57.08** ↓ **0.12**	**3.90**(**11.59**×)	**1.92**(**5.83**×)	↓ 0.12
ULAT [19]	WRN-34-20	85.64	56.82	57.40	57.00	56.86	44.96	11.18	**56.76** ↓ **0.10**	**3.88**(**11.60**×)	**1.90**(**5.89**×)	↓ 0.10
MART [46] †	WRN-28-10	87.50	65.04	58.09	56.80	56.29	44.60	11.10	**56.20** ↓ **0.09**	**3.86**(**11.55**×)	**1.89**(**5.93**×)	↓ 8.84
Pre-training [22]†	WRN-34-10	87.11	57.40	56.43	55.32	54.92	43.40	10.80	**54.76** ↓ **0.16**	**3.73**(**11.64**×)	**1.83**(**5.90**×)	↓ 2.64
Proxy [38]	ResNet-18	84.38	55.60	56.31	54.98	54.43	43.21	10.71	**54.35** ↓ **0.08**	**3.75**(**11.52**×)	**1.84**(**5.81**×)	↓ 1.25
AT_HE [35]	WRN-34-20	85.14	62.14	55.33	54.21	53.74	43.00	10.69	**53.67** ↓ **0.07**	**3.68**(**11.68**×)	**1.81**(**5.91**×)	↓ 8.47
LBGAT [8]‡	WRN-34-20	88.70	53.57	54.69	53.90	53.57	43.11	10.58	**53.46** ↓ **0.11**	**3.69**(**11.63**×)	**1.81**(**5.80**×)	↓ 0.11
FAT [58]	WRN-34-10	84.52	53.51	54.46	53.83	53.51	42.94	10.54	**53.42** ↓ **0.09**	**3.68**(**11.72**×)	**1.81**(**5.83**×)	↓ 0.09
Overfitting [37]	WRN-34-20	85.34	58.00	55.21	53.95	53.42	42.10	10.50	**53.33** ↓ **0.09**	**3.66**(**11.50**×)	**1.80**(**5.83**×)	↓ 4.67
Self-adaptive [24]‡	WRN-34-10	83.48	58.03	54.39	53.62	53.33	42.10	10.50	**53.20** ↓ **0.13**	**3.66**(**11.50**×)	**1.80**(**5.83**×)	↓ 4.83
TRADES [56]‡	WRN-34-10	84.92	56.43	54.02	53.31	53.08	42.00	10.40	**53.01** ↓ **0.07**	**3.63**(**11.57**×)	**1.78**(**5.75**×)	↓ 3.42
LBGAT [8]‡	WRN-34-10	88.22	52.86	54.37	53.26	52.86	41.80	10.30	**52.76** ↓ **0.10**	**3.64**(**11.48**×)	**1.79**(**5.79**×)	↓ 0.10
OAAT [1]	ResNet-18	80.24	51.06	51.69	51.28	51.06	40.54	10.21	**51.02** ↓ **0.04**	**3.51**(**11.53**×)	**1.72**(**5.93**×)	↓ 0.04
SAT [40]	WRN-34-10	86.84	50.72	52.95	51.38	50.72	40.14	10.01	**50.62** ↓ **0.10**	**3.50**(**11.46**×)	**1.72**(**5.81**×)	↓ 0.10
Robustness [13]	ResNet-50	87.03	53.29	52.19	50.14	49.21	39.10	9.80	**49.16** ↓ **0.05**	**3.42**(**11.43**×)	**1.68**(**5.83**×)	↓ 4.13
YOPO [53]	WRN-34-10	87.20	47.98	47.11	45.57	44.83	35.60	9.00	**44.77** ↓ **0.06**	**3.09**(**11.52**×)	**1.52**(**5.92**×)	↓ 3.21
MMA [10]	WRN-28-4	84.36	47.18	47.78	42.42	41.51	33.30	8.60	**41.27** ↓ **0.24**	**3.17**(**10.50**×)	**1.66**(**5.19**×)	↓ 5.85
DNR [28]	ResNet-18	87.32	40.41	42.15	41.01	40.41	32.81	8.72	**40.26** ↓ **0.15**	**2.81**(**11.67**×)	**1.38**(**6.32**×)	↓ 5.93
CNL [3]‡	ResNet-18	81.30	79.67	40.26	40.23	40.22	32.70	8.70	**39.83** ↓ **0.39**	**2.74**(**11.93**×)	**1.34**(**6.49**×)	↓ 39.84
Feature Scatter [54]	WRN-28-10	89.98	60.60	54.63	42.91	36.62	30.00	8.20	**36.31** ↓ **0.33**	**11.02**(**2.72**×)	**5.44**(**1.51**×)	↓ 24.33
Interpolation [55]	WRN-28-10	90.25	68.70	66.72	49.35	36.45	30.00	8.50	**36.21** ↓ **0.24**	**11.21**(**2.64**×)	**5.52**(**1.54**×)	↓ 32.32
Sensible [26]	WRN-34-10	91.51	57.23	56.04	43.15	34.22	28.20	7.80	**34.00** ↓ **0.22**	**10.66**(**2.65**×)	**5.25**(**1.49**×)	↓ 23.23
Regularization [25]	ResNet-18	90.84	77.68	52.77	19.73	1.35	3.10	2.30	**0.89** ↓ **0.46**	**2.24**(**1.38**×)	**1.09**(**2.11**×)	↓ 76.79

CIFAR-100 Defense Method	Model	Clean	Nominal	PGD	ODI	AA			A^3			Δ
		acc	acc	acc	acc	acc	→	←	acc	→	←	acc
ULAT [19]†	WRN-70-16	69.15	36.88	38.64	37.41	36.88	29.84	7.42	**36.86** ↓ **0.02**	**2.56**(**11.64**×)	**1.25**(**5.92**×)	↓ 0.02
Fixing Data [36]	WRN-70-16	63.56	34.64	35.95	34.98	34.64	28.02	6.96	**34.55** ↓ **0.09**	**2.38**(**11.76**×)	**1.16**(**6.00**×)	↓ 0.04
Fixing Data [36]	WRN-28-10	62.41	32.06	33.39	32.36	32.06	25.53	6.48	**32.00** ↓ **0.06**	**2.24**(**11.38**×)	**1.10**(**5.90**×)	↓ 0.06
OAAT [1]	WRN-34-10	65.73	30.35	31.62	30.93	30.35	24.34	6.11	**30.31** ↓ **0.04**	**2.18**(**11.14**×)	**1.07**(**5.70**×)	↓ 0.04
LBGAT [8]‡	WRN-34-20	62.55	30.20	31.65	30.49	30.20	23.97	6.10	**30.12** ↓ **0.08**	**2.16**(**11.11**×)	**1.05**(**5.80**×)	↓ 0.08
ULAT [19]	WRN-70-16	60.86	30.03	31.03	30.41	30.03	23.93	6.09	**29.99** ↓ **0.04**	**2.13**(**11.23**×)	**1.04**(**5.86**×)	↓ 0.04
LBGAT [8]‡	WRN-34-10	60.64	29.33	30.56	29.63	29.33	23.21	5.94	**29.18** ↓ **0.15**	**2.11**(**11.00**×)	**1.03**(**5.77**×)	↓ 0.15
AWP [48]	WRN-34-10	60.38	28.86	30.70	29.45	28.86	23.01	5.84	**28.78** ↓ **0.08**	**2.10**(**10.96**×)	**1.02**(**5.72**×)	↓ 0.08
Pre-training [22]	WRN-28-10	59.23	28.42	30.56	29.13	28.42	22.74	5.73	**28.31** ↓ **0.11**	**2.08**(**10.93**×)	**1.02**(**5.61**×)	↓ 0.11
OAAT [1]	ResNet18	62.02	27.14	27.90	27.47	27.14	21.74	5.61	**27.09** ↓ **0.05**	**2.34**(**9.29**×)	**1.15**(**4.88**×)	↓ 0.05
SAT [40]	WRN-34-10	62.82	24.57	26.69	25.43	24.57	19.70	5.10	**24.51** ↓ **0.06**	**1.90**(**10.36**×)	**0.93**(**5.48**×)	↓ 0.06
Overfitting [37]	PAResNet-18	53.83	18.95	20.15	19.39	18.95	15.28	4.00	**18.90** ↓ **0.05**	**1.64**(**9.32**×)	**0.80**(**5.00**×)	↓ 0.05

Table 1. Comparison of robust accuracy (%) under the attack of A^3, PGD, ODI, and AutoAttack(AA) across various defense strategies. The "acc" column shows the robust accuracies of different models. The "Nominal" column shows the robust accuracies reported by defense models. The "Δ" column shows the difference between the robust accuracies of "Nominal" and A^3. The "→" column shows the iteration number of forward propagation (million), while the "←" column shows the iteration number of backward propagation (million). Models marked with † were additionally trained with unlabeled datasets. We used ϵ = 8/255 except for models marked with ‡ , which used ϵ = 0.031 as originally reported by the authors. Notably, the "**acc**" column of A^3 shows the difference between the robust accuracies of AA and A^3, the "←" and "→" columns of A^3 show the speedup factors of A^3 relative to AA.

tested with the official adversarial robustness evaluation platform ARES. The competition has three stages. Stage 1 and stage 2 employed 13 and 2 defense models trained on CIFAR-10 and ImageNet datasets, respectively. The ℓ_∞ constraint is set to $8/255$, $4/255$ on CIFAR-10 and ImageNet, respectively. Methods for all participants tested on CIFAR-10 dataset and $1,000$ random images of the ImageNet validation set to evaluate the final score. The evalu-

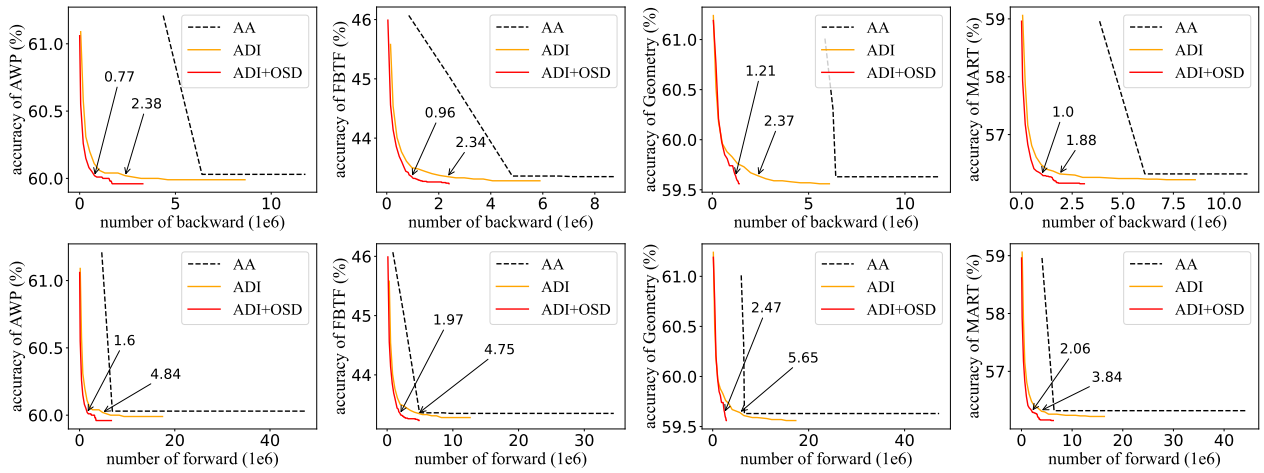

Figure 3. Comparisons of performance of ADI+OSD, ADI and AA on 4 defenders. For each defender, we separately record the number of forward propagation and back propagation by columns. The horizontal axes show the number of back propagation (top) and forward propagation (bottom). The vertical axes show the percentage of remaining unsuccessful examples. The iteration numbers needed for ADI+OSD and ADI to defeat AA are also marked.

Defense method	**R-ADI**	**ODI**	**ADI**
Geometry	60.00 ↑ 0.09	59.91	**59.73 ↓ 0.18**
RST	59.81 ↑ 0.01	59.80	**59.63 ↓ 0.17**
Proxy	59.65 ↑ 0.25	59.40	**59.22 ↓ 0.18**
HYDRA	57.47 ↑ 0.11	57.36	**57.26 ↓ 0.10**
MART	56.86 ↑ 0.28	56.58	**56.48 ↓ 0.10**
Pre-training	55.48 ↑ 0.37	55.11	**54.96 ↓ 0.15**
Self-adaptive	53.59 ↑ 0.11	53.48	**53.33 ↓ 0.15**
Feature Scatter	40.90 ↑ 2.22	38.68	**38.29 ↓ 0.39**
Interpolation	55.47 ↑ 12.95	42.52	**40.01 ↓ 2.51**
Sensible	41.14 ↑ 3.19	37.95	**37.19 ↓ 0.76**
Regularization	24.82 ↑ 14.83	9.99	**7.75 ↓ 2.24**

Table 2. Comparisons of robust accuracy (%) under attack of ADI, naïve ODI and R-ADI across various defense strategies. R-ADI obtains the worst results because it chose the worst starting points.

Attack method	RLPE	Self-adaptive	OAAT	Feature Scatter
I-FGSM	60.77	54.38	51.74	55.00
I-FGSM+ADI	59.66 ↓ 1.12	53.36 ↓ 1.02	51.09 ↓ 0.65	38.24 ↓ 16.76
I-FGSM+ADI+OSD	**59.50 ↓ 1.28**	**53.24 ↓ 1.14**	**51.01 ↓ 0.73**	**36.70 ↓ 18.30**
PGD	60.65	54.29	51.5	52.98
PGD+ADI	59.61 ↓ 1.04	53.37 ↓ 0.92	51.08 ↓ 0.42	38.08 ↓ 14.90
PGD+ADI+OSD	**59.51 ↓ 1.14**	**53.23 ↓ 1.06**	**51.04 ↓ 0.46**	**36.72 ↓ 16.26**
MT-PGD	60.51	54.08	51.59	50.92
MT-PGD+ADI	60.11 ↓ 0.40	53.69 ↓ 0.39	51.50 ↓ 0.09	38.41 ↓ 12.51
MT-PGD+ADI+OSD	**59.71 ↓ 0.80**	**53.34 ↓ 0.74**	**51.18 ↓ 0.41**	**36.74 ↓ 14.18**
ODI-PGD	59.74	53.49	51.13	38.56
ODI-PGD+ADI	59.65 ↓ 0.09	53.34 ↓ 0.15	51.09 ↓ 0.04	37.92 ↓ 0.64
ODI-PGD+ADI+OSD	**59.51 ↓ 0.23**	**53.26 ↓ 0.23**	**51.00 ↓ 0.13**	**36.75 ↓ 1.17**

Table 3. Effectiveness of the two modules ADI and OSD on 4 different attack methods.

ation score is defined as the average misclassification rate. Finally, we obtained 51.10% score in the final stage and achieved first place among 1681 teams.

5. Conclusion

In this paper, we find that model-agnostic initial directions drawn from uniform distributions result in unsatisfactory robust accuracy, and naïve iterative strategy leads to unreliable attacks. We propose a novel approach called Adaptive Auto Attack (A^3) to address these issues, which adopts adaptive direction initialization and an online statistics-based discarding strategy to achieve efficient and reliable robustness evaluation. Extensive experiments demonstrate the effectiveness of A^3. Particularly, we achieve lower robust accuracy in all cases by consuming much fewer iterations than existing models, *e.g.*, $1/10$ on average ($10\times$ speed up). We won first place out of 1,681 teams in CVPR 2021 White-box Adversarial Attacks on Defense Models competitions with this method.

6. Broader Impacts

Performing reliable robustness evaluation helps distinguish good from bad defenses to resist against the widespread adversarial examples. Our research proposes a practical robustness evaluation method. On the positive side, our method enables us to identify advanced defenses to defend against adaptive attacks, preventing critical safety systems from crashing. On the negative side, malicious users can exploit our method to attack the system, raising a security risk. We will continue to expand the scope of our evaluation for advanced defenses in the future.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No. 62020106008, No. 62122018, No. 61772116, No. 61872064), Sichuan Science and Technology Program (Grant No.2019JDTD0005).

References

[1] Sravanti Addepalli, Samyak Jain, Gaurang Sriramanan, Shivangi Khare, and Venkatesh Babu Radhakrishnan. Towards achieving adversarial robustness beyond perceptual limits. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021. 4, 7

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018. 2

[3] Matan Atzmon, Niv Haim, Lior Yariv, Ofer Israelov, Haggai Maron, and Yaron Lipman. Controlling neural level sets. In *NeurIPS*, 2019. 7

[4] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *SP*, 2017. 1

[5] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C. Duchi, and Percy Liang. Unlabeled data improves adversarial robustness. In *NeurIPS*, 2019. 7

[6] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *ICML*, 2020. 2

[7] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, pages 2206–2216. PMLR, 2020. 1, 2, 3, 6

[8] Jiequan Cui, Shu Liu, Liwei Wang, and Jiaya Jia. Learnable boundary guided adversarial training. In *ICCV*, pages 15721–15730, October 2021. 7

[9] Guneet S. Dhillon, Kamyar Azizzadenesheli, Zachary C. Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Animashree Anandkumar. Stochastic activation pruning for robust adversarial defense. In *ICLR*, 2018. 1

[10] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. MMA training: Direct input space margin maximization through adversarial training. In *ICLR*, 2020. 7

[11] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, 2018. 1

[12] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *CVPR*, 2019. 1, 2

[13] Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. 4, 7

[14] Lianli Gao, Yaya Cheng, Qilong Zhang, Xing Xu, and Jingkuan Song. Feature space targeted attacks by statistic alignment. In *IJCAI*, 2021. 1

[15] Lianli Gao, Qilong Zhang, Jingkuan Song, Xianglong Liu, and Heng Tao Shen. Patch-wise attack for fooling deep neural network. In *ECCV*, 2020. 1

[16] Lianli Gao, Qilong Zhang, Jingkuan Song, and Heng Tao Shen. Patch-wise++ perturbation for adversarial targeted attacks. *CoRR*, abs/2012.15503, 2020. 1

[17] Zhitao Gong, Wenlu Wang, and Wei-Shinn Ku. Adversarial and clean data are not twins. *arXiv preprint arXiv:1704.04960*, 2017. 1

[18] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 1

[19] Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy A. Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *CoRR*, abs/2010.03593, 2020. 7

[20] Sven Gowal, Jonathan Uesato, Chongli Qin, Po-Sen Huang, Timothy A. Mann, and Pushmeet Kohli. An alternative surrogate loss for pgd-based adversarial testing. *arXiv preprint arXiv:1910.09338*, 2019. 1, 2, 6

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[22] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *ICML*, 2019. 4, 7

[23] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 1

[24] Lang Huang, Chao Zhang, and Hongyang Zhang. Self-adaptive training: beyond empirical risk minimization. In *NeurIPS*, 2020. 7

[25] Charles Jin and Martin Rinard. Manifold regularization for adversarial robustness. *CoRR*, abs/2003.04286, 2020. 4, 7

[26] Jungeum Kim and Xiao Wang. Sensible adversarial learning. 2019. 7

[27] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[28] Souvik Kundu, Mahdi Nazemi, Peter A. Beerel, and Massoud Pedram. DNR: A tunable robust pruning framework through dynamic network rewiring of dnns. In *ASPDAC*, pages 344–350, 2021. 7

[29] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *ICLR Workshop*, 2017. 6

[30] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E. Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. In *ICLR*, 2020. 1

[31] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018. 1, 2, 3, 6

[32] Xiaofeng Mao, Yuefeng Chen, Shuhui Wang, Hang Su, Yuan He, and Hui Xue. Composite adversarial attacks. In *AAAI*, 2021. 2

[33] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *CVPR*, 2016. 1

[34] Linh Nguyen, Sky Wang, and Arunesh Sinha. A learning and masking approach to secure learning. In *GameSec*, 2018. 4

[35] Tianyu Pang, Xiao Yang, Yinpeng Dong, Taufik Xu, Jun Zhu, and Hang Su. Boosting adversarial training with hypersphere embedding. In *NeurIPS*, 2020. 4, 7

[36] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan A. Calian, Florian Stimberg, Olivia Wiles, and Timothy A. Mann. Fixing data augmentation to improve adversarial robustness. *CoRR*, abs/2103.01946, 2021. 7

[37] Leslie Rice, Eric Wong, and J. Zico Kolter. Overfitting in adversarially robust deep learning. In *ICML*, 2020. 7

[38] Vikash Sehwag, Saeed Mahloujifar, Tinashe Handina, Sihui Dai, Chong Xiang, Mung Chiang, and Prateek Mittal. Improving adversarial robustness using proxy distributions. *CoRR*, abs/2104.09425, 2021. 4, 7

[39] Vikash Sehwag, Shiqi Wang, Prateek Mittal, and Suman Jana. HYDRA: pruning adversarially robust neural networks. In *NeurIPS*, 2020. 3, 7

[40] Chawin Sitawarin, Supriyo Chakraborty, and David A. Wagner. Improving adversarial robustness through progressive hardening. *CoRR*, abs/2003.09347, 2020. 7

[41] Kaustubh Sridhar, Oleg Sokolsky, Insup Lee, and James Weimer. Robust learning via persistency of excitation. *CoRR*, abs/2106.02078, 2021. 4, 7

[42] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 1

[43] Yusuke Tashiro, Yang Song, and Stefano Ermon. Diversity can be transferred: Output diversification for white- and black-box attacks. *arXiv preprint arXiv:2003.06878*, 2020. 1, 2, 3, 6

[44] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020. 1, 2

[45] Jonathan Uesato, Brendan O'donoghue, Pushmeet Kohli, and Aaron Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *ICML*, 2018. 1, 2

[46] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *ICLR*, 2020. 4, 7

[47] Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020. 3

[48] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *NeurIPS*, 2020. 4, 7

[49] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan L. Yuille. Mitigating adversarial effects through randomization. In *ICLR*, 2018. 1

[50] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L. Yuille. Improving transferability of adversarial examples with input diversity. In *CVPR*, 2019. 1

[51] Yunrui Yu, Xitong Gao, and Cheng-Zhong Xu. Lafeat: Piercing through adversarial defenses with latent features. In *CVPR*, 2021. 2

[52] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 1

[53] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. In *NeurIPS*, pages 227–238, 2019. 7

[54] Haichao Zhang and Jianyu Wang. Defense against adversarial attacks using feature scattering-based adversarial training. In *NeurIPS*, 2019. 4, 7

[55] Haichao Zhang and Wei Xu. Adversarial interpolation training: A simple approach for improving model robustness. 2019. 4, 7

[56] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *ICML*, pages 7472–7482, 2019. 4, 7

[57] Ji Zhang, Jingkuan Song, Yazhou Yao, and Lianli Gao. Curriculum-based meta-learning. In *ACM MM*, pages 1838–1846, 2021. 1

[58] Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan S. Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In *ICML*, pages 11278–11287, 2020. 4, 7

[59] Jingfeng Zhang, Jianing Zhu, Gang Niu, Bo Han, Masashi Sugiyama, and Mohan S. Kankanhalli. Geometry-aware instance-reweighted adversarial training. In *ICLR*, 2021. 3, 7

[60] Qilong Zhang, Xiaodan Li, Yuefeng Chen, Jingkuan Song, Lianli Gao, Yuan He, and Hui Xue. Beyond imagenet attack: Towards crafting adversarial examples for black-box domains. In *ICLR*, 2022. 1

[61] Qilong Zhang, Chaoning Zhang, Chaoqun Li, Jingkuan Song, Lianli Gao, and Heng Tao Shen. Practical no-box adversarial attacks with training-free hybrid image transformation. *CoRR*, abs/2203.04607, 2022. 1

[62] Zhengyu Zhao, Zhuoran Liu, and Martha A. Larson. Towards large yet imperceptible adversarial image perturbations with perceptual color distance. In *CVPR*, 2020. 1