Rethinking the Adversarial Robustness of Multi-Exit Neural Networks in an Attack-Defense Game

Keyizhi Xu Chi Zhang Zhan Chen Zhongyuan Wang Chunxia Xiao Chao Liang*
School of Computer Science, Wuhan University
National Engineering Research Center for Multimedia Software (NERCMS)
Hubei Key Laboratory of Multimedia and Network Communication Engineering, Wuhan
{xukeyizhi,zhangchii,chenzhan,cxxiao,cliang}@whu.edu.cn, wzy_hope@163.com

Abstract

Multi-exit neural networks represent a promising approach to enhancing model inference efficiency, yet like common neural networks, they suffer from significantly reduced robustness against adversarial attacks. While some defense methods have been raised to strengthen the adversarial robustness of multi-exit neural networks, we identify a long-neglected flaw in the evaluation of previous studies: simply using a fixed set of exits for attack may lead to an overestimation of their defense capacity. *Based on this finding, our work explores the following three key aspects in the adversarial robustness of multi-exit neural networks: (1) we discover that a mismatch of the network exits used by the attacker and defender is responsible for the overestimated robustness of previous defense methods; (2) by finding the best strategy in a two-player zero-sum game, we propose* AIMER *as an improved evaluation scheme to measure the intrinsic robustness of multi-exit neural networks; (3) going further, we introduce* NEED *defense method under the evaluation of AIMER that can optimize the defender's strategy by finding a Nash equilibrium of the game. Experiments over 3 datasets, 7 architectures, 7 attacks and 4 baselines show that AIMER evaluates the robustness 13.52% lower than previous methods under AutoAttack, while the robust performance of NEED surpasses single-exit networks of the same backbones by 5.58% maximally.*

1. Introduction

Deep neural networks have achieved remarkable advancements in the field of computer vision, yet researchers are drawn to two pressing issues. First, the computational cost escalates as the networks grow deeper, leading to the rise of multi-exit neural networks [11, 12, 33, 37, 43]. These networks utilize an early-exit mechanism to produce results

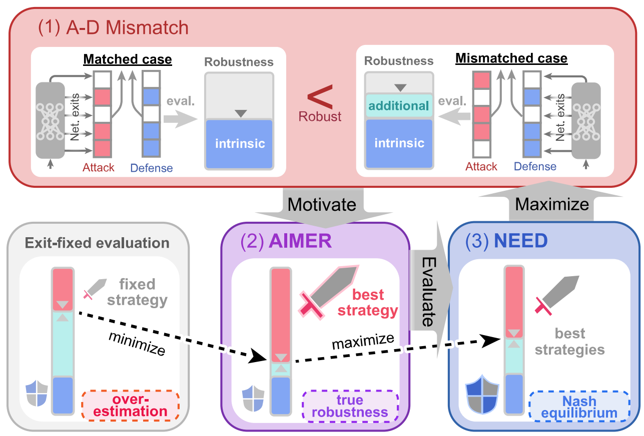


Figure 1. The triple focuses of this paper and their relationship. (1) the A-D mismatch phenomenon we find; (2) The improved evaluation scheme AIMER we propose; (3) The NEED method to optimize the defense under the evaluation of AIMER. (2) and (3) are principled by game theory.

from shallower branches, maintaining accuracy while reducing computational load. Second, the vulnerability of neural networks to adversarial attacks [2, 22, 32, 42] poses a significant challenge, where small, carefully crafted perturbations can be added to input data to deceive the predictions of models. In recent years, enhancing the robustness of models under adversarial attacks has thus emerged as a pivotal research topic [9, 19, 22, 27, 41, 47].

Inspired by the above work, increasing efforts have been devoted to the study of the adversarial robustness of multi-exit neural networks [3, 10, 13, 15, 16]. However, we identify a subtle defect long-neglected in the current robustness evaluation: *simply using fixed network exits as the targets of attack results in insufficient flexibility that unfairly weakens the attacker while favoring the defender.* This might lead to an overestimation in the robustness evaluation of multi-exit neural networks. In this paper, by delving into the following

*Chao Liang is the corresponding author.

three challenging problems, we strive to expose the negative consequence of such flawed exit-fixed evaluation and resort to game theory to amend it.

What does exit-fixed robustness evaluation lead to? Through experimental investigations of multi-exit neural networks, we observe a notable phenomenon, which we term as *Attack-Defense* (A-D) *mismatch*. In multi-exit neural networks, both the attacker and defender have the freedom of choosing which exit (or an ensemble) they use for generating adversarial examples or inference. However, when the attacker avoids the exact ensemble of exits the defender uses for inference (*i.e.*, a mismatch), the evaluated robustness is always higher than that of the matching case (Figure 1), bringing additional robustness apart from the intrinsic robustness obtained from adversarial training. Especially when the exits for attack are fixed, it is quite easy to cause an A-D mismatch by detouring with a different inference strategy. Therefore, we argue that such exit-fixed evaluation exacerbates the additional robustness brought by A-D mismatch, leading to an overestimation of defense capacity long-neglected by previous researchers.

Can we reduce A-D mismatch during the evaluation? Aware of the drawbacks of using exit-fixed robustness evaluation, we aim to find a better attack scheme that can reduce A-D mismatch during robustness evaluation. However, due to the uncertainty of the defender's choice of exits, this task can be quite challenging. To approach the tricky problem, we seek inspiration from game theory [35], model the adversarial attack and defense of multi-exit neural networks as a two-player zero-sum game, and identify the best strategy for the attacker as the criterion for evaluating adversarial robustness. We refer to this white-box evaluation scheme as *AdaptIve evaluation of Multi-Exit Robustness* (AIMER). It does not involve any modifications to attack algorithms but rather optimizes the choice of victim exits. Considering the robust performance of a network remains constant under a fixed attack algorithm, AIMER can reduce the additional robustness and thus more accurately reflect the network's intrinsic robustness.

Is it still possible to utilize A-D mismatch under AIMER? Though A-D mismatch is largely avoided under the more stringent evaluation of AIMER, it cannot be completely eliminated due to the uncertainty of the attack-defense game. To maximize the robustness of defense in the worst-case evaluation, we devise the *Nash Equilibrium Enhanced Defense* (NEED) method to reach the minimax point of the game. Specifically, NEED operates a stochastic strategy inferring with ensembles of exits with a certain probability, making both the defender and attacker perform their best strategies by seeking a Nash equilibrium [25].

The efficacy of both AIMER and NEED are verified with extensive experiments, covering different network architectures, datasets, attack algorithms and adversarial training methods. An illustration of the main focuses of this paper and their relationship can be found in Figure 1, and our contributions can be summarized as follows:

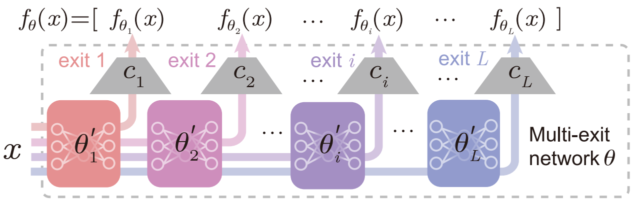


Figure 2. The structure of a multi-exit neural network.

- We find the A-D mismatch phenomenon in the robustness evaluation of multi-exit neural networks, which explains the overestimated robustness of defense in previous work.
- We raise a more stringent scheme AIMER to evaluate the adversarial robustness of multi-exit neural networks, where the attacker operates his best strategy in the attack-defense game.
- We devise a NEED method for defense by finding a mixed-strategy Nash equilibrium, which maximizes the robustness brought by A-D mismatch under AIMER.
- We conduct extensive experiments covering 7 network architectures, 3 datasets, 7 attack settings and combine NEED with 4 adversarial training methods, which consistently demonstrate the efficacy of AIMER and NEED.

2. Preliminaries

A multi-exit neural network can be equivalently modeled as a set of single-exit networks partially sharing their parameters. Suppose a multi-exit neural network θ with L exits is divided into L sequential blocks $[\theta'_i]_{i=1}^L$ and the final-exit classifier c_L. Each subnetwork of θ can be expressed as θ_i, where $\theta_i = [\theta'_1, \cdots, \theta'_i, c_i]$ with c_i being the classifier in the i-th exit of θ. Given an input x, the output is formulated as a group of predictions from the subnetworks: $f_\theta(x) = [f_{\theta_i}(x)]_{i=1}^L$. The detailed structure of a multi-exit neural network is depicted in Figure 2.

This paper focuses on the adversarial attack and defense of multi-exit neural networks, which are very flexible due to the multiple prediction outcomes. The defender makes an inference by choosing from or aggregating the results in $f_\theta(x)$, and some typical inference strategies are specified in Appendix D. The attacker also has ample choices of attack schemes to fool the model. Following the previous work [16], adversarial attacks for multi-exit neural networks come mainly in three forms, *i.e.*, single attack, average attack, and max-average attack:

$$x^{\text{adv}}_{\text{sin},i} = \underset{x' \in \{x' : |x'-x|_\infty < \epsilon\}}{\arg\max} |\mathcal{L}(f_{\theta_i}(x'), y)| \quad (1)$$

$$x^{\text{adv}}_{\text{avg}} = \underset{x' \in \{x' : |x'-x|_\infty < \epsilon\}}{\arg\max} \left| \frac{1}{L} \sum_{i=1}^{L} \mathcal{L}(f_{\theta_i}(x'), y) \right| \quad (2)$$

$$x^{\text{adv}}_{\max} = x^{\text{adv}}_{\sin,i^*} \text{ where } i^* = \arg\max_i \left| \frac{1}{L} \sum_{j=1}^{L} \mathcal{L}(f_{\theta_j}(x^{\text{adv}}_{\sin,i}), y) \right| \tag{3}$$

A detailed explanation of the formulations can be found in Appendix E. Generally, These three recipes of attacks are simple in form and easy to implement; however, they only consider rather limited scenarios for attack (*i.e.*, using either a single exit or all the exits for attack), leaving the remaining overlooked (*e.g.*, picking several particular exits for attack). Unfortunately, the evaluation in previous attempts to improve the robustness of multi-exit neural networks [3, 10, 16] was limited to these paradigms, without considering the risks above. Although they have undertaken valuable research into the robustness of multi-exit neural networks, we argue that the evaluation results using limited recipes of victim exits cannot truthfully reflect the intrinsic robustness of these defense methods. This is what motivates us to conduct a more in-depth exploration of this issue.

3. Methodology

In this section, we clarify the problem setup and then sequentially detail the main components of our work, *i.e.*, the Attack-Defense (A-D) mismatch phenomenon we identify, the **A**dapt**I**ve evaluation of **M**ulti-**E**xit **R**obustness (AIMER) and the **N**ash **E**quilibrium **E**nhanced **D**efense (NEED) method.

3.1. Problem Setup

To better convey the following concepts, we define a new attack form for multi-exit neural networks dubbed *partial attack*. Let a set of exit indices $E_a = \{n : n \in \{i\}_{i=1}^{L}\}$, $E_a \neq \varnothing$ denotes an ensemble of exits selected by the attacker, an adversarial example generated by partial attack is formulated as:

$$x^{\text{adv}}_{\text{par},E_a} = \underset{x' \in \{x' : |x'-x|_\infty < \epsilon\}}{\arg\max} \left| \frac{1}{|E_a|} \sum_{i \in E_a} \mathcal{L}(f_{\theta_i}(x'), y) \right| \tag{4}$$

It allows the attacker a to select any ensemble of network exits for attack, which unifies single attack (Equation 1) and average attack (Equation 2) and, in the meantime, considers more possibilities. Similarly, the defender d also has the freedom to choose any ensemble of exits $E_d = \{n : n \in \{i\}_{i=1}^{L}\}$ and $E_d \neq \varnothing$ to infer with the mean of logits.

Threat Model. This paper focuses on the white-box adversarial robustness of multi-exit neural networks with the following setups:

- The gradient information from every network exit can be utilized for the generation of adversarial examples.
- We assume that a and d decide their strategies beforehand, are aware of the probabilistic strategies of each other, and are not allowed to alter their strategies during evaluation.
- a first generates adversarial examples following the attack strategy, and then uses the adversarial examples to challenge d that independently makes the inference.
- Although a and d are aware of the probabilistic strategies of each other, a has no access to the specific exit ensemble E_d being selected for inference by d.

3.2. Attack-Defense Mismatch

Following the above setup of attack and defense, we carry out an empirical study into the adversarial robustness of multi-exit neural networks. Specifically, we test the robust accuracy scores using different partial inferences under different partial attacks. $\text{Acc}(\theta, E_a, E_d)$ evaluates the robust accuracy of multi-exit neural network θ under partial attack with E_a and the defender uses partial inference with E_d. Delving into the example displayed in Figure 3, we make the following observations:

Remark 3.1. *When* $E' \subset E$, $\text{Acc}(\theta, E, E) \leq \text{Acc}(\theta, E', E)$, i.e., *using fewer exits for attack than for inference weakens the attack. The unattacked exits in the ensemble can partially mitigate the attack received by other exits.*

Remark 3.2. *When* $E' \supset E$, $\text{Acc}(\theta, E, E) \leq \text{Acc}(\theta, E', E)$, i.e., *using more exits for attack than for inference also weakens the attack, for the impact is dispersed onto more inactive exits in inference.*

Remark 3.3. *When* $E' \cap E = \varnothing$, $\text{Acc}(\theta, E, E) \leq \text{Acc}(\theta, E', E)$. *Using completely different exits for attack from those for inference weakens the attack, for the exits for inference are not directly attacked. The decrease in accuracy is simply due to the shared model parameters or transferability.*

Remark 3.4. *When* $E' \cap E \neq \varnothing$, $|E' \cup E| > \max(|E'|, |E|)$, $\text{Acc}(\theta, E, E) \leq \text{Acc}(\theta, E', E)$. *Using exits partially different from those for inference weakens the attack, suffering from both the unattacked exits and the dispersed impact in Remark 3.1 and 3.2.*

Attack-Defense (A-D) mismatch happens when $E_d \neq E_a$ (correspondingly, A-D match when $E_d = E_a$), consisting of the 4 situations in the remarks above. Thereby we summarize them into a more concise assumption about the robust performance of multi-exit neural networks:

Assumption 3.5 (A-D Mismatch)**.** *We assume that when* $E' \neq E$, $\text{Acc}(\theta, E, E) \leq \text{Acc}(\theta, E', E)$.

It indicates that the most effective partial attack for a multi-exit neural network is to attack exactly the same ensemble of exits as the defender uses for inference, which is an A-D match case. As a result, for the defender, evading such precisely "matching" attacks and taking advantage of A-D mismatch makes a cunning yet indeed effective defensive

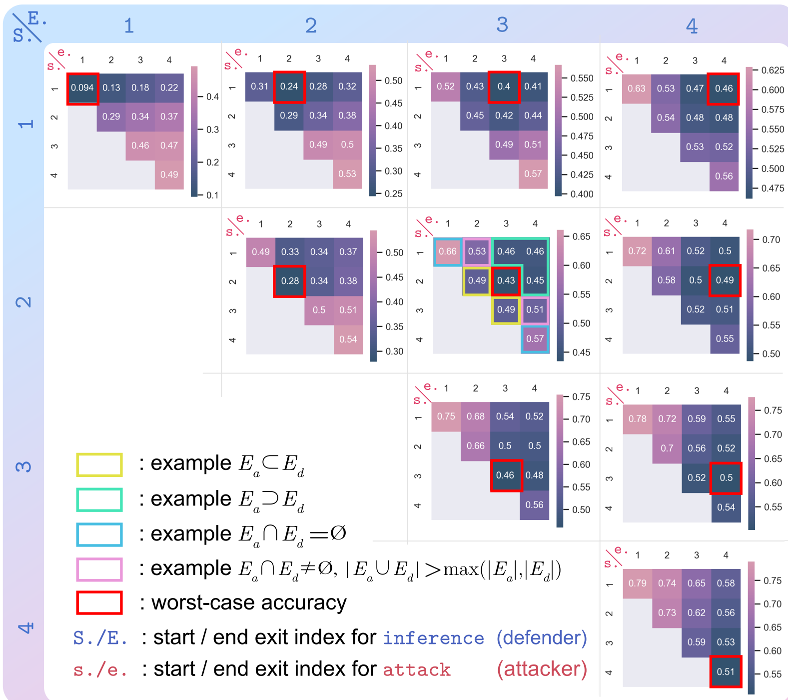


Figure 3. A demonstration of A-D mismatch. The accuracy scores of a 4-exit ResNet-18 under PGD-20 attack are plotted, enumerating all the cases in which the attacker and defender use continous exit ensembles.

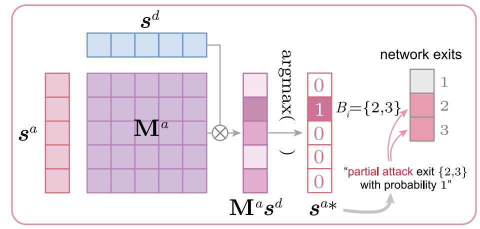


Figure 4. The principle of finding the best strategy for the attacker with AIMER.

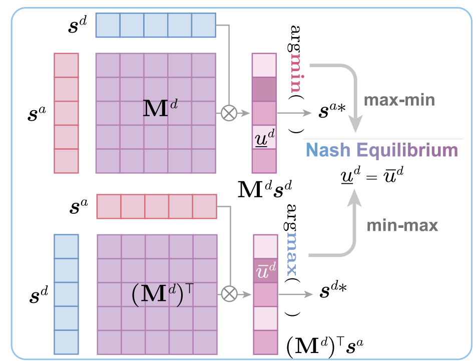


Figure 5. The principle of finding the best strategy for the defender in the Nash equilibrium with NEED.

tactic, which can provide an additional portion of robustness apart from the intrinsic robustness of the networks as depicted in Figure 1.

Despite the seemingly enhanced robustness thanks to A-D mismatch, the crux of the issue lies in that a savvy attacker, fully aware of the defender's strategy, will strive to minimize the occurrence of mismatch. However, the practice of using three fixed attack recipes to evaluate multi-exit neural networks, as demonstrated in previous works such as [3, 10, 16], has largely overlooked the impact of A-D mismatch. We believe this is problematic for evaluation since the additional robust scores brought by A-D mismatch obscure the intrinsic robustness of the networks.

3.3. Adaptive Evaluation of Multi-Exit Robustness

Through the phenomenon above, mismatched choice of exits between the attacker and defender is responsible for an overestimation in adversarial robustness. Therefore, to more accurately reflect the intrinsic robustness of multi-exit neural networks, a desirable evaluation should maximally hit exits used by d for inference. To this end, we attempt to provide a solution called AIMER that models the problem with game theory [35] and seeks the best strategy for the attacker to reduce the impact of A-D mismatch.

Model setup. The white-box adversarial attack and defense of multi-exit neural network θ can be modeled as a *two-player complete information static game* G, with a set of players $\mathcal{P} = \{a, d\}$, where a denotes the attacker and d denotes the defender. Suppose a performs partial attacks and d uses partial inference, then the action space for these two players is $\mathcal{A} = \{E : E \in \{n : n \in \{i\}_{i=1}^{L}\}, |E| > 0\}$. Note that $|\mathcal{A}| = 2^L - 1$, indicating each player has $2^L - 1$ types of actions to take.

Payoff functions. Since d endeavors to make the network reliable enough under attacks, *i.e.* to maximize the accuracy score, while a aims to do quite the opposite, the payoff function for a and d when a attacks E_a and d infers with E_d can be formulated into a zero-sum game:

$$\pi^a(E_a, E_d) = -\pi^d(E_a, E_d) = -\mathrm{Acc}(\theta, E_a, E_d) \quad (5)$$

The function values of π^a and π^d under every E_a and E_d constitute payoff matrices $\mathbf{M}^a, \mathbf{M}^d \in \mathbb{R}^{|\mathcal{A}| \times |\mathcal{A}|}$, and $\mathbf{M}^a = -\mathbf{M}^d$. In practical scenarios, due to the expensive cost to have complete tests of the network (see Appendix F.2), it is impossible to precisely understand the payoff matrix of this game. Therefore, we employ an approximation approach, selecting a subset of the dataset for an average attack to ascertain the characteristics of the network (Algorithm 3 in Appendix). Subsequent calculations are then based on this approximate matrix $\hat{\mathbf{M}}^a = -\hat{\mathbf{M}}^d \in \mathbb{R}^{|\mathcal{A}| \times |\mathcal{A}|}$.

The best strategy for the attacker. Aware of the payoff matrix $\hat{\mathbf{M}}^a$, a can calculate his best strategy in G with the following method. We default the strategy in G in a mixed form, *i.e.*, performing each action in $\mathcal{A}$ with a certain probability. Let the strategies of a and d be $\boldsymbol{s}^a = [s_1^a, s_2^a, \cdots, s_{|\mathcal{A}|}^a]^\top$ and $\boldsymbol{s}^d = [s_1^d, s_2^d, \cdots, s_{|\mathcal{A}|}^d]^\top$, which satisfy $\boldsymbol{s}^a, \boldsymbol{s}^d \in \mathcal{S}, \mathcal{S} = \{\boldsymbol{s} \in \mathbb{R}^{|\mathcal{A}|} : s_i \geq 0, \sum_{i=1}^{|\mathcal{A}|} s_i = 1\}$, representing the probability vector of a and d selecting the corresponding ensemble of exits. In such a case, the objective of each player is to maximize his own expected payoff $u^a(\boldsymbol{s}^a, \boldsymbol{s}^d) = (\boldsymbol{s}^a)^\top \hat{\mathbf{M}}^a \boldsymbol{s}^d$ or $u^d(\boldsymbol{s}^a, \boldsymbol{s}^d) = -u^a(\boldsymbol{s}^a, \boldsymbol{s}^d)$.

As shown in Figure 4, a maximizes his expected payoff via allocating all the probability to the action corresponding to the maximum value in vector $\hat{\mathbf{M}}^a \boldsymbol{s}^d$. Also, when multiple actions correspond to the same maximum value, a will not favor any particular one but will randomly choose among them. Therefore, a's best strategy can be represented as a probabilistic vector $\boldsymbol{s}^{a*} = [s_i^{a*}]_{i=1}^{|\mathcal{A}|}$, where

$$s_i^{a*} = \begin{cases} 1/|I^*|, & i \in I^* = \arg\max_{i \leq |\mathcal{A}|} (\hat{\mathbf{M}}^a \boldsymbol{s}^d)_i \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

and $(\cdot)_i$ denotes the i-th element of a vector. This formulation obtains the corresponding best strategy $\boldsymbol{s}^{a*}$ once the strategy of the defender $\boldsymbol{s}^d$ is given. We categorize the calculation of $\boldsymbol{s}^d$ into three cases (see Appendix D for details):

- For static inference, d simply uses a fixed exit or ensemble for inference: $\boldsymbol{s}^d = [0, \cdots, 0, 1, 0, \cdots, 0]^\top$, with the probability of corresponding action set to 1.
- For dynamic inference, d follows a certain rule to choose exits and it is difficult to model his strategy. In this case we test the frequency of using each exit under an average attack as a surrogate for $\boldsymbol{s}^d$.
- For random inference, d picks his exits or ensemble with $\boldsymbol{p}^d$. Since in a white-box setting $\boldsymbol{p}^d$ is common knowledge, we can easily obtain $\boldsymbol{s}^d = \boldsymbol{p}^d$.

With the best strategy of a, one can generate the adversarial example $x_{\text{AIMER}}^{\text{adv}}$ for AIMER evaluation (Algorithm 1 in Appendix) by the following formulation, where $\text{random}(C, \boldsymbol{p})$ denotes random choice from set C according to the probability distribution $\boldsymbol{p}$:

$$x_{\text{AIMER}}^{\text{adv}} = x_{\text{par},\text{random}(\mathcal{A}, \boldsymbol{s}^{a*})}^{\text{adv}} \tag{7}$$

3.4. Nash Equilibrium Enhanced Defense

From the attacker's perspective, A-D mismatch should be minimized to obtain a more accurate evaluation (Section 4.2); yet on the opposite side, A-D mismatch can also be utilized by the defender to confuse the attacker: avoiding the exits chosen by the attacker makes the gradients in the attacks less effective (see Appendix C.2 for detailed discussions). Specifically, under the strict evaluation of AIMER, the key to increasing the mismatch is to find the minimax point of the adversarial game. Thus, we intuitively devise the Nash Equilibrium Enhanced Defense (NEED) for multi-exit neural networks as a robust inference strategy for the defender. It seeks the Nash Equilibrium (NE) [25] of the adversarial game, where both players are performing their best strategies.

As shown in Figure 5, we formulate the NE in this two-player zero-sum attack-defense game with the following description. Given that in a zero-sum game, both players seek their best strategy while assuming the opponent is also using their best strategy (which is the most disadvantageous to each other), we can view this as an optimization problem aimed at maximizing the expected payoff in the worst-case scenario. Consider the defender's payoff as the objective to optimize, a is faced with a maximin problem, *i.e.*, while d strives to maximize the payoff, a seeks the tight lower bound $\underline{u}^d$ in the formula.

$$\begin{aligned} \min_{\boldsymbol{s}^a \in \mathcal{S}} (\boldsymbol{s}^a)^\top \hat{\mathbf{M}}^d \boldsymbol{s}^d &= \min_{i \leq |\mathcal{A}|} (\hat{\mathbf{M}}^d \boldsymbol{s}^d)_i \\ &= \max\{\underline{u}^d \in \mathbb{R} | \hat{\mathbf{M}}^d \boldsymbol{s}^d \succeq \underline{u}^d \cdot \mathbf{1}\} \end{aligned} \tag{8}$$

where $\mathbf{1} = [1, \cdots, 1]^\top \in \mathbb{R}^{|\mathcal{A}|}$. Conversely, the defender deals with a minimax problem seeking the tight upper bound $\overline{u}^d$ in the formula:

$$\begin{aligned} \max_{\boldsymbol{s}^d \in \mathcal{S}} (\boldsymbol{s}^a)^\top \hat{\mathbf{M}}^d \boldsymbol{s}^d &= \max_{i \leq |\mathcal{A}|} ((\hat{\mathbf{M}}^d)^\top \boldsymbol{s}^a)_i \\ &= \min\{\overline{u}^d \in \mathbb{R} | (\hat{\mathbf{M}}^d)^\top \boldsymbol{s}^a \preceq \overline{u}^d \cdot \mathbf{1}\} \end{aligned} \tag{9}$$

By the Minimax Theorem [34], the NE can be finally achieved in the two-player zero-sum attack-defense game, where the best strategy of the defender $\boldsymbol{s}^{d*}$ can be solved. Given the page limit, we encourage interested readers to consult Theorem A.3, its proof and the solution of NE provided in Appendix A, which systematically explain why $\boldsymbol{s}^{d*}$ is optimal and how it can be obtained.

4. Experiments

In this section, we first provide our experiment setup (Section 4.1). Next, we show our experimental results and analysis of the performance of AIMER evaluation scheme (Section 4.2) and NEED defense method (Section 4.3). Finally, we compare the computational cost of AIMER (Section 4.4). Due to page limit, more detailed settings and additional results are deferred to Appendix B and F.

4.1. Experiment Setup

Dataset and network architectures. We consider three datasets for evaluation: SVHN [26], CIFAR-10 [20] and Tiny ImageNet [5]. For multi-exit neural networks, we directly modify common networks into multi-exit versions, including VGG-16 [31], ResNet-18 [14], WideResNet-34-10 [46], ViT-B/16 [7] and employ existing multi-exit architectures MSDNet [17], RANet [43] and L2W-DEN [12].

Adversarial attacks. We apply the following 7 attack settings, with all of them restricted by l_∞ perturbation bound

Table 1. Robust accuracy scores (%) obtained by different evaluation schemes. The lowest scores of each column are set in **bold**. Results on more architectures/datasets, and under LAFIT [45] attack can be found in Appendix B.

Method	Network	Dataset	Evaluation	FGSM	PGD-20	PGD-100	EoT-PGD-20	VMI-FGSM	AutoAttack
Static (3/4)	ResNet-18 (4 exits)	CIFAR-10	Single attack	60.29 ± 0.00	56.04 ± 0.08	54.12 ± 0.12	54.42 ± 0.06	55.30 ± 0.05	59.34 ± 0.02
			Average attack	56.54 ± 0.00	52.09 ± 0.04	50.51 ± 0.04	50.72 ± 0.04	51.38 ± 0.05	56.49 ± 0.03
			Max-average attack	54.32 ± 0.20	50.26 ± 0.34	49.72 ± 0.05	48.96 ± 0.33	49.36 ± 0.29	55.92 ± 0.10
			AIMER (ours)	**52.81** ± 0.00	**45.85** ± 0.05	**43.21** ± 0.01	**43.60** ± 0.03	**45.10** ± 0.03	**42.40** ± 0.03
Random	VGG-16 (5 exits)	SVHN	Single attack	66.57 ± 0.19	52.77 ± 0.46	45.04 ± 0.04	47.60 ± 0.66	48.04 ± 0.66	46.63 ± 0.09
			Average attack	62.87 ± 0.07	47.49 ± 0.18	40.86 ± 0.06	42.42 ± 0.13	44.00 ± 0.09	43.42 ± 0.04
			Max-average attack	62.11 ± 0.07	45.90 ± 0.18	39.17 ± 0.04	40.40 ± 0.11	42.45 ± 0.09	46.31 ± 0.05
			AIMER (ours)	**60.87** ± 0.00	**43.61** ± 0.04	**37.70** ± 0.03	**38.55** ± 0.03	**40.87** ± 0.02	**40.70** ± 0.10
Dynamic [16]	VGG-16 (5 exits)	SVHN	Single attack	62.90 ± 0.00	44.81 ± 0.05	37.08 ± 0.04	39.23 ± 0.03	41.17 ± 0.03	39.59 ± 0.12
			Average attack	62.01 ± 0.00	44.81 ± 0.05	37.93 ± 0.05	39.23 ± 0.03	41.17 ± 0.03	39.97 ± 0.08
			Max-average attack	59.82 ± 0.00	43.53 ± 0.08	35.98 ± 0.06	37.78 ± 0.11	39.70 ± 0.02	39.50 ± 0.05
			AIMER (ours)	**59.65** ± 0.00	**42.30** ± 0.04	**35.69** ± 0.03	**37.30** ± 0.04	**39.56** ± 0.04	**38.56** ± 0.06
Dynamic [3]	MSDNet (5 exits)	CIFAR-10	Single attack	51.32 ± 0.00	42.31 ± 0.06	38.79 ± 0.05	39.34 ± 0.08	40.31 ± 0.05	35.21 ± 0.06
			Average attack	54.70 ± 0.00	43.65 ± 0.12	38.40 ± 0.05	39.71 ± 0.11	41.81 ± 0.03	35.10 ± 0.04
			Max-average attack	60.17 ± 0.00	48.96 ± 0.24	43.72 ± 0.03	44.98 ± 0.18	52.90 ± 0.32	34.22 ± 0.12
			AIMER (ours)	**51.04** ± 0.00	**40.47** ± 0.15	**33.73** ± 0.07	**35.53** ± 0.10	**37.69** ± 0.05	**33.64** ± 0.08
Dynamic [3]	ViT-B/16 [7] (4 exits)	CIFAR-10	Single attack	58.76 ± 0.00	56.71 ± 0.05	49.79 ± 0.04	56.31 ± 0.04	56.43 ± 0.03	59.55 ± 0.10
			Average attack	55.19 ± 0.00	52.09 ± 0.03	50.78 ± 0.06	50.91 ± 0.08	51.47 ± 0.05	56.58 ± 0.07
			Max-average attack	53.63 ± 0.00	50.57 ± 0.04	49.53 ± 0.05	49.88 ± 0.10	50.22 ± 0.06	55.29 ± 0.09
			AIMER (ours)	**53.45** ± 0.00	**50.47** ± 0.08	**49.10** ± 0.03	**49.67** ± 0.04	**50.17** ± 0.04	**54.98** ± 0.05

$\epsilon = 8/255$: FGSM [9], PGD-20 (which means PGD attack with 20 perturbation steps), PGD-100, EoT-PGD-20 [1], VMI-FGSM [36], AutoAttack [4], and LAFIT [45]. Among them, EoT-PGD addresses the stochastic behavior of the networks, while VMI-FGSM enhances the transferability of attacks among different network exits.

Evaluation protocol. Given the unique nature of multi-exit neural networks, we employ a distinct approach for evaluation compared to traditional networks. Overall, we assume that all attacks are conducted under the *white-box* setup in Section 3.1. The attacker first chooses his exit ensemble to generate the adversarial examples, and the defender then chooses his exit ensemble for inference. Considering the possible randomness in evaluation, we obtain the results by repeating the same test 5 times and report the mean value and standard deviation.

4.2. Evaluation with AIMER

This section primarily validates the effectiveness of the AIMER evaluation scheme. We select single attack (against the last exit; see Appendix B for other exits), average attack and max-average attack in [16] as our baselines, and use these four schemes to evaluate different adversarial defense methods for multi-exit neural networks including [3, 16]. Additionally, we construct two ad-hoc models with static and random inference strategies. For the static inference method, we use the third exit for inference; for the random inference method, we use the 3rd and 5th exits for inference, with probability $[0.5, 0.5]$ respectively.

An ideal scheme for robustness evaluation should maximally reduce the impact of A-D mismatch and achieve a lower accuracy than others. In Table 1, it can be observed that AIMER measures lower robustness compared with other evaluation schemes in all the cases. The margin is largest in evaluating static and random defense methods, reducing the robustness score under AutoAttack by 13.52% compared with max-average attack.

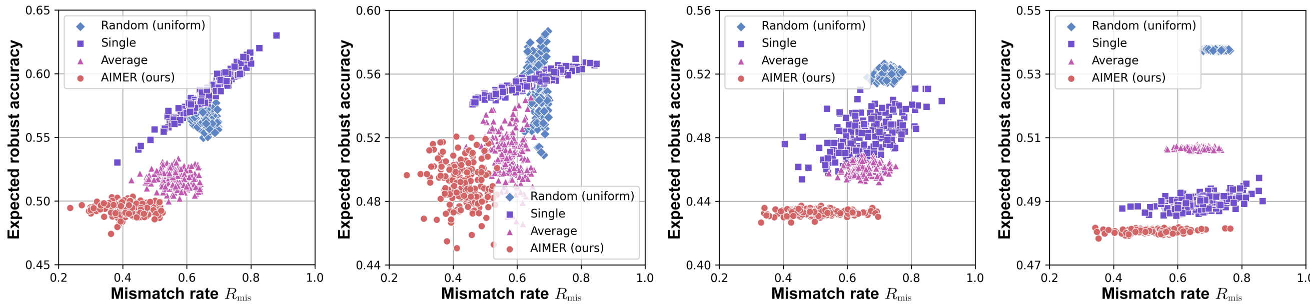
Noticeably, in the evaluation of [3], max-average attack does not necessarily outperform single or average attack. This reveals its limitation in averaging the adversarial loss on all exits, for it might be inconsistent with the objective of seeking the "best" single attack in some cases. Another key insight from the results is that AutoAttack is not necessarily the strongest attack against multi-exit neural networks, probably because the black-box ingredients in the algorithm fail to generalize on the unattacked inference exits.

We are also highly interested in whether AIMER can truly reduce the occurrence of A-D mismatch. Therefore, we first define the following metrics to measure the mismatch rate of two exit ensembles (r_{mis}) and two strategies (R_{mis}) for attack and defense:

$$r_{\text{mis}}(E_{a,i}, E_{d,j}) = 1 - \frac{|E_{a,i} \cap E_{d,j}|}{|E_{a,i} \cup E_{d,j}|} \tag{10}$$

$$R_{\text{mis}}(\boldsymbol{s}^a, \boldsymbol{s}^d) = \sum_{i=1}^{|\mathcal{A}|} \sum_{j=1}^{|\mathcal{A}|} s_i^a s_j^d r_{\text{mis}}(E_{a,i}, E_{d,j}) \tag{11}$$

Then we uniformly generate 200 random strategies for the defender, and find the corresponding strategies for the attacker using the following 4 schemes: (1) a random strategy that uniformly attacks every possible ensemble, (2) a single attack strategy only considering the main exit, (3)



(a) ResNet-18, SVHN (b) ResNet-18, CIFAR-10 (c) VGG-16, SVHN (d) VGG-16, CIFAR-10

Figure 6. Mismatch rate R_{mis} and expected robust accuracy of different attack schemes on different network architectures datasets. Expected robust accuracy is calculated with the payoff matrix.

Table 2. The Accuracy (%) of different defense strategies evaluated with AIMER. R_{mis} indicates the mismatch rate; Robust accuracy is obtained under PGD-20 attack. The best result of each column is set in **bold**.

Method	R_{mis}	Clean Acc.	Robust Acc.
Network: *ResNet-18 (4 exits)*, Dataset: *CIFAR-10*			
Single-exit	-	84.37 ± 0.00	49.82 ± 0.00
Multi-exit (static)	0.00	**84.83** ± 0.00	50.65 ± 0.04
Multi-exit (dynamic)	0.38	83.02 ± 0.00	51.30 ± 0.12
Multi-exit (NEED)	0.43	83.60 ± 0.00	**52.77** ± 0.30
Network: *VGG-16 (5 exits)*, Dataset: *SVHN*			
Single-exit	-	89.66 ± 0.00	43.35 ± 0.00
Multi-exit (static)	0.00	92.40 ± 0.00	42.33 ± 0.03
Multi-exit (dynamic)	0.36	92.19 ± 0.00	42.33 ± 0.04
Multi-exit (NEED)	0.46	**93.70** ± 0.00	**45.32** ± 0.16

an average strategy only considering using an ensemble of all exits, and (4) a strategy found by AIMER. Finally, we plot the mismatch rate R_{mis} and the expected robust accuracy of the strategy pairs in Figure 6. According to the results, AIMER greatly reduces both the robust accuracy and the mismatch rate compared with other schemes. Notably, none of the 4 schemes tested makes any change to the attack algorithms, which maintains the intrinsic robustness of the network. Therefore, we can assume that the robust accuracy reduced by AIMER is an extra portion that is closely related to the A-D mismatch phenomenon.

4.3. Defense with NEED

In this section, we primarily showcase the experimental results demonstrating the efficacy of NEED in enhancing the adversarial robustness of networks. We conducted tests on two different network architectures (ResNet-18 and VGG-16), assessing 4 scenarios for both clean accuracy and robust accuracy (where robust accuracy is obtained using AIMER evaluation based on PGD-20 attacks): (1) multi-exit neural networks using a static strategy for inference with the main exit; (2) multi-exit neural networks employing a dynamic strategy for inference; (3) multi-exit neural networks using the NEED method for inference; and (4) single-exit networks, where the AIMER evaluation reverts to standard evaluation.

Interestingly, according to the results in Table 2, with the aid of the NEED-enhanced multi-exit structure, the robust accuracy can even surpass that of single-exit networks. This suggests that NEED is not just a strategy for multi-exit neural networks but also has the potential to be a method for *enhancing general adversarial defense* performance through modifications in network structure.

To validate this viewpoint, we conduct further research, integrating it with various types of adversarial training methods including PGD-AT [22], TRADES [47], MART [38], and FAT [48]. Testing with different attack algorithms, the results in Table 3 consistently demonstrate stable improvements. Results on more network architectures and datasets can be found in Appendix B.

4.4. Computational Cost

This section compares the computational cost of AIMER and max-average attack that have similar performance in some cases. In Table 4, we list three aspects of the evaluation schemes: the per-example cost of each attack algorithm, the cost of pre-processing and the cost of a complete evaluation (including pre-processing and evaluation with every attack algorithm for 5 runs). From the results in the table, it is evident that AIMER has a significant advantage in both single-sample evaluation and overall evaluation time cost. This strongly indicates that AIMER is not only more accurate in reflecting the network's intrinsic adversarial robustness but also more cost-friendly.

5. Related Work and Discussions

Game theory for adversarial robustness. Game theory [35] has been applied in various fields of computer science [8, 21, 23], yet there is limited previous work looking into adversarial robustness from a game-theoretic perspective [28, 29]. The most recent work is [24], which models

Table 3. Accuracy (%) under different attacks when combining NEED with AT methods on the **VGG-16** model and **CIFAR-10** dataset. Better results are set in **bold**.

Method	Clean	FGSM	PGD-20	PGD-100	EoT-PGD-20	VMI-FGSM	AutoAttack
Standard	**92.24** $\pm$ 0.00	8.37 $\pm$ 0.00	0.11 $\pm$ 0.01	0.00 $\pm$ 0.00	0.08 $\pm$ 0.02	0.03 $\pm$ 0.00	0.00 $\pm$ 0.00
Standard + NEED	91.30 $\pm$ 0.20	**18.33** $\pm$ 2.29	**1.70** $\pm$ 0.44	**2.34** $\pm$ 0.31	**4.12** $\pm$ 0.19	**3.02** $\pm$ 0.28	**0.29** $\pm$ 0.03
PGD-AT [22]	**77.30** $\pm$ 0.00	52.50 $\pm$ 0.00	47.86 $\pm$ 0.03	46.13 $\pm$ 0.05	46.43 $\pm$ 0.03	47.17 $\pm$ 0.03	43.09 $\pm$ 0.02
PGD-AT + NEED	75.17 $\pm$ 0.12	**54.31** $\pm$ 0.16	**50.61** $\pm$ 0.11	**47.03** $\pm$ 0.09	**46.63** $\pm$ 0.14	**47.29** $\pm$ 0.24	**46.07** $\pm$ 0.15
TRADES [47]	79.26 $\pm$ 0.00	52.60 $\pm$ 0.00	47.30 $\pm$ 0.04	45.48 $\pm$ 0.03	45.83 $\pm$ 0.02	46.76 $\pm$ 0.02	42.06 $\pm$ 0.02
TRADES + NEED	**81.77** $\pm$ 0.04	**53.42** $\pm$ 0.07	**48.26** $\pm$ 0.07	**45.87** $\pm$ 0.16	**45.92** $\pm$ 0.05	**47.63** $\pm$ 0.06	**47.77** $\pm$ 0.24
MART [38]	**76.15** $\pm$ 0.00	51.10 $\pm$ 0.00	44.88 $\pm$ 0.03	42.44 $\pm$ 0.03	42.86 $\pm$ 0.06	44.01 $\pm$ 0.02	38.58 $\pm$ 0.02
MART + NEED	76.12 $\pm$ 0.12	**51.60** $\pm$ 0.08	**45.58** $\pm$ 0.06	**43.90** $\pm$ 0.04	**44.07** $\pm$ 0.08	**44.34** $\pm$ 0.11	**43.58** $\pm$ 0.07
FAT [48]	**83.65** $\pm$ 0.00	50.42 $\pm$ 0.00	44.73 $\pm$ 0.09	42.73 $\pm$ 0.03	42.84 $\pm$ 0.06	43.35 $\pm$ 0.04	38.03 $\pm$ 0.02
FAT + NEED	82.65 $\pm$ 0.19	**54.18** $\pm$ 0.30	**47.76** $\pm$ 0.12	**43.80** $\pm$ 0.19	**45.62** $\pm$ 0.28	**46.88** $\pm$ 0.23	**43.61** $\pm$ 0.10

Table 4. Computational cost (ms) of different methods on on **ResNet-18** backbone and **CIFAR-10** dataset. Experiments are conducted on a single NVIDIA RTX A6000. Lower results of each row are set in **bold**.

Evaluation Process		Max-average	AIMER
Single sample	FGSM	3.0964×10^0	$\mathbf{2.4472} \times 10^0$
	PGD-20	1.3540×10^1	$\mathbf{3.8754} \times 10^0$
	PGD-100	1.1438×10^2	$\mathbf{1.4013} \times 10^1$
	EoT-PGD-20	2.5308×10^1	$\mathbf{7.5075} \times 10^0$
	VMI-FGSM	7.3650×10^1	$\mathbf{1.9740} \times 10^1$
	AutoAttack	1.9590×10^3	$\mathbf{5.5118} \times 10^2$
Pre-processing		$\mathbf{0.0000} \times 10^0$	1.0187×10^5
Complete evaluation		1.0945×10^8	$\mathbf{3.0040} \times 10^7$

the attack and defense of randomized classifiers into a game and identifies the mixed Nash equilibrium [25]. Despite a similar framework of game theory, this paper for the first time studies the unexplored A-D mismatch problem and has essentially different motivation, purpose, methodology, and design of experiments from previous work, identifying a more direct application of theory to realistic problems. More discussion can be found in Appendix G.

Adversarial robustness of multi-exit neural networks. multi-exit neural networks for efficient inference [11, 12, 40, 43] and their adversarial robustness have attracted increasing research interest. [16] is the first to adversarially train an input-adaptive multi-exit neural network; [3] proposes a fast adversarial training method for multi-exit neural networks with reduced time complexity; [10] employs knowledge distillation to prompt each exit to produce orthogonal results. Unlike previous works, our paper focuses on the phenomenon of A-D mismatch that reveals the possible flaws in the evaluation of these works. Compared with existing techniques like EoT [1] that address the randomness in the networks, AIMER pioneers a novel perspective and a tailored solution for multi-exit neural networks (further discussion can be found in Appendix C.1). We believe that our findings and methodology can provide a more rigorous evaluation on the research of multi-exit robustness.

Adversarial training. Adversarial Training (AT) [22] has greatly advanced in recent years and has become the most widely researched defense against adversarial attacks. Improved regularization methods like TRADES [47], MART [38] and FAT [48] seek to achieve a better balance between accuracy and robustness; Methods like [18, 30] are devoted to faster AT. As a promising defense method, AT is also combined with other types of defenses such as feature-level robustness [6, 19, 39] and input transformation methods [44]. By default, the finding and methodology in this paper are based on AT-enhanced multi-exit neural networks.

6. Conclusion and Future Work

In this paper, we identify A-D mismatch as another source of adversarial robustness apart from the intrinsic robustness of multi-exit neural networks. Taking this finding into consideration, we devise a game-theoretically principled methodology for the adversarial attack and defense of multi-exit neural networks: AIMER evaluation with an enhanced strength of attack minimizes the mismatch and more accurately reflects the true adversarial robustness, while NEED defense under AIMER evaluation can still maximally confound the attacker with the best defense strategy in the game. The experimental results over different datasets, attack algorithms, and network architectures fully demonstrate the effectiveness of the methods.

While there exist several possible limitations in this work (Appendix H), we believe the problems identified and the methods proposed in this paper have not been thoroughly considered and empirically verified before, which provides a novel perspective for the research of adversarial robustness of multi-exit neural networks. Also, we expect that our work can serve as a constant reminder for researchers of adversarial defense methods: *when one seeks to prove the strength of his shield, he must sharpen the blade of offense.*

Acknowledgement

This work is supported in part by the National Natural Science Foundation of China (No. 62372339, 62371350, and 62372336). We would like to sincerely thank Prof. Xiang Sun from the Economics and Management School, Wuhan University for his insightful discussions and guidance. The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University.

References

[1] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018. 6, 8, 5, 10

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

[3] Sihong Chen, Haojing Shen, Ran Wang, and Xizhao Wang. Towards improving fast adversarial training in multi-exit network. *Neural Networks*, 150:1–11, 2022. 1, 3, 4, 6, 8

[4] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020. 6, 10

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 5

[6] Minjing Dong and Chang Xu. Adversarial robustness via random projection filters. In *CVPR*, 2023. 8

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 5, 6, 9

[8] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 7

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 1, 6

[10] Seokil Ham, Jungwuk Park, Dong-Jun Han, and Jaekyun Moon. Neo-kd: Knowledge-distillation-based adversarial training for robust multi-exit neural networks. In *NeurIPS*, 2023. 1, 3, 4, 8, 6

[11] Dong-Jun Han, Jungwuk Park, Seokil Ham, Namjin Lee, and Jaekyun Moon. Improving low-latency predictions in multi-exit neural networks via block-dependent losses. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. 1, 8

[12] Yizeng Han, Yifan Pu, Zihang Lai, Chaofei Wang, Shiji Song, Junfeng Cao, Wenhui Huang, Chao Deng, and Gao Huang. Learning to weight samples for dynamic early-exiting networks. In *ECCV*, 2022. 1, 5, 8, 3, 4, 6, 9

[13] Mirazul Haque, Anki Chauhan, Cong Liu, and Wei Yang. Ilfo: Adversarial attack on adaptive neural networks. In *CVPR*, 2020. 1

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5, 9

[15] Sanghyun Hong, Yiğitcan Kaya, Ionuţ-Vlad Modoranu, and Tudor Dumitraş. A panda? no, it's a sloth: Slowdown attacks on adaptive multi-exit neural network inference. In *ICLR*, 2020. 1

[16] Ting-Kuei Hu, Tianlong Chen, Haotao Wang, and Zhangyang Wang. Triple wins: Boosting accuracy, robustness and efficiency together by enabling input-adaptive inference. In *ICLR*, 2020. 1, 2, 3, 4, 6, 8, 10

[17] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q Weinberger. Multi-scale dense networks for resource efficient image classification. In *ICLR*, 2018. 5, 9, 10

[18] Xiaojun Jia, Yong Zhang, Baoyuan Wu, Jue Wang, and Xiaochun Cao. Boosting fast adversarial training with learnable adversarial initialization. *IEEE Transactions on Image Processing*, 31:4417–4430, 2022. 8

[19] Woo Jae Kim, Yoonki Cho, Junsik Jung, and Sung-Eui Yoon. Feature separation and recalibration for adversarial robustness. In *CVPR*, 2023. 1, 8

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

[21] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings 1994*, pages 157–163. Morgan Kaufmann, 1994. 7

[22] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICML*, 2018. 1, 7, 8, 5, 10

[23] Mohammad Hossein Manshaei, Quanyan Zhu, Tansu Alpcan, Tamer Bacşar, and Jean-Pierre Hubaux. Game theory meets network security and privacy. *ACM Computing Surveys (CSUR)*, 45(3):25, 2013. 7

[24] Laurent Meunier, Meyer Scetbon, Rafael B Pinot, Jamal Atif, and Yann Chevaleyre. Mixed nash equilibria in the adversarial examples game. In *ICML*, 2021. 7, 11

[25] John F. Nash. The nash equilibrium: A perspective. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950. 2, 5, 8

[26] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 5

[27] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. In *ICML*, 2022. 1

[28] Seong Joon Oh, Mario Fritz, and Bernt Schiele. Adversarial image perturbation for privacy protection a game theory perspective. In *ICCV*, 2017. 7

[29] Ambar Pal and René Vidal. A game theoretic analysis of additive adversarial attacks and defenses. In *NeurIPS*, 2020. 7

[30] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *NeurIPS*, 2019. 8

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 5, 9

[32] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. 1, 9

[33] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *ICPR*, 2016. 1

[34] J v. Neumann. Zur theorie der gesellschaftsspiele. *Mathematische annalen*, 100(1):295–320, 1928. 5

[35] John v. Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ, 1944. 2, 4, 7

[36] Xiaosen Wang and Kun He. Enhancing the transferability of adversarial attacks through variance tuning. In *CVPR*, 2021. 6, 10

[37] Xinglu Wang and Yingming Li. Harmonized dense knowledge distillation training for multi-exit architectures. In *AAAI*, 2021. 1

[38] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *ICLR*, 2019. 7, 8, 5

[39] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *CVPR*, 2019. 8

[40] Qunliang Xing, Mai Xu, Tianyi Li, and Zhenyu Guan. Early exit or not: Resource-efficient blind quality enhancement for compressed images. In *ECCV*, 2020. 8

[41] Keyizhi Xu, Zhan Chen, Zhongyuan Wang, Chunxia Xiao, and Chao Liang. Towards robust adversarial purification for face recognition under intensity-unknown attacks. *IEEE Transactions on Information Forensics and Security*, 2024. 1

[42] Keyizhi Xu, Yajuan Lu, Zhongyuan Wang, and Chao Liang. A survey of adversarial examples in computer vision: Attack, defense, and beyond. *Wuhan University Journal of Natural Science*, 30(1):1–20, 2025. 1

[43] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution adaptive networks for efficient inference. In *CVPR*, 2020. 1, 5, 8, 3, 4, 6, 9

[44] Xuwang Yin, Shiying Li, and Gustavo K Rohde. Learning energy-based models with adversarial training. In *ECCV*, 2022. 8

[45] Yunrui Yu, Xitong Gao, and Cheng-Zhong Xu. Lafit: Efficient and reliable evaluation of adversarial defenses with latent features. *IEEE TPAMI*, 2023. 6, 4, 10

[46] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 5, 3, 9

[47] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *ICML*, 2019. 1, 7, 8, 5

[48] Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In *ICML*, 2020. 7, 8, 5