PROARD: PROGRESSIVE ADVERSARIAL ROBUSTNESS DISTILLATION: PROVIDE WIDE RANGE OF ROBUST STUDENTS

Seyedhamidreza Mousavi
Mälardalen University
seyedhamidreza.mousavi@mdu.se

Seyedali Mousavi
Mälardalen University
seyedali.mousavi@mdu.se

Masoud Daneshtalab
Mälardalen University
masoud.daneshtalab@mdu.se

ABSTRACT

Adversarial Robustness Distillation (ARD) has emerged as an effective method to enhance the robustness of lightweight deep neural networks against adversarial attacks. Current ARD approaches have leveraged a large robust teacher network to train one robust lightweight student. However, due to the diverse range of edge devices and resource constraints, current approaches require training a new student network from scratch to meet specific constraints, leading to substantial computational costs and increased CO_2 emissions.

This paper proposes Progressive Adversarial Robustness Distillation (ProARD), enabling the efficient one-time training of a dynamic network that supports a diverse range of accurate and robust student networks without requiring retraining. We first make a dynamic deep neural network based on dynamic layers by encompassing variations in width, depth, and expansion in each design stage to support a wide range of architectures ($> 10^{19}$). Then, we consider the student network with the largest size as the dynamic teacher network. ProARD trains this dynamic network using a weight-sharing mechanism to jointly optimize the dynamic teacher network and its internal student networks. However, due to the high computational cost of calculating exact gradients for all the students within the dynamic network, a sampling mechanism is required to select a subset of students. We show that random student sampling in each iteration fails to produce accurate and robust students. ProARD employs a progressive sampling strategy that gradually reduces the size of student networks in three steps during training while applying robustness distillation between the dynamic teacher network and the selected students. Finally, we leverage a multi-objective evolutionary algorithm based on a proposed accuracy-robustness predictor to identify optimal architectures that balance accuracy, robustness, and efficiency.

Through the experiments, we show that ProARD reduces the computational cost by $60\times$ and improves accuracy and robustness by 13% and 14%, respectively, compared to random sampling. We also demonstrate that our accuracy-robustness predictor can estimate the accuracy and robustness of test student networks with root mean squared errors of 0.0073 and 0.0072, respectively.

Keywords Adversarial Robustness, Robustness Distillation, Dynamic Network

1 Introduction

Deep Neural Networks (DNNs) are highly effective in solving complex tasks, including image classification [38], object detection [39], and image segmentation [40]. Recently, DNNs have been widely used in security-critical applications, such as self-driving cars [41], face recognition [42], and medical diagnosis [70], where robustness to input perturbations is the primary concern. Despite their strengths, DNNs are susceptible to subtle and imperceptible changes in input data, referred to as adversarial attacks [43]. These vulnerabilities pose significant challenges for security-critical applications.

Different strategies have been suggested to counter adversarial attacks and enhance the robustness of DNNs. Among them, Adversarial Training (AT) is currently the most effective method for robustification [4]. AT methods aim to train an adversarially robust DNN such that its predictions are locally invariant to a small neighborhood of inputs. However, AT requires a well-designed, high-capacity network architecture, and its effectiveness is limited in lightweight networks due to their limited capacity [44]. To address this issue, the Adversarial Robustness Distillation (ARD) technique has been designed to transfer robustness from a high-capacity robust teacher network to a lightweight student network [79]. Current ARD methods are limited to training a single student network and primarily focus on finding the best training loss function based on the alignment between teacher and student networks [80]. Consequently, addressing the heterogeneity of edge devices and resource constraints requires retraining various network architectures tailored to each deployment scenario. Therefore, as the number of deployment scenarios (N) increases, the total cost of designing a lightweight network grows linearly (i.e. $O(N)$). This approach poses significant challenges for practical implementation across varying hardware configurations, while also incurring substantial computational costs and contributing to increased CO_2 emissions. Given this limitation, we pose the following question: *How can we take advantage of adversarial robustness distillation to provide a wide range of robust students efficiently without retraining them?*

This paper introduces a Progressive Adversarial Robustness Distillation (ProARD) training strategy to train a wide range of student network architectures suitable for different deployment scenarios. ProARD makes a dynamic deep neural network based on dynamic layers, including variations in width (kernel size), depth, and expansion in each design stage to support a wide range of architectures. To train this dynamic network, we consider the largest configuration of the dynamic network as the dynamic teacher and jointly train the dynamic teacher and students in a weight-sharing mechanism. However, due to the huge number of students in the dynamic network, it is computationally prohibitive to compute the exact gradient based on all students (more than 10^{19} students). We address this by sampling a subset of students by progressively reducing their size in each training iteration, and we show that this method works better than random sampling. After training the dynamic network by ProARD, we utilize a multi-objective evolutionary search algorithm alongside an accuracy-robustness predictor to identify the optimal student effectively. Our contributions can be summarized as follows:

- We present a novel method called Progressive Adversarial Robustness Distillation (ProARD), designed to train a dynamic network including a wide range of students by progressive sampling.
- We propose a joint accuracy-robustness predictor to efficiently estimate the robustness and accuracy, leveraging it in the multi-objective evolutionary algorithm to quickly find the best architecture based on the resource constraints without retraining.
- We extensively evaluate the distribution of the robustness and accuracy of students and the effectiveness of our accuracy-robustness predictor and multi-objective search mechanism on the CIFAR-10 and CIFAR-100 datasets with the dynamic networks based on ResNet and MobileNet architectures.

2 Related Work

2.1 Adversarial Attacks & Training

Adversarial attacks create subtle alterations in input data to intentionally mislead DNNs into making incorrect predictions. Common techniques for generating these attacks include the Fast Gradient Sign Method (FGSM) [1], Projected Gradient Descent (PGD) [3], and the Carlini & Wagner (CW) attack [12], among others [17]. Furthermore, AutoAttack (AA) [13], a set of four techniques combining white-box and black-box methods, has shown remarkable effectiveness in generating adversarial attacks. Adversarial training is the current state-of-the-art defense method against adversarial attacks, aiming to create robust DNNs. The earliest adversarial training approach leveraged both clean and adversarial images to train a robust DNN [1]. Adversarial training can be reformulated as a min-max optimization problem, where the network is trained exclusively on adversarial examples [3]. TRADES [4] regularizes the loss function for clean data by incorporating a robust loss term and making a trade-off between them. TRADES variants have also been proposed to consider regularization terms and reduce the distance between the distribution of natural data and their adversarial counterparts [5, 14]. Although adversarial training methods work well for high-capacity network architectures, their effectiveness is limited in lightweight networks due to their limited capacity [44].

2.2 Adversarial Robustness Distillation

Adversarial Robustness distillation (ARD) [11, 21, 22, 71, 74] is a knowledge distillation technique designed to transfer robustness from a large, robust network to a smaller, more efficient network. ARD [21] has shown that robustness distillation can produce a student with greater robustness than training from scratch. It encourages the student network
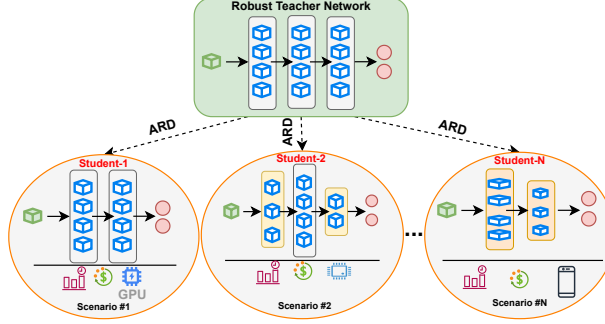
Figure 1: Adversarial Robustness distillation(ARD) for different deployment scenarios (GPU, CPU, Mobile, etc)

to mimic the teacher logits within an ϵ-ball around the data points. IAD [22] focuses on the reliability of the teacher and the student network trust in the teacher network based on the performance of the student on average and natural data. RSLAD [11] introduces the concept of robust soft labels (RSL) generated by the robust teacher, providing an effective and robust representation of the student network. PeerAiD [71] is an adversarial distillation approach that simultaneously trains the peer network and the student network, enabling the peer network to specialize in defending the student network. MTARD [67] utilizes multiple teachers to guide smaller networks in an adversarial setting. AdaAD [68] introduces an adaptive knowledge distillation method that also considers the teacher in the optimization process. SmaraAD [74] aligns the attribution regions of the student with those of the teacher network, facilitating a closer correspondence between the outputs of the teacher and student networks. Although adversarial robustness distillation methods provide a robust lightweight student network, these methods provide a single robust student network at each run and require the training process to be repeated for each scenario with different resource constraints. ProARD introduces a progressive robustness distillation approach to develop a dynamic robust network that encompasses a diverse set of robust student networks, adaptable to various deployment scenarios. ProARD works independently of specific robustness distillation methods; in this paper, we adopt RSLAD [11] as the baseline.

3 Research Motivation

Adversarial robustness distillation methods utilize a robust teacher network to train a robust student. However, the training process must be repeated for a variety of edge devices and deployment scenarios, which is both time-intensive and resource-demanding. As shown in Fig. 1, robustness distillation for different deployment scenarios requires retraining for each hardware device and set of resource constraints, resulting in significant computational costs and substantial CO_2 emissions.

To address this limitation, the primary motivation of ProARD is to train a robust dynamic network from which multiple student networks can be extracted without requiring retraining. To achieve this, we design a dynamic robust network with dynamic layers capable of varying widths (kernel sizes), depths, and expansion, enabling flexibility in its structure. The dynamic networks are built using bottleneck residual blocks (Dynamic ResNet) and inverted bottleneck blocks (Dynamic MobileNet), incorporating diverse configuration parameters at each design stage. For training, we first adversarially train the largest student network within the dynamic network using the TRADE [4] method. This largest network then serves as a dynamic teacher, guiding the training of randomly sampled student networks in each iteration. The student networks share their weights with the dynamic teacher network to ensure consistency. Given the enormous number of possible student networks within the dynamic network (exceeding 10^{19} networks), it is computationally infeasible to calculate exact gradients for all networks and share their weights. To overcome this, we randomly sample student networks during each training iteration. The red points in Fig. 2 depict the accuracy-robustness distribution of $2,000$ student networks trained via random sampling during each iteration within the Dynamic ResNet network on the CIFAR-10 dataset. The green point represents the robustness and accuracy of the ResNet-50 base network, which is adversarially trained using the TRADES [4] algorithm on the CIFAR-10 dataset. Most student networks extracted from the dynamic network trained using the random sampling strategy exhibit significantly lower accuracy and robustness compared to ResNet-50. The most accurate student network has 13.34% and 12.58% lower accuracy compared to ResNet-50. This indicates that the random sampling and weight-sharing strategy for training the dynamic network fails to produce robust and accurate student networks. To address this shortcoming, a new training strategy is necessary to enhance the robustness and accuracy of student networks. ProARD introduces a novel progressive robustness distillation approach designed to overcome these limitations and improve the performance of student networks. The blue points in
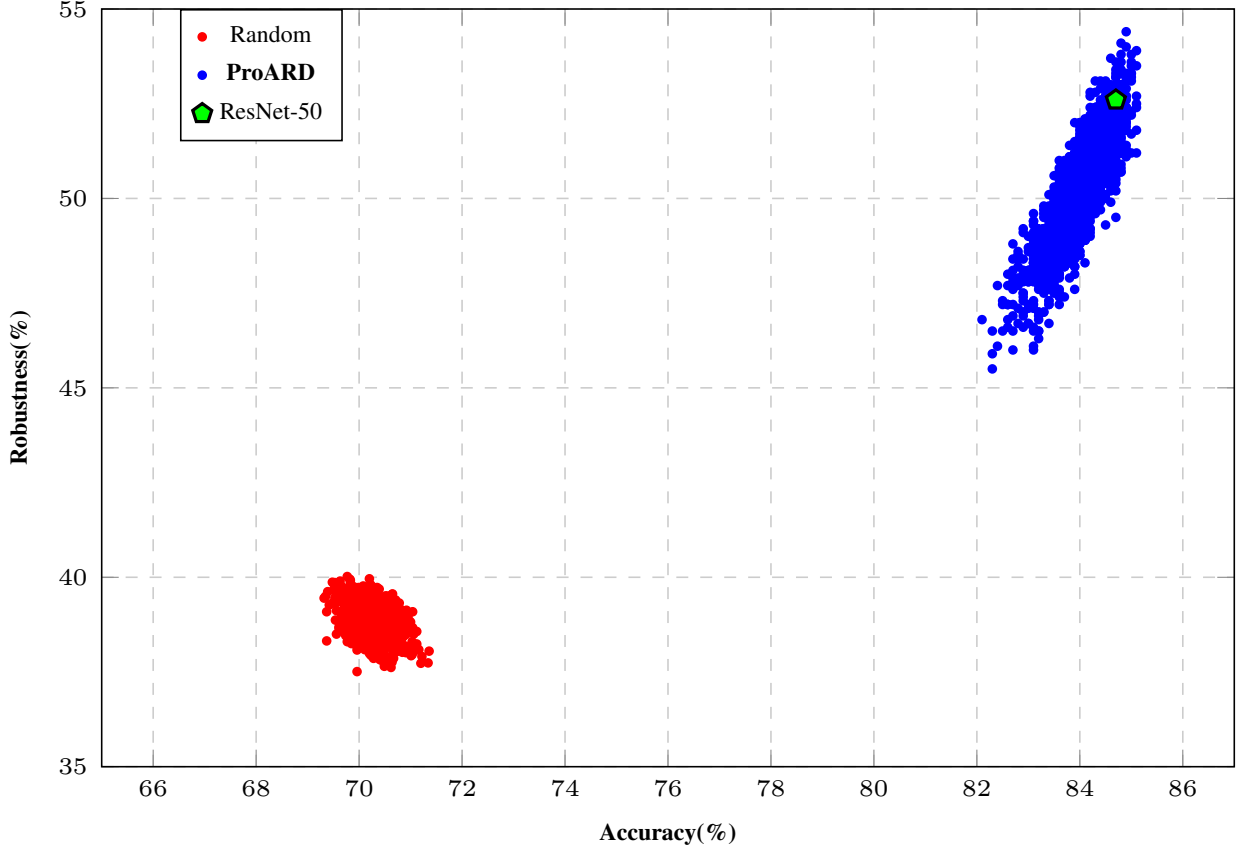
Figure 2: The accuracy-robustness distribution of students when trained with random sampling (red) and our ProARD (blue). For evaluation, we use the PGD attack with $\epsilon = 0.031$ and step-size $= 0.0078$. The green point shows the accuracy and robustness of the ResNet-50 on the CIFAR-10 dataset.

Fig. 2 show the distribution of $2,000$ student networks trained via our proposed method (ProARD) that can provide accurate and robust students.
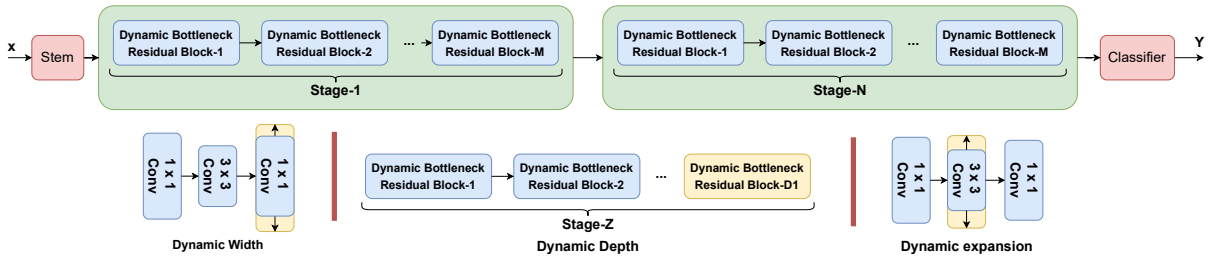
4 Method

Figure 3: The architecture of Dynamic ResNet with dynamic bottleneck residual blocks. Each dynamic bottleneck residual block has dynamic width and expansion and each stage has a dynamic depth.

4.1 Preliminaries

Adversarial training can be formulated as the following min-max optimization problem:

$$\underbrace{\min_{\mathbf{w}} \mathbb{E}_{(x,y)\sim P}\mathcal{L}_{\min}(f_{\mathbf{w}}(x+\delta), y)}_{\textbf{Outer minimization}}$$

$$\text{s.t.} \quad \delta = \underbrace{\max_{||\delta||_p \leq \epsilon} \mathbb{E}_{(x,y)\sim P}\mathcal{L}_{\max}(f_{\mathbf{w}}(x+\delta), y)}_{\textbf{Inner maximization}} \tag{1}$$

Where P is the data distribution, $(x \in \mathcal{X}, y \in \mathcal{Y})$ is a pair of input features and labels, $f_{\mathbf{w}}(.)$ is the DNN with parameters $\mathbf{w}$, and δ is the perturbation within the bounded l_p distance (ϵ). $\mathcal{L}_{\min}$ and $\mathcal{L}_{\max}$ are the loss functions for the inner and outer optimization problems, respectively. The Cross-Entropy (CE) loss function is specified as $\mathcal{L}_{\min}$, while $\mathcal{L}_{\max}$ may vary depending on the chosen method. For example, SAT [3] and TRADE [4] methods employ cross-entropy loss and KL divergence as $\mathcal{L}_{\max}$.

Adversarial robustness distillation (ARD) has been introduced to take advantage of the performance of a robust teacher network. It first trained a large capacity teacher network $T(.)$ with adversarial training following Eq. 1, then the teacher used to provide soft labels with natural data (x_i) and adversarial example ($x_i + \delta$) of the student network $S(.)$. By using different loss functions for $\mathcal{L}_{\min}$ and $\mathcal{L}_{\max}$, we can formulate different robustness distillation methods. However, they can train only one student network, and the distillation process must be repeated for each deployment scenario. Our method is independent of the robustness distillation methods and we used the RSLAD [11] adversarial distillation which considers the following loss functions for $\mathcal{L}_{\min}$ and $\mathcal{L}_{\max}$:

$$\mathcal{L}_{\min} = (1-\alpha) \cdot \mathbf{KL}(S(x), T(x)) + \alpha \cdot \mathbf{KL}(S(x+\delta), T(x))$$
$$\mathcal{L}_{\max} = \mathbf{KL}(S(x+\delta), T(x)) \tag{2}$$

Where $\mathbf{KL}$ is the KL-divergence. $S(x)$ and $T(x)$ indicate the student and teacher with parameters $\boldsymbol{w_s}$ and $\boldsymbol{w_t}$ respectively.

4.2 Problem Formulation

DNNs are organized into multiple design stages, with each stage consisting of a specific arrangement of layers. A dynamic network architecture can be constructed by varying configuration parameters such as width (kernel size), depth, and expansion across these stages and layers. Assuming the largest dynamic network architecture, configured with the maximum values for these parameters, serves as the dynamic teacher with trainable parameters $\boldsymbol{w_t}$. The dynamic network encompasses a diverse set of student architectural configurations, denoted as $\mathcal{S}$, from which various student networks s can be selected. The primary objective is to train a robust dynamic teacher network and utilize a robustness distillation method to ensure robustness across all students. To achieve a wide range of robust student networks, we aim to solve the following optimization problem:

$$\min_{\boldsymbol{w_t}} \mathbb{E}_{s\in\mathcal{S}}\mathbb{E}_{(x,y)\sim P}\mathcal{L}_{\min}(s^{\boldsymbol{w_t^s}}(x+\delta), y)$$

$$\text{s.t.} \quad \delta = \max_{||\delta||_p \leq \epsilon} \mathbb{E}_{(x,y)\sim P}\mathcal{L}_{\max}(s^{\boldsymbol{w_t^s}}(x+\delta), y)) \tag{3}$$

Where $s^{\boldsymbol{w_t^s}}$ refers to trainable parameters of the student network chosen according to the s configuration and a specific selection scheme. The primary goal of the training process is to adversarially optimize the dynamic network so that every student architecture achieves a similar level of accuracy and robustness. To solve the optimization problem (3), two approaches can be considered. The first approach involves selecting and training all the student networks within the dynamic network from scratch. However, with over 10^{19} possible student networks, this solution is computationally infeasible. Moreover, as demonstrated in the motivation section, randomly selecting student networks for training fails to achieve satisfactory accuracy and robustness. To address these challenges, a novel method is required to train both the dynamic network and its student networks effectively. The proposed solution involves constructing a dynamic network and progressively selecting student networks from it during each training iteration, as described in the following sections.

4.3 Dynamic Network Architecture

To support a diverse range of student network architectures, we made a dynamic network based on dynamic layers which cover different networks based on three dimensions (i.e., depth, width (kernel size), and expansion) in a deep neural
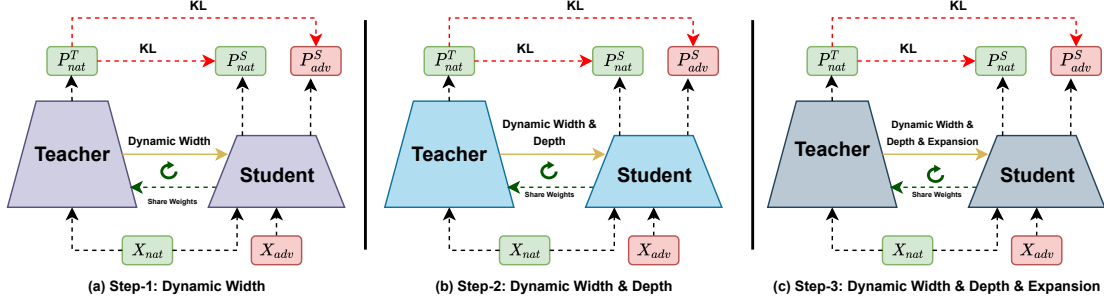
Figure 4: ProARD: Three steps Progressive Adversarial Robustness Distillation Framework. (a) Step-1: train dynamic width (b) Step-2: train dynamic width and depth, and (c) Step-3: train dynamic width, depth, and expansion.

network. We construct a Dynamic ResNet by incorporating variations in width, depth, and expansion in bottleneck residual blocks, while we make Dynamic MobileNet by utilizing kernel size, depth, and expansion in inverted bottleneck blocks. Following the common practice of DNN networks, we divide each DNN into a sequence of different stages with multiple layers that gradually reduce the size of the feature map and increase the number of channels. To make a dynamic network architecture, we allow each stage to have an arbitrary number of depths (dynamic depth). We allow each layer to have an arbitrary number of channels, arbitrary width (for Dynamic ResNet) and kernel size (for Dynamic MobileNet) (dynamic expansion, dynamic width, and dynamic kernels) in each stage. Fig 3 indicates the architecture of our Dynamic ResNet based on the dynamic bottleneck residual blocks. We can consider different widths and expansions by altering the number of output channels in the last conv and middle conv layers in the bottleneck residual block and using different depths by reducing the number of blocks in each stage. With this design for a DNN with 5 stages, if we select the depth of each stage from $\{2, 3, 4\}$ and use 3 different values for the width and expansion of each layer in the stages, then we have roughly $((3 \times 3)^2 + (3 \times 3)^3 + (3 \times 3)^4)^5 \approx 2 \times 10^{19}$ student network architectures in our dynamic network. The architecture of Dynamic MobileNet is similar to that in Fig 3, using inverted bottleneck blocks. The only modification is replacing the dynamic width with a dynamic kernel size in this network.

4.4 Progressive Adversarial Robustness Distillation

To efficiently train student networks within the dynamic network, we propose Progressive Adversarial Robustness Distillation (ProARD), which enables the joint training of the dynamic teacher network—the largest student network within the dynamic architecture—and a wide range of student networks. In ProARD, we start by selecting the maximum values for the configuration parameters of the dynamic network to construct the largest network. We adversarially train the largest network and utilize it as the dynamic teacher network for distillation. To overcome the sampling challenges discussed in the motivation section, we adopt a progressive adversarial robustness distillation approach that progressively samples student network architectures from large to small in three steps. In the first step, we fix the depth and expansion, extracting different student networks by varying the width (kernel size) of the bottleneck residual block (inverted bottleneck block) in each iteration. Robustness distillation based on the RSLAD (Eq 2) is applied between the dynamic teacher and students. After training, the student parameters are shared with the dynamic teacher network. In the second step, we fix the expansion while varying the width (kernel size) and depth to train student networks using robustness distillation, then share the weights with the dynamic teacher network. In the third step, we extract student networks by varying all three parameters, train them, and apply the weight-sharing. Fig 4 illustrates the progressive robustness distillation mechanism used to build a dynamic network that supports a wide range of robust and lightweight student networks.

4.5 Multi-objective Search

After training the dynamic network using ProARD, we obtain a diverse set of robust student networks within the dynamic teacher network. The next step is to identify the best student network based on the given resource constraints and deployment scenario. We employ a multi-objective search mechanism that takes robustness, accuracy, and efficiency (i.e., number of FLOPs) into account to identify the optimal student network. However, computing robustness and accuracy during the search process is time-consuming, making it impractical to evaluate robustness for a wide range of students. To address this challenge, we propose an accuracy-robustness predictor network that provides quick feedback for the search process. Specifically, we sample $2K$ student network architectures from the dynamic network and measure their robustness and accuracy by directly evaluating them. This dataset ([student architecture, robustness, accuracy]) is then used to train a predictor that can estimate the robustness and accuracy of any given student network.
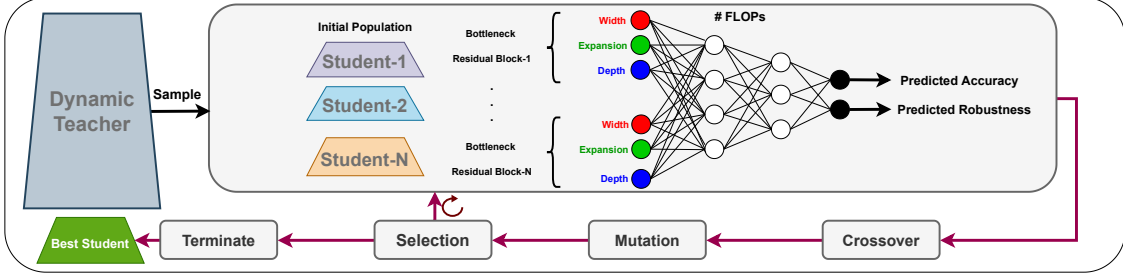
Figure 5: Multi-objectice Search engine consists of our proposed accuracy-robustness predictor

For the specification of the student architectures, we leverage the width (kernel size), depth, and expansion values in each stage of design and layers. With the predictor in place, the multi-objective search process can efficiently find the best student architecture for a given deployment scenario, considering FLOPs, robustness, and accuracy. By training the predictor just once, we reduce the search cost, and it remains constant regardless of the deployment scenario. Fig 5 illustrates the architecture of our accuracy-robustness predictor and the multi-objective search to identify the best student network.

5 Exprimental Results

In this section, we compare random sampling-based robustness distillation training with our proposed ProARD approach for addressing the optimization problem outlined in Eq. 3. We then demonstrate the effectiveness of our accuracy-robustness predictor within the multi-objective evolutionary algorithm, showing that our method can efficiently identify the optimal student. Additionally, we assess the adversarial robustness and accuracy of the optimal student architecture under various white-box attacks. We use TRADES [4] and RSLAD [11] as baselines for adversarial training and distillation. However, our training approach remains independent of the choice of the training and distillation methods. We design dynamic networks using bottleneck residual blocks and inverted bottleneck blocks to support both ResNet and MobileNet architectures. In this work, the dynamic ResNet network is parameterized by Depth (D), Width (W), and Expansion (E), while for the dynamic MobileNet network, we replace the width with Kernel size(K) in its configuration parameters. The experiments are conducted on the CIFAR-10 and CIFAR-100 datasets.

5.0.1 Training and Evaluation Details

To train the dynamic network, we first adversarially train the largest dynamic teacher network using the TRADES [4] method for 300 epochs. Next, we allocate 120 epochs for each step of training dynamic width (kernel size), depth, and expansion for the student networks, sharing the weights with the dynamic network. The training process uses the stochastic gradient descent (SGD) optimizer with an initial learning rate of 0.01, a momentum of 0.9, and a weight decay of $2e-4$. The batch size is set to 128. For the PGD attack used during TRADES training, we adopt the l_∞ norm with 10 iterations, a step size of $2/255$, and $\epsilon = 8/255$. In ProARD, we consider a list of values for one configuration parameter (e.g., depth, width (kernel size), and expansion) at each stage of the dynamic network. For instance, we consider $W = \{0.65, 0.8, 1.0\}$, $D = \{0, 1, 2\}$, and $E = \{0.2, 0.25, 0.35\}$ for a dynamic ResNet network. For the dynamic MobileNet network, we used $K = \{3, 5, 7\}$ instead of width in the stage design. Our method is agnostic to the choice of loss function used in robustness distillation. For this study, we employ RSLAD [11], which leverages robust soft-label adversarial distillation. After training the dynamic network, we evaluate each student's architecture against 2 adversarial attacks: FGSM, PGD20. Maximum perturbation used for evaluation is also set to $\epsilon = 8/255$ and 20 perturbation iterations.

5.0.2 Multi-objective Search

For the multi-objective search, we use the NSGA-II evolutionary algorithm, which leverages non-dominated sorting and crowding distance mechanisms to tackle the optimization problem. The configuration parameters in each stage and layer of each architecture are encoded as a genotype representation. The objective function combines predicted robustness and predicted accuracy, while the search is constrained by the number of FLOPs. Given the computational complexity of directly evaluating robustness and accuracy, we develop an accuracy-robustness predictor based on a simple fully-connected neural network. To train the predictor, we generate a dataset comprising 2,000 samples of diverse network architectures and compute their robustness and accuracy under specific adversarial attack settings (PGD ($\epsilon = 0.031, iter = 20, step-size = 0.0071$). The predictor is trained to efficiently estimate the robustness and

accuracy of any given architecture configuration. Using NSGA-II, we identify the optimal student network architecture that strikes the best balance between robustness and accuracy, subject to a specified number of FLOPs constraint.

5.1 Random sampling vs Progressive

We compare the random sampling robustness distillation approach that randomly selects some student networks and solves the optimization problem(Eq. 3) to train the dynamic network with our ProARD method. In Fig 7 we plot the accuracy-robustness distribution of 1000 different student architectures extracted from Dynamic ResNet and Dynamic MobileNet on the CIFAR-10 dataset after training. ProARD significantly improves the accuracy and robustness of the student networks in comparison with random sampling. ProARD achieves a 13% and 10% accuracy improvement by selecting the most accurate student and a 14% and 5% robustness improvement when considering the most robust student network in both dynamic networks. Fig 6 reports the accuracy and robustness of student networks extracted from Dynamic ResNet trained on the CIFAR-10 dataset with ProARD and random sampling. Due to space limitations, we select 3 students for comparison by varying the width of the dynamic ResNet with fixed depth and expansion. ProARD yields better accuracy and robustness for the selected students compared to random sampling. For example, the student with $D = 2, E = 0.35,$ and $W = 0.65$ delivers nearly the same accuracy and robustness as the largest network (green line), but with fewer parameters. Fig 8 shows the total cost required for training student networks across 50 different deployment scenarios (edge devices). ProARD reduces the total cost (GPU hours) by a factor of 60 compared to RSLAD by training a dynamic network and efficiently extracting students for each device.

5.2 Accuracy-Robustness Predictor

For the multi-objective evolutionary search, a fitness function is required. However, evaluating accuracy and robustness for different architectures is time-consuming. To address this, we train a prediction model based on a fully connected network to predict the accuracy and robustness of each architecture. To create a dataset for training, we extract 2000 student networks from the dynamic network, generate a feature vector based on the width (kernel), depth, and expansion of each stage, and use this vector as the features of the architecture. We then evaluate the accuracy and robustness of these networks, using them as labels for training. The fully connected network is trained for 30 epochs. We compare the accuracy of our predictor across 300 different architectures, evaluating the actual accuracy and robustness of each and comparing them with the predicted values. Fig 9 shows the difference between the real accuracy and robustness and the predicted values. Our predictor can estimate accuracy and robustness for test architectures with a root mean squared error (RMSE) of 0.0076 for accuracy and 0.0072 for robustness.
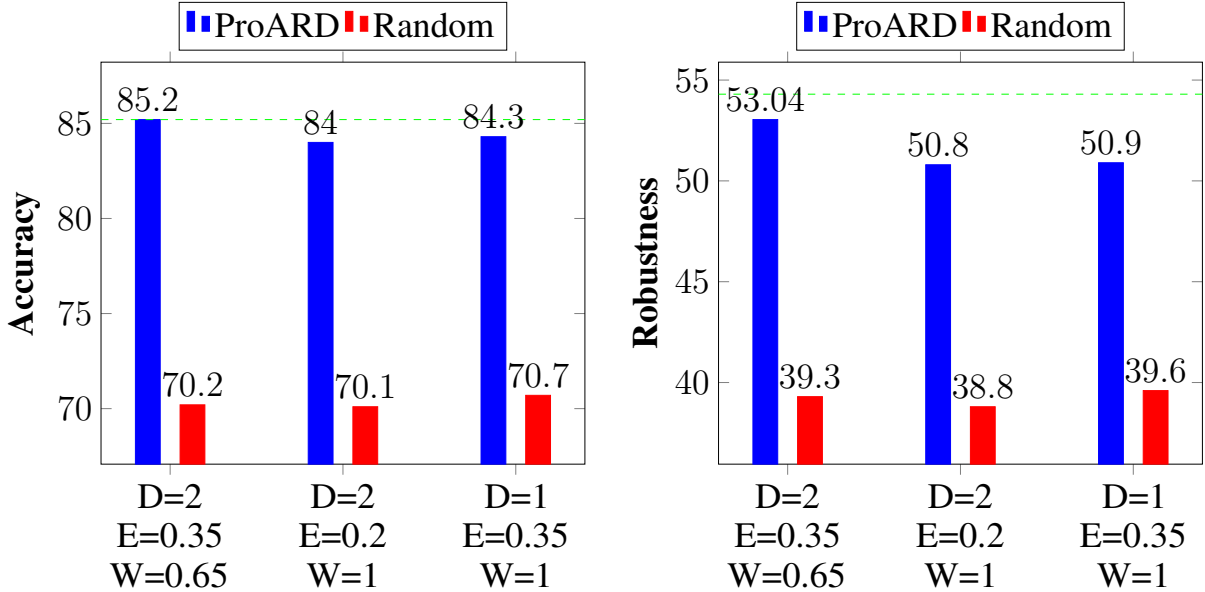

Figure 6: (Left) Accuracy and (Right) Robustness for three students extracted from the Dynamic ResNet on the CIFAR-10 dataset. The green line shows the performance of largest student network architecture trained with TRADES [4]
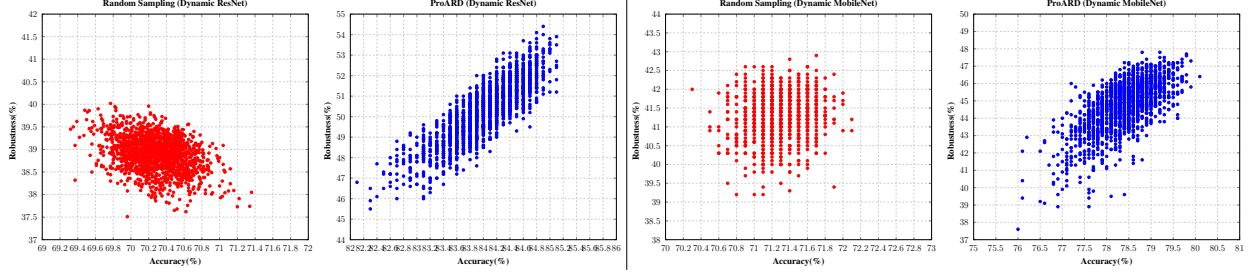
Figure 7: The accuracy-robustness distribution of students for dynamic networks with random sampling and ProARD on CIFAR-10 dataset: (Left) Dynamic ResNet, (Right) Dynamic MobileNet.

5.3 Search student networks

To find the best architecture based on accuracy, robustness, and a given FLOPs constraint, we use NSGA-II as the multi-objective search framework. We begin with a random initial population and perform 100 iterations with a mutation rate of 0.1 to identify the optimal student architecture. To accelerate the search process, we use the predictor trained in the previous step to evaluate all individuals in the population. Fig 10 shows the accuracy and robustness of the first population and the final student networks after 100 generations. NSGA-II successfully identifies the best architecture, achieving better robustness and accuracy while maintaining the same number of FLOPs. In this case, we set our FLOPs constraint to be less than 2000. We choose FLOPs as a metric because it is independent of hardware-specific architectural details and effectively reflects how well a system can leverage parallelism, a fundamental characteristic of modern accelerators.

5.4 White-box Attacks

We evaluate the accuracy and robustness of the best student network identified through the multi-objective search for white-box attacks and present the results in Table 1 for CIFAR-10 and CIFAR-100 datasets. We report the best robustness validated by FGSM and PGD-20 attack methods. We utilize Dynamic ResNet and Dynamic MobileNet as the dynamic architectures. These results suggest that ProARD delivers equal or superior accuracy and robustness for student networks with the same number of FLOPs, without requiring retraining, by simply performing a quick search within the dynamic network. ProARD identifies models with a higher number of parameters while maintaining the same FLOPs, effectively increasing model capacity. This aligns with previous research suggesting that increasing network capacity can enhance robustness [82].

6 Conclusion

In this paper, we address the challenge of training a vast number of student networks (over 10^{19}) for robustness distillation. We introduce a dynamic network that supports diverse configurations, including variations in depth, width (kernel size), and expansion. Training each student individually is computationally infeasible, and random sampling combined with weight sharing fails to produce accurate and robust networks. To overcome these limitations, we propose Progressive Adversarial Robustness Distillation (ProARD), a novel approach that efficiently trains a dynamic network to generate robust student networks without retraining. ProARD employs a progressive sampling during training, coupled with a multi-objective search powered by an accuracy-robustness predictor, to quickly identify optimal architectures
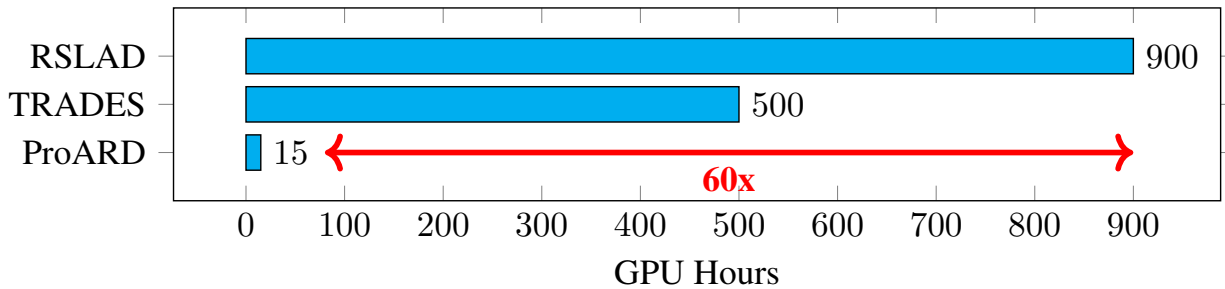

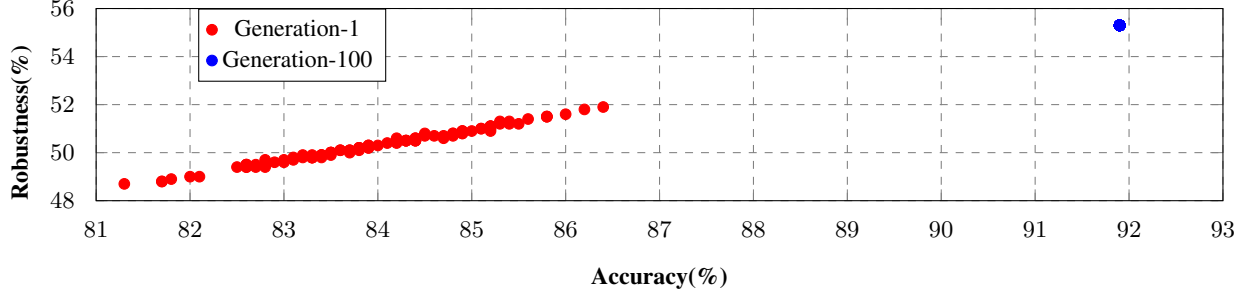Figure 8: Total Cost (GPU Hours) For 50 edge devices

Figure 10: Representation of the first-generation and final student networks found by NSGA-II, based on our accuracy-robustness predictor, for a fixed FLOPs constraint in the dynamic ResNet on the CIFAR-10 dataset.

tailored to specific resource constraints. Our results demonstrate that ProARD reduces training costs while delivering students with higher accuracy and improved robustness compared to random sampling. ProARD offers an efficient framework for training robust dynamic networks, enabling the rapid selection of optimized architectures for various hardware constraints without the need for retraining. This significantly reduces computational costs and supports scalable deployment across diverse devices.

References

[1] Goodfellow, I., Shlens, J. & Szegedy, C. Explaining and harnessing adversarial examples. *ArXiv Preprint ArXiv:1412.6572*. (2014)

[2] Kannan, H., Kurakin, A. & Goodfellow, I. Adversarial logit pairing. *ArXiv Preprint ArXiv:1803.06373*. (2018)

[3] Madry, A., Makelov, A., Schmidt, L., Tsipras, D. & Vladu, A. Towards deep learning models resistant to adversarial attacks. *ArXiv Preprint ArXiv:1706.06083*. (2017)

[4] Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L. & Jordan, M. Theoretically principled trade-off between robustness and accuracy. *International Conference On Machine Learning*. pp. 7472-7482 (2019)

[5] Cui, J., Liu, S., Wang, L. & Jia, J. Learnable boundary guided adversarial training. *Proceedings Of The IEEE/CVF International Conference On Computer Vision*. pp. 15721-15730 (2021)

[6] Zhang, J., Xu, X., Han, B., Niu, G., Cui, L., Sugiyama, M. & Kankanhalli, M. Attacks which do not kill training make adversarial learning stronger. *International Conference On Machine Learning*. pp. 11278-11287 (2020)

[7] Xie, C. & Yuille, A. Intriguing properties of adversarial training at scale. *ArXiv Preprint ArXiv:1906.03787*. (2019)
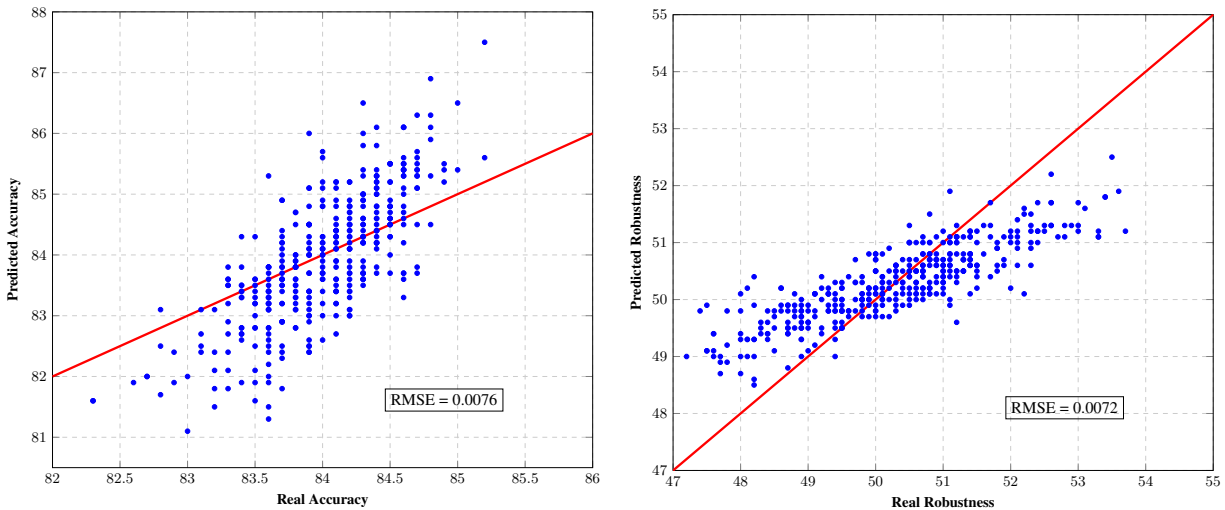
Figure 9: Evaluation performance of the accuracy-robustness predictor on 300 different test student architectures from Dyanmic ResNet on CIFAR-10 dataset.

Table 1: Accuracy and white-box robustness results on CIFAR-10 and CIFAR-100 datasets. The best results are **blodfaced**. For RSLAD, we used WideResNet-34-10 as the teacher network

Dataset	Networks	#Params(M)	Methods	Natural Acc.	FGSM	PGD20	Training
CIFAR10	ResNet50	23.52	TRADES	84.7	59.7	52.6	Yes (train from scratch)
	MBV3	4.21	TRADES	80.0	52.2	48.6	Yes (train from scratch)
	ResNet50 (WideResNet)	23.52	RSLAD	86.0	59.4	52.6	Yes (train from scratch)
	MBV3 (WideResNet)	4.21	RSLAD	**81.2**	54.5	50.5	Yes (train from scratch)
	Dyn-Resnet50	24.94	**ProARD**	**87.0**	**62.8**	**54.2**	**No (quick search)**
	Dyn-MBV3	5.28	**ProARD**	80.6	**55.0**	**50.9**	**No (quick search)**
CIFAR100	ResNet50	23.71	TRADES	54.6	28.7	25.8	Yes (train from scratch)
	MBV3	4.33	TRADES	53.4	28.9	27.2	Yes (train from scratch)
	ResNet50 (WideResNet)	23.71	RSLAD	55.3	**29.6**	**26.5**	Yes (train from scratch)
	MBV3 (WideResNet)	4.33	RSLAD	54.9	**29.1**	**28.7**	Yes (train from scratch)
	Dyn-Resnet50	27.24	**ProARD**	**60.1**	29.4	25.9	**No (quick search)**
	Dyn-MBV3	5.41	**ProARD**	**55.3**	28.6	27.8	**No (quick search)**

[8] Huang, H., Wang, Y., Erfani, S., Gu, Q., Bailey, J. & Ma, X. Exploring architectural ingredients of adversarially robust deep neural networks. *Advances In Neural Information Processing Systems*. **34** pp. 5545-5559 (2021)

[9] Rice, L., Wong, E. & Kolter, Z. Overfitting in adversarially robust deep learning. *International Conference On Machine Learning*. pp. 8093-8104 (2020)

[10] Cai, H., Gan, C., Lin, J. & Han, S. Network augmentation for tiny deep learning. *ArXiv Preprint ArXiv:2110.08890*. (2021)

[11] Zi, B., Zhao, S., Ma, X. & Jiang, Y. Revisiting adversarial robustness distillation: Robust soft labels make student better. *Proceedings Of The IEEE/CVF International Conference On Computer Vision*. pp. 16443-16452 (2021)

[12] Carlini, N. & Wagner, D. Towards evaluating the robustness of neural networks. *2017 Ieee Symposium On Security And Privacy (sp)*. pp. 39-57 (2017)

[13] Croce, F. & Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. *International Conference On Machine Learning*. pp. 2206-2216 (2020)

[14] Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X. & Gu, Q. Improving adversarial robustness requires revisiting misclassified examples. *International Conference On Learning Representations*. (2020)

[15] Zhang, J., Zhu, J., Niu, G., Han, B., Sugiyama, M. & Kankanhalli, M. Geometry-aware instance-reweighted adversarial training. *ArXiv Preprint ArXiv:2010.01736*. (2020)

[16] Yu, C., Han, B., Shen, L., Yu, J., Gong, C., Gong, M. & Liu, T. Understanding robust overfitting of adversarial training and beyond. *International Conference On Machine Learning*. pp. 25595-25610 (2022)

[17] Costa, J., Roxo, T., Proença, H. Inácio, P. How deep learning sees the world: A survey on adversarial attacks & defenses. *IEEE Access*. (2024)

[18] Rade, R. & Moosavi-Dezfooli, S. Helper-based adversarial training: Reducing excessive margin to achieve a better accuracy vs. robustness trade-off. *ICML 2021 Workshop On Adversarial Machine Learning*. (2021)

[19] Shafahi, A., Najibi, M., Ghiasi, M., Xu, Z., Dickerson, J., Studer, C., Davis, L., Taylor, G. & Goldstein, T. Adversarial training for free!. *Advances In Neural Information Processing Systems*. **32** (2019)

[20] Wong, E., Rice, L. & Kolter, J. Fast is better than free: Revisiting adversarial training. *ArXiv Preprint ArXiv:2001.03994*. (2020)

[21] Goldblum, M., Fowl, L., Feizi, S. & Goldstein, T. Adversarially robust distillation. *Proceedings Of The AAAI Conference On Artificial Intelligence*. **34**, 3996-4003 (2020)

[22] Zhu, J., Yao, J., Han, B., Zhang, J., Liu, T., Niu, G., Zhou, J., Xu, J. & Yang, H. Reliable adversarial distillation with unreliable teachers. *ArXiv Preprint ArXiv:2106.04928*. (2021)

[23] Zoph, B., Vasudevan, V., Shlens, J. & Le, Q. Learning transferable architectures for scalable image recognition. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*. pp. 8697-8710 (2018)

[24] Zoph, B. Neural architecture search with reinforcement learning. *ArXiv Preprint ArXiv:1611.01578*. (2016)

[25] Liu, H., Simonyan, K. & Yang, Y. Darts: Differentiable architecture search. *ArXiv Preprint ArXiv:1806.09055*. (2018)

[26] Chitty-Venkata, K. & Somani, A. Neural architecture search survey: A hardware perspective. *ACM Computing Surveys*. **55**, 1-36 (2022)

[27] Cai, H., Zhu, L. & Han, S. Proxylessnas: Direct neural architecture search on target task and hardware. *ArXiv Preprint ArXiv:1812.00332*. (2018)

[28] Chitty-Venkata, K., Emani, M., Vishwanath, V. & Somani, A. Neural architecture search benchmarks: Insights and survey. *IEEE Access*. **11** pp. 25217-25236 (2023)

[29] Mao, Y., Zhong, G., Wang, Y. & Deng, Z. Differentiable light-weight architecture search. *2021 IEEE International Conference On Multimedia And Expo (ICME)*. pp. 1-6 (2021)

[30] Cai, H., Gan, C., Wang, T., Zhang, Z. & Han, S. Once-for-all: Train one network and specialize it for efficient deployment. *ArXiv Preprint ArXiv:1908.09791*. (2019)

[31] Sahni, M., Varshini, S., Khare, A. & Tumanov, A. CompOFA: Compound once-for-all networks for faster multi-platform deployment. *ArXiv Preprint ArXiv:2104.12642*. (2021)

[32] Girard, M., Quétu, V., Tardieu, S., Nguyen, V. & Tartaglione, E. Memory-Optimized Once-For-All Network. *ArXiv Preprint ArXiv:2409.05900*. (2024)

[33] Sakuma, Y., Ishii, M. & Narihira, T. DetOFA: Efficient Training of Once-for-All Networks for Object Detection using Path Filter. *Proceedings Of The IEEE/CVF International Conference On Computer Vision*. pp. 1333-1342 (2023)

[34] Wang, H., Wu, Z., Liu, Z., Cai, H., Zhu, L., Gan, C. & Han, S. Hat: Hardware-aware transformers for efficient natural language processing. *ArXiv Preprint ArXiv:2005.14187*. (2020)

[35] Tang, H., Liu, Z., Zhao, S., Lin, Y., Lin, J., Wang, H. & Han, S. Searching efficient 3d architectures with sparse point-voxel convolution. *European Conference On Computer Vision*. pp. 685-702 (2020)

[36] Lin, J., Zhang, R., Ganz, F., Han, S. & Zhu, J. Anycost gans for interactive image synthesis and editing. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 14986-14996 (2021)

[37] Wang, Y., Li, M., Cai, H., Chen, W. & Han, S. Lite pose: Efficient architecture design for 2d human pose estimation. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 13126-13136 (2022)

[38] Krizhevsky, A., Sutskever, I. & Hinton, G. Imagenet classification with deep convolutional neural networks. *Advances In Neural Information Processing Systems*. **25** (2012)

[39] Zou, Z., Chen, K., Shi, Z., Guo, Y. & Ye, J. Object detection in 20 years: A survey. *Proceedings Of The IEEE*. **111**, 257-276 (2023)

[40] Zou, X., Yang, J., Zhang, H., Li, F., Li, L., Wang, J., Wang, L., Gao, J. & Lee, Y. Segment everything everywhere all at once. *Advances In Neural Information Processing Systems*. **36** (2024)

[41] Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T. & Song, D. Robust physical-world attacks on deep learning visual classification. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*. pp. 1625-1634 (2018)

[42] Sharif, M., Bhagavatula, S., Bauer, L. & Reiter, M. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. *Proceedings Of The 2016 Acm Sigsac Conference On Computer And Communications Security*. pp. 1528-1540 (2016)

[43] Szegedy, C. Intriguing properties of neural networks. *ArXiv Preprint ArXiv:1312.6199*. (2013)

[44] Ou, Y., Feng, Y. & Sun, Y. Towards Accurate and Robust Architectures via Neural Architecture Search. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 5967-5976 (2024)

[45] Mok, J., Na, B., Choe, H. & Yoon, S. AdvRush: Searching for adversarially robust neural architectures. *Proceedings Of The IEEE/CVF International Conference On Computer Vision*. pp. 12322-12332 (2021)

[46] Hosseini, R., Yang, X. & Xie, P. Dsrna: Differentiable search of robust neural architectures. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 6196-6205 (2021)

[47] Yue, Z., Lin, B., Zhang, Y. & Liang, C. Effective, efficient and robust neural architecture search. *2022 International Joint Conference On Neural Networks (IJCNN)*. pp. 1-8 (2022)

[48] Cheng, Z., Li, Y., Dong, M., Su, X., You, S. & Xu, C. Neural architecture search for wide spectrum adversarial robustness. *Proceedings Of The AAAI Conference On Artificial Intelligence*. **37**, 442-451 (2023)

[49] Ou, Y., Xie, X., Gao, S., Sun, Y., Tan, K. & Lv, J. Differentiable search of accurate and robust architectures. *ArXiv Preprint ArXiv:2212.14049*. (2022)

[50] Li, Y., Yang, Z., Wang, Y. & Xu, C. Neural architecture dilation for adversarial robustness. *Advances In Neural Information Processing Systems*. **34** pp. 29578-29589 (2021)

[51] Guo, M., Yang, Y., Xu, R., Liu, Z. & Lin, D. When nas meets robustness: In search of robust architectures against adversarial attacks. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 631-640 (2020)

[52] Xie, G., Wang, J., Yu, G., Lyu, J., Zheng, F. & Jin, Y. Tiny adversarial multi-objective one-shot neural architecture search. *Complex & Intelligent Systems*. **9**, 6117-6138 (2023)

[53] Feng, Y., Lv, Z., Chen, H., Gao, S., An, F. & Sun, Y. LRNAS: Differentiable Searching for Adversarially Robust Lightweight Neural Architecture. *IEEE Transactions On Neural Networks And Learning Systems*. (2024)

[54] Wu, D., Wang, Y., Xia, S., Bailey, J. & Ma, X. Skip connections matter: On the transferability of adversarial examples generated with resnets. *ArXiv Preprint ArXiv:2002.05990*. (2020)

[55] Shao, R., Shi, Z., Yi, J., Chen, P. & Hsieh, C. On the adversarial robustness of vision transformers. *ArXiv Preprint ArXiv:2103.15670*. (2021)

[56] Zhu, X., Li, J., Liu, Y. & Wang, W. Robust Neural Architecture Search. *ArXiv Preprint ArXiv:2304.02845*. (2023)

[57] Dong, M., Li, Y., Wang, Y. & Xu, C. Adversarially robust neural architectures. *ArXiv Preprint ArXiv:2009.00902*. (2020)

[58] Chen, H., Zhang, B., Xue, S., Gong, X., Liu, H., Ji, R. & Doermann, D. Anti-bandit neural architecture search for model defense. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*. pp. 70-85 (2020)

[59] Du, X., Zhang, J., Han, B., Liu, T., Rong, Y., Niu, G., Huang, J. & Sugiyama, M. Learning diverse-structured networks for adversarial robustness. *International Conference On Machine Learning*. pp. 2880-2891 (2021)

[60] Ha, H., Kim, M. & Hwang, S. Generalizable lightweight proxy for robust NAS against diverse perturbations. *Advances In Neural Information Processing Systems*. **36** (2024)

[61] Wu, Y., Liu, F., Simon-Gabriel, C., Chrysos, G. & Cevher, V. Robust NAS under adversarial training: benchmark, theory, and beyond. *ArXiv Preprint ArXiv:2403.13134*. (2024)

[62] Jung, S., Lukasik, J. & Keuper, M. Neural architecture design and robustness: A dataset. *ArXiv Preprint ArXiv:2306.06712*. (2023)

[63] Peng, S., Xu, W., Cornelius, C., Li, K., Duggal, R., Chau, D. & Martin, J. Robarch: Designing robust architectures against adversarial attacks. *ArXiv Preprint ArXiv:2301.03110*. (2023)

[64] Huang, S., Lu, Z., Deb, K. & Boddeti, V. Revisiting residual networks for adversarial robustness. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 8202-8211 (2023)

[65] Yang, S., Sun, X., Xu, K., Liu, Y., Tian, Y. & Zhang, X. Hybrid Architecture-Based Evolutionary Robust Neural Architecture Search. *IEEE Transactions On Emerging Topics In Computational Intelligence*. (2024)

[66] Feng, Y., Lv, Z., Chen, H., Gao, S., An, F. & Sun, Y. LRNAS: Differentiable Searching for Adversarially Robust Lightweight Neural Architecture. *IEEE Transactions On Neural Networks And Learning Systems*. (2024)

[67] Zhao, S., Yu, J., Sun, Z., Zhang, B. & Wei, X. Enhanced accuracy and robustness via multi-teacher adversarial distillation. *European Conference On Computer Vision*. pp. 585-602 (2022)

[68] Huang, B., Chen, M., Wang, Y., Lu, J., Cheng, M. & Wang, W. Boosting accuracy and robustness of student models via adaptive adversarial distillation. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 24668-24677 (2023)

[69] Vaswani, A. Attention is all you need. *Advances In Neural Information Processing Systems*. (2017)

[70] Ma, X., Niu, Y., Gu, L., Wang, Y., Zhao, Y., Bailey, J. & Lu, F. Understanding adversarial attacks on deep learning based medical image analysis systems. *Pattern Recognition*. **110** pp. 107332 (2021)

[71] Jung, J., Jang, H., Song, J. & Lee, J. PeerAiD: Improving Adversarial Distillation from a Specialized Peer Tutor. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 24482-24491 (2024)

[72] Takahashi, T., Yamada, M., Yamanaka, Y. & Yamashita, T. ARDIR: Improving Robustness using Knowledge Distillation of Internal Representation. *ArXiv Preprint ArXiv:2211.00239*. (2022)

[73] Zhou, Y., Zhang, Y., Zhang, L. & Hua, Z. DERD: data-free adversarial robustness distillation through self-adversarial teacher group. *Proceedings Of The 32nd ACM International Conference On Multimedia*. pp. 10055-10064 (2024)

[74] Yin, S., Xiao, Z., Song, M. & Long, J. Adversarial Distillation Based on Slack Matching and Attribution Region Alignment. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 24605-24614 (2024)

[75] Deng, J., Palmer, A., Mahmood, R., Rathbun, E., Bi, J., Mahmood, K. & Aguiar, D. Distilling Adversarial Robustness Using Heterogeneous Teachers. *ArXiv Preprint ArXiv:2402.15586*. (2024)

[76] Shao, R., Yi, J., Chen, P. & Hsieh, C. How and when adversarial robustness transfers in knowledge distillation?. *ArXiv Preprint ArXiv:2110.12072*. (2021)

[77] Kuang, H., Liu, H., Wu, Y., Satoh, S. & Ji, R. Improving adversarial robustness via information bottleneck distillation. *Advances In Neural Information Processing Systems*. **36** pp. 10796-10813 (2023)

[78] Ham, S., Park, J., Han, D. & Moon, J. NEO-KD: knowledge-distillation-based adversarial training for robust multi-exit neural networks. *Advances In Neural Information Processing Systems*. **36** (2024)

[79] Dong, J., Koniusz, P., Chen, J., Wang, Z. & Ong, Y. Robust Distillation via Untargeted and Targeted Intermediate Adversarial Samples. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 28432-28442 (2024)

[80] Dong, J., Koniusz, P., Chen, J. & Ong, Y. Adversarially Robust Distillation by Reducing the Student-Teacher Variance Gap. *European Conference On Computer Vision*. pp. 92-111 (2025)

[81] Zhou, Y., Zhang, Y., Zhang, L. & Hua, Z. DERD: data-free adversarial robustness distillation through self-adversarial teacher group. *Proceedings Of The 32nd ACM International Conference On Multimedia*. pp. 10055-10064 (2024)

[82] Bubeck, S. & Sellke, M. A universal law of robustness via isoperimetry. *Advances In Neural Information Processing Systems*. **34** pp. 28811-28822 (2021)