ROBUST NEURAL ARCHITECTURE SEARCH BY CROSS-LAYER KNOWLEDGE DISTILLATION

Utkarsh Nath, Yancheng Wang & Yingzhen Yang
School of Computing and Augmented Intelligence
Arizona State University
Tempe, AZ 85281, USA
{unath, ywan1053, yyang409}@asu.edu

ABSTRACT

Deep Neural Networks are vulnerable to adversarial attacks. Neural Architecture Search (NAS), one of the driving tools of deep neural networks, demonstrates superior performance in prediction accuracy in various machine learning applications. However, it is unclear how it performs against adversarial attacks. Given the presence of a robust teacher, it would be interesting to investigate if NAS would produce robust neural architecture by inheriting robustness from the teacher. In this paper, we propose Robust Neural Architecture Search by Cross-Layer Knowledge Distillation (RNAS-CL), a novel NAS algorithm that improves the robustness of NAS by learning from a robust teacher through cross-layer knowledge distillation. Unlike previous knowledge distillation methods that encourage close student/teacher output only in the last layer, RNAS-CL automatically searches for the best teacher layer to supervise each student layer. Experimental result evidences the effectiveness of RNAS-CL and shows that RNAS-CL produces small and robust neural architecture.

1 INTRODUCTION

Neural Architecture Search (NAS), one of the most promising driving tools with state-of-the-art performance of deep neural networks in various tasks such as computer vision and natural language processing, has been attracting a lot of attention in recent years. NAS automatically searches for neural architecture according to user-specified criteria without human intervention, thus avoiding the time-consuming and burdensome manual design of neural architecture. Most recent studies (Liu et al., 2019; Cai et al., 2019; Wu et al., 2019; Wan et al., 2020) encode architectures as a weight-sharing super-net and optimize the weights using gradient descent. Architectures found by NAS exhibit two significant advantages. First, they achieve SOTA performance for various computer vision tasks. Second, the architectures found by NAS are efficient in terms of speed and size. Both advantages make NAS incredibly useful for real-world applications. However, most NAS methods are designed to optimize accuracy, parameters, or FLOPs. It is not clear how these architectures perform against adversarial attacks. In this paper, we propose RNAS-CL, a NAS method that jointly optimizes accuracy, latency, and robustness against adversarial attacks without robust training.

Adversarial attacks are performed by adding adversarial samples, for example, adding small sophisticated perturbations to the clean image, such that the model misclassifies the image. It is widely accepted that deep learning models are susceptible to adversarial attacks (Szegedy et al., 2014). Therefore, it is critical to analyze the robustness of models against adversarial attacks. Adversarial training (Goodfellow et al., 2015) is the most standard defense mechanism against adversarial attacks. Other types of defense mechanisms include models trained by losses or regularizations (Cissé et al., 2017; Pang et al., 2020), transforming inputs before feeding to model (Guo et al., 2018; Xie et al., 2019), and using model ensemble (Kurakin et al., 2018; Liu et al., 2018).

Orthogonal to these methods, recent research (Madry et al., 2018; Guo et al., 2020; Su et al., 2018; Xie & Yuille, 2020; Huang et al., 2021) found an intrinsic influence of network architecture on adversarial robustness. Inspired by this idea, we propose Robust Knowledge Distillation for Neural Architecture Search (RNAS-CL), to the best of our knowledge, the first NAS method that uses

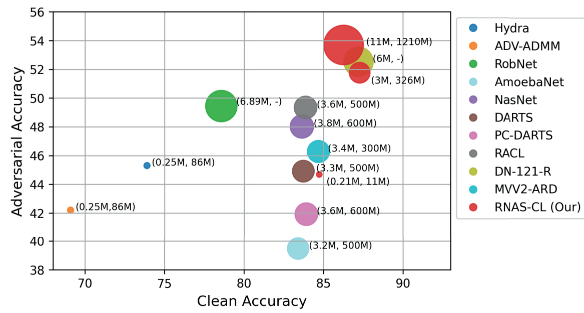


Figure 1: The figure compares various SOTA efficient and robust methods on CIFAR-10. Clean Accuracy represents top-1 accuracy on clean images. Adversarial Accuracy represents top-1 accuracy on images perturbed by PGD attack. A larger marker size indicates larger architecture. The numbers in brackets represent the number of parameters and MACs, respectively.

knowledge distilled from a robust teacher model to find a robust architecture. Knowledge distillation transfers knowledge from a complex teacher model to a small student model. In standard knowledge distillation (Hinton et al., 2015), outputs from the teacher model are used as ”soft labels” to train the student model. However, apart from the final teacher outputs, intermediate layers contain rich attention information. Different intermediate layers attend to different parts of the input object (Zagoruyko & Komodakis, 2017).

Hence, we ask the question: *can a robust teacher improve the robustness of the student model by providing information about where to look, i.e., where to pay attention?* The proposed RNAS-CL gives affirmative answers to the above question. In RNAS-CL, apart from learning from the output of the robust teacher model, each layer in the student learns ”where to look” from the layers in the teacher model. We hypothesize that learning where to pay attention from a robust teacher will inherently make the student model more robust to adversarial attacks. However, the teacher and student might have a different number of layers. This leads us to another question regarding how to map a student layer to its corresponding teacher layer that it should learn from. In RNAS-CL, apart from searching the architecture of the student model, we search for the perfect tutor (teacher) layer for each student layer.

Let us consider a teacher (T) and student (S) model with n_t and n_s layers, respectively. T_i, S_i are the i-th teacher and student layer, respectively. In RNAS-CL, each student layer S_i is associated with n_t gumbel weights, and each gumbel weight corresponds to each teacher layer. Intuitively, each gumbel weight indicates the weight of the connection between the student layer and each teacher layer. In the search phase, besides optimizing the architectural weights, we optimize these gumbel weights to find the perfect teacher layer. We hope the teacher to teach ”where to pay attention.” Therefore, by virtue of our RNAS-CL loss function for each student-teacher layer pair, each student layers learns robustness from a properly and automatically chosen teacher layer by maximizing the similarity of its attention map to that of its teacher layer.

1.1 Contributions

Below are the main contributions of this work.

1. **Adversarial robust NAS.** RNAS-CL optimizes neural architecture to achieve a good tradeoff between robustness and prediction accuracy in a differentiable manner. To the best of our knowledge, RNAS-CL is the first NAS method that optimizes robustness and prediction accuracy without robust training. Leveraging the penalty on model size/inference cost, the neural architecture found by RNAS-CL is compact compared to competing NAS methods. We compare RNAS-CL with other computationally efficient and robust models (Sehwag et al., 2020; Ye et al., 2019; Gui et al., 2019; Goldblum et al., 2020; Dong et al., 2020; Huang et al., 2021). Compared to these models, similar sized RNAS-CL models achieve up to $\sim 10\%$ higher clean accuracy and up to $\sim 5\%$ higher PGD accuracy on CIFAR-10 dataset.
2. **Cross-Layer Knowledge Distillation.** Our work advances the research of Knowledge Distillation (KD) using NAS. In particular, while conventional KD only uses fixed connections between

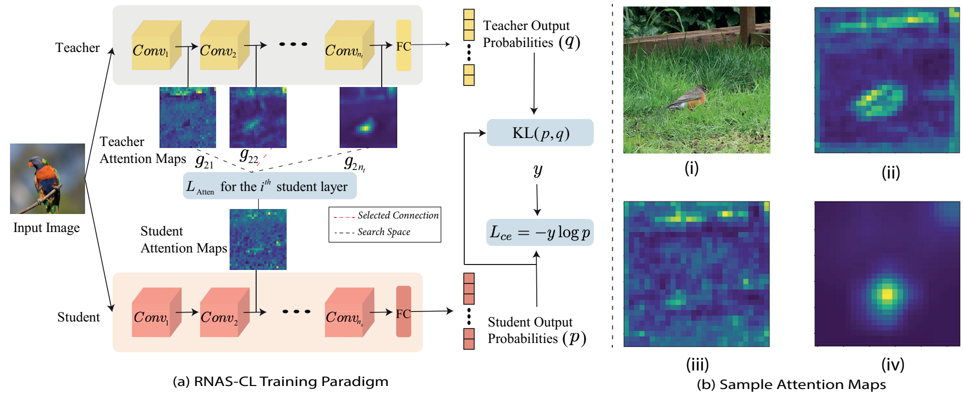


Figure 2: (a) Training paradigm based on RNAS-CL. We connect attention maps from each student layer to each robust teacher layer. For each student layer, we search for the optimum teacher layer. g_{ij} represents gumbel weights associated between i^{th} student layer and j^{th} teacher layer. RNAS-CL induces robustness to the student model by searching for the optimum teacher layer. We also search for the number of filters in each layer to build an efficient model inspired by FBNetV2 (Wan et al., 2020). (b) Sample attention maps corresponding to input Image (i) from low-level (ii), mid-level (iii), and high-level (iv) convolution layers.

teacher and student models to guide the student model, RNAS-CL extends the teaching scheme to learnable connections between layers of the teacher and the student models.

2 Related Work

2.1 Knowledge Distillation

Since (Hinton et al., 2015), numerous KD variants (Romero et al., 2015; Yim et al., 2017; Zagoruyko & Komodakis, 2017; Li et al., 2019; Tian et al., 2020a; Sun et al., 2019) based on feature map, attention map, or contrastive learning have been proposed. (Romero et al., 2015) introduced intermediate-level hints from a single teacher layer to guide the student model training. Moving a step further, (Yim et al., 2017), (Zagoruyko & Komodakis, 2017) and (Li et al., 2019) used information from multiple teacher layers to guide students' training. Contrary to the above methods, which map few teacher-student layers or blocks. We map all student layers to a teacher layer. We propose RNAS-CL to search for the perfect tutor layer for each student layer. Similar to (Zagoruyko & Komodakis, 2017), we minimize the distance between mapped student-teacher attention maps.

2.2 Efficient and Robust Models

Research community has extensively researched building efficient and adversarially robust models individually. However, few works combine both domains, building an efficient and adversarially robust model. (Sehwag et al., 2020) propose to make the pruning technique aware of the robust training objective. They formulate pruning as an empirical risk minimization (ERM) problem and integrate it with a robust training objective. (Huang et al., 2021) investigated the impact of network width and depth configurations on the robustness of adversarial trained DNNs. They observed that reducing capacity at last blocks improves adversarial robustness. (Goldblum et al., 2020), propose Adversarially Robust Distillation (ARD), where they encourage student networks to mimic their teacher's output within an ϵ-ball of training samples. Furthermore, there are few NAS methods (Yue et al., 2022; Ning et al., 2020; Xie et al., 2021) that jointly optimises accuracy, latency and robustness. Compared to these methods, similar-sized RNAS-CL models achieve both higher clean and robust accuracy.

3 Robust Knowledge Distillation for Neural Architecture Search

We use knowledge distilled from a robust teacher model to search for a robust and efficient architecture. In standard knowledge distillation, outputs from the teacher model are used as ”soft labels” to train the student model. However, apart from the final teacher outputs, intermediate features constitute important attention information. Different intermediate layers ”attend” to different parts of the input object. In RNAS-CL, apart from learning from the teacher's soft labels, the method learns from intermediate teacher layers where to pay attention, i.e., each student layer is mapped to a robust teacher layer to learn where to look. In Section 3.1, we discuss how we define attention maps. In our method, we search for the perfect tutor for each layer. Furthermore, along with increasing the robustness, we are also interested in searching for an efficient architecture. In Section 3.2 and 3.3, we discuss our tutor and architecture search algorithm. In Section 3.4, we discuss our searching and training optimization objectives.

3.1 Attention Map

We are interested in learning where to pay attention from a robust teacher model. Let us consider a convolution layer with activation tensor $A \in R^{C\times H\times W}$ where C is the number of channels, and H and W are spatial dimensions. We define a mapping function $\mathcal{F} : R^{C\times H\times W} \longrightarrow R^{H\times W}$ that takes A as input and outputs an attention map $\mathcal{F}(A) \in R^{H\times W}$ by $[\mathcal{F}(A)]_{hw} = \sum_{c=1}^{C} A^2_{c,h,w}$, where $A_{c,h,w}$ represents the element of A with channel coordinate c and spatial coordinates h and w.

We use activation-based mapping function $\mathcal{F}$ as proposed in (Zagoruyko & Komodakis, 2017). The mapping function $\mathcal{F}$ is applied to activation tensors after each convolution layer to generate an attention map. We visualized few attention maps in Figure 2(b). RNAS-CL intends to find a teacher layer, referred to as a tutor, for each student layer such that the student layer's attention map is similar to that of its tutor in the teacher model. The student attention map may differ in dimension compared to that of its tutor. To address this issue, we interpolate all attention maps to a common dimension.

3.2 Tutor Search

As described above, we aim to find a tutor (teacher layer) for each student layer, which teaches where to pay attention. However, each student layer can choose any tutor, resulting in an exponentially large search space. For example, the search space for a student model with 20 layers and a teacher model with 50 layers is of size 50^{20}. In order to address the computational issue, we employ Gumbel-Softmax (Jang et al., 2017) to search for the tutor for each student layer in a differentiable manner. Given network parameter $v = [v_1, \dots, v_n]$ and a constant τ. The Gumbel-Softmax function is defined as $g(v) = [g_1, \dots, g_n]$ where $g_i = \frac{\exp[(v_i+\epsilon_i)/\tau]}{\sum_i \exp[(v_i+\epsilon_i)/\tau]}$ and $\epsilon_i \sim N(0,1)$ is the uniform random noise, which is also referred to as Gumbel noise. When $\tau \to 0$, Gumbel-Softmax tends to the $\arg\max$ function. Gumbel-Softmax is a “re-parametrization trick”, that can be regarded as a differentiable approximation to the argmax function.

Now consider a teacher T and student S model with n_t and n_s number of layers, respectively. A^i_t and A^i_s are the i^{th} activation tensors of teacher and student layers. In RNAS-CL, each student layer (i) is associated with n_t Gumbel weights (g_i) such that $g_i \in R^{1\times n_t}$. Let g_{ij} be the Gumbel weight associated with i^{th} student and j^{th} teacher layer. Then the attention loss is defined as

$$L_{\text{Attn}}(A_t, A_s) = \frac{1}{n_s \times n_t} \sum_{i=0}^{n_s} \sum_{j=0}^{n_t} g_{ij} \left|\left| \frac{\mathcal{F}(A^i_s)}{||\mathcal{F}(A^i_s)||_2} - \frac{\mathcal{F}(A^j_t)}{||\mathcal{F}(A^j_t)||_2} \right|\right|_2, \quad (1)$$

where A_s and A_t are activation tensors for all student and teacher convolution layers. $\mathcal{F}$ is the mapping function as defined in Section 3.1. $\|\cdot\|_2$ is the ℓ^2-norm. We exponentially decay the temperature τ of Gumbel-Softmax while searching, leading to an encoding close to a one-hot vector.

3.3 Architecture Search

Apart from searching the tutor for each layer, we are interested in building efficient architecture with low latency. Inspired by FBNetV2 (Wan et al., 2020), we search for the optimal number of filters, or the number of output channels, for each convolution block. Let $A = \{f_1, f_2, ..., f_n\}$ be the choices of filters and $\{z_1, z_2, ..., z_n\}$ be their corresponding outputs for a convolution block. Then the cumulative output is defined as $Z = \sum_{i=1}^{n} g_w^{(i)} z_i$, where $g_w^{(i)}$ is the Gumbel weight corresponding to i^{th} filter choice. The number of FLOPs is optimized so as to ensure low latency. The FLOPs are proportional to the number of filters, and the cumulative number of filters is a function of Gumbel weights. As a result, the FLOPs can be optimized in a differential manner using SGD.

3.4 RNAS-CL Loss

Following the convention of state-of-the-art NAS methods (Liu et al., 2019; Wu et al., 2019; Wan et al., 2020), RNAS-CL has searching and training phases. In the search phase, Gumbel weights and other model parameters are updated at each epoch of SGD, where the Gumbel weights correspond to the intermediate student-teacher connection (3.1) and the filter choices (3.3). The weights are optimised using our RNAS-CL search loss defined by (2).

Let y be the ground-truth one-hot encoded vector, p and q be output probabilities of the student and teacher network and A_s, A_t be activation tensors for all student and teacher convolution layers. Then the RNAS-CL search loss is defined as

$$L(y, p, q, A_t, A_s) = (-y \log p + KL(p, q) + \gamma_s L_{\text{Attn}}(A_t, A_s))n_f, \tag{2}$$

where $KL(p, q) = \sum_i p_i \log \frac{p_i}{q_i}$ is the Kullback–Leibler(KL) divergence between two probability measures. L_{Attn} is the attention loss as defined in (1) and γ_s is a normalization constant. n_f represents latency, which is optimized in a differential manner following (Wan et al., 2020).

After the search phase, a tutor is selected as the j^* teacher layer with $j^* = \arg\max_j g_{ij}$ for each student layer i. In addition, the filter choices described in Section 3.3 for neural architecture are decided as the one corresponding to the maximum Gumbel weight for each convolution block. We then start the training phase, where the searched architecture is trained using the RNAS-CL train loss defined below.

$$L(y, p, q, A_t, A_s) = -y \log p + KL(p, q) + \gamma_t L_{Attn}(A_t, A_s), \tag{3}$$

where y, p, q, A_t, A_s are same as in (2) and γ_t is a normalization constant. Note that, g_i in L_{Attn} is a one-hot vector. Thus, each student attention map is optimized w.r.t. to a single tutor layer.

4 Experiments

In this paper, we evaluate RNAS-CL on two public benchmarks for image classification. (1) *CIFAR-10* - a collection of $60k$ images in 10 classes (Krizhevsky, 2009). (2) *ImageNet-100* - a subset of ImageNet-1k dataset (Russakovsky et al., 2015) with 100 classes and about $130k$ images (Tian et al., 2020b). We use standard data augmentation techniques for each dataset, such as random-resize cropping and random flipping. We train different architectures found by RNAS-CL on both CIFAR-10 and ImageNet-100. On each dataset, we first perform the searching step. We train our model using RNAS-CL search loss (2). We search for the channel number and the connected teacher layer at each student layer. We conduct experiments with different search spaces and various robust teacher models. In this section, we refer to our model by *RNAS-CL-X-T* where X represents our search space, and T represents the robust teacher model. Detailed search space is the appendix (Section A.4). We use 4 robust teacher model, ResNet-50, ResNet-18, WideResNet-50, and WideResNet-34, which are referred to as R-50, R-18, WRT-50, and WRT-34. For example, RNAS-CL-S3-R-18 represents a model trained in the S3 search space using an adversarially robust ResNet-18 model. The hyper-parameters for our training have been discussed in the appendix (Section A.1). For robustness evaluation, we choose three powerful attacks including FGSM (Goodfellow et al., 2015), MI-FGSM (Dong et al., 2018) and PGD (Madry et al., 2018), CW (Carlini & Wagner, 2017). Consistent with

Method	Clean Acc	FSGM	PGD^{20}	MI-FGSM	# Params (M)	MACs (M)
Without Adversarial Training						
DARTS (Liu et al., 2019)	97.03	42.48	7.09	0.28	3.3	500^*
PC-DARTS (Xu et al., 2020)	97.05	49.18	9.84	1.21	3.6	600^*
RACL (Dong et al., 2020)	97.44	50.53	1.93	4.68	3.6	500^*
AmoebaNet (Real et al., 2019)	97.39	44.79	0.25	0.80	3.2	500^*
NasNet (Zoph et al., 2018)	97.37	47.53	0.42	1.01	3.8	600^*
MVV2-ARD (Goldblum et al., 2020)	76.13	-	38.21	-	3.4	300
E2RNAS-C16 (Yue et al., 2022)	93.97	-	6.76	-	0.44	-
RNAS-CL-S3-WRT-34 (Our)	**89.4**	44.95	**34.3**	38.92	**0.11**	**6.64**
RNAS-CL-S5-WRT-34 (Our)	90.4	46.72	35.59	40.57	0.21	11.02
RNAS-CL-S7-WRT-34 (Our)	90.62	48.93	37.24	42.27	0.32	15.58
RNAS-CL-M-WRT-34 (Our)	92.46	50.51	39.84	44.54	3	326
RNAS-CL-L–WRT-34 (Our)	**92.6**	**52.37**	**41.9**	**46.66**	11	1210
With Adversarial Training						
Hydra ResNet 18 (Sehwag et al., 2020)	69	-	41.6	-	0.11	37.63
Hydra ResNet 34 (Sehwag et al., 2020)	71.8	-	44.4	-	0.21	75.43
Hydra ResNet 50 (Sehwag et al., 2020)	73.9	-	45.3	-	0.25	85.92
ADV-ADMM ResNet 18 (Ye et al., 2019)	58.7	-	36.1	-	0.11	37.63
ADV-ADMM ResNet 34 (Ye et al., 2019)	68.8	-	41.5	-	0.21	75.43
ADV-ADMM ResNet 50 (Ye et al., 2019)	69.1	-	42.2	-	0.25	85.92
RobNet-Small (Guo et al., 2020)	78.05	53.93	48.32	48.98	4.41	-
RobNet-Medium (Guo et al., 2020)	78.33	54.55	49.13	49.34	5.66	-
RobNet-Large (Guo et al., 2020)	78.57	54.98	49.44	49.92	6.89	-
NasNet (Zoph et al., 2018)	83.66	55.67	48.02	53.05	3.8	600^*
DARTS (Liu et al., 2019)	83.75	55.75	44.91	51.63	3.3	500^*
PC-DARTS (Xu et al., 2020)	83.94	52.67	41.92	49.09	3.6	600^*
RACL (Dong et al., 2020)	83.89	57.44	49.34	54.73	3.6	500^*
VGG-11-R (Huang et al., 2021)	79.63	57.35	43.93	-	5.83	-
DN-121-R (Huang et al., 2021)	87.22	**67.12**	52.52	-	6	-
DARTS-R (Huang et al., 2021)	87.2	66.74	52.36	-	2.53	-
MVV2-ARD (Goldblum et al., 2020)	84.70	-	46.28	-	3.4	300
MSRobNet-1000 (Ning et al., 2020)	84.5	59.6	52.7	-	3.16	-
MSRobNet-2000 (Ning et al., 2020)	85.7	60.6	53.6	-	6.46	-
$S^2_{8/255}$ (Xie et al., 2021)	76.54	-	31.83	-	1.68	-
RNAS-CL-S3-WRT-34 (Our)	83.45	50.67	43.07	43.98	**0.11**	**6.64**
RNAS-CL-S5-WRT-34 (Our)	84.75	51.99	44.68	46.3	0.21	11.02
RNAS-CL-S7-WRT-34 (Our)	85.81	49.11	43.24	45.53	0.32	15.58
RNAS-CL-M-WRT-34 (Our)	**87.29**	59.71	**51.76**	53.43	3	326
RNAS-CL-L-WRT-34 (Our)	86.28	61.12	**53.69**	**55.07**	11	1210

Table 1: The table shows performance of various efficient and robust methods on CIFAR-10 dataset. Clean Acc represents top-1 accuracy on clean images. FSGM, PGD^{20}, MI-FGSM represents top-1 accuracy on images perturbed by the corresponding attacks. PGD^{20} represents 20 step PGD attack. $*$ represents approximate values. Columns with unreported values are represented by -.

adversarial literature (Madry et al., 2018; Zhang et al., 2019b), the adversarial perturbation is considered under l_∞ norm with a total perturbation scale of 8/255 (0.031). We discuss our results on ImageNet-100 dataset in the appendix (Section A.3).

4.1 Compare Efficient and Robust CIFAR-10 models

In this section, we compare the robustness of our method against other SOTA efficient and robust models. In Table 1, we compare RNAS-CL to both efficient models trained with and without adversarial training. All RNAS-CL models are trained with robust WideResNet-34 (Rice et al., 2020) as the teacher model. RNAS-CL significantly outperforms all models trained without adversarial training in terms of adversarial accuracy. While being significantly smaller, our models achieve significantly higher adversarial accuracy when compared to models trained without adversarial training. For example, *RNAS-CL-S7-WRT-34* achieves more than 28% higher PGD accuracy compared to most of the other methods. Compared to MVVV2-ARD, *RNAS-CL-S7-WRT-34* achieves $\sim 1\%$ lower PGD accuracy; however, it exceeds MVVV2-ARD by 14.5% in clean accuracy while being $10\times$ smaller. A similar-sized model, for example, *RNAS-CL-M-WRT-34*, exceeds both clean and PGD accuracy by 16.5% and 1.43%.

Next, we compare RNAS-CL against adversarially trained robust models. For a fair comparison, after the training stage, we train our RNAS-CL models with the TRADES optimization objective for 20 epochs. For retraining, the cross-entropy term in (3) is replaced by TRADES optimization objective. Adversarially training RNAS-CL models improve its adversarial accuracy. RNAS-CL models achieve similar or higher adversarial accuracy compared to other adversarially trained models. However, RNAS-CL models are much smaller and achieve significantly higher clean accuracy. For example, in Table 1, RNAS-CL-M-WRT-34 achieves similar or higher adversarial accuracy than most other methods while being smaller and significantly exceeding in terms of clean accuracy. Similar results can also be visualized in Figure 1. In Figure 1, RNAS-CL models are on the top right corner of the plot, representing the models with the highest clean and adversarial accuracy.

5 Conclusion

In this paper, we propose Robust Neural Architecture Search by Cross-Layer Knowledge Distillation (RNAS-CL), a novel NAS algorithm that improves the robustness of the student model by learning from a robust teacher through cross-layer knowledge distillation. RNAS-CL optimizes neural architecture to achieve a good tradeoff between robustness and clean accuracy in a differentiable manner without robust training. RNAS-CL extends conventional knowledge distillation by learning student-teacher cross-connections. We show that models obtained by RNAS-CL outperform all models obtained without robust training in terms of adversarial robustness. We show that adding adversarial training can further increase the adversarial robustness of RNAS-CL models. After robust training, RNAS-CL achieves similar adversarial robustness compared to models obtained via robust training while outperforming them in terms of clean accuracy.

References

Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. In *7th International Conference on Learning Representations, ICLR*, 2019.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy*, pp. 39–57. IEEE, 2017.

Moustapha Cissé, Piotr Bojanowski, Edouard Grave, Yann N. Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning, ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 854–863. PMLR, 2017.

Minjing Dong, Yanxi Li, Yunhe Wang, and Chang Xu. Adversarially robust neural architectures. *arXiv preprint arXiv:2009.00902*, 2020.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, pp. 9185–9193. Computer Vision Foundation / IEEE Computer Society, 2018.

Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. URL https://github.com/MadryLab/robustness.

Micah Goldblum, Liam Fowl, Soheil Feizi, and Tom Goldstein. Adversarially robust distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 3996–4003, 2020.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR*, 2015.

Shupeng Gui, Haotao Wang, Haichuan Yang, Chen Yu, Zhangyang Wang, and Ji Liu. Model compression with adversarial robustness: A unified optimization framework. In *Annual Conference on Neural Information Processing Systems*, pp. 1283–1294, 2019.

Chuan Guo, Mayank Rana, Moustapha Cissé, and Laurens van der Maaten. Countering adversarial images using input transformations. In *6th International Conference on Learning Representations, ICLR*, 2018.

Minghao Guo, Yuzhe Yang, Rui Xu, Ziwei Liu, and Dahua Lin. When NAS meets robustness: In search of robust architectures against adversarial attacks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 628–637, 2020.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

Song Han, Xingyu Liu, Huizi Mao, Jing Pu, Ardavan Pedram, Mark A. Horowitz, and William J. Dally. EIE: efficient inference engine on compressed deep neural network. In *ISCA*, pp. 243–254. IEEE Computer Society, 2016.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

Hanxun Huang, Yisen Wang, Sarah M. Erfani, Quanquan Gu, James Bailey, and Xingjun Ma. Exploring architectural ingredients of adversarially robust deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 5545–5559, 2021.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR*, 2017.

A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, Univ. Toronto, 2009.

Alexey Kurakin, Ian Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, et al. Adversarial attacks and defences competition. In *The NIPS'17 Competition: Building Intelligent Systems*, pp. 195–231. Springer, 2018.

Hao-Ting Li, Shih-Chieh Lin, Cheng-Yeh Chen, and Chen-Kuo Chiang. Layer-level knowledge distillation for deep neural network learning. *Applied Sciences*, 9(10), 2019.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *7th International Conference on Learning Representations, ICLR*, 2019.

Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *European Conference on Computer Vision, ECCV*, pp. 381–397, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR*, 2018.

Xuefei Ning, Junbo Zhao, Wenshuo Li, Tianchen Zhao, Yin Zheng, Huazhong Yang, and Yu Wang. Discovering robust convolutional architecture at targeted capacity: A multi-shot approach. *arXiv preprint arXiv:2012.11835*, 2020.

Tianyu Pang, Kun Xu, Yinpeng Dong, Chao Du, Ning Chen, and Jun Zhu. Rethinking softmax cross-entropy loss for adversarial robustness. In *8th International Conference on Learning Representations, ICLR*, 2020.

Jongsoo Park, Sheng R. Li, Wei Wen, Ping Tak Peter Tang, Hai Li, Yiran Chen, and Pradeep Dubey. Faster cnns with direct sparse convolutions and guided pruning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=rJPcZ3txx.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4780–4789, 2019.

Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pp. 8093–8104. PMLR, 2020.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *3rd International Conference on Learning Representations, ICLR*, 2015.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Vikash Sehwag, Shiqi Wang, Prateek Mittal, and Suman Jana. HYDRA: pruning adversarially robust neural networks. In *Annual Conference on Neural Information Processing Systems*, 2020.

Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao. Is robustness the cost of accuracy? - A comprehensive study on the robustness of 18 deep image classification models. In *European Conference on Computer Vision, ECCV*, pp. 644–661, 2018.

Dawei Sun, Anbang Yao, Aojun Zhou, and Hao Zhao. Deeply-supervised knowledge synergy. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 6997–7006, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR*, 2014.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *8th International Conference on Learning Representations, ICLR*, 2020a.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European conference on computer vision*, pp. 776–794. Springer, 2020b.

Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, Peter Vajda, and Joseph E. Gonzalez. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 12962–12971, 2020.

Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 10734–10742, 2019.

Cihang Xie and Alan L. Yuille. Intriguing properties of adversarial training at scale. In *8th International Conference on Learning Representations, ICLR*, 2020.

Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L. Yuille. Improving transferability of adversarial examples with input diversity. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 2730–2739, 2019.

Guoyang Xie, Jinbao Wang, Guo Yu, Feng Zheng, and Yaochu Jin. Tiny adversarial mulit-objective oneshot neural architecture search. *arXiv preprint arXiv:2103.00363*, 2021.

Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. Pc-darts: Partial channel connections for memory-efficient architecture search. In *ICLR*. OpenReview.net, 2020.

Shaokai Ye, Xue Lin, Kaidi Xu, Sijia Liu, Hao Cheng, Jan-Henrik Lambrechts, Huan Zhang, Aojun Zhou, Kaisheng Ma, and Yanzhi Wang. Adversarial robustness vs. model compression, or both? In *IEEE/CVF International Conference on Computer Vision, ICCV*, pp. 111–120, 2019.

Junho Yim, Donggyu Joo, Ji-Hoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 7130–7138, 2017.

Zhixiong Yue, Baijiong Lin, Yu Zhang, and Christy Liang. Effective, efficient and robust neural architecture search. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2022.

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *5th International Conference on Learning Representations, ICLR*, 2017.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7472–7482. PMLR, 2019a.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7472–7482. PMLR, 2019b.

Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8697–8710, 2018.

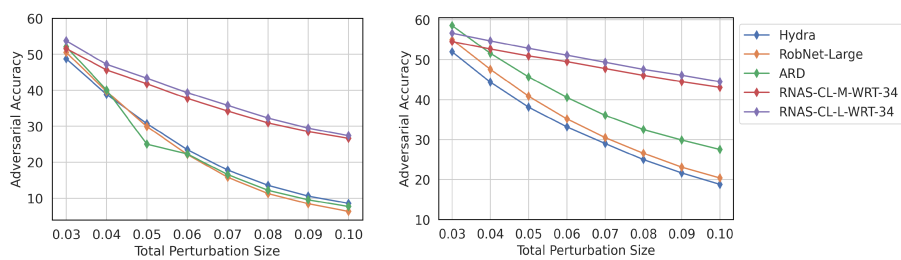
A APPENDIX

A.1 HYPER-PARAMETERS

In this section, we discuss the hyper-parameters used for our training. For both datasets, we use SGD optimizer. For CIFAR-10, default values of momentum and weight decay are set to 0.9 and $2e-4$, respectively. The batch size is set to 128. We train our model for 100 epochs in both the searching and training phases. The learning rate is initialized as 0.1, and reduced by a factor of 10 after the 75^{th} and the 90^{th} epoch. For ImageNet-100, default values of momentum and weight decay are set to 0.9 and $4e-5$, respectively. The batch size is set to 256. The learning rate is initialized as 0.05 and annealed down to zero following a cosine schedule. After the search stage which takes 100 epochs, the searched architecture is trained from scratch using RNAS-CL train loss (3) for 200 epochs. Following the settings of FBNetV2, the temperature (τ) in Gumbel-Softmax is initialized as 5.0 and exponentially annealed by $e^{-0.045}$ every epoch in the search phase. The hyper-parameter λ_s and λ_t in (2, 3) is selected from a candidate set $\{0.01, 0.1, 0.1, 1.0, 10, 100\}$. Both λ_s and λ_t are set to 1.0 for all experiments. In the search phase for each batch, we use 80% of the data to optimize the model weights and the remaining 20% data to optimize architectural weights which are Gumbel weights.

A.2 COMPARISON AGAINST VARIOUS PERTURBATION BUDGET

To further illustrate the effectiveness of RNAS-CL, we compare RNAS-CL with previously proposed defense mechanisms against various perturbation budgets. In Figure 3, we compare various methods against PGD and FSGM attacks. For both attacks, RNAS-CL outperforms its counterparts at all perturbations. RNAS-CL significantly outperforms other methods as perturbation size increases. For $\epsilon = 0.1$, RNAS-CL exceeds other methods by ∼20% for both PGD and FSGM attacks.



(a) Adversarial Accuracy on CIFAR-10 under different PGD total perturbationsize

(b) Adversarial Accuracy on CIFAR-10 under different FGSM total perturbationsize

Figure 3: Robustness evaluation under different perturbation sizes for PGD and FGSM attacks.

A.3 COMPARE EFFICIENT AND ROBUST IMAGENET-100 MODELS

Method	Clean	PGD^{20}	# Params (M)	MACs (M)
Hydra (ResNet-18) - 90% (Sehwag et al., 2020)	59.96	29.79	**1.1**	1200
LWM (ResNet-18) - 90% (Han et al., 2015)	59.02	27.67	1.1	1200
RNAS-CL-I-R-18	85.22	8.3	3.94	**241.98**
RNAS-CL-I-R-50	**85.98**	5.08	3.96	244.76
RNAS-CL-I-WRT-50	85.46	3.36	4.01	255.37
RNAS-CL-I-R-18 + TRADES	78.94	29.02	3.94	**241.98**
RNAS-CL-I-R-50 + TRADES	79.95	**32.44**	3.96	244.76
RNAS-CL-I-WRT-50 + TRADES	79.42	28.06	4.01	255.37

Table 2: Performance of various efficient and robust methods on ImageNet-100 dataset. Clean Acc and Adv Acc are the same as that in Table 1. All MACs were calculated without special hardware (Han et al., 2016) or special software (Park et al., 2017)

We compare RNAS-CL to adversarially robust pruning methods on ImageNet-100 dataset, with results shown in Table 2. RNAS-CL models are trained with three different robust teachers, ResNet-18, ResNet-50, and WideResNet-50, with the ImageNet pre-trained (Engstrom et al., 2019) being the robust teacher. It is observed that RNAS-CL models consistently exceed other models by $\sim 25\%$ in terms of clean accuracy while exibiting adversarial robustness. In Table 2, both Hydra and LWM were adversarially trained using TRADES (Zhang et al., 2019a). For a fair comparison, after the regular training stage without TRADES, we retrain our RNAS-CL models with the TRADES optimization objective. We replace the cross-entropy term in (3) by the TRADES optimization objective. With such training, RNAS-CL achieves similar or higher adversarial accuracy while significantly outperforming Hydra and LWM in clean accuracy with only a fraction of MACs.

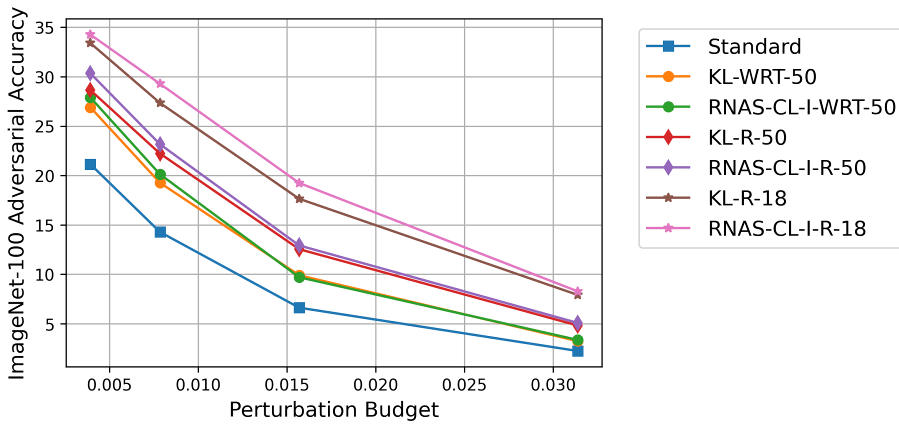


Figure 4: Adversarial accuracy of various models at various perturbation budgets on the ImageNet-100 dataset.

We further study adversarial accuracy at various perturbation budgets for three different teacher models. As illustrated in Figure 4, RNAS-CL exceeds its counterpart in adversarial accuracy at various perturbation budgets for all teacher models on the ImageNet-100 dataset. This demonstrates the significance of cross-layer connections in RNAS-CL.

A.4 Architecture

In this section, we discuss architectures for various proposed super-nets used in RNAS-CL for CIFAR-10 and ImageNet-100 datasets. Table 3 describes the super-nets used for CIFAR-10. We use super-nets with three blocks. Super-nets used for ImageNet-100 are described in Table 4. For ImageNet-100, the number of blocks varies from 3 to 5.

Search Space for CIFAR-10				
Search Space	Depth	Stage 1	Stage 2	Stage 3
RNAS-CL-S3	3-3-3	16, 12	32, 28, 24, 20	64, 60, 56, 52
RNAS-CL-S5	5-5-5	16, 12	32, 28, 24, 20	64, 60, 56, 52
RNAS-CL-S7	7-7-7	16, 12	32, 28, 24, 20	64, 60, 56, 52
RNAS-CL-M	9-7-1	80, 76	160, 156, 152, 148	128, 124, 120, 116
RNAS-CL-L	9-7-1	160, 156	320, 316, 312, 308	256, 252, 248, 244

Table 3: The table describes the search space for CIFAR-10. Depth represents the depth of each stage. For example, 3-3-3 represents three convolution blocks in each stage. All search spaces have three stages. Stage 1, Stage 2, and Stage 3 represent the filter choices for their respective stages. For example, at stage 3 of RNAS-CL-S3, for each convolution block, we search between 4 output channels (64, 60, 56, 52).

Search Space for ImageNet-100						
Search Space	Depth	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
RNAS-CL-IS	3-3-3	28, 24, 20, 16	40, 36, 32, 28	96, 88, 80, 72, 64, 56, 48		
RNAS-CL-IM	3-3-3-4	28, 24, 20, 16	40, 36, 32, 28	96, 88, 80, 72, 64, 56, 48	128 120, 108, 100, 92, 84, 76, 68	
RNAS-CL-I	3-3-3-4-4	28, 24, 20, 16	40, 36, 32, 28	96, 88, 80, 72, 64, 56, 48	128 120, 108, 100, 92, 84, 76, 68	216, 208, 200, 192, 184,176, 168, 160, 152, 144,136, 128, 120, 108

Table 4: The table describes the search space for ImageNet-100. Similar to Table 3, depth represents the depth of each stage. For ImageNet-100, we have up to 5 stages. Stage 1, Stage 2, Stage 3, Stage 4, and Stage 5 represent the filter choices for their respective stages. For example, in stage 1, for each convolution block, we search for its channel within 4 output channel options $(28, 24, 20, 16)$.

A.5 Architecture Search by FBNetV2

RNAS-CL builds both an efficient and adversarially robust deep learning model. In this work, we use the training paradigm of FBNetV2 to search for efficient models. In Figure 5, we illustrate the searching process for neural architecture at a single convolution layer. Each filter choice is attached with a Gumbel weight. These Gumbel weights are optimized to select an efficient model.

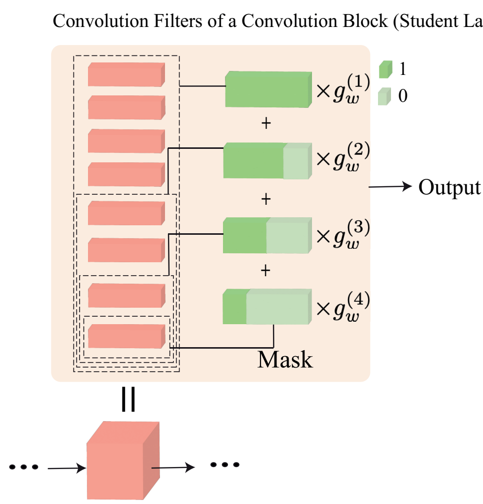


Figure 5: Illustration of searching for the neural architecture of each layer of student model using the searching mechanism in FBNetV2. g_w^i represents gumbel weights associated with each mask.