Adversarial Attacks on ML Defense Models Competition

Yinpeng Dong[1,3], Qi-An Fu[1], Xiao Yang[1], Wenzhao Xiang[4], Tianyu Pang[1], Hang Su[1], Jun Zhu[1,3],
Jiayu Tang[3], Yuefeng Chen[2], XiaoFeng Mao[2], Yuan He[2], Hui Xue[2], Chao Li[2],
Ye Liu[5], Qilong Zhang[5], Lianli Gao[5], Yunrui Yu[6], Xitong Gao[7], Zhe Zhao[8], Daquan Lin[8],
Jiadong Lin[9], Chuanbiao Song[9], Zihao Wang[10], Zhennan Wu[10], Yang Guo[11],
Jiequan Cui[12], Xiaogang Xu[12], Pengguang Chen[12]

[1] Tsinghua University [2] Alibaba Group [3] RealAI [4] Shanghai Jiao Tong University
[5] University of Electronic Science and Technology of China [6] University of Macau
[7] Chinese Academy of Sciences [8] ShanghaiTech University
[9] Huazhong University of Science and Technology [10] Indiana University Bloomington
[11] University of Wisconsin–Madison [12] The Chinese University of Hong Kong

{dyp17, qaf19, yangxiao19, pty17}@mails.tsinghua.edu.cn, {suhangss, dcszj}@tsinghua.edu.cn

{yuefeng.chenyf, mxf164419, heyuan.hy, hui.xueh, lizhao.lz}@alibaba-inc.com

Abstract

Due to the vulnerability of deep neural networks (DNNs) to adversarial examples, a large number of defense techniques have been proposed to alleviate this problem in recent years. However, the progress of building more robust models is usually hampered by the incomplete or incorrect robustness evaluation. To accelerate the research on reliable evaluation of adversarial robustness of the current defense models in image classification, the TSAIL group at Tsinghua University and the Alibaba Security group organized this competition along with a CVPR 2021 workshop on adversarial machine learning (https://aisecure-workshop.github.io/amlcvpr2021/). The purpose of this competition is to motivate novel attack algorithms to evaluate adversarial robustness more effectively and reliably. The participants were encouraged to develop stronger white-box attack algorithms to find the worst-case robustness of different defenses. This competition was conducted on an adversarial robustness evaluation platform — ARES (https://github.com/thu-ml/ares), and is held on the TianChi platform (https://tianchi.aliyun.com/competition/entrance/531847/introduction) as one of the series of AI Security Challengers Program. After the competition, we summarized the results and established a new adversarial robustness benchmark at https://ml.cs.tsinghua.edu.cn/ares-bench/, which allows users to upload adversarial attack algorithms and defense models for evaluation.

1. Introduction

Despite the remarkable success of deep neural networks (DNNs) in various applications [15], these models are vulnerable to adversarial examples [3, 12, 16, 24, 38], which are maliciously generated by adding imperceptible perturbations to normal examples but can cause erroneous predictions. As the vulnerability of DNNs raises concerns in various security-sensitive applications, a large number of adversarial defense methods have been proposed, including randomization [6, 48], image denoising [26], ensemble learning [32, 41], adversarial detection [25, 30], and adversarial training [16, 28], and certified defenses [45]. However, most of the defenses have soon been shown to be ineffective against stronger or adaptive attacks [2, 13, 40, 42, 50], making it challenging to understand the effects of the current defenses and identify the progress of the field. Among these defenses, adversarial training is arguably the most effective approach [2, 11], which trains the network on the adversarial examples generated by different attacks instead of the natural examples [28].

The most widely adopted approach to evaluate adversarial robustness is using adversarial attacks. One of the most popular attacks is the *projected gradient descent* (PGD) method [28], which iteratively generates an adversarial example by performing gradient updates to maximize a classification loss (e.g., cross-entropy loss) w.r.t. input. Based on PGD, recent improvements have been made in different aspects, including using different loss functions [3, 7, 18], adjusting the step size during adversarial attacks [7], and in-

troducing new strategies of initialization [39]. For example, AutoAttack [7] establishes a stronger baseline for reliable evaluation of adversarial robustness, which is composed of an ensemble of four attack methods, including APGD_{CE}, APGD_{DLR}, FAB, and Square Attack. The output diversified initialization (ODI) [39] method proposes to improve the diversity of adversarial examples in the output space by adopting a new initialization strategy.

To further accelerate the research on reliable evaluation of adversarial robustness of different defense models, we organized this competition in which the participants were required to develop effective white-box attack algorithms. We focus on image classification models since most adversarial defenses are developed on typical image benchmarks, e.g., CIFAR-10 [23] and ImageNet [8]. We only consider adversarial training models as the evaluated defenses since 1) these models are the state-of-the-art defenses; 2) these models do not have randomness in their predictions; and 3) the model gradient can be calculated without causing nonexistence of gradients. The whole competition consisted of three stages. In the first stage, we evaluated the submissions of white-box attacks on 15 defense models, including 13 models on CIFAR-10 and 2 models on ImageNet. We used the first 1,000 images of the CIFAR-10 test set and randomly chose 1,000 images of the ILSVRC 2012 validation set for evaluation. The top-100 teams of the first stage can enter the second one. In the second stage, we evaluated submissions on another set of 15 secret defense models with the same images, to ensure that the submissions cannot adapt to the details of the defense models. The top-20 teams can enter the final stage, in which we evaluated the final submissions using the whole CIFAR-10 test set containing 10,000 images and randomly chosen 10,000 images from the ILSVRC 2012 validation set. We evaluated the submissions of white-box attacks by the misclassification rate of the defense models (higher is better). Below we present more details about the competition and the top-6 solutions.

2. Competition Details

This competition was conducted on an adversarial robustness evaluation platform — ARES (https://github.com/thu-ml/ares) [11]. The participants needed to implement white-box attack algorithms following the *Attacker* class in ARES. The whole competition consisted of three stages with different models and datasets.

2.1. Models and Datasets

Stage I. In the first stage, we evaluated the submissions of white-box attacks on 15 defense models, which include 13 models trained on CIFAR-10 and 2 models trained on ImageNet. The models are shown in Table 1. We used the first 1,000 images from the CIFAR-10 test set and randomly chose 1,000 images from the ILSVRC 2012 validation set for evaluating the submissions on the public leaderboard.

Stage II. The top-100 teams in Stage I can enter Stage II. In this stage, we evaluated submissions on another set of 15 secret models. The models are shown in Table 1. The datasets are the same as Stage I.

Final Stage. We evaluated the final scores of the top-20 teams in Stage II. Each participant can select two submissions for the final evaluation. The models are the same as those used in Stage II. We used all the 10,000 images from the CIFAR-10 test set and randomly chosen 10,000 images from the ILSVRC 2012 validation set for final evaluations.

2.2. Evaluation

The participants were required to submit the source code of their developed white-box attacks. The submissions were evaluated under the untargeted ℓ_∞-norm threat models. For CIFAR-10, the perturbation budget is $\epsilon = 8/255$. For ImageNet, the perturbation budget is $\epsilon = 4/255$. The submitted attacks have access to the white-box models, but are limited to using their logit outputs. The participants can design losses and attack algorithms based on the logits for gradient calculations. We encouraged the participants to develop "general" attack algorithms, which means that they are not specified to each model. We evaluated the submitted attacks by the misclassification rate of the defenses models (higher is better), which is computed by the following formula

$$Score(A) = \frac{1}{|\mathcal{M}|} \sum_{M_i \in \mathcal{M}} \frac{1}{|\mathcal{D}|} \sum_{(x_j, y_j) \in \mathcal{D}} \mathbf{1}(M_i(A(x_j)) \neq y_j), \tag{1}$$

where A is an attack method that returns the adversarial example given a natural one, M_i is a defense model, and (x_j, y_j) is an image-label pair. The participants needed to ensure that $(A(x_j))$ returns an adversarial example whose distance from x_j is smaller than ϵ, otherwise we would clip the adversarial example within the range.

2.3. Additional Restrictions

To save computational cost and make a fair comparison between different attack methods, we restricted the runtime of the attack submissions. The average number of backward gradient calculations per image should be less than 100; the average number of forward model predictions should be less than 200. The total runtime of a submission for all models should be less than 3 hours on a Tesla V100 GPU (as a comparison, the baseline PGD-100 attack needs less than 2 hours for all models). When the attack runs backward gradient calculation, running forward model prediction is necessary, such that each backward gradient calculation would consume one backward gradient calculation quota and one forward model prediction quota, even if the attack does not explicit run the forward model predictions.

Stage I			Stage II & Final Stage		
Defense Model	Architecture	Dataset	Defense Model	Architecture	Dataset
Madry et al. (2018) [28]	WRN-28-10	CIFAR-10	Wang et al. (2020) [44]	WRN-28-10	CIFAR-10
Carmon et al. (2019) [4]	WRN-28-10	CIFAR-10	Ding et al. (2020) [9]	WRN-28-10	CIFAR-10
Zhang et al. (2019) [53]	WRN-34-10	CIFAR-10	Wang & Zhang (2019) [43]	WRN-28-10	CIFAR-10
Zhang & Wang (2019) [52]	WRN-28-10	CIFAR-10	Huang et al. (2020) [20]	WRN-34-10	CIFAR-10
Hendrycks et al. (2019) [19]	WRN-28-10	CIFAR-10	Chen et al. (2020) [5]	(3×) ResNet-50	CIFAR-10
Pang et al. (2020a) [34]	WRN-34-10	CIFAR-10	Gowal et al. (2020) [17]	WRN-28-10	CIFAR-10
Rice et al. (2020) [35]	WRN-34-10	CIFAR-10	Alayrac et al. (2019) [1]	WRN-106-8	CIFAR-10
Pang et al. (2020b) [31]	ResNet-v2-47	CIFAR-10	Dong et al. (2020) [10]	WRN-28-10	CIFAR-10
Wong et al. (2020) [46]	ResNet-18	CIFAR-10	Pang et al. (2021) [33]	DenseNet-121	CIFAR-10
Shafahi et al. (2019) [37]	WRN-34-10	CIFAR-10	Pang et al. (2021) [33]	DenseNet-201	CIFAR-10
Wu et al. (2020) [47]	WRN-28-10	CIFAR-10	Pang et al. (2021) [33]	DPN	CIFAR-10
Sehwag et al. (2020) [36]	WRN-28-10	CIFAR-10	Pang et al. (2021) [33]	WRN-34-10	CIFAR-10
Pang et al. (2021) [33]	WRN-34-10	CIFAR-10	Pang et al. (2021) [33]	WRN-34-10	CIFAR-10
Wong et al. (2020) [46]	ResNet-50	ImageNet	Pang et al. (2020a) [34]	ResNet-50	ImageNet
Shafahi et al. (2019) [37]	ResNet-50	ImageNet	Xie et al. (2019) [49]	ResNet-152	ImageNet

Table 1. The defense models used in the competition.

Team	**Stage I**	**Stage II**	**Final Stage**
green hand	**52.53**	**50.93**	**51.104**
UM-SIAT	52.39	50.65	50.961
S3L	52.13	50.56	50.955
BorderLine	52.40	50.69	50.895
Kanra	52.18	50.65	50.888
BalaBala2020	51.89	50.55	50.887

Table 2. Competition results of the top-6 teams.

3. Competition Results

There are more than 1,600 teams participating the competition. The total number of submissions is over 2,500. We received about 100 high-quality attack algorithms during the competition. The scores of the top-6 submissions are shown in Table 2. The final scores of the top-6 submissions are very close. They all lie within $[50.887, 51.104]$. After the competition, we summarized the results and established a new adversarial robustness benchmark at https://ml.cs.tsinghua.edu.cn/ares-bench/. This benchmark includes several typical attacks (e.g., PGD-100, MIM-100, CW-100) as well as the top-5 attacks in the competition to evaluate adversarial robustness of different defense models. This benchmark also allows users to upload adversarial attacks and defenses for evaluation. We will further maintain the benchmark results after the competition for evaluating and comparing future attack or defense methods.

4. Top Scoring Submissions

4.1. 1st place: green hand

Team members: Ye Liu, Qilong Zhang, Lianli Gao

4.1.1 Method

Since the numbers of backward gradient calculations and the forward model predictions are limited, we need to make effective use of it. We adopt ODI-PGD [39] as the baseline method and make several improvements upon it.

Step decay: The fixed step size is sub-optimal, so we gradually reduce the step size from the maximum to one hundred-th of the maximum step size during the attack. Specifically, we use the SGDR [27] method in the step adjustment strategy. In each restart, we utilize a large step size at the beginning and then quickly reduce the step size so that we can converge to local maximum of loss. The overall step size can be formalized as:

$$\eta_i = \frac{1}{2}(\eta_{max} - \eta_{min})\left(1 + \cos\left(\frac{i \bmod T}{T}\pi\right)\right) + \eta_{min}, \tag{2}$$

where η_i indicates the step size of the i-th iteration, η_{min} represents the smallest step size, η_{max} represents the maximum step size, i denotes the current iteration number, and T represents the total number of iterations.

Delete difficult examples: The attack difficulty of different examples is different, and we cannot successfully attack all examples. Therefore, we do not attack the difficult examples. Specifically, we judge the difficulty of the examples based on the loss. If the loss is relatively small, we think the example is more difficult to attack, if the loss is relatively large, we think the example is easier to attack successfully.

Iteration increase: In order to efficiently use the number of iterations, we use multiple restarts. At the beginning, the number of iterations for restart is smaller. As the number of restarts increases, the number of iterations included in restart also increases.

Bias output diversity initialization: The output diversity initialization method is moving in a random direction. We propose a biased output diversity initialization method, which can move in a better direction to improve the efficiency of the attack.

4.1.2 Submission Details and Results

Submission details: The number of restarts is 17, the number of iterations is [10-60], maximum step size η_{max} is 8/255, smallest step size η_{min} is $0.001 \times \eta_{max}$.
Results: The preliminary score is 52.53, the semifinal score is 50.93, the final score is 51.104. All scores are ranked first.

4.2. 2nd place: team UM-SIAT

Team members: Yunrui Yu, Xitong Gao

4.2.1 Method

We used the new surrogate loss function introduced in our earlier work [51], which can improve attack success rate and convergence rate with this function when compared to the original loss function used to train the model. For this competition, we divided the model into robust models and non-robust ones, and designed different strategies for each type. In order to maximize the score, we additionally explored the use of step-size schedules, ensemble strategies, and adapts the number of iterations for different image samples. The next section will explain our method in detail.

4.2.2 Submission Details and Results

We found that for many of the defense models, the output from the standard softmax cross-entropy loss function is often easily saturated, *i.e.*, it is prone to underflow in floating-point arithmetic, or produce negligible back-propagation signals. In some cases, attacks based on back-propagation on this function will fail. To overcome this, we used the LAFEAT loss function introduced in our earlier work [51], which constrains the range of the difference between the maximum and the second largest value in the logits output of the model:

$$\mathcal{L}^{\mathrm{sur}}(\mathbf{z}, \mathbf{y}) \triangleq \mathcal{L}^{\mathrm{sce}}\big(\mathbf{z} \big/ \big(\mathbf{y}^\top \mathbf{z} - \max\left((1-\mathbf{y}) \cdot \mathbf{z}\right)\big), \mathbf{y}\big)\,, \tag{3}$$

where $\mathbf{z}$ is the pre-logits network output, $\mathbf{y}$ is the one-hot ground-truth label, and $\cdot$ denotes the element-wise product. Using this loss function, gradient-based attacks can in general converge significantly faster, and improve the success rate of attacks.

The second tactic we adopted in our method is targeted variants of the above loss function. Instead of maximizing the loss of the ground-truth label, the targeted variant minimizes the loss of the target label:

$$\mathcal{L}^{\mathrm{sur}}_{\boldsymbol{\tau}}(\mathbf{z}, \mathbf{y}) \triangleq -\mathcal{L}^{\mathrm{sce}}\Big(\frac{\mathbf{z}}{t}\big(\mathbf{y}^\top \mathbf{z} - \max\left((1-\mathbf{y}) \cdot \mathbf{z}\right)\big)^{-1}, \boldsymbol{\tau}\Big)\,. \tag{4}$$

We found that the targeted variants could further reduce the accuracy of the non-robust models. Based on this finding, we designed a simple tactic to automatically distinguish if the model is robust against our baseline strategy, and adjust our strategy accordingly. We designed a 10-iteration attack which uses the untargeted surrogate loss, followed by the top-3 targets of the DLR loss [7]. We classify models as non-robust, if the robustness of the model is dropped sharply during the targeted phase. We divide the remaining number of gradient-iteration budget by 9 to attack the non-robust model by the top-9 targets of the new loss function. We will use all remaining iterations to attack robust models using the untargeted loss function. As the different attack settings are carried out in sequence, we allow new attack settings to continue working on the previous perturbation generated by the previous setting.

We explored different choices of step size schedules, and used $\cos(\frac{4i}{I})$ as the final version, where i is the current iteration count and I is the number of iterations of the current attack. After each attack iteration, if the sample has been attacked successfully, we will stop further attacks on the sample and allocate the remaining iterations to outstanding samples that have not been attacked successfully.

To ensure that the attack strategy can generalize to new models, we search for the optimal configuration with results that are relatively stable under local hyperparameter changes.

Result: The final score is 50.961.

4.3. 3rd place: team S3L

Team members: Zhe Zhao and Daquan Lin

4.3.1 Method

The critical point of this competition is the limitation on queries, which requires the attacker to construct more efficient attack methods. We conducted a simple test of each adversarially trained model using PGD attack, and found that these models often have similar characteristics when facing an adversarial attack: about 50% of the benign samples can be easily attacked successfully, about 40% of the samples are almost impossible to find adversarial examples, the remaining 10% are the key to compete with other teams, and these images have adversarial examples, but require a lot of iterations to find them.

We designed our method for this competition from two perspectives.

1. For images that are easy to attack: finding adversarial examples using as few iterations as possible.

2. For images that cannot be attacked, finding and excluding them using as few iterations as possible.

The key point of our method is to estimate *local robustness* effectively and efficiently.

Definition 4.1 (Local Robustness) *Given a sample input x, a DNN $\mathcal{D}$ and a perturbation threshold ϵ, $\mathcal{D}$ is ϵ -local robust iff for any sample input x' such that $\|x - x'\|_p \leq \delta$, we have $\mathcal{D}(x) = \mathcal{D}(x')$, where $\|\cdot\|_p$ is the p -norm to measure the distance between two images.*

There are some techniques based on software analysis and verification (e.g., interval analysis, abstract interpretation, etc. [21, 54]) that can derive the robust accuracy of a neural network soundly. The robust accuracy refers to the accuracy that a neural network can achieve under the local robustness constraint. In this competition the models are white-box models, but the organizers restrict the attackers to use only gradient data. Thus we cannot use the verification techniques mentioned above. Therefore, we propose a novel attack method, named outside-inside attack (OIA), to estimate whether an input has the potential to be successfully attacked.[1] The inspiration of OIA comes from [55], which uses attack costs to estimate local robustness.

OIA first attacks the benign samples using a perturbation larger than ϵ (e.g., $2 \times \epsilon$), and subsequently projects the perturbation to ϵ under L_∞ constraint. In the first step, if the image is successfully attacked with a larger perturbation, we consider it to have an adversarial example in the ϵ interval. While for a failed attack, we think the input has no adversarial example in the ϵ interval, i.e., satisfying local robustness. For these images, we remove them from the subsequent iterations, thus saving the iterations for those images that are likely to be successfully attacked.

4.3.2 Submission Details and Results

For OIA, we used a 5-step BIM attack with $\alpha = \epsilon/2$. Subsequently, we used ODI-PGD attack [39]. Referring to the hyperparameters provided by [39], we performed a 2-step ODI attack with $\alpha = \epsilon$. Then a 20-step PGD search was executed. It is worth mentioning that we use the method recommended in AutoAttack [7] to select the optimal perturbation obtained during the previous attack as the starting point for the next round.

Using OIA before ODI-PGD attack, we can optimize the two perspectives mentioned in Sec. 4.3.1 at the same time. First, OIA can quickly attack those vulnerable samples whose perturbations are also easily effective after clipping. In addition, OIA uses only a few iterations to filter out a large number of images that cannot be successfully attacked under L_∞ constraint. However, OIA only provides an approximate estimate of local robustness and cannot give a theoretical guarantee, but from our statistical results, the false negative rate of this method is only about 0.1‰.

During the implementation of PGD attack, we hard-coded the number of iterations and step size decay. In the initial few restarts, we use a smaller number of iterations to search quickly, and in the subsequent restarts, we increase the number of iterations. During the iterations, we keep reducing the step size to avoid oscillations. Inspired by DFA [14], we constrain the minimum step-size in the reducing process to avoid accuracy problems. In the end, we got 50.955 points and took third place.

4.4. 4th place: team BorderLine

Team members: Jiadong Lin and Chuanbiao Song

4.4.1 Method

In this section, we introduce the random real target (RRT) attack method, which won the 4th place in the competition. The idea is to develop a novel sampling strategy for the initial perturbed points to improve the efficiency of adversarial attack.

Random sampling is crucial for the optimization-based white-box attacks, which helps to find a diverse set of initial perturbed points for the attack success. Many sampling strategies have been used to improve the attack efficiency, such as uniform sampling, Gaussian sampling and output diversified sampling (ODS) [39]. Currently, ODS is widely used to randomly restart for the white-box attacks and leads to achieve the best performance among the sampling strategies. The basic idea of ODS is randomly specify a direction in the output space and then perform gradient-based optimization to generate a perturbation in the input space. However in practice, the randomly direction in the output space may not be a great representation of the effective and true direction, which limits the attack strength.

In order to overcome this drawback, we propose random real target (RRT) sampling, a novel sampling strategy that attempts to enhance the sampling diversity in the space of the target model's outputs while maintain its authenticity. RTT and ODS differ in two ways. First, RRT uses the logits of the real target image as the optimization direction, while ODS uses the random noises sampled from the uniform distribution. Second, RRT uses cosine similarity as the distance for the optimization, while ODS directly uses the sampled direction as the distance.

Given an original image x_{ori}, a random real target image x_{tar}, we define the perturbation vector of RRT as follows:

$$v_{RRT}(x_{ori}, x_{tar}) = \nabla_{x_{ori}}\left(\frac{f(x_{ori}) \cdot f(x_{tar})}{\|f(x_{ori})\|_2 * \|f(x_{tar})\|_2}\right),$$

[1] The paper based on OIA is in preparation and will be available soon.

where f is a classifier that maps the input image $x \in [0,1]^D$ to the logits $z \in \mathbb{R}^C$, D is input dimension and C is the number of classes.

In the competition, we utilize RRT for initialization to generate diversified initial perturbed points and maintain the representative and authenticity of the logits of the initial perturbed points. Following ODS, we perform RRT initialization by the gradient-based optimization. Given an original image $x_0 = x_{ori}$ and a random real target image x_{tar}, we try to find a restart point x via the following iterative update:

$$x_{t+1} = \Pi_{\mathcal{B}_\epsilon^\infty(x_{ori})}\left(x_t + \alpha\,\mathrm{sign}(v_{RRT}(x_t, x_{tar}))\right),$$

where $\mathcal{B}_\epsilon^\infty(x) := \{x' : \|x' - x\|_\infty \leq \epsilon\}$ is the set of allowed perturbations, and $\Pi_{\mathcal{B}_\epsilon^\infty(x)}$ indicates the projection of the set $\mathcal{B}_\epsilon^\infty(x)$. Similar to ODS, we randomly sample a new target image x_{tar} for each random restart for the attack. After obtaining the starting points, we perform the projected gradient descent (PGD) attack to generate adversarial examples.

4.4.2 Submission Details and Results

In this section, we summarize the submission details including hyper-parameters setting and some tricks, and the submission results.

Iterative number. Iterative number is a crucial parameter for adversarial attacks. In this competition, an average of 100 iterations were allowed per sample. We set the iterative number of RRT initialization as 2 and the iterative number of PGD attack as 18.

Learning rate. Following the MD attack method [22], we use a large step size ($2 \times \epsilon$) in the first stage (steps 0-4) for a better exploration, and a small step size ($0.25 \cdot \epsilon$) in the second stage (steps 5-17) to ensure a stable optimization.

Momentum update. Following the MIM attack [12], we also integrated the momentum idea into PGD to accelerate the attack. Since in the first stage, the update direction is highly inaccurate, we only perform momentum update in the second stage (step 5-17).

Multi-Targeted attack. Following the MT attack [18], at each restart, we pick a new target class for targeted attack. We first sort the target classes based on the logit outputs, from the highest to the bottom. Then, in turn, the target class is selected to attack until the allowed number of iterations are run out.

Submission Results. As shown in Table 3, we report the major milestone for the changes in attack method. We can see that, at the Stage I, the ODI, MT and MIM strategies can boost the attack success of the PGD attack, and our RRT provides better performance than the ODI, which we can attribute to the diverse and real initial perturbed points.

4.5. 5th place: team Kanra

Team members: Zihao Wang, Zhennan Wu, Yang Guo

Scenario	Stage I	Stage II	Final
RRT+PGD+MT+MIM	52.40	50.69	50.895
ODI+PGD+MT+MIM	52.23	-	-
ODI+PGD+MT	52.09	-	-
ODI+PGD	51.76	-	-
PGD	46.18	-	-

Table 3. Major Millstone for the changes of method

4.5.1 Method

Since the average number of backward gradient calculations per image and forward model predictions are restricted, to improve the efficiency of the attack, we propose Fast Restart Projected Gradient Descent (FR-PGD), a two-phase strategy that can efficiently utilize the limited resources.

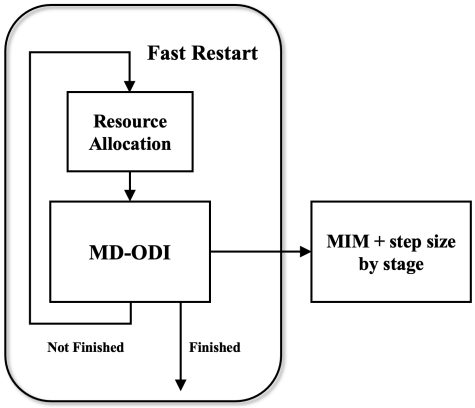

Figure 1. Workflow of Fast Restart Projected Gradient Descent.

Figure 1 demonstrates a high-level workflow of our attack. The attack can be divided into two phases: Fast Restart phase and multi-step convergence phase. Compared with previous restart methods, each restart of FR-PGD takes only a small number of backward gradient calculations. Therefore, the number of restarts can be guaranteed even in the case of resource constraints. Then, several adversarial samples from the Fast Restart phase (the number of the samples depends on how many samples have not been successfully attacked in the restart phase) with the highest Loss value are sent to the multi-step convergence phase. Finally, the convergence results are returned.

It is worth noting that although we choose the highest Loss value each time, there is only a positive correlation between the Loss value and the final success probability of the attack to a certain extent, but not absolutely. The more the number of backward gradient calculations in the restart phase, the stronger the correlation. However, this will lead to a reduction in the number of restarts or the steps

of convergence, since the resources are limited. Therefore, it needs to be weighed according to resources.

4.5.2 Submission Details and Results

Specifically, a Output Diversified Initialization (ODI) [39] version of Margin Decomposition (MD) attack [22] is used in the Fast Restart phase. The loss function in each restart is defined as follows:

$$\mathbf{x}_{k+1} = \Pi_\epsilon(\mathbf{x}_k + \alpha \cdot \text{sign}(\nabla_{\mathbf{x}} \ell_k^r(\mathbf{x}_k, y))), \quad (5)$$

$$\ell_k^r(\mathbf{x}_k, y) = \begin{cases} \mathbf{z}_{max} & \text{if } k < \frac{K}{2} \text{ and } r \bmod 2 = 0 \\ -\mathbf{z}_y & \text{if } k < \frac{K}{2} \text{ and } r \bmod 2 = 1 \\ \mathbf{z}_{max} - \mathbf{z}_y & \text{if } k \geq \frac{K}{2}, \end{cases}$$

where, $k \in \{1, \ldots, K\}$ is the number of backward gradient calculations in the restart phase, $r \in \{1, \ldots, n\}$ is the r-th restart, mod is the modulo operation for alternating optimization, and ℓ_k^r defines the loss function used at the k-th step and r-th restart. The loss function switches from the individual terms back to the full margin loss at step $\frac{K}{2}$. In each restart, we perform 2 steps of ODI and 4 steps of gradient ascents. After each restart, the computing resources are reallocated. Specifically, the resources of the samples that have been successfully attacked are allocated to the samples that have not been done, so that the unfinished samples can get more number of restarts.

In the multi-step convergence phase, Momentum Iterative Method (MIM) [12] with the C&W Loss [3] is performed. We adopt scheduled step size instead of fixed one. Because we found that starting from large step size brings better results. We set the initial step size η_0 as $\eta_0 = \epsilon$. We update the step size to $\epsilon/3, \epsilon/8$ at $k = 0.25N, 0.5N$, respectively.

The ratio of resources of the Fast Restart phase and the multi-step convergence phase is set to 4:1 empirically. Our method got a score of 50.888 in the Final Stage.

4.6. 6th place: team BalaBala2020

Team members: Jiequan Cui, Xiaogang Xu, Pengguang Chen

4.6.1 Method

The team proposed a novel attack strategy called *Difficulty-Hierarchical Attack (DH Attack)* for attacking target models with a constrained number of attack iterations. In this strategy, more attack iterations will be conducted for the robust samples that are harder to perturb, improving the integral attack success rate. Moreover, a global perturbation [29] is utilized, attacking a part of samples at the beginning of each attack iteration and allowing more attack iterations for robust samples. Furthermore, several effective tricks are explored and adopted in DH Attack.

Hierarchical Attack. To optimally utilize the constrained attack iterations, the team distributes the number of attack iteration to each sample according to their robustness towards adversarial perturbations. The attack is conducted hierarchically with multiple restart times, and once one sample is successfully attacked, the attack process for this sample will be canceled, preserving more attack iterations for robust samples.

Global Perturbation. Moreover, the team utilizes the transferability of adversarial attacks and creates a global perturbation by accumulating the perturbations of different samples. The creation of global perturbation does not involve the backward of the network, and it can be utilized at the beginning of each restart to attack a part of the samples.

Other Tricks.

I. Loss function ensemble. The team observed that different robust models are sensitive to different loss functions for generating adversarial examples. After each restart with ODI initialization [39], DH Attack would uniformly choose loss function from {MarginLoss [3], DLRLoss [7]}.

II. Decaying step-size. Adopting cosine schedule for step-size is better than a fixed value.

III. ODI initialization with restricted search directions. In the ODI algorithm, the generated initialization with one random vector uniformly exists in the whole search space. It keeps the diversity of initialization at the cost of attack efficiency. In experiments, the team observed that, for most successfully attacked images, the corresponding adversarial examples are misclassified into the first or the second most possible classes finally. Based on this phenomenon, after each ODI initialization, DH Attack would apply 5 attack iterations with a multi-targeted attack loss, i.e., summing the targeted loss for the first and the second most possible classes that could be misclassified into.

4.6.2 Submission Details and Results

For ODI initialization, the team applied 5 iterations to update. The initial step-size is 4/255 and is decayed with a cosine schedule for each restart. In the beginning, the team utilized a 10 iterations attack to filter the easiest samples that can be attacked. We use 20 iterations for all the remaining restart if the backward quota is enough. The team achieved a score of 50.887 for the final testing and rank sixth.

5. Conclusion

We successfully organized the *Adversarial Attacks on ML Defense Models* competition, which attracted thousands of teams to participate. We obtained many effective attack algorithms to evaluate adversarial robustness reliably and

correctly, which helps to accelerate the research on robustness evaluation. The technical details of the top scoring submissions are presented in this paper. We also established an adversarial robustness benchmark to summarize the results, which could be further used for newly developed attack and defense methods.

References

[1] Jean-Baptiste Alayrac, Jonathan Uesato, Po-Sen Huang, Alhussein Fawzi, Robert Stanforth, and Pushmeet Kohli. Are labels required for improving adversarial robustness? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 12192–12202, 2019. 3

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2018. 1

[3] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017. 1, 7

[4] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, Percy Liang, and John C Duchi. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 3

[5] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 699–708, 2020. 3

[6] Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning (ICML)*, 2019. 1

[7] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning (ICML)*, pages 2206–2216, 2020. 1, 2, 4, 5, 7

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 2

[9] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. In *International Conference on Learning Representations (ICLR)*, 2020. 3

[10] Yinpeng Dong, Zhijie Deng, Tianyu Pang, Hang Su, and Jun Zhu. Adversarial distributional training for robust deep learning. In *Advances in Neural Information Processing Systems*, 2020. 3

[11] Yinpeng Dong, Qi-An Fu, Xiao Yang, Tianyu Pang, Hang Su, Zihao Xiao, and Jun Zhu. Benchmarking adversarial robustness on image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 321–331, 2020. 1, 2

[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 (CVPR)*, pages 9185–9193, 2018. 1, 6, 7

[13] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4312–4321, 2019. 1

[14] Yuchao Duan, Zhe Zhao, Lei Bu, and Fu Song. Things you may not know about adversarial example: A black-box adversarial image attack. *arXiv preprint arXiv:1905.07672*, 2019. 5

[15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org. 1

[16] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015. 1

[17] 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. 3

[18] Sven Gowal, Jonathan Uesato, Chongli Qin, Po-Sen Huang, Timothy Mann, and Pushmeet Kohli. An alternative surrogate loss for pgd-based adversarial testing. *arXiv preprint arXiv:2006.13726*, 2020. 1, 6

[19] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning (ICML)*, pages 2712–2721, 2019. 3

[20] Lang Huang, Chao Zhang, and Hongyang Zhang. Self-adaptive training: beyond empirical risk minimization. *arXiv preprint arXiv:2002.10319*, 2020. 3

[21] Xiaowei Huang, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xinping Yi. A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability. *Computer Science Review*, 37:100270, 2020. 5

[22] Linxi Jiang, Xingjun Ma, Zejia Weng, James Bailey, and Yu-Gang Jiang. Imbalanced gradients: A new cause of overestimated adversarial robustness. *arXiv preprint arXiv:2006.13726*, 2020. 6, 7

[23] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 2

[24] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *International Conference on Learning Representations (ICLR) Workshops*, 2017. 1

[25] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 1

[26] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[27] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 3

[28] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018. 1, 3

[29] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 7

[30] Tianyu Pang, Chao Du, Yinpeng Dong, and Jun Zhu. Towards robust detection of adversarial examples. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4579–4589, 2018. 1

[31] Tianyu Pang, Kun Xu, Yinpeng Dong, Chao Du, Ning Chen, and Jun Zhu. Rethinking softmax cross-entropy loss for adversarial robustness. In *International Conference on Learning Representations (ICLR)*, 2020. 3

[32] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning (ICML)*, 2019. 1

[33] Tianyu Pang, Xiao Yang, Yinpeng Dong, Hang Su, and Jun Zhu. Bag of tricks for adversarial training. In *International Conference on Learning Representations (ICLR)*, 2021. 3

[34] Tianyu Pang, Xiao Yang, Yinpeng Dong, Kun Xu, Hang Su, and Jun Zhu. Boosting adversarial training with hypersphere embedding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[35] Leslie Rice, Eric Wong, and J Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning (ICML)*, 2020. 3

[36] Vikash Sehwag, Shiqi Wang, Prateek Mittal, and Suman Jana. Hydra: Pruning adversarially robust neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[37] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 3

[38] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014. 1

[39] Yusuke Tashiro, Yang Song, and Stefano Ermon. Diversity can be transferred: Output diversification for white- and black-box attacks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4536–4548, 2020. 2, 3, 5, 7

[40] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1

[41] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017. 1

[42] Jonathan Uesato, Brendan O'Donoghue, Aaron van den Oord, and Pushmeet Kohli. Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning (ICML)*, 2018. 1

[43] Jianyu Wang and Haichao Zhang. Bilateral adversarial training: Towards fast training of more robust models against adversarial attacks. In *IEEE International Conference on Computer Vision (ICCV)*, pages 6629–6638, 2019. 3

[44] 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 (ICLR)*, 2020. 3

[45] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning (ICML)*, 2018. 1

[46] Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations (ICLR)*, 2020. 3

[47] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[48] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations (ICLR)*, 2018. 1

[49] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[50] Xiao Yang, Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Boosting transferability of targeted adversarial examples via hierarchical generative networks. *arXiv preprint arXiv:2107.01809*, 2021. 1

[51] Yunrui Yu, Xitong Gao, and Cheng-Zhong Xu. LAFEAT: Piercing through adversarial defenses with latent features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 4

[52] Haichao Zhang and Jianyu Wang. Defense against adversarial attacks using feature scattering-based adversarial training. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1829–1839, 2019. 3

[53] 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 *International Conference on Machine Learning (ICML)*, 2019. 3

[54] Yedi Zhang, Zhe Zhao, Guangke Chen, Fu Song, and Taolue Chen. Bdd4bnn: A bdd-based quantitative analy-

sis framework for binarized neural networks. *arXiv preprint arXiv:2103.07224*, 2021. 5

[55] Zhe Zhao, Guangke Chen, Jingyi Wang, Yiwei Yang, Fu Song, and Jun Sun. Attack as defense: Characterizing adversarial examples using robustness. *arXiv preprint arXiv:2103.07633*, 2021. 5